Package 'ggstatsplot'

October 22, 2026

Type Package

Title 'ggplot2' Based Plots with Statistical Details

Version 1.1.4

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

Description Extension of 'ggplot2', 'ggstatsplot' creates graphics with details from statistical tests included in the plots themselves. It provides an easier syntax to generate information-rich plots for statistical analysis of continuous (violin plots, scatterplots, histograms, dot plots, dot-and-whisker plots) or categorical (pie and bar charts) data. Currently, it supports the most common types of statistical approaches and tests: parametric, nonparametric, robust, and Bayesian versions of t-test/ANOVA, correlation analyses, contingency table analysis, meta-analysis, and regression analyses. References: Patil (2021) <doi:10.21105/joss.03167>.

License MIT + file LICENSE

URL https://www.indrapatil.com/ggstatsplot/, https://github.com/IndrajeetPatil/ggstatsplot

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

Depends R (>= 4.5.0)

Imports correlation (>= 0.8.8), datawizard (>= 1.4.0), dplyr (>= 1.2.1), forcats (>= 1.0.1), ggcorrplot (>= 0.3.0), ggplot2 (>= 4.0.3), ggrepel (>= 0.9.8), ggside (>= 0.4.1), ggsignif (>= 0.6.4), glue (>= 1.8.1), insight (>= 1.5.4), paletteer (>= 1.7.0), parameters (>= 0.29.3), patchwork (>= 1.3.2), performance (>= 0.18.2), purrr (>= 1.2.2), rlang (>= 1.3.0), statsExpressions (>= 2.2.1), tidyr (>= 1.3.2), utils

Suggests afex (>= 1.5-1), BayesFactor (>= 0.9.12-4.8), bayestestR (>= 0.19.0), gapminder (>= 1.0.1), knitr (>= 1.52), lme4 (>= 2.0-6), MASS (>= 7.3-66), metaBMA (>= 0.6.9), metafor (>= 5.2-1), metaplus (>= 1.0-10), patrick (>= 0.3.1), psych (>= 2.6.9), rmarkdown (>= 2.32), rstantools (>= 2.7.1), stats, survival (>= 3.8-12), testthat (>= 3.3.2), tibble (>= 3.3.1), vdiffr (>= 1.0.9), withr (>= 3.0.3), WRS2 (>= 1.1-7)

VignetteBuilder knitr

Config/Needs/check anthonynorth/roxyglobals, any::patrick, any::vdiffr

Config/roxygen2/version 8.1.1

Config/roxyglobals/unique TRUE

Config/testthat/edition 3

Config/testthat/parallel true

Config/testthat/start-first ggbarstats, ggpiestats, pairwise-ggsignif

Encoding UTF-8

Language en-US

LazyData true

NeedsCompilation no

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

Repository CRAN

Date/Publication 2026-10-22 13:10:02 UTC

Contents

bugs_long . . . 3
combine_plots . . . 4
extract_stats . . . 5
ggbarstats . . . 7
ggbetweenstats . . . 12
ggcoefstats . . . 20
ggcorrmat . . . 24
ggdotplotstats . . . 28
gghistostats . . . 32
ggpiestats . . . 36
ggscatterstats . . . 41
ggwithinstats . . . 46
grouped_ggbarstats . . . 54
grouped_ggbetweenstats . . . 57
grouped_ggcorrmat . . . 61
grouped_ggdotplotstats . . . 64
grouped_gghistostats . . . 67
grouped_ggpiestats . . . 70
grouped_ggscatterstats . . . 73
grouped_ggwithinstats . . . 76
iris_long . . . 80
movies_long . . . 81
theme_ggstatsplot . . . 82
Titanic_full . . . 82

Index **84**

bugs_long *Tidy version of the "Bugs" dataset*

Description

Tidy version of the "Bugs" dataset

Usage

```
bugs_long
```

Format

A data frame with 372 rows and 6 variables

- subject. Dummy identity number for each participant.
- gender. Participant's gender (Female, Male).
- region. Region of the world the participant was from.
- education. Level of education.
- condition. Condition of the experiment the participant gave rating for (**LDLF**: low disgustingness and low frighteningness; **HDLF**: high disgustingness and low frighteningness; **LDHF**: low disgustingness and high frighteningness; **HDHF**: high disgustingness and high frighteningness).
- desire. The desire to kill an arthropod was indicated on a scale from 0 to 10.

Details

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

References

Ryan, R. S., Wilde, M., & Crist, S. (2013). Compared to a small, supervised lab experiment, a large, unsupervised web-based experiment on a previously unknown effect has benefits that outweigh its potential costs. *Computers in Human Behavior*, *29*(4), 1295-1301.

Examples

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

combine_plots *Combining and arranging multiple plots in a grid*

Description

Wrapper around `patchwork::wrap_plots()` that returns a combined grid of plots with annotations. It is mostly useful with {ggstatsplot} plots and is exported only for backward compatibility; to create a grid of plots, it is **highly recommended** that you use {patchwork} directly.

Usage

```
combine_plots(
  plotlist,
  plotgrid.args = list(),
  annotation.args = list(),
  guides = "collect",
  ...
)
```

Arguments

plotlist	A list containing ggplot objects.
plotgrid.args	A list of additional arguments passed to `patchwork::wrap_plots()`, except for guides, which has its own argument.
annotation.args	A list of additional arguments passed to `patchwork::plot_annotation()`.
guides	A string specifying how guides should be treated in the layout. 'collect' will collect guides below to the given nesting level, removing duplicates. 'keep' will stop collection at this level and let guides be placed alongside their plot. auto will allow guides to be collected if a upper level tries, but place them alongside the plot if not. If you modify default guide "position" with theme(legend.position=...) while also collecting guides you must apply that change to the overall patchwork (see example).
...	Currently ignored.

Value

A patchwork object: a combined plot with annotation labels.

Examples

```
library(ggplot2)

p1 <- ggplot(
  data = subset(iris, Species == "setosa"),
  aes(x = Sepal.Length, y = Sepal.Width)
) +
```

```
  geom_point() +
  labs(title = "setosa")

p2 <- ggplot(
  data = subset(iris, Species == "versicolor"),
  aes(x = Sepal.Length, y = Sepal.Width)
) +
  geom_point() +
  labs(title = "versicolor")

# combining the plot with a title and a caption
combine_plots(
  plotlist = list(p1, p2),
  plotgrid.args = list(nrow = 1),
  annotation.args = list(
    tag_levels = "a",
    title = "Dataset: Iris Flower dataset",
    subtitle = "Edgar Anderson collected this data",
    caption = "Note: Only two species of flower are displayed",
    theme = theme(
      plot.subtitle = element_text(size = 20),
      plot.title = element_text(size = 30)
    )
  )
)
```

`extract_stats` *Extracting data frames or expressions from* {ggstatsplot} *plots*

Description

Extracting data frames or expressions from {ggstatsplot} plots

Usage

```
extract_stats(p)

extract_subtitle(p)

extract_caption(p)
```

Arguments

p	A plot from {ggstatsplot}.

Details

These helpers retrieve the statistical data frames and expressions retained by {ggstatsplot} plots. If you only need tabular results, call the underlying analysis functions directly: {statsExpressions}

for statistical tests, parameters::model_parameters() for model coefficients, and performance::model_performance() for model fit indices.

ggcorrmat() does not retain statistical data frames, so all elements returned by extract_stats() are NULL. Use correlation::correlation() to obtain its statistical results as data.

Value

For extract_stats(), a list with the following elements, each either a data frame or NULL if not relevant for the given plot:

- subtitle_data: results of the test displayed in the subtitle
- caption_data: results of the test displayed in the caption (typically, the Bayesian test)
- pairwise_comparisons_data: results of pairwise comparisons
- descriptive_data: counts and percentages (ggpiestats() and ggbarstats())
- one_sample_data: proportion tests for each level of y (ggpiestats() and ggbarstats())
- tidy_data: tidy model parameters (ggcoefstats())
- glance_data: model performance indices (ggcoefstats())

For extract_subtitle() and extract_caption(), the statistical expression used in the subtitle or caption, respectively (or NULL if unavailable). User-supplied subtitle and caption text is not returned.

For plots from grouped_* functions, a list with one such element per plot.

Examples

```
set.seed(123)

# non-grouped plot
p1 <- ggbetweenstats(mtcars, cyl, mpg)

# grouped plot
p2 <- grouped_ggbarstats(Titanic_full, Survived, Sex, grouping.var = Age)

# extracting expressions -----------------------------

extract_subtitle(p1)
extract_caption(p1)

extract_subtitle(p2)
extract_caption(p2)

# extracting data frames -----------------------------

extract_stats(p1)

extract_stats(p2)
```

ggbarstats *Stacked bar charts with statistical tests*

Description

Bar charts for categorical data with statistical details included in the plot as a subtitle.

Usage

```
ggbarstats(
  data,
  x,
  y = NULL,
  counts = NULL,
  type = "parametric",
  paired = FALSE,
  results.subtitle = TRUE,
  label = "percentage",
  label.args = list(alpha = 1, fill = "white"),
  sample.size.label.args = list(size = 4),
  digits = 2L,
  proportion.test = results.subtitle,
  digits.perc = 0L,
  bf.message = TRUE,
  ratio = NULL,
  alternative = "two.sided",
  conf.level = 0.95,
  p.adjust.method = "holm",
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  legend.title = NULL,
  xlab = NULL,
  ylab = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  palette = "ggthemes::gdoc",
  ggplot.component = NULL,
  ...
)
```

Arguments

data	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from {dplyr} should be ungrouped before they are entered as data.

`x` The variable to use as the **rows** in the contingency table. Note that empty factor levels are *not* dropped: they are shown in the legend and counted (with zero observations) in the statistical test. Use `droplevels()` beforehand if this is not intended.

`y` The variable to use as the **columns** in the contingency table. If `NULL` (default), a one-sample proportion test (a goodness of fit test) is run for the `x` variable. Otherwise, an appropriate association test is run.

`counts` The variable in data containing counts, or `NULL` if each row represents a single observation.

`type` A character specifying the type of statistical approach:

- `"parametric"`
- `"nonparametric"`
- `"robust"`
- `"bayes"`

You can specify just the initial letter (e.g. `"np"` or `"bf"`). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as `"Bayes"`) falls back to `"parametric"` without a warning.

`paired` Logical indicating whether data came from a within-subjects or repeated measures design study (Default: `FALSE`). Only relevant for two-way tables (i.e., when `y` is supplied), and paired designs are only supported for frequentist tests (McNemar's test). For a two-way table with `type = "bayes"`, `paired` is ignored and the Bayesian test of independence for unpaired data is run instead. For one-way tables (`y = NULL`), `paired` has no effect.

`results.subtitle`
Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`label` Character that decides what each label displays: `"percentage"` (default), `"counts"`, or `"both"`.

`label.args` A list of additional aesthetic arguments passed to `ggplot2::geom_label()`.

`sample.size.label.args`
A list of additional aesthetic arguments passed to `ggplot2::geom_text()`.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`proportion.test`
Logical that decides whether a proportion test for `x` is carried out for each level of `y`. Defaults to `results.subtitle`. `ggbarstats()` displays only the *p*-values from this test. Always `FALSE` when `type = "bayes"`, when `y` is `NULL`, or when `x` has only one level.

`digits.perc` Number of decimal places for percentage labels (Default: `0L`).

bf.message	Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for non-Bayesian types and unpaired designs (paired = FALSE) (Default: TRUE).
ratio	A vector of proportions: the expected proportions for the proportion test (should sum to 1). Default is NULL, which means the null is equal theoretical proportions across the levels of the nominal variable. E.g., ratio = c(0.5, 0.5) for two levels, ratio = c(0.25, 0.25, 0.25, 0.25) for four levels, etc.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
conf.level	Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).
p.adjust.method	Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".
title	The text for the plot title.
subtitle	The text for the plot subtitle. Used only if results.subtitle = FALSE.
caption	The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see bf.message).
legend.title	Title text for the legend. If NULL (default), the name of the x variable is used.
xlab	Label for x axis. If NULL (default), the name of the y variable is used (no label if y is NULL), since bars are drawn for each level of y.
ylab	Label for y axis. If NULL (default), no label is shown.
ggtheme	A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirty can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyei removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.
palette	Name of the discrete palette in "package::palette" format to be used for coloring (via {paletteer}). Run View(paletteer::palettes_d_names) to see all available options.
ggplot.component	A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.
...	Currently ignored.

Details

Rows with a missing value in x, y, or counts are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggbarstats.html

Value

A ggplot object, which can be further modified with {ggplot2} functions. Use extract_stats() to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	geom used	argument for further modification
bars	`ggplot2::geom_col()`	`NA`
descriptive labels	`ggplot2::geom_label()`	`label.args`
sample size labels	`ggplot2::geom_text()`	`sample.size.label.args`

Contingency table analyses

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

There are no dedicated non-parametric or robust contingency table analyses. `type = "nonparametric"` and `type = "robust"` are accepted, but run the same frequentist tests as `type = "parametric"` (the "Parametric/Non-parametric" rows below).

Two-way table:

Hypothesis testing

Type	Design	Test	Function used
Parametric/Non-parametric	Unpaired	Pearson's chi-squared test	`stats::chisq.test()`
Bayesian	Unpaired	Bayesian Pearson's chi-squared test	`BayesFactor::contingencyTableBF()`
Parametric/Non-parametric	Paired	McNemar's chi-squared test	`stats::mcnemar.test()`
Bayesian	Paired	No	No

Effect size estimation

Type	Design	Effect size	CI available?	Function used
Parametric/Non-parametric	Unpaired	Cramer's *V*	Yes	`effectsize::cramers_v()`
Bayesian	Unpaired	Cramer's *V*	Yes	`effectsize::cramers_v()`
Parametric/Non-parametric	Paired	Cohen's *g*	Yes	`effectsize::cohens_g()`
Bayesian	Paired	No	No	No

Paired Bayesian analysis is not supported: for a two-way table with `type = "bayes"`, the `paired` argument is ignored and the unpaired Bayesian test is run instead.

One-way table:

Hypothesis testing

Type	Test	Function used

Parametric/Non-parametric	Goodness of fit chi-squared test	`stats::chisq.test()`
Bayesian	Bayesian Goodness of fit chi-squared test	(custom)

Effect size estimation

Type	Effect size	CI available?	Function used
Parametric/Non-parametric	Pearson's *C*	Yes	`effectsize::pearsons_c()`
Bayesian	No	No	No

Pairwise comparisons

When there is a two-way table, `x` has more than two levels, and `paired = FALSE`, pairwise contingency table analyses (Fisher's exact tests) are computed using `statsExpressions::pairwise_contingency_table()`. These results are **not** displayed in the plot: unlike the brackets in box/violin plots, pie and bar charts (which are designed to stay visually congruent) have no natural or established way to show pairwise comparisons. Retrieve them as a data frame with `extract_stats(plot)$pairwise_comparisons_data`.

See Also

`grouped_ggbarstats()`, `ggpiestats()`, `grouped_ggpiestats()`

Examples

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

# one sample goodness of fit proportion test
p <- ggbarstats(mtcars, vs)

# looking at the plot
p

# extracting details from statistical tests
extract_stats(p)

# association test (or contingency table analysis)
ggbarstats(mtcars, vs, cyl)

# with 3+ x levels, pairwise comparisons are available
ggbarstats(mtcars, cyl, am)

# Bayesian test
ggbarstats(mtcars, vs, cyl, type = "bayes")

# using pre-aggregated data with counts
ggbarstats(as.data.frame(Titanic), x = Survived, y = Sex, counts = Freq)
```

ggbetweenstats *Box/Violin plots for between-subjects comparisons*

Description

A combination of box and violin plots along with jittered data points for between-subjects designs with statistical details included in the plot as a subtitle.

Usage

```
ggbetweenstats(
  data,
  x,
  y,
  type = "parametric",
  pairwise.display = "significant",
  pairwise.alpha = 0.05,
  p.adjust.method = "holm",
  bf.prior = 0.707,
  bf.message = TRUE,
  results.subtitle = TRUE,
  xlab = NULL,
  ylab = NULL,
  caption = NULL,
  title = NULL,
  subtitle = NULL,
  digits = 2L,
  conf.level = 0.95,
  tr = 0.2,
  alternative = "two.sided",
  centrality.plotting = TRUE,
  centrality.type = type,
  centrality.point.args = list(size = 5, color = "darkred"),
  centrality.label.args = list(size = 3, nudge_x = 0.4, segment.linetype = 4,
    min.segment.length = 0),
 point.args = list(position = ggplot2::position_jitterdodge(dodge.width = 0.6), alpha =
    0.4, size = 3, stroke = 0, na.rm = TRUE),
  boxplot.args = list(width = 0.3, alpha = 0.2, na.rm = TRUE),
  violin.args = list(width = 0.5, alpha = 0.2, na.rm = TRUE),
  ggsignif.args = list(textsize = 3, tip_length = 0.01, na.rm = TRUE),
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  palette = "ggthemes::gdoc",
  ggplot.component = NULL,
  ...
)
```

Arguments

data	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from {dplyr} should be ungrouped before they are entered as data.
x	The grouping (or independent) variable from data.
y	The response (or outcome or dependent) variable from data.
type	A character specifying the type of statistical approach: • "parametric" • "nonparametric" • "robust" • "bayes" You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.
pairwise.display	Decides *which* pairwise comparisons to display. Available options are: • "significant" (abbreviation accepted: "s") • "non-significant" (abbreviation accepted: "ns") • "all" • "none" Use this to keep the plot uncluttered when many groups are compared. Pairwise comparisons are carried out (and displayed) only when x has three or more levels. For type = "bayes", there are no *p*-values, so all comparisons are displayed unless this is "none".
pairwise.alpha	Numeric alpha threshold used to decide which pairwise comparisons are displayed when pairwise.display = "significant" or pairwise.display = "non-significant" (Default: 0.05). It is compared against *p*-values adjusted using p.adjust.method.
p.adjust.method	Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".
bf.prior	A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.
bf.message	Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: TRUE).
results.subtitle	Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: TRUE). Setting this to FALSE also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`xlab` Label for `x` axis variable. If `NULL` (default), variable name for `x` will be used.

`ylab` Label for `y` axis variable. If `NULL` (default), variable name for `y` will be used.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`title` The text for the plot title.

`subtitle` The text for the plot subtitle. Used only if `results.subtitle = FALSE`.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

`tr` Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`alternative` a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.

`centrality.plotting`

Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use `centrality.type` to choose a different centrality measure.

`centrality.type`

Decides which centrality measure is displayed. Defaults to the same value as `type`. Can be one of:

- `"parametric"` (for **mean**)
- `"nonparametric"` (for **median**)
- `"robust"` (for **trimmed mean**)
- `"bayes"` (for **MAP estimator**)

As with `type`, abbreviations are accepted.

`centrality.point.args`, `centrality.label.args`

A list of additional aesthetic arguments passed to `ggplot2::geom_point()` and `ggrepel::geom_label_repel()`, respectively, which display the centrality measure.

`point.args` A list of additional aesthetic arguments passed to `ggplot2::geom_point()`.

`boxplot.args` A list of additional aesthetic arguments passed to `ggplot2::geom_boxplot()`. By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). Use `coef` to change this, e.g., `boxplot.args = list(coef = 3)` extends the whiskers to the most extreme observations within 3 * IQR of the hinges.

violin.args	A list of additional aesthetic arguments passed to ggplot2::geom_violin().
ggsignif.args	A list of additional aesthetic arguments passed to ggsignif::geom_signif().
ggtheme	A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirty can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyei removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.
palette	Name of the discrete palette in "package::palette" format to be used for coloring (via {paletteer}). Run View(paletteer::palettes_d_names) to see all available options.
ggplot.component	A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.
...	Currently ignored.

Details

Rows with a missing value in either x or y are removed before plotting and analysis, and unused levels of x are dropped.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggbetweenstats.html

Value

A ggplot object, which can be further modified with {ggplot2} functions. Use extract_stats() to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	geom used	argument for further modification
raw data	ggplot2::geom_point()	point.args
box plot	ggplot2::geom_boxplot()	boxplot.args
density plot	ggplot2::geom_violin()	violin.args
centrality measure point	ggplot2::geom_point()	centrality.point.args
centrality measure label	ggrepel::geom_label_repel()	centrality.label.args
pairwise comparisons	ggsignif::geom_signif()	ggsignif.args

Statistical defaults

This function uses statistically justified defaults that are not user-configurable:

- For **parametric tests**, effect sizes use Hedges' *g* for two groups and omega-squared for three or more groups. These estimators correct for positive bias, which can be especially relevant in small samples.

- **Welch's *t*-test and one-way test** are used instead of Student's versions, so equal variances are not assumed.

Users who need non-default values for these settings can call `{statsExpressions}` directly.

Centrality measures

Type	Measure	Function used
Parametric	mean	`datawizard::describe_distribution()`
Non-parametric	median	`datawizard::describe_distribution()`
Robust	trimmed mean	`datawizard::describe_distribution()`
Bayesian	MAP	`datawizard::describe_distribution()`

Two-sample tests

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Between-subjects:

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	2	Student's or Welch's *t*-test	`stats::t.test()`
Non-parametric	2	Mann-Whitney *U* test	`stats::wilcox.test()`
Robust	2	Yuen's test for trimmed means	`WRS2::yuen()`
Bayesian	2	Student's *t*-test	`BayesFactor::ttestBF()`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Function used
Parametric	2	Cohen's *d*, Hedges' *g*	Yes	`effectsize:`
Non-parametric	2	*r* (rank-biserial correlation)	Yes	`effectsize:`
Robust	2	Algina-Keselman-Penfield robust standardized difference	Yes	`WRS2::akp.e`
Bayesian	2	difference	Yes	`bayestestR:`

Within-subjects:

Paired data: See `complete_subjects()` for subject matching, duplicate observations, and removal of incomplete subjects.

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	2	Student's *t*-test	`stats::t.test()`
Non-parametric	2	Wilcoxon signed-rank test	`stats::wilcox.test()`
Robust	2	Yuen's test on trimmed means for dependent samples	`WRS2::yuend()`
Bayesian	2	Student's *t*-test	`BayesFactor::ttestBF()`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Function used
Parametric	2	Cohen's *d*, Hedges' *g*	Yes	`effectsize:`
Non-parametric	2	*r* (rank-biserial correlation)	Yes	`effectsize:`
Robust	2	Algina-Keselman-Penfield robust standardized difference	Yes	`WRS2::wmcpA`
Bayesian	2	difference	Yes	`bayestestR:`

One-way ANOVA

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Between-subjects:

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	> 2	Fisher's or Welch's one-way ANOVA	`stats::oneway.test()`
Non-parametric	> 2	Kruskal-Wallis one-way ANOVA	`stats::kruskal.test()`
Robust	> 2	Heteroscedastic one-way ANOVA for trimmed means	`WRS2::t1way()`
Bayesian	> 2	Fisher's ANOVA	`BayesFactor::anovaBF()`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Function used
Parametric	> 2	partial eta-squared, partial omega-squared	Yes	`effectsize::omega_square`
Non-parametric	> 2	rank epsilon squared	Yes	`effectsize::rank_epsilo`
Robust	> 2	Explanatory measure of effect size	Yes	`WRS2::t1way()`
Bayesian	> 2	Bayesian R-squared	Yes	`performance::r2_bayes()`

Within-subjects:

Paired data: See `complete_subjects()` for subject matching, duplicate observations, and removal of incomplete subjects.

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	> 2	One-way repeated measures ANOVA	`afex::aov_ez`
Non-parametric	> 2	Friedman rank sum test	`stats::friedm`
Robust	> 2	Heteroscedastic one-way repeated measures ANOVA for trimmed means	`WRS2::rmanov`
Bayesian	> 2	One-way repeated measures ANOVA	`BayesFactor::`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Funct
Parametric	> 2	partial eta-squared, partial omega-squared	Yes	`effec`
Non-parametric	> 2	Kendall's coefficient of concordance	Yes	`effec`
Robust	> 2	Algina-Keselman-Penfield robust standardized difference average	Yes	`WRS2::`
Bayesian	> 2	Bayesian R-squared	Yes	`perfo`

Pairwise comparison tests

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Between-subjects:

Hypothesis testing

Type	Equal variance?	Test	*p*-value adjustment?	Function used
Parametric	No	Games-Howell test	Yes	`PMCMRplus::gamesHowellTest()`
Parametric	Yes	Student's *t*-test	Yes	`stats::pairwise.t.test()`
Non-parametric	No	Dunn test	Yes	`PMCMRplus::kwAllPairsDunnTest`
Robust	No	Yuen's trimmed means test	Yes	`WRS2::lincon()`
Bayesian	NA	Student's *t*-test	NA	`BayesFactor::ttestBF()`

Effect size estimation

Not supported.

Within-subjects:

Paired data: See `complete_subjects()` for subject matching, duplicate observations, and removal of incomplete subjects.

Hypothesis testing

Type	Test	*p*-value adjustment?	Function used
Parametric	Student's *t*-test	Yes	`stats::pairwise.t.test()`
Non-parametric	Durbin-Conover test	Yes	`PMCMRplus::durbinAllPairsTest()`
Robust	Yuen's trimmed means test	Yes	`WRS2::rmmcp()`
Bayesian	Student's *t*-test	NA	`BayesFactor::ttestBF()`

Effect size estimation
Not supported.

See Also

grouped_ggbetweenstats(), ggwithinstats(), grouped_ggwithinstats()

Examples

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

p <- ggbetweenstats(mtcars, am, mpg)
p

# extracting details from statistical tests
extract_stats(p)

# show non-significant pairwise comparisons (needs 3+ groups for ggsignif)
ggbetweenstats(mtcars, cyl, mpg, pairwise.display = "non-significant")

# show all pairwise comparisons
ggbetweenstats(mtcars, cyl, mpg, pairwise.display = "all")

# use a stricter alpha threshold for significant pairwise comparisons
ggbetweenstats(mtcars, cyl, mpg, pairwise.alpha = 0.001)

# modifying defaults
ggbetweenstats(
  morley,
  x    = Expt,
  y    = Speed,
  type = "robust",
  xlab = "The experiment number",
  ylab = "Speed-of-light measurement"
)

# you can remove a specific geom to reduce complexity of the plot
ggbetweenstats(
  mtcars,
  am,
  wt,
  # to remove violin plot
  violin.args = list(width = 0, linewidth = 0, colour = NA),
  # to remove boxplot
  boxplot.args = list(width = 0),
  # to remove points
  point.args = list(alpha = 0)
)
```

`ggcoefstats` *Dot-and-whisker plots for regression analyses*

Description

Plot with the regression coefficients' point estimates as dots with confidence interval whiskers and other statistical details included as labels.

Although the statistical models displayed in the plot may differ based on the class of models being investigated, there are a few aspects of the plot that will be invariant across models:

- Each dot represents an **estimate**, and its whiskers the **confidence interval** (`95%` by default). Estimates can be effect sizes (for tests based on the `F`-statistic) or regression coefficients (for tests based on the `t`-, `chi^2`-, or `z`-statistic), etc. The confidence intervals can be asymmetric if bootstrapping was used.
- The label attached to each dot provides details from the statistical test, typically the estimate, statistic, and *p*-value.
- The caption contains model diagnostics, if available, that can be useful for model selection: the smaller the Akaike's Information Criterion (**AIC**) and the Bayesian Information Criterion (**BIC**) values, the "better" the model.

Usage

```
ggcoefstats(
  x,
  statistic = NULL,
  conf.int = TRUE,
  conf.level = 0.95,
  digits = 2L,
  exclude.intercept = FALSE,
  effectsize.type = "omega",
  meta.analytic.effect = FALSE,
  meta.type = "parametric",
  bf.message = TRUE,
  sort = "none",
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  only.significant = FALSE,
  point.args = list(size = 3, color = "blue", na.rm = TRUE),
  errorbar.args = list(width = 0, na.rm = TRUE),
  vline = TRUE,
  vline.args = list(linewidth = 1, linetype = "dashed"),
  stats.labels = TRUE,
  stats.label.color = NULL,
```

```
  stats.label.args = list(size = 3, direction = "y", min.segment.length = 0, na.rm =
    TRUE),
  palette = "ggthemes::gdoc",
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  ...
)
```

Arguments

x	A model object to be tidied, or a tidy data frame from a regression model. Function internally uses parameters::model_parameters() to get a tidy data frame. If a data frame, it *must* contain a column named estimate (estimates of coefficients or other quantities of interest). A term column (names of predictors) is recommended; if absent, terms are named term_1, term_2, etc. Columns conf.low and conf.high are needed for error bars, statistic and p.value for labels, and std.error for meta-analysis.
statistic	Relevant statistic for the model ("t", "f", "z", or "chi") in the label. Relevant only if x is a *data frame*; if it is NULL (default), no labels are shown for data frames.
conf.int	Logical that decides whether to display confidence intervals as error bars (Default: TRUE).
conf.level	Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).
digits	Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).
exclude.intercept	Logical that decides whether the intercept should be excluded from the plot (Default: FALSE).
effectsize.type	Effect size for ANOVA-like objects, passed as the es_type argument of parameters::model_parameter (Default: "omega", the unbiased estimator).
meta.analytic.effect	Logical that decides whether to display a subtitle with results from a meta-analysis via linear (mixed-effects) models (Default: FALSE). If TRUE, subtitle is ignored. This is mostly relevant if a data frame with estimates and their standard errors is entered.
meta.type	Type of statistics used to carry out random-effects meta-analysis. If "parametric" (default), metafor::rma() will be used. If "robust", metaplus::metaplus() will be used. If "bayes", metaBMA::meta_random() will be used.
bf.message	Logical that decides whether results from running a Bayesian meta-analysis assuming that the effect size *d* varies across studies with standard deviation *t* (i.e., a random-effects analysis) are displayed in the caption. Relevant only if meta.analytic.effect = TRUE and meta.type = "parametric" (Default: TRUE).

sort	If "none" (default), do not sort; if "ascending" or "descending", sort by increasing or decreasing coefficient value, respectively.
xlab	Label for x axis. If NULL (default), "estimate" is used.
ylab	Label for y axis. If NULL (default), "term" is used.
title	The text for the plot title.
subtitle	The text for the plot subtitle. Ignored if meta.analytic.effect = TRUE.
caption	The text for the plot caption. It will be replaced by model fit indices (AIC and BIC), if available, or by the Bayesian meta-analysis results (see bf.message).
only.significant	If TRUE, only stats labels for significant effects ($p < 0.05$) are shown (Default: FALSE). This can be helpful when a large number of regression coefficients are to be displayed in a single plot.
point.args	A list of additional aesthetic arguments passed to ggplot2::geom_point().
errorbar.args	A list of additional arguments passed to ggplot2::geom_errorbar().
vline	Logical that decides whether to display a vertical line at 0 (Default: TRUE).
vline.args	A list of additional arguments passed to ggplot2::geom_vline().
stats.labels	Logical that decides whether the statistic and p-value for each coefficient are attached to its dot as a label using {ggrepel} (Default: TRUE).
stats.label.color	Color for the labels: either a single color or a vector with one color per term. If set to NULL (default), colors will be chosen from the specified palette.
stats.label.args	A list of additional arguments passed to ggrepel::geom_label_repel().
palette	Name of the discrete palette in "package::palette" format to be used for coloring (via {paletteer}). Run View(paletteer::palettes_d_names) to see all available options.
ggtheme	A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirty can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyei removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.
...	Additional arguments passed to parameters::model_parameters().

Details

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggcoefstats.html

Value

A ggplot object, which can be further modified with {ggplot2} functions. Use extract_stats() to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	geom used	argument for further modification
regression estimate	`ggplot2::geom_point()`	`point.args`
error bars	`ggplot2::geom_errorbar()`	`errorbar.args`
vertical line	`ggplot2::geom_vline()`	`vline.args`
label with statistical details	`ggrepel::geom_label_repel()`	`stats.label.args`

Random-effects meta-analysis

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Hypothesis testing and effect size estimation

Type	Test	Effect size	CI available?	Function used
Parametric	Meta-analysis via random-effects models	*beta*	Yes	`metafor::rma()`
Robust	Meta-analysis via robust random-effects models	*beta*	Yes	`metaplus::metaplus()`
Bayesian	Meta-analysis via Bayesian random-effects models	*beta*	Yes	`metaBMA::meta_random()`

Note

1. For meta-analysis, you will be asked to install the needed package (`{metafor}`, `{metaplus}`, or `{metaBMA}`) if it is unavailable.
2. Rows with a missing `estimate` are always removed. If labels are requested, rows where `statistic`, `std.error`, or `p.value` is missing are kept but get no label.
3. New model classes are frequently added to `{easystats}` packages, so it can help to install their development versions with `easystats::install_latest()`.

Examples

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

# model object
mod <- lm(formula = mpg ~ cyl * am, data = mtcars)

# creating a plot
p <- ggcoefstats(mod)

# looking at the plot
p
```

```
# extracting details from statistical tests
extract_stats(p)

# exclude intercept from the plot
ggcoefstats(mod, exclude.intercept = TRUE)

# only show significant labels
ggcoefstats(mod, only.significant = TRUE)

# ANOVA model (F-statistic)
ggcoefstats(aov(mpg ~ cyl * am, data = mtcars))

# a tidy data frame can also be passed directly (model-free use)
ggcoefstats(data.frame(term = c("a", "b", "c"), estimate = c(0.5, -0.2, 1.1)))

# without a `term` column (auto-generated)
ggcoefstats(data.frame(estimate = c(0.5, -0.2, 1.1)))

# tidy data frames can also include stats-label inputs directly
df_tidy <- parameters::model_parameters(stats::lm(wt ~ am * cyl, mtcars), ci = 0.95)
names(df_tidy) <- c(
  "term", "estimate", "std.error", "conf.level", "conf.low",
  "conf.high", "statistic", "df.error", "p.value"
)
df_tidy$p.value[2L] <- 0.42

ggcoefstats(
  df_tidy,
  statistic = "t",
  only.significant = TRUE,
  stats.label.color = c("firebrick", "grey50", "forestgreen", "navy")
)


# further arguments can be passed to `parameters::model_parameters()`
library(lme4)
ggcoefstats(lmer(Reaction ~ Days + (Days | Subject), sleepstudy), effects = "fixed")
```

---

ggcorrmat *Visualization of a correlation matrix*

Description

Correlation matrix containing results from pairwise correlation tests. If you want a data frame of (grouped) correlation matrix, use `correlation::correlation()` instead. It can also do grouped analysis when used with output from `dplyr::group_by()`.

Usage

```
ggcorrmat(
  data,
  cor.vars = NULL,
  cor.vars.names = NULL,
  matrix.type = "upper",
  type = "parametric",
  tr = 0.2,
  partial = FALSE,
  digits = 2L,
  sig.level = 0.05,
  conf.level = 0.95,
  bf.prior = 0.707,
  p.adjust.method = "holm",
  colors = c("#EA4335", "white", "#4285F4"),
  pch = "cross",
 ggcorrplot.args = list(method = "square", outline.color = "black", pch.cex = 14),
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  ggplot.component = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  ...
)
```

Arguments

`data`	A data frame from which variables specified are to be taken.
`cor.vars`	Variables for which the correlation matrix is to be computed and visualized, specified using `{tidyselect}` syntax (e.g., `c(var1, var2)` or `starts_with("Sepal")`). If `NULL` (default), all numeric variables from `data` will be used.
`cor.vars.names`	Optional list of names to be used for `cor.vars`. The names should be entered in the same order.
`matrix.type`	Character, `"upper"` (default), `"lower"`, or `"full"`, to display the upper triangular, lower triangular, or full matrix, respectively.
`type`	A character specifying the type of statistical approach: • `"parametric"` • `"nonparametric"` • `"robust"` • `"bayes"` You can specify just the initial letter (e.g. `"np"` or `"bf"`). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as `"Bayes"`) falls back to `"parametric"` without a warning.
`tr`	Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`partial` Can be `TRUE` for partial correlations. For Bayesian partial correlations, "full" instead of pseudo-Bayesian partial correlations (i.e., Bayesian correlation based on frequentist partialization) are returned.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`sig.level` Significance level (Default: `0.05`). If the (adjusted; see `p.adjust.method`) *p*-value is bigger than `sig.level`, then the corresponding correlation coefficient is regarded as insignificant and flagged as such in the plot. Ignored for `type = "bayes"`.

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: `"medium"`, `"wide"`, and `"ultrawide"`, corresponding to *r* scale values of `1/2`, `sqrt(2)/2`, and `1`, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

`p.adjust.method` Adjustment method for *p*-values for multiple comparisons. Possible methods are: `"holm"` (default), `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, `"none"`.

`colors` A character vector of exactly three colors for the gradient: low (negative correlations), mid (zero), and high (positive correlations). Must be a **diverging** palette so that the sign of the correlation is visually obvious. Default: `c("#EA4335", "white", "#4285F4")` (red-white-blue).

`pch` Decides the point shape to be used for insignificant correlation coefficients. Default: `pch = "cross"`. The caption explaining the insignificance marker is added only for `pch = "cross"` (or `4`).

`ggcorrplot.args` A list of additional (mostly aesthetic) arguments passed to `ggcorrplot::ggcorrplot()`. Do not include the following arguments, which are set internally: `corr`, `p.mat`, `sig.level`, `ggtheme`, `colors`, `type`, `lab`, `pch`, `legend.title`, `digits`.

`ggtheme` A {ggplot2} theme (Default: `theme_ggstatsplot()`). Any {ggplot2} theme (e.g., `ggplot2::theme_bw()`) or theme from an extension package (e.g., `ggthemes::theme_fivethirty` can be used. Note that some themes remove plot details. For example, `ggthemes::theme_fivethirtyei` removes the secondary Y-axis, which `ggbetweenstats()` uses to show details about pairwise comparisons.

`ggplot.component` A {ggplot2} component, or a list of components, to add to the plot (Default: `NULL`). This is mostly helpful for the `grouped_` variants, where it is applied to each individual plot.

`title` The text for the plot title.

`subtitle` The text for the plot subtitle.

`caption` The text for the plot caption. If the insignificance marker caption is shown (see `pch`), this text is displayed above it.

... Additional arguments (currently ignored).

Details

Missing values are handled using pairwise deletion (each correlation uses all complete pairs of observations). If sample sizes then differ across pairs, the legend shows their minimum, mode, and maximum. Partial correlations use only complete cases.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggcorrmat.html

Value

A `ggplot` object, which can be further modified with `{ggplot2}` functions. Note that `extract_stats()` does not return statistical details for this plot; use `correlation::correlation()` instead.

Summary of graphics

graphical element	`geom` used	argument for further modification
correlation matrix	`ggcorrplot::ggcorrplot()`	`ggcorrplot.args`

Correlation analyses

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Hypothesis testing and effect size estimation

Type	Test	CI available?	Function used
Parametric	Pearson's correlation coefficient	Yes	`correlation::correlation()`
Non-parametric	Spearman's rank correlation coefficient	Yes	`correlation::correlation()`
Robust	Winsorized Pearson's correlation coefficient	Yes	`correlation::correlation()`
Bayesian	Bayesian Pearson's correlation coefficient	Yes	`correlation::correlation()`

See Also

`grouped_ggcorrmat()`, `ggscatterstats()`, `grouped_ggscatterstats()`

Examples

```
set.seed(123)
ggcorrmat(iris)

# with data containing NAs (uses pairwise complete observations)
ggcorrmat(airquality)

# selecting specific variables
ggcorrmat(iris, cor.vars = c(Sepal.Length, Petal.Length, Petal.Width))
```

---

ggdotplotstats *Dot plot/chart for labeled numeric data*

Description

A dot chart (as described by William S. Cleveland) with statistical details from one-sample test.

The point estimate (and associated uncertainty) displayed depends on the type of statistics selected:

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Usage

```
ggdotplotstats(
  data,
  x,
  y,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  type = "parametric",
  test.value = 0,
  alternative = "two.sided",
  bf.prior = 0.707,
  bf.message = TRUE,
  conf.int = TRUE,
  conf.level = 0.95,
  tr = 0.2,
  digits = 2L,
  results.subtitle = TRUE,
  point.args = list(color = "black", size = 3, shape = 16),
  errorbar.args = list(width = 0, na.rm = TRUE),
```

```
  centrality.plotting = TRUE,
  centrality.type = type,
 centrality.line.args = list(color = "blue", linewidth = 1, linetype = "dashed"),
  ggplot.component = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  ...
)
```

Arguments

data	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from {dplyr} should be ungrouped before they are entered as data.
x	A numeric variable from the data frame data.
y	Label or grouping variable. Each level of y is shown as a single dot: if a level has several rows, x is summarized using the centrality measure for the chosen type (see above), and the one-sample test is then carried out on these per-level summaries (i.e., the sample size is the number of levels of y).
xlab	Label for x axis variable. If NULL (default), variable name for x will be used.
ylab	Label for y axis variable. If NULL (default), variable name for y will be used.
title	The text for the plot title.
subtitle	The text for the plot subtitle. Used only if results.subtitle = FALSE.
caption	The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see bf.message).
type	A character specifying the type of statistical approach: • "parametric" • "nonparametric" • "robust" • "bayes" You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.
test.value	A number indicating the true value of the mean (Default: 0).
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.
bf.prior	A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.
bf.message	Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: TRUE).

`conf.int` Logical that decides whether to display confidence intervals as error bars (Default: `TRUE`).

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

`tr` Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`point.args` A list of additional aesthetic arguments passed to `ggplot2::geom_point()`.

`errorbar.args` A list of additional arguments passed to `ggplot2::geom_errorbar()`.

`centrality.plotting` Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use `centrality.type` to choose a different centrality measure.

`centrality.type` Decides which centrality measure is displayed. Defaults to the same value as `type`. Can be one of:

- `"parametric"` (for **mean**)
- `"nonparametric"` (for **median**)
- `"robust"` (for **trimmed mean**)
- `"bayes"` (for **MAP estimator**)

As with `type`, abbreviations are accepted.

`centrality.line.args` A list of additional aesthetic arguments passed to `ggplot2::geom_vline()`, which displays the line for the centrality measure.

`ggplot.component` A {ggplot2} component, or a list of components, to add to the plot (Default: `NULL`). This is mostly helpful for the `grouped_` variants, where it is applied to each individual plot.

`ggtheme` A {ggplot2} theme (Default: `theme_ggstatsplot()`). Any {ggplot2} theme (e.g., `ggplot2::theme_bw()`) or theme from an extension package (e.g., `ggthemes::theme_fivethirty`

can be used. Note that some themes remove plot details. For example, `ggthemes::theme_fivethirtyei` removes the secondary Y-axis, which `ggbetweenstats()` uses to show details about pairwise comparisons.

... Currently ignored.

Details

Rows with a missing value in either `x` or `y` are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggdotplotstats.html

Value

A `ggplot` object, which can be further modified with `{ggplot2}` functions. Use `extract_stats()` to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	`geom` used	argument for further modification
point estimates	`ggplot2::geom_point()`	`point.args`
error bars	`ggplot2::geom_errorbar()`	`errorbar.args`
centrality measure line	`ggplot2::geom_vline()`	`centrality.line.args`

One-sample tests

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Hypothesis testing

Type	Test	Function used
Parametric	One-sample Student's *t*-test	`stats::t.test()`
Non-parametric	One-sample Wilcoxon test	`stats::wilcox.test()`
Robust	Bootstrap-*t* method for one-sample test	`WRS2::trimcibt()`
Bayesian	One-sample Student's *t*-test	`BayesFactor::ttestBF()`

Effect size estimation

Type	Effect size	CI available?	Function used
Parametric	Cohen's *d*, Hedges' *g*	Yes	`effectsize::cohens_d()`, `effectsize::hedges_g()`
Non-parametric	*r* (rank-biserial correlation)	Yes	`effectsize::rank_biserial()`

Robust	trimmed mean	Yes	`WRS2::trimcibt()`
Bayesian	difference	Yes	`bayestestR::describe_posterior()`

See Also

`grouped_ggdotplotstats()`, `gghistostats()`, `grouped_gghistostats()`

Examples

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

# creating a plot
p <- ggdotplotstats(
  data = ggplot2::mpg,
  x = cty,
  y = manufacturer,
  title = "Fuel economy data",
  xlab = "city miles per gallon"
)

# looking at the plot
p

# extracting details from statistical tests
extract_stats(p)
```

---

gghistostats *Histogram for distribution of a numeric variable*

Description

Histogram with statistical details from one-sample test included in the plot as a subtitle.

Usage

```
gghistostats(
  data,
  x,
  binwidth = NULL,
  xlab = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  type = "parametric",
  test.value = 0,
```

```
  alternative = "two.sided",
  bf.prior = 0.707,
  bf.message = TRUE,
  conf.level = 0.95,
  tr = 0.2,
  digits = 2L,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  results.subtitle = TRUE,
  bin.args = list(color = "black", fill = "grey50", alpha = 0.7),
  centrality.plotting = TRUE,
  centrality.type = type,
 centrality.line.args = list(color = "blue", linewidth = 1, linetype = "dashed"),
  ggplot.component = NULL,
  ...
)
```

Arguments

data	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from {dplyr} should be ungrouped before they are entered as data.
x	A numeric variable from the data frame data.
binwidth	The width of the histogram bins. Can be specified as a numeric value, or a function that calculates width from x. The default is to use (max(x) - min(x)) / sqrt(N). If all values of x are identical, 0.1 (or abs(x) / 1000, if larger) is used instead. You should always check this value and explore multiple widths to find the one that best illustrates the stories in your data.
xlab	Label for x axis variable. If NULL (default), variable name for x will be used.
title	The text for the plot title.
subtitle	The text for the plot subtitle. Used only if results.subtitle = FALSE.
caption	The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see bf.message).
type	A character specifying the type of statistical approach: • "parametric" • "nonparametric" • "robust" • "bayes" You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.
test.value	A number indicating the true value of the mean (Default: 0).
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: `"medium"`, `"wide"`, and `"ultrawide"`, corresponding to *r* scale values of `1/2`, `sqrt(2)/2`, and `1`, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

`bf.message` Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: `TRUE`).

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

`tr` Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`ggtheme` A `{ggplot2}` theme (Default: `theme_ggstatsplot()`). Any `{ggplot2}` theme (e.g., `ggplot2::theme_bw()`) or theme from an extension package (e.g., `ggthemes::theme_fivethirty` can be used. Note that some themes remove plot details. For example, `ggthemes::theme_fivethirtyei` removes the secondary Y-axis, which `ggbetweenstats()` uses to show details about pairwise comparisons.

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`bin.args` A list of additional aesthetic arguments passed to `ggplot2::stat_bin()`, which displays the bins. Use the dedicated `binwidth` argument (not this list) to set the bin width.

`centrality.plotting` Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use `centrality.type` to choose a different centrality measure.

`centrality.type` Decides which centrality measure is displayed. Defaults to the same value as `type`. Can be one of:

- `"parametric"` (for **mean**)
- `"nonparametric"` (for **median**)
- `"robust"` (for **trimmed mean**)
- `"bayes"` (for **MAP estimator**)

As with type, abbreviations are accepted.

centrality.line.args
: A list of additional aesthetic arguments passed to ggplot2::geom_vline(), which displays the line for the centrality measure.

ggplot.component
: A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.

...
: Currently ignored.

Details

Missing values in x are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/gghistostats.html

Value

A ggplot object, which can be further modified with {ggplot2} functions. Use extract_stats() to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	geom used	argument for further modification
histogram bin	ggplot2::stat_bin()	bin.args
centrality measure line	ggplot2::geom_vline()	centrality.line.args

One-sample tests

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Hypothesis testing

Type	Test	Function used
Parametric	One-sample Student's *t*-test	stats::t.test()
Non-parametric	One-sample Wilcoxon test	stats::wilcox.test()
Robust	Bootstrap-*t* method for one-sample test	WRS2::trimcibt()
Bayesian	One-sample Student's *t*-test	BayesFactor::ttestBF()

Effect size estimation

Type	Effect size	CI available?	Function used
Parametric	Cohen's *d*, Hedges' *g*	Yes	`effectsize::cohens_d()`, `effectsize::hedges_g()`
Non-parametric	*r* (rank-biserial correlation)	Yes	`effectsize::rank_biserial()`
Robust	trimmed mean	Yes	`WRS2::trimcibt()`
Bayesian	difference	Yes	`bayestestR::describe_posterior()`

See Also

`grouped_gghistostats()`, `ggdotplotstats()`, `grouped_ggdotplotstats()`

Examples

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

# creating a plot
p <- gghistostats(
  data            = ToothGrowth,
  x               = len,
  xlab            = "Tooth length",
  centrality.type = "np"
)

# looking at the plot
p

# extracting details from statistical tests
extract_stats(p)
```

---

ggpiestats *Pie charts with statistical tests*

Description

Pie charts for categorical data with statistical details included in the plot as a subtitle.

Usage

```
ggpiestats(
  data,
  x,
  y = NULL,
  counts = NULL,
  type = "parametric",
  paired = FALSE,
  results.subtitle = TRUE,
```

```
  label = "percentage",
  label.args = list(direction = "both"),
  label.repel = FALSE,
  digits = 2L,
  proportion.test = results.subtitle,
  digits.perc = 0L,
  bf.message = TRUE,
  ratio = NULL,
  alternative = "two.sided",
  conf.level = 0.95,
  p.adjust.method = "holm",
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  legend.title = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  palette = "ggthemes::gdoc",
  ggplot.component = NULL,
  ...
)
```

Arguments

`data`	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from `{dplyr}` should be ungrouped before they are entered as `data`.
`x`	The variable to use as the **rows** in the contingency table. Note that empty factor levels are *not* dropped: they are shown in the legend and counted (with zero observations) in the statistical test. Use `droplevels()` beforehand if this is not intended.
`y`	The variable to use as the **columns** in the contingency table. If `NULL` (default), a one-sample proportion test (a goodness of fit test) is run for the `x` variable. Otherwise, an appropriate association test is run.
`counts`	The variable in data containing counts, or `NULL` if each row represents a single observation.
`type`	A character specifying the type of statistical approach: • `"parametric"` • `"nonparametric"` • `"robust"` • `"bayes"` You can specify just the initial letter (e.g. `"np"` or `"bf"`). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as `"Bayes"`) falls back to `"parametric"` without a warning.
`paired`	Logical indicating whether data came from a within-subjects or repeated measures design study (Default: `FALSE`). Only relevant for two-way tables (i.e.,

when y is supplied), and paired designs are only supported for frequentist tests (McNemar's test). For a two-way table with type = "bayes", paired is ignored and the Bayesian test of independence for unpaired data is run instead. For one-way tables (y = NULL), paired has no effect.

results.subtitle
: Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: TRUE). Setting this to FALSE also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

label
: Character that decides what each label displays: "percentage" (default), "counts", or "both".

label.args
: A list of additional aesthetic arguments passed to ggplot2::geom_label() (or ggrepel::geom_label_repel() if label.repel = TRUE).

label.repel
: Logical that decides whether labels are repelled using {ggrepel} to avoid overlaps (Default: FALSE).

digits
: Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

proportion.test
: Logical that decides whether a proportion test for x is carried out for each level of y. Defaults to results.subtitle. ggbarstats() displays only the *p*-values from this test. Always FALSE when type = "bayes", when y is NULL, or when x has only one level.

digits.perc
: Number of decimal places for percentage labels (Default: 0L).

bf.message
: Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for non-Bayesian types and unpaired designs (paired = FALSE) (Default: TRUE).

ratio
: A vector of proportions: the expected proportions for the proportion test (should sum to 1). Default is NULL, which means the null is equal theoretical proportions across the levels of the nominal variable. E.g., ratio = c(0.5, 0.5) for two levels, ratio = c(0.25, 0.25, 0.25, 0.25) for four levels, etc.

alternative
: a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

conf.level
: Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).

p.adjust.method
: Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".

title
: The text for the plot title.

subtitle
: The text for the plot subtitle. Used only if results.subtitle = FALSE.

caption
: The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see bf.message).

legend.title
: Title text for the legend. If NULL (default), the name of the x variable is used.

ggtheme	A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirty can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyei removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.
palette	Name of the discrete palette in "package::palette" format to be used for coloring (via {paletteer}). Run View(paletteer::palettes_d_names) to see all available options.
ggplot.component	A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.
...	Currently ignored.

Details

Rows with a missing value in x, y, or counts are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggpiestats.html

Value

A ggplot object, which can be further modified with {ggplot2} functions. Use extract_stats() to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	geom used	argument for further modification
pie slices	ggplot2::geom_col()	NA
labels	ggplot2::geom_label()/ggrepel::geom_label_repel()	label.args

Pairwise comparisons

When there is a two-way table, x has more than two levels, and paired = FALSE, pairwise contingency table analyses (Fisher's exact tests) are computed using statsExpressions::pairwise_contingency_table(). These results are **not** displayed in the plot: unlike the brackets in box/violin plots, pie and bar charts (which are designed to stay visually congruent) have no natural or established way to show pairwise comparisons. Retrieve them as a data frame with extract_stats(plot)$pairwise_comparisons_data.

Contingency table analyses

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate

- the functions used internally to compute these details

There are no dedicated non-parametric or robust contingency table analyses. `type = "nonparametric"` and `type = "robust"` are accepted, but run the same frequentist tests as `type = "parametric"` (the "Parametric/Non-parametric" rows below).

Two-way table:

Hypothesis testing

Type	Design	Test	Function used
Parametric/Non-parametric	Unpaired	Pearson's chi-squared test	`stats::chisq.test()`
Bayesian	Unpaired	Bayesian Pearson's chi-squared test	`BayesFactor::contingencyTableBF()`
Parametric/Non-parametric	Paired	McNemar's chi-squared test	`stats::mcnemar.test()`
Bayesian	Paired	No	No

Effect size estimation

Type	Design	Effect size	CI available?	Function used
Parametric/Non-parametric	Unpaired	Cramer's *V*	Yes	`effectsize::cramers_v()`
Bayesian	Unpaired	Cramer's *V*	Yes	`effectsize::cramers_v()`
Parametric/Non-parametric	Paired	Cohen's *g*	Yes	`effectsize::cohens_g()`
Bayesian	Paired	No	No	No

Paired Bayesian analysis is not supported: for a two-way table with `type = "bayes"`, the `paired` argument is ignored and the unpaired Bayesian test is run instead.

One-way table:

Hypothesis testing

Type	Test	Function used
Parametric/Non-parametric	Goodness of fit chi-squared test	`stats::chisq.test()`
Bayesian	Bayesian Goodness of fit chi-squared test	(custom)

Effect size estimation

Type	Effect size	CI available?	Function used
Parametric/Non-parametric	Pearson's *C*	Yes	`effectsize::pearsons_c()`
Bayesian	No	No	No

See Also

`grouped_ggpiestats()`, `ggbarstats()`, `grouped_ggbarstats()`

Examples

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

# one sample goodness of fit proportion test
p <- ggpiestats(mtcars, vs)

# looking at the plot
p

# extracting details from statistical tests
extract_stats(p)

# association test (or contingency table analysis)
ggpiestats(mtcars, vs, cyl)

# Bayesian test
ggpiestats(mtcars, vs, cyl, type = "bayes")

# with repelled labels to avoid overlapping
ggpiestats(mtcars, vs, label.repel = TRUE)

# show counts instead of percentages
ggpiestats(mtcars, vs, label = "counts")

# show both counts and percentages
ggpiestats(mtcars, vs, label = "both")

# using pre-aggregated data with counts
ggpiestats(as.data.frame(Titanic), Survived, counts = Freq)
```

---

ggscatterstats *Scatterplot with marginal distributions and statistical results*

Description

Scatterplots from {ggplot2} combined with marginal distribution plots with statistical details.

Usage

```
ggscatterstats(
  data,
  x,
  y,
  type = "parametric",
  conf.level = 0.95,
  bf.prior = 0.707,
  bf.message = TRUE,
```

```
  tr = 0.2,
  digits = 2L,
  results.subtitle = TRUE,
  label.var = NULL,
  label.expression = NULL,
  marginal = TRUE,
  point.args = list(size = 3, alpha = 0.4, stroke = 0),
  point.width.jitter = 0,
  point.height.jitter = 0,
  point.label.args = list(size = 3, max.overlaps = 1e+06),
 smooth.line.args = list(linewidth = 1.5, color = "blue", method = "lm", formula = y ~
    x),
  xsidehistogram.args = list(fill = "#4285F4", color = "black", na.rm = TRUE),
  ysidehistogram.args = list(fill = "#EA4335", color = "black", na.rm = TRUE),
  xsidehistogram.scale = list(),
  ysidehistogram.scale = list(),
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  ggplot.component = NULL,
  ...
)
```

Arguments

`data`	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from `{dplyr}` should be ungrouped before they are entered as `data`.
`x`	The column in `data` containing the explanatory variable to be plotted on the x-axis.
`y`	The column in `data` containing the response (outcome) variable to be plotted on the y-axis.
`type`	A character specifying the type of statistical approach: • `"parametric"` • `"nonparametric"` • `"robust"` • `"bayes"` You can specify just the initial letter (e.g. `"np"` or `"bf"`). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as `"Bayes"`) falls back to `"parametric"` without a warning.
`conf.level`	Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

bf.prior
: A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

bf.message
: Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: TRUE).

tr
: Trim level for the mean when carrying out robust tests. In case of an error, try reducing the value of tr, which is by default set to 0.2. Lowering the value might help.

digits
: Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

results.subtitle
: Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: TRUE). Setting this to FALSE also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

label.var
: Variable to use for point labels entered as a symbol (e.g. var1).

label.expression
: An expression evaluating to a logical vector that determines the subset of data points to label (e.g. y < 4 & z < 20). While using this argument with purrr::pmap(), you will have to provide a quoted expression (e.g. quote(y < 4 & z < 20)).

marginal
: Logical that decides whether marginal distributions are plotted on the axes using {ggside} (Default: TRUE).

point.args
: A list of additional aesthetic arguments passed to ggplot2::geom_point().

point.width.jitter, point.height.jitter
: Amount of jitter (in data units) in x and y direction, respectively, passed to ggplot2::position_jitter() (Default: 0, i.e., no jitter). Do not specify the jitter in point.args, because it is shared by the **points** and their **labels**.

point.label.args
: A list of additional aesthetic arguments passed to ggrepel::geom_label_repel(), which displays the point labels.

smooth.line.args
: A list of additional aesthetic arguments passed to ggplot2::geom_smooth(), which displays the regression line. Its confidence band uses conf.level.

xsidehistogram.args, ysidehistogram.args
: A list of arguments passed to ggside::geom_xsidehistogram() and ggside::geom_ysidehistogram(respectively, to change the marginal histograms.

xsidehistogram.scale, ysidehistogram.scale
: A list of arguments passed to ggside::scale_xsidey_continuous() and ggside::scale_ysidex_con respectively, to control the scale of marginal histograms (e.g., breaks, limits, transform). Default is list() (no modifications).

xlab
: Label for x axis variable. If NULL (default), variable name for x will be used.

ylab	Label for y axis variable. If NULL (default), variable name for y will be used.
title	The text for the plot title.
subtitle	The text for the plot subtitle. Used only if results.subtitle = FALSE.
caption	The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see bf.message).
ggtheme	A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirty can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyei removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.
ggplot.component	A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.
...	Additional arguments (currently ignored).

Details

Rows with a missing value in either x or y are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggscatterstats.html

Value

A ggplot object, which can be further modified with {ggplot2} functions. Use extract_stats() to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	geom used	argument for further modi
raw data	ggplot2::geom_point()	point.args
labels for raw data	ggrepel::geom_label_repel()	point.label.args
smooth line	ggplot2::geom_smooth()	smooth.line.args
marginal histograms	ggside::geom_xsidehistogram(), ggside::geom_ysidehistogram()	xsidehistogram.args, y

Correlation analyses

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Hypothesis testing and effect size estimation

Type	Test	CI available?	Function used
Parametric	Pearson's correlation coefficient	Yes	`correlation::correlation()`
Non-parametric	Spearman's rank correlation coefficient	Yes	`correlation::correlation()`
Robust	Winsorized Pearson's correlation coefficient	Yes	`correlation::correlation()`
Bayesian	Bayesian Pearson's correlation coefficient	Yes	`correlation::correlation()`

Note

Point labels use `ggrepel::geom_label_repel()` to keep them from overlapping, so plotting can become very slow (and the plot file large) if many labels overlap.

See Also

`grouped_ggscatterstats()`, `ggcorrmat()`, `grouped_ggcorrmat()`

Examples

```
set.seed(123)

# creating a plot
p <- ggscatterstats(
  iris,
  x = Sepal.Width,
  y = Petal.Length,
  label.var = Species,
  label.expression = Sepal.Length > 7.6
) +
  ggplot2::geom_rug(sides = "b")

# looking at the plot
p

# extracting details from statistical tests
extract_stats(p)

# customize marginal histogram bins and scales
ggscatterstats(
  mtcars,
  x = wt,
  y = mpg,
  results.subtitle = FALSE,
 xsidehistogram.args = list(fill = "#4285F4", color = "black", na.rm = TRUE, binwidth = 0.5),
  ysidehistogram.args = list(fill = "#EA4335", color = "black", na.rm = TRUE, bins = 15),
  xsidehistogram.scale = list(breaks = seq(0, 15, 5)),
  ysidehistogram.scale = list(breaks = seq(0, 15, 5))
)
```

ggwithinstats *Box/Violin plots for repeated measures comparisons*

Description

A combination of box and violin plots along with raw (unjittered) data points for within-subjects designs with statistical details included in the plot as a subtitle.

The data must be in *long* format: one row per subject per level of x (see bugs_long for an example).

Usage

```
ggwithinstats(
  data,
  x,
  y,
  type = "parametric",
  subject.id = NULL,
  pairwise.display = "significant",
  pairwise.alpha = 0.05,
  p.adjust.method = "holm",
  bf.prior = 0.707,
  bf.message = TRUE,
  results.subtitle = TRUE,
  xlab = NULL,
  ylab = NULL,
  caption = NULL,
  title = NULL,
  subtitle = NULL,
  digits = 2L,
  conf.level = 0.95,
  tr = 0.2,
  alternative = "two.sided",
  centrality.plotting = TRUE,
  centrality.type = type,
  centrality.point.args = list(size = 5, color = "darkred"),
  centrality.label.args = list(size = 3, nudge_x = 0.4, segment.linetype = 4),
  centrality.path = TRUE,
  centrality.path.args = list(linewidth = 1, color = "red", alpha = 0.5),
  point.args = list(size = 3, alpha = 0.5, na.rm = TRUE),
  point.path = TRUE,
  point.path.args = list(alpha = 0.5, linetype = "dashed"),
  boxplot.args = list(width = 0.2, alpha = 0.5, na.rm = TRUE),
  violin.args = list(width = 0.5, alpha = 0.2, na.rm = TRUE),
  ggsignif.args = list(textsize = 3, tip_length = 0.01, na.rm = TRUE),
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  palette = "ggthemes::gdoc",
```

```
  ggplot.component = NULL,
  ...
)
```

Arguments

`data`	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from `{dplyr}` should be ungrouped before they are entered as `data`.
`x`	The condition (or repeated-measures) variable from `data`.
`y`	The response (or outcome or dependent) variable from `data`.
`type`	A character specifying the type of statistical approach: • `"parametric"` • `"nonparametric"` • `"robust"` • `"bayes"` You can specify just the initial letter (e.g. `"np"` or `"bf"`). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as `"Bayes"`) falls back to `"parametric"` without a warning.
`subject.id`	The subject (or repeated measures) identifier from `data`. Each subject must appear at most once per level of `x`. If `NULL` (default), observations are paired by their row order within each level of `x`. **If you are not sure the data are sorted this way, specify this argument.** Rows with a missing `x` or `subject.id` are ignored, and subjects with a missing (`NA`) dependent variable in any condition are dropped list-wise (see `statsExpressions::complete_subjects()`), so the plot, its labels, and the statistical results are all based on the same complete subjects.
`pairwise.display`	Decides *which* pairwise comparisons to display. Available options are: • `"significant"` (abbreviation accepted: `"s"`) • `"non-significant"` (abbreviation accepted: `"ns"`) • `"all"` • `"none"` Use this to keep the plot uncluttered when many groups are compared. Pairwise comparisons are carried out (and displayed) only when `x` has three or more levels. For `type = "bayes"`, there are no *p*-values, so all comparisons are displayed unless this is `"none"`.
`pairwise.alpha`	Numeric alpha threshold used to decide which pairwise comparisons are displayed when `pairwise.display = "significant"` or `pairwise.display = "non-significant"` (Default: `0.05`). It is compared against *p*-values adjusted using `p.adjust.method`.
`p.adjust.method`	Adjustment method for *p*-values for multiple comparisons. Possible methods are: `"holm"` (default), `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, `"none"`.

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: `"medium"`, `"wide"`, and `"ultrawide"`, corresponding to *r* scale values of `1/2`, `sqrt(2)/2`, and `1`, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

`bf.message` Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: `TRUE`).

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`xlab` Label for `x` axis variable. If `NULL` (default), variable name for `x` will be used.

`ylab` Label for `y` axis variable. If `NULL` (default), variable name for `y` will be used.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`title` The text for the plot title.

`subtitle` The text for the plot subtitle. Used only if `results.subtitle = FALSE`.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

`tr` Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`alternative` a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.

`centrality.plotting` Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use `centrality.type` to choose a different centrality measure.

`centrality.type` Decides which centrality measure is displayed. Defaults to the same value as `type`. Can be one of:

- `"parametric"` (for **mean**)
- `"nonparametric"` (for **median**)
- `"robust"` (for **trimmed mean**)

- "bayes" (for **MAP estimator**)

As with type, abbreviations are accepted.

centrality.point.args, centrality.label.args
: A list of additional aesthetic arguments passed to ggplot2::geom_point() and ggrepel::geom_label_repel(), respectively, which display the centrality measure.

centrality.path.args, point.path.args
: A list of additional aesthetic arguments passed to ggplot2::geom_path() connecting centrality measures and raw data points, respectively.

point.args
: A list of additional aesthetic arguments passed to ggplot2::geom_point().

point.path, centrality.path
: Logical that decides whether individual data points and centrality measures, respectively, are connected using ggplot2::geom_path() (Default: TRUE). point.path is relevant only when there are two groups (i.e., for a *t*-test). With many data points, consider point.path = FALSE, as these lines can overwhelm the plot.

boxplot.args
: A list of additional aesthetic arguments passed to ggplot2::geom_boxplot(). By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). Use coef to change this, e.g., boxplot.args = list(coef = 3) extends the whiskers to the most extreme observations within 3 * IQR of the hinges.

violin.args
: A list of additional aesthetic arguments passed to ggplot2::geom_violin().

ggsignif.args
: A list of additional aesthetic arguments passed to ggsignif::geom_signif().

ggtheme
: A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirty can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyei removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.

palette
: Name of the discrete palette in "package::palette" format to be used for coloring (via {paletteer}). Run View(paletteer::palettes_d_names) to see all available options.

ggplot.component
: A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.

...
: Currently ignored.

Details

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggwithinstats.html

Value

A ggplot object, which can be further modified with {ggplot2} functions. Use extract_stats() to get the statistical details shown in the plot as data frames.

Summary of graphics

graphical element	geom used	argument for further modification
raw data	`ggplot2::geom_point()`	`point.args`
point path	`ggplot2::geom_path()`	`point.path.args`
box plot	`ggplot2::geom_boxplot()`	`boxplot.args`
density plot	`ggplot2::geom_violin()`	`violin.args`
centrality measure point	`ggplot2::geom_point()`	`centrality.point.args`
centrality measure point path	`ggplot2::geom_path()`	`centrality.path.args`
centrality measure label	`ggrepel::geom_label_repel()`	`centrality.label.args`
pairwise comparisons	`ggsignif::geom_signif()`	`ggsignif.args`

Centrality measures

Type	Measure	Function used
Parametric	mean	`datawizard::describe_distribution()`
Non-parametric	median	`datawizard::describe_distribution()`
Robust	trimmed mean	`datawizard::describe_distribution()`
Bayesian	MAP	`datawizard::describe_distribution()`

Two-sample tests

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Between-subjects:

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	2	Student's or Welch's *t*-test	`stats::t.test()`
Non-parametric	2	Mann-Whitney *U* test	`stats::wilcox.test()`
Robust	2	Yuen's test for trimmed means	`WRS2::yuen()`
Bayesian	2	Student's *t*-test	`BayesFactor::ttestBF()`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Function used

Parametric	2	Cohen's *d*, Hedges' *g*	Yes	`effectsize:`
Non-parametric	2	*r* (rank-biserial correlation)	Yes	`effectsize:`
Robust	2	Algina-Keselman-Penfield robust standardized difference	Yes	`WRS2::akp.e`
Bayesian	2	difference	Yes	`bayestestR:`

Within-subjects:

Paired data: See `complete_subjects()` for subject matching, duplicate observations, and removal of incomplete subjects.

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	2	Student's *t*-test	`stats::t.test()`
Non-parametric	2	Wilcoxon signed-rank test	`stats::wilcox.test()`
Robust	2	Yuen's test on trimmed means for dependent samples	`WRS2::yuend()`
Bayesian	2	Student's *t*-test	`BayesFactor::ttestBF()`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Function used
Parametric	2	Cohen's *d*, Hedges' *g*	Yes	`effectsize:`
Non-parametric	2	*r* (rank-biserial correlation)	Yes	`effectsize:`
Robust	2	Algina-Keselman-Penfield robust standardized difference	Yes	`WRS2::wmcpA`
Bayesian	2	difference	Yes	`bayestestR:`

One-way ANOVA

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Between-subjects:

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	> 2	Fisher's or Welch's one-way ANOVA	`stats::oneway.test()`
Non-parametric	> 2	Kruskal-Wallis one-way ANOVA	`stats::kruskal.test()`
Robust	> 2	Heteroscedastic one-way ANOVA for trimmed means	`WRS2::t1way()`
Bayesian	> 2	Fisher's ANOVA	`BayesFactor::anovaBF()`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Function used
Parametric	> 2	partial eta-squared, partial omega-squared	Yes	`effectsize::omega_square`
Non-parametric	> 2	rank epsilon squared	Yes	`effectsize::rank_epsilon`
Robust	> 2	Explanatory measure of effect size	Yes	`WRS2::t1way()`
Bayesian	> 2	Bayesian R-squared	Yes	`performance::r2_bayes()`

Within-subjects:

Paired data: See `complete_subjects()` for subject matching, duplicate observations, and removal of incomplete subjects.

Hypothesis testing

Type	No. of groups	Test	Function used
Parametric	> 2	One-way repeated measures ANOVA	`afex::aov_ez`
Non-parametric	> 2	Friedman rank sum test	`stats::friedc`
Robust	> 2	Heteroscedastic one-way repeated measures ANOVA for trimmed means	`WRS2::rmanov`
Bayesian	> 2	One-way repeated measures ANOVA	`BayesFactor:`

Effect size estimation

Type	No. of groups	Effect size	CI available?	Funct
Parametric	> 2	partial eta-squared, partial omega-squared	Yes	`effec`
Non-parametric	> 2	Kendall's coefficient of concordance	Yes	`effec`
Robust	> 2	Algina-Keselman-Penfield robust standardized difference average	Yes	`WRS2:`
Bayesian	> 2	Bayesian R-squared	Yes	`perfo`

Pairwise comparison tests

Below is a summary of:

- the statistical test carried out for inferential statistics
- the type of effect size estimate and a measure of uncertainty for this estimate
- the functions used internally to compute these details

Between-subjects:

Hypothesis testing

Type	Equal variance?	Test	*p*-value adjustment?	Function used
Parametric	No	Games-Howell test	Yes	`PMCMRplus::gamesHowellTest()`
Parametric	Yes	Student's *t*-test	Yes	`stats::pairwise.t.test()`
Non-parametric	No	Dunn test	Yes	`PMCMRplus::kwAllPairsDunnTes`
Robust	No	Yuen's trimmed means test	Yes	`WRS2::lincon()`
Bayesian	NA	Student's *t*-test	NA	`BayesFactor::ttestBF()`

Effect size estimation

Not supported.

Within-subjects:

Paired data: See `complete_subjects()` for subject matching, duplicate observations, and removal of incomplete subjects.

Hypothesis testing

Type	Test	*p*-value adjustment?	Function used
Parametric	Student's *t*-test	Yes	`stats::pairwise.t.test()`
Non-parametric	Durbin-Conover test	Yes	`PMCMRplus::durbinAllPairsTest()`
Robust	Yuen's trimmed means test	Yes	`WRS2::rmmcp()`
Bayesian	Student's *t*-test	NA	`BayesFactor::ttestBF()`

Effect size estimation

Not supported.

See Also

`grouped_ggwithinstats()`, `ggbetweenstats()`, `grouped_ggbetweenstats()`

Examples

```
# for reproducibility
set.seed(123)
library(dplyr, warn.conflicts = FALSE)

# create a plot
p <- ggwithinstats(
  data       = filter(bugs_long, condition %in% c("HDHF", "HDLF")),
  x          = condition,
  y          = desire,
  type       = "np",
  subject.id = subject
)

# looking at the plot
p

# extracting details from statistical tests
extract_stats(p)

# if the data are already arranged in repeated-measures order, `subject.id`
# can be omitted
ggwithinstats(
  data             = filter(bugs_long, condition %in% c("HDHF", "HDLF")),
  x                = condition,
  y                = desire,
  results.subtitle = FALSE
)

# use a stricter alpha threshold for significant pairwise comparisons
ggwithinstats(
```

```
  data = bugs_long,
  x = condition,
  y = desire,
  subject.id = subject,
  pairwise.alpha = 0.001
)

# modifying defaults
ggwithinstats(
  data       = bugs_long,
  x          = condition,
  y          = desire,
  type       = "robust",
  subject.id = subject
)

# you can remove a specific geom to reduce complexity of the plot
ggwithinstats(
  data = bugs_long,
  x = condition,
  y = desire,
  subject.id = subject,
  # to remove violin plot
  violin.args = list(width = 0, linewidth = 0, colour = NA),
  # to remove boxplot
  boxplot.args = list(width = 0),
  # to remove points
  point.args = list(alpha = 0)
)
```

`grouped_ggbarstats` *Grouped bar charts with statistical tests*

Description

Applies `ggbarstats()` across all levels of `grouping.var` and combines the resulting plots using `combine_plots()`.

Usage

```
grouped_ggbarstats(
  data,
  ...,
  grouping.var,
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

data	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from {dplyr} should be ungrouped before they are entered as data.
...	Arguments passed on to ggbarstats label.args A list of additional aesthetic arguments passed to ggplot2::geom_label(). sample.size.label.args A list of additional aesthetic arguments passed to ggplot2::geom_text(). xlab Label for x axis. If NULL (default), the name of the y variable is used (no label if y is NULL), since bars are drawn for each level of y. ylab Label for y axis. If NULL (default), no label is shown. x The variable to use as the **rows** in the contingency table. Note that empty factor levels are *not* dropped: they are shown in the legend and counted (with zero observations) in the statistical test. Use droplevels() beforehand if this is not intended. y The variable to use as the **columns** in the contingency table. If NULL (default), a one-sample proportion test (a goodness of fit test) is run for the x variable. Otherwise, an appropriate association test is run. proportion.test Logical that decides whether a proportion test for x is carried out for each level of y. Defaults to results.subtitle. ggbarstats() displays only the *p*-values from this test. Always FALSE when type = "bayes", when y is NULL, or when x has only one level. bf.message Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for non-Bayesian types and unpaired designs (paired = FALSE) (Default: TRUE). digits.perc Number of decimal places for percentage labels (Default: 0L). label Character that decides what each label displays: "percentage" (default), "counts", or "both". legend.title Title text for the legend. If NULL (default), the name of the x variable is used. p.adjust.method Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". results.subtitle Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: TRUE). Setting this to FALSE also skips the automatic Bayes Factor caption. Other plot details have their own arguments. subtitle The text for the plot subtitle. Used only if results.subtitle = FALSE. caption The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see bf.message). ggplot.component A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.

palette Name of the discrete palette in "package::palette" format to be used for coloring (via {paletteer}). Run View(paletteer::palettes_d_names) to see all available options.

ggtheme A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirtyeight()) can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyeight() removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.

type A character specifying the type of statistical approach:

- "parametric"
- "nonparametric"
- "robust"
- "bayes"

You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.

digits Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

conf.level Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).

alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

paired Logical indicating whether data came from a within-subjects or repeated measures design study (Default: FALSE). Only relevant for two-way tables (i.e., when y is supplied), and paired designs are only supported for frequentist tests (McNemar's test). For a two-way table with type = "bayes", paired is ignored and the Bayesian test of independence for unpaired data is run instead. For one-way tables (y = NULL), paired has no effect.

counts The variable in data containing counts, or NULL if each row represents a single observation.

ratio A vector of proportions: the expected proportions for the proportion test (should sum to 1). Default is NULL, which means the null is equal theoretical proportions across the levels of the nominal variable. E.g., ratio = c(0.5, 0.5) for two levels, ratio = c(0.25, 0.25, 0.25, 0.25) for four levels, etc.

grouping.var A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides, which has its own argument.

annotation.args
: A list of additional arguments passed to patchwork::plot_annotation().

Details

Rows with a missing value in x, y, or counts are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggbarstats.html

Value

A patchwork object combining one plot per level of grouping.var (see combine_plots()). Applying extract_stats() to it returns a list with the statistical details for each of these plots.

See Also

ggbarstats(), ggpiestats(), grouped_ggpiestats()

Examples

```
set.seed(123)
# grouped one-sample proportion test
grouped_ggbarstats(
  data = mtcars,
  x = cyl,
  grouping.var = am,
  annotation.args = list(title = "Cylinder distribution by transmission type")
)
```

---

grouped_ggbetweenstats

Grouped box/violin plots for between-subjects comparisons

Description

Applies ggbetweenstats() across all levels of grouping.var and combines the resulting plots using combine_plots().

Usage

```
grouped_ggbetweenstats(
  data,
  ...,
  grouping.var,
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data` A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from `{dplyr}` should be ungrouped before they are entered as `data`.

`...` Arguments passed on to `ggbetweenstats`

`xlab` Label for `x` axis variable. If `NULL` (default), variable name for `x` will be used.

`ylab` Label for `y` axis variable. If `NULL` (default), variable name for `y` will be used.

`p.adjust.method` Adjustment method for *p*-values for multiple comparisons. Possible methods are: `"holm"` (default), `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, `"none"`.

`pairwise.display` Decides *which* pairwise comparisons to display. Available options are:

- `"significant"` (abbreviation accepted: `"s"`)
- `"non-significant"` (abbreviation accepted: `"ns"`)
- `"all"`
- `"none"`

Use this to keep the plot uncluttered when many groups are compared. Pairwise comparisons are carried out (and displayed) only when `x` has three or more levels. For `type = "bayes"`, there are no *p*-values, so all comparisons are displayed unless this is `"none"`.

`pairwise.alpha` Numeric alpha threshold used to decide which pairwise comparisons are displayed when `pairwise.display = "significant"` or `pairwise.display = "non-significant"` (Default: `0.05`). It is compared against *p*-values adjusted using `p.adjust.method`.

`bf.message` Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: `TRUE`).

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`subtitle` The text for the plot subtitle. Used only if `results.subtitle = FALSE`.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`centrality.plotting` Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use centrality.type to choose a different centrality measure.

centrality.type Decides which centrality measure is displayed. Defaults to the same value as type. Can be one of:

- "parametric" (for **mean**)
- "nonparametric" (for **median**)
- "robust" (for **trimmed mean**)
- "bayes" (for **MAP estimator**)

As with type, abbreviations are accepted.

point.args A list of additional aesthetic arguments passed to ggplot2::geom_point().

boxplot.args A list of additional aesthetic arguments passed to ggplot2::geom_boxplot(). By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). Use coef to change this, e.g., boxplot.args = list(coef = 3) extends the whiskers to the most extreme observations within 3 * IQR of the hinges.

violin.args A list of additional aesthetic arguments passed to ggplot2::geom_violin().

ggplot.component A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.

palette Name of the discrete palette in "package::palette" format to be used for coloring (via {paletteer}). Run View(paletteer::palettes_d_names) to see all available options.

centrality.point.args,centrality.label.args A list of additional aesthetic arguments passed to ggplot2::geom_point() and ggrepel::geom_label_repel(), respectively, which display the centrality measure.

ggsignif.args A list of additional aesthetic arguments passed to ggsignif::geom_signif().

ggtheme A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirtyeight()) can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyeight() removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.

x The grouping (or independent) variable from data.

y The response (or outcome or dependent) variable from data.

type A character specifying the type of statistical approach:

- "parametric"
- "nonparametric"
- "robust"
- "bayes"

You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.

digits Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: `"medium"`, `"wide"`, and `"ultrawide"`, corresponding to *r* scale values of `1/2`, `sqrt(2)/2`, and `1`, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

`tr` Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`alternative` a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.

`grouping.var` A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.

`plotgrid.args` A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides`, which has its own argument.

`annotation.args`
A `list` of additional arguments passed to `patchwork::plot_annotation()`.

Value

A `patchwork` object combining one plot per level of `grouping.var` (see `combine_plots()`). Applying `extract_stats()` to it returns a list with the statistical details for each of these plots.

See Also

`ggbetweenstats()`, `ggwithinstats()`, `grouped_ggwithinstats()`

Examples

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

library(dplyr, warn.conflicts = FALSE)
library(ggplot2)

grouped_ggbetweenstats(
  data = filter(ggplot2::mpg, drv != "4"),
  x = year,
  y = hwy,
  grouping.var = drv
)

# modifying individual plots using `ggplot.component` argument
grouped_ggbetweenstats(
  data = filter(
```

```
    movies_long,
    genre %in% c("Action", "Comedy"),
    mpaa %in% c("R", "PG")
  ),
  x = genre,
  y = rating,
  grouping.var = mpaa,
  ggplot.component = scale_y_continuous(
    breaks = seq(1, 9, 1),
    limits = c(1, 9)
  ),
  annotation.args = list(title = "Ratings by genre for different MPAA ratings")
)
```

`grouped_ggcorrmat` *Grouped visualizations of correlation matrices*

Description

Applies `ggcorrmat()` across all levels of `grouping.var` and combines the resulting plots using `combine_plots()`.

Usage

```
grouped_ggcorrmat(
  data,
  ...,
  grouping.var,
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A data frame from which variables specified are to be taken.
`...`	Arguments passed on to `ggcorrmat`
	`cor.vars` Variables for which the correlation matrix is to be computed and visualized, specified using `{tidyselect}` syntax (e.g., `c(var1, var2)` or `starts_with("Sepal")`). If `NULL` (default), all numeric variables from `data` will be used.
	`cor.vars.names` Optional list of names to be used for `cor.vars`. The names should be entered in the same order.
	`partial` Can be `TRUE` for partial correlations. For Bayesian partial correlations, "full" instead of pseudo-Bayesian partial correlations (i.e., Bayesian correlation based on frequentist partialization) are returned.
	`matrix.type` Character, `"upper"` (default), `"lower"`, or `"full"`, to display the upper triangular, lower triangular, or full matrix, respectively.

`sig.level` Significance level (Default: `0.05`). If the (adjusted; see `p.adjust.method`) *p*-value is bigger than `sig.level`, then the corresponding correlation coefficient is regarded as insignificant and flagged as such in the plot. Ignored for `type = "bayes"`.

`pch` Decides the point shape to be used for insignificant correlation coefficients. Default: `pch = "cross"`. The caption explaining the insignificance marker is added only for `pch = "cross"` (or 4).

`colors` A character vector of exactly three colors for the gradient: low (negative correlations), mid (zero), and high (positive correlations). Must be a **diverging** palette so that the sign of the correlation is visually obvious. Default: `c("#EA4335", "white", "#4285F4")` (red-white-blue).

`ggcorrplot.args` A list of additional (mostly aesthetic) arguments passed to `ggcorrplot::ggcorrplot()`. Do not include the following arguments, which are set internally: `corr`, `p.mat`, `sig.level`, `ggtheme`, `colors`, `type`, `lab`, `pch`, `legend.title`, `digits`.

`subtitle` The text for the plot subtitle.

`caption` The text for the plot caption. If the insignificance marker caption is shown (see `pch`), this text is displayed above it.

`type` A character specifying the type of statistical approach:

- `"parametric"`
- `"nonparametric"`
- `"robust"`
- `"bayes"`

You can specify just the initial letter (e.g. `"np"` or `"bf"`). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as `"Bayes"`) falls back to `"parametric"` without a warning.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`conf.level` Scalar between `0` and 1 (default: `95%` confidence/credible intervals, `0.95`).

`tr` Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`bf.prior` A number between `0.5` and 2 (default `0.707`), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: `"medium"`, `"wide"`, and `"ultrawide"`, corresponding to *r* scale values of `1/2`, `sqrt(2)/2`, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

`p.adjust.method` Adjustment method for *p*-values for multiple comparisons. Possible methods are: `"holm"` (default), `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, `"none"`.

ggplot.component A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.

ggtheme A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirtyeight()) can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyeight() removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.

grouping.var A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides, which has its own argument.

annotation.args A list of additional arguments passed to patchwork::plot_annotation().

Details

Missing values are handled using pairwise deletion (each correlation uses all complete pairs of observations). If sample sizes then differ across pairs, the legend shows their minimum, mode, and maximum. Partial correlations use only complete cases.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggcorrmat.html

Value

A patchwork object combining one plot per level of grouping.var (see combine_plots()).

See Also

ggcorrmat(), ggscatterstats(), grouped_ggscatterstats()

Examples

```
set.seed(123)

grouped_ggcorrmat(
  data = iris,
  grouping.var = Species,
  type = "robust",
  colors = c("#0072B2", "white", "#D55E00"),
  plotgrid.args = list(ncol = 1L),
  annotation.args = list(tag_levels = "i")
)
```

grouped_ggdotplotstats *Grouped dot plots/charts for labeled numeric data*

Description

Applies ggdotplotstats() across all levels of grouping.var and combines the resulting plots using combine_plots().

Usage

```
grouped_ggdotplotstats(
  data,
  ...,
  grouping.var,
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

data	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from {dplyr} should be ungrouped before they are entered as data.
...	Arguments passed on to ggdotplotstats y Label or grouping variable. Each level of y is shown as a single dot: if a level has several rows, x is summarized using the centrality measure for the chosen type (see above), and the one-sample test is then carried out on these per-level summaries (i.e., the sample size is the number of levels of y). centrality.line.args A list of additional aesthetic arguments passed to ggplot2::geom_vline(), which displays the line for the centrality measure. x A numeric variable from the data frame data. type A character specifying the type of statistical approach: • "parametric" • "nonparametric" • "robust" • "bayes" You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning. test.value A number indicating the true value of the mean (Default: 0).

`alternative` a character string specifying the alternative hypothesis, must be one of `"two.sided"` (default), `"greater"` or `"less"`. You can specify just the initial letter.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g. `digits = "scientific4"` to have scientific notation with 4 decimal places, or `digits = "signif5"` for 5 significant figures (see also `signif()`).

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`).

`tr` Trim level for the mean when carrying out `robust` tests. In case of an error, try reducing the value of `tr`, which is by default set to `0.2`. Lowering the value might help.

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: `"medium"`, `"wide"`, and `"ultrawide"`, corresponding to *r* scale values of `1/2`, `sqrt(2)/2`, and `1`, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

`xlab` Label for `x` axis variable. If `NULL` (default), variable name for `x` will be used.

`bf.message` Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: `TRUE`).

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`subtitle` The text for the plot subtitle. Used only if `results.subtitle = FALSE`.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`centrality.plotting` Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use `centrality.type` to choose a different centrality measure.

`centrality.type` Decides which centrality measure is displayed. Defaults to the same value as `type`. Can be one of:

- `"parametric"` (for **mean**)
- `"nonparametric"` (for **median**)
- `"robust"` (for **trimmed mean**)
- `"bayes"` (for **MAP estimator**)

As with type, abbreviations are accepted.

ggplot.component A {ggplot2} component, or a list of components, to add to the plot (Default: NULL). This is mostly helpful for the grouped_ variants, where it is applied to each individual plot.

ggtheme A {ggplot2} theme (Default: theme_ggstatsplot()). Any {ggplot2} theme (e.g., ggplot2::theme_bw()) or theme from an extension package (e.g., ggthemes::theme_fivethirtyeight()) can be used. Note that some themes remove plot details. For example, ggthemes::theme_fivethirtyeight() removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.

ylab Label for y axis variable. If NULL (default), variable name for y will be used.

point.args A list of additional aesthetic arguments passed to ggplot2::geom_point().

conf.int Logical that decides whether to display confidence intervals as error bars (Default: TRUE).

errorbar.args A list of additional arguments passed to ggplot2::geom_errorbar().

grouping.var A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides, which has its own argument.

annotation.args A list of additional arguments passed to patchwork::plot_annotation().

Details

Rows with a missing value in either x or y are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggdotplotstats.html

Value

A patchwork object combining one plot per level of grouping.var (see combine_plots()). Applying extract_stats() to it returns a list with the statistical details for each of these plots.

See Also

ggdotplotstats(), gghistostats(), grouped_gghistostats()

Examples

```
# for reproducibility
set.seed(123)
library(dplyr, warn.conflicts = FALSE)

# drop the 5-cylinder cars, which have very few observations
df <- filter(ggplot2::mpg, cyl %in% c("4", "6", "8"))
```

```
grouped_ggdotplotstats(
  data            = df,
  x               = cty,
  y               = manufacturer,
  grouping.var    = cyl,
  test.value      = 15.5,
  annotation.args = list(title = "City mileage by manufacturer for different cylinders")
)
```

`grouped_gghistostats` *Grouped histograms for distribution of a numeric variable*

Description

Applies `gghistostats()` across all levels of `grouping.var` and combines the resulting plots using `combine_plots()`.

Usage

```
grouped_gghistostats(
  data,
  x,
  grouping.var,
  binwidth = NULL,
  plotgrid.args = list(),
  annotation.args = list(),
  ...
)
```

Arguments

`data`	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from `{dplyr}` should be ungrouped before they are entered as `data`.
`x`	A numeric variable from the data frame `data`.
`grouping.var`	A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.
`binwidth`	The width of the histogram bins (see `gghistostats()`). If `NULL` (default), it is computed once from the non-missing values of `x` in the full data (not per group), so that all plots share the same bin width.
`plotgrid.args`	A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides`, which has its own argument.
`annotation.args`	A `list` of additional arguments passed to `patchwork::plot_annotation()`.

... Arguments passed on to gghistostats

bin.args A list of additional aesthetic arguments passed to ggplot2::stat_bin(), which displays the bins. Use the dedicated binwidth argument (not this list) to set the bin width.

centrality.line.args A list of additional aesthetic arguments passed to ggplot2::geom_vline(), which displays the line for the centrality measure.

type A character specifying the type of statistical approach:

- "parametric"
- "nonparametric"
- "robust"
- "bayes"

You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.

test.value A number indicating the true value of the mean (Default: 0).

alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

digits Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

conf.level Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).

tr Trim level for the mean when carrying out robust tests. In case of an error, try reducing the value of tr, which is by default set to 0.2. Lowering the value might help.

bf.prior A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

xlab Label for x axis variable. If NULL (default), variable name for x will be used.

bf.message Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: TRUE).

results.subtitle Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: TRUE). Setting this to FALSE also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

subtitle The text for the plot subtitle. Used only if results.subtitle = FALSE.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`centrality.plotting` Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use `centrality.type` to choose a different centrality measure.

`centrality.type` Decides which centrality measure is displayed. Defaults to the same value as `type`. Can be one of:

- `"parametric"` (for **mean**)
- `"nonparametric"` (for **median**)
- `"robust"` (for **trimmed mean**)
- `"bayes"` (for **MAP estimator**)

As with `type`, abbreviations are accepted.

`ggplot.component` A `{ggplot2}` component, or a list of components, to add to the plot (Default: `NULL`). This is mostly helpful for the `grouped_` variants, where it is applied to each individual plot.

`ggtheme` A `{ggplot2}` theme (Default: `theme_ggstatsplot()`). Any `{ggplot2}` theme (e.g., `ggplot2::theme_bw()`) or theme from an extension package (e.g., `ggthemes::theme_fivethirtyeight()`) can be used. Note that some themes remove plot details. For example, `ggthemes::theme_fivethirtyeight()` removes the secondary Y-axis, which `ggbetweenstats()` uses to show details about pairwise comparisons.

Details

Missing values in `x` are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/gghistostats.html

Value

A `patchwork` object combining one plot per level of `grouping.var` (see `combine_plots()`). Applying `extract_stats()` to it returns a list with the statistical details for each of these plots.

See Also

`gghistostats()`, `ggdotplotstats()`, `grouped_ggdotplotstats()`

Examples

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

```
grouped_gghistostats(
  data            = iris,
  x               = Sepal.Length,
  test.value      = 5,
  grouping.var    = Species,
  plotgrid.args   = list(nrow = 1),
  annotation.args = list(tag_levels = "i")
)
```

`grouped_ggpiestats` *Grouped pie charts with statistical tests*

Description

Applies `ggpiestats()` across all levels of `grouping.var` and combines the resulting plots using `combine_plots()`.

Usage

```
grouped_ggpiestats(
  data,
  ...,
  grouping.var,
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from `{dplyr}` should be ungrouped before they are entered as `data`.
`...`	Arguments passed on to `ggpiestats`
	`x` The variable to use as the **rows** in the contingency table. Note that empty factor levels are *not* dropped: they are shown in the legend and counted (with zero observations) in the statistical test. Use `droplevels()` beforehand if this is not intended.
	`y` The variable to use as the **columns** in the contingency table. If `NULL` (default), a one-sample proportion test (a goodness of fit test) is run for the `x` variable. Otherwise, an appropriate association test is run.
	`proportion.test` Logical that decides whether a proportion test for `x` is carried out for each level of `y`. Defaults to `results.subtitle`. `ggbarstats()` displays only the *p*-values from this test. Always `FALSE` when `type = "bayes"`, when `y` is `NULL`, or when `x` has only one level.

`bf.message` Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for non-Bayesian types and unpaired designs (`paired = FALSE`) (Default: `TRUE`).

`digits.perc` Number of decimal places for percentage labels (Default: `0L`).

`label` Character that decides what each label displays: `"percentage"` (default), `"counts"`, or `"both"`.

`label.args` A list of additional aesthetic arguments passed to `ggplot2::geom_label()` (or `ggrepel::geom_label_repel()` if `label.repel = TRUE`).

`label.repel` Logical that decides whether labels are repelled using {ggrepel} to avoid overlaps (Default: `FALSE`).

`legend.title` Title text for the legend. If `NULL` (default), the name of the `x` variable is used.

`p.adjust.method` Adjustment method for *p*-values for multiple comparisons. Possible methods are: `"holm"` (default), `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, `"none"`.

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`subtitle` The text for the plot subtitle. Used only if `results.subtitle = FALSE`.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`ggplot.component` A {ggplot2} component, or a list of components, to add to the plot (Default: `NULL`). This is mostly helpful for the `grouped_` variants, where it is applied to each individual plot.

`palette` Name of the discrete palette in `"package::palette"` format to be used for coloring (via {paletteer}). Run `View(paletteer::palettes_d_names)` to see all available options.

`ggtheme` A {ggplot2} theme (Default: `theme_ggstatsplot()`). Any {ggplot2} theme (e.g., `ggplot2::theme_bw()`) or theme from an extension package (e.g., `ggthemes::theme_fivethirtyeight()`) can be used. Note that some themes remove plot details. For example, `ggthemes::theme_fivethirtyeight()` removes the secondary Y-axis, which `ggbetweenstats()` uses to show details about pairwise comparisons.

`type` A character specifying the type of statistical approach:

- `"parametric"`
- `"nonparametric"`
- `"robust"`
- `"bayes"`

You can specify just the initial letter (e.g. `"np"` or `"bf"`). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as `"Bayes"`) falls back to `"parametric"` without a warning.

`digits` Number of digits for rounding or significant figures. May also be `"signif"` to return significant figures or `"scientific"` to return scientific notation. Control the number of digits by adding the value as suffix, e.g.

digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

conf.level Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).

alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

paired Logical indicating whether data came from a within-subjects or repeated measures design study (Default: FALSE). Only relevant for two-way tables (i.e., when y is supplied), and paired designs are only supported for frequentist tests (McNemar's test). For a two-way table with type = "bayes", paired is ignored and the Bayesian test of independence for unpaired data is run instead. For one-way tables (y = NULL), paired has no effect.

counts The variable in data containing counts, or NULL if each row represents a single observation.

ratio A vector of proportions: the expected proportions for the proportion test (should sum to 1). Default is NULL, which means the null is equal theoretical proportions across the levels of the nominal variable. E.g., ratio = c(0.5, 0.5) for two levels, ratio = c(0.25, 0.25, 0.25, 0.25) for four levels, etc.

grouping.var A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides, which has its own argument.

annotation.args A list of additional arguments passed to patchwork::plot_annotation().

Details

Rows with a missing value in x, y, or counts are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggpiestats.html

Value

A patchwork object combining one plot per level of grouping.var (see combine_plots()). Applying extract_stats() to it returns a list with the statistical details for each of these plots.

See Also

ggpiestats(), ggbarstats(), grouped_ggbarstats()

Examples

```
set.seed(123)
# grouped one-sample proportion test
```

```
grouped_ggpiestats(
  data = mtcars,
  x = cyl,
  grouping.var = am,
  annotation.args = list(title = "Cylinder distribution by transmission type")
)
```

`grouped_ggscatterstats` *Grouped scatterplots with marginal distributions and statistical results*

Description

Applies `ggscatterstats()` across all levels of `grouping.var` and combines the resulting plots using `combine_plots()`.

Usage

```
grouped_ggscatterstats(
  data,
  ...,
  grouping.var,
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from `{dplyr}` should be ungrouped before they are entered as `data`.
`...`	Arguments passed on to `ggscatterstats` `label.var` Variable to use for point labels entered as a symbol (e.g. `var1`). `label.expression` An expression evaluating to a logical vector that determines the subset of data points to label (e.g. `y < 4 & z < 20`). While using this argument with `purrr::pmap()`, you will have to provide a quoted expression (e.g. `quote(y < 4 & z < 20)`). `point.label.args` A list of additional aesthetic arguments passed to `ggrepel::geom_label_repel()`, which displays the point labels. `smooth.line.args` A list of additional aesthetic arguments passed to `ggplot2::geom_smooth()`, which displays the regression line. Its confidence band uses `conf.level`. `marginal` Logical that decides whether marginal distributions are plotted on the axes using `{ggside}` (Default: `TRUE`).

point.width.jitter,point.height.jitter Amount of jitter (in data units) in x and y direction, respectively, passed to ggplot2::position_jitter() (Default: 0, i.e., no jitter). Do not specify the jitter in point.args, because it is shared by the **points** and their **labels**.

xsidehistogram.args,ysidehistogram.args A list of arguments passed to ggside::geom_xsidehistogram() and ggside::geom_ysidehistogram(), respectively, to change the marginal histograms.

xsidehistogram.scale,ysidehistogram.scale A list of arguments passed to ggside::scale_xsidey_continuous() and ggside::scale_ysidex_continuous(), respectively, to control the scale of marginal histograms (e.g., breaks, limits, transform). Default is list() (no modifications).

x The column in data containing the explanatory variable to be plotted on the x-axis.

y The column in data containing the response (outcome) variable to be plotted on the y-axis.

type A character specifying the type of statistical approach:

- "parametric"
- "nonparametric"
- "robust"
- "bayes"

You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.

digits Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

conf.level Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).

tr Trim level for the mean when carrying out robust tests. In case of an error, try reducing the value of tr, which is by default set to 0.2. Lowering the value might help.

bf.prior A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

xlab Label for x axis variable. If NULL (default), variable name for x will be used.

ylab Label for y axis variable. If NULL (default), variable name for y will be used.

bf.message Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: TRUE).

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`subtitle` The text for the plot subtitle. Used only if `results.subtitle = FALSE`.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`point.args` A list of additional aesthetic arguments passed to `ggplot2::geom_point()`.

`ggplot.component` A {ggplot2} component, or a list of components, to add to the plot (Default: `NULL`). This is mostly helpful for the `grouped_` variants, where it is applied to each individual plot.

`ggtheme` A {ggplot2} theme (Default: `theme_ggstatsplot()`). Any {ggplot2} theme (e.g., `ggplot2::theme_bw()`) or theme from an extension package (e.g., `ggthemes::theme_fivethirtyeight()`) can be used. Note that some themes remove plot details. For example, `ggthemes::theme_fivethirtyeight()` removes the secondary Y-axis, which `ggbetweenstats()` uses to show details about pairwise comparisons.

`grouping.var` A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.

`plotgrid.args` A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides`, which has its own argument.

`annotation.args`
A `list` of additional arguments passed to `patchwork::plot_annotation()`.

Details

Rows with a missing value in either `x` or `y` are removed before plotting and analysis.

For details, see: https://www.indrapatil.com/ggstatsplot/articles/web_only/ggscatterstats.html

Value

A `patchwork` object combining one plot per level of `grouping.var` (see `combine_plots()`). Applying `extract_stats()` to it returns a list with the statistical details for each of these plots.

See Also

`ggscatterstats()`, `ggcorrmat()`, `grouped_ggcorrmat()`

Examples

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

library(dplyr, warn.conflicts = FALSE)
library(ggplot2)
```

```
grouped_ggscatterstats(
  data             = filter(movies_long, genre == "Comedy" | genre == "Drama"),
  x                = length,
  y                = rating,
  type             = "robust",
  grouping.var     = genre,
  ggplot.component = list(geom_rug(sides = "b"))
)

# label points matching an expression
grouped_ggscatterstats(
  data             = filter(ggplot2::mpg, cyl != 5),
  x                = displ,
  y                = hwy,
  grouping.var     = cyl,
  type             = "robust",
  label.var        = manufacturer,
  label.expression = hwy > 25 & displ > 2.5,
  ggplot.component = scale_y_continuous(sec.axis = dup_axis())
)

# label all points
grouped_ggscatterstats(
  data            = filter(movies_long, rating == 7, genre %in% c("Drama", "Comedy")),
  x               = budget,
  y               = length,
  grouping.var    = genre,
  bf.message      = FALSE,
  label.var       = "title",
  annotation.args = list(tag_levels = "a")
)
```

---

grouped_ggwithinstats *Grouped box/violin plots for repeated measures comparisons*

Description

Applies ggwithinstats() across all levels of grouping.var and combines the resulting plots using combine_plots().

Usage

```
grouped_ggwithinstats(
  data,
  ...,
  grouping.var,
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

data	A data frame (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix, table, array, etc.) will **not** be accepted. Additionally, grouped data frames from {dplyr} should be ungrouped before they are entered as data.
...	Arguments passed on to ggwithinstats

x The condition (or repeated-measures) variable from data.

point.path,centrality.path Logical that decides whether individual data points and centrality measures, respectively, are connected using ggplot2::geom_path() (Default: TRUE). point.path is relevant only when there are two groups (i.e., for a *t*-test). With many data points, consider point.path = FALSE, as these lines can overwhelm the plot.

centrality.path.args,point.path.args A list of additional aesthetic arguments passed to ggplot2::geom_path() connecting centrality measures and raw data points, respectively.

subject.id The subject (or repeated measures) identifier from data. Each subject must appear at most once per level of x. If NULL (default), observations are paired by their row order within each level of x. **If you are not sure the data are sorted this way, specify this argument.** Rows with a missing x or subject.id are ignored, and subjects with a missing (NA) dependent variable in any condition are dropped list-wise (see statsExpressions::complete_subjects()), so the plot, its labels, and the statistical results are all based on the same complete subjects.

xlab Label for x axis variable. If NULL (default), variable name for x will be used.

ylab Label for y axis variable. If NULL (default), variable name for y will be used.

p.adjust.method Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".

pairwise.display Decides *which* pairwise comparisons to display. Available options are:

- "significant" (abbreviation accepted: "s")
- "non-significant" (abbreviation accepted: "ns")
- "all"
- "none"

Use this to keep the plot uncluttered when many groups are compared. Pairwise comparisons are carried out (and displayed) only when x has three or more levels. For type = "bayes", there are no *p*-values, so all comparisons are displayed unless this is "none".

pairwise.alpha Numeric alpha threshold used to decide which pairwise comparisons are displayed when pairwise.display = "significant" or pairwise.display = "non-significant" (Default: 0.05). It is compared against *p*-values adjusted using p.adjust.method.

`bf.message` Logical that decides whether to display the Bayes Factor in favor of the *null* hypothesis as a caption. Relevant only for parametric tests (Default: `TRUE`).

`results.subtitle` Logical that decides whether the results of statistical tests are displayed as a subtitle (Default: `TRUE`). Setting this to `FALSE` also skips the automatic Bayes Factor caption. Other plot details have their own arguments.

`subtitle` The text for the plot subtitle. Used only if `results.subtitle = FALSE`.

`caption` The text for the plot caption. It will be replaced by the Bayes Factor caption whenever that is displayed (see `bf.message`).

`centrality.plotting` Logical that decides whether central tendency measure is to be displayed as a point with a label (Default: `TRUE`). Function decides which central tendency measure to show depending on the `type` argument.

- **mean** for parametric statistics
- **median** for non-parametric statistics
- **trimmed mean** for robust statistics
- **MAP estimator** for Bayesian statistics

Use `centrality.type` to choose a different centrality measure.

`centrality.type` Decides which centrality measure is displayed. Defaults to the same value as `type`. Can be one of:

- `"parametric"` (for **mean**)
- `"nonparametric"` (for **median**)
- `"robust"` (for **trimmed mean**)
- `"bayes"` (for **MAP estimator**)

As with `type`, abbreviations are accepted.

`point.args` A list of additional aesthetic arguments passed to `ggplot2::geom_point()`.

`boxplot.args` A list of additional aesthetic arguments passed to `ggplot2::geom_boxplot()`. By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). Use `coef` to change this, e.g., `boxplot.args = list(coef = 3)` extends the whiskers to the most extreme observations within 3 * IQR of the hinges.

`violin.args` A list of additional aesthetic arguments passed to `ggplot2::geom_violin()`.

`ggplot.component` A {ggplot2} component, or a list of components, to add to the plot (Default: `NULL`). This is mostly helpful for the `grouped_` variants, where it is applied to each individual plot.

`palette` Name of the discrete palette in `"package::palette"` format to be used for coloring (via {paletteer}). Run `View(paletteer::palettes_d_names)` to see all available options.

`centrality.point.args,centrality.label.args` A list of additional aesthetic arguments passed to `ggplot2::geom_point()` and `ggrepel::geom_label_repel()`, respectively, which display the centrality measure.

`ggsignif.args` A list of additional aesthetic arguments passed to `ggsignif::geom_signif()`.

`ggtheme` A {ggplot2} theme (Default: `theme_ggstatsplot()`). Any {ggplot2} theme (e.g., `ggplot2::theme_bw()`) or theme from an extension package (e.g., `ggthemes::theme_fivethirtyeight()`) can be used. Note that

some themes remove plot details. For example, ggthemes::theme_fivethirtyeight() removes the secondary Y-axis, which ggbetweenstats() uses to show details about pairwise comparisons.

y The response (or outcome or dependent) variable from data.

type A character specifying the type of statistical approach:

- "parametric"
- "nonparametric"
- "robust"
- "bayes"

You can specify just the initial letter (e.g. "np" or "bf"). Matching is on the initial lowercase letter only, so any other value (including capitalized values such as "Bayes") falls back to "parametric" without a warning.

digits Number of digits for rounding or significant figures. May also be "signif" to return significant figures or "scientific" to return scientific notation. Control the number of digits by adding the value as suffix, e.g. digits = "scientific4" to have scientific notation with 4 decimal places, or digits = "signif5" for 5 significant figures (see also signif()).

conf.level Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95).

bf.prior A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.

tr Trim level for the mean when carrying out robust tests. In case of an error, try reducing the value of tr, which is by default set to 0.2. Lowering the value might help.

alternative a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less". You can specify just the initial letter.

grouping.var A single grouping variable. A separate plot is created for each of its levels (in factor level order, or in order of appearance for character variables). Rows with a missing value in this variable are removed.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides, which has its own argument.

annotation.args A list of additional arguments passed to patchwork::plot_annotation().

Value

A patchwork object combining one plot per level of grouping.var (see combine_plots()). Applying extract_stats() to it returns a list with the statistical details for each of these plots.

See Also

ggwithinstats(), ggbetweenstats(), grouped_ggbetweenstats()

Examples

```
# for reproducibility
set.seed(123)
library(dplyr, warn.conflicts = FALSE)
library(ggplot2)

# the most basic function call
grouped_ggwithinstats(
  data             = filter(bugs_long, condition %in% c("HDHF", "HDLF")),
  x                = condition,
  y                = desire,
  subject.id       = subject,
  grouping.var     = gender,
  type             = "np",
  # additional modifications for **each** plot using `{ggplot2}` functions
  ggplot.component = scale_y_continuous(breaks = seq(0, 10, 1), limits = c(0, 10)),
  annotation.args  = list(title = "Desire ratings by condition for each gender")
)
```

---

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

Description

Edgar Anderson's Iris Data in long format

Usage

```
iris_long
```

Format

A data frame with 600 rows and 6 variables

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

Details

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

This is a modified dataset from `{datasets}` package.

Examples

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

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

Description

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

Usage

```
movies_long
```

Format

A data frame with 1,579 rows and 8 variables

- title. Title of the movie.
- year. Year of release.
- budget. Total budget (if known) in US dollars.
- length. Length in minutes.
- rating. Average IMDB user rating.
- votes. Number of IMDB users who rated this movie.
- mpaa. MPAA rating.
- genre. Different genres of movies (Action, Action Comedy, Action Drama, Animated, Comedy, Comedy Drama, Drama, RomCom, Romance Drama).

Details

Modified dataset from `{ggplot2movies}` package.

The internet movie database (IMDB) is a website devoted to collecting movie data supplied by studios and fans. It claims to be the biggest movie database on the web and is run by Amazon.

Source

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

Examples

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

theme_ggstatsplot *Default theme used in* {ggstatsplot}

Description

Theme used by default in all {ggstatsplot} plots, built on ggplot2::theme_bw(). Use the ggtheme argument of any {ggstatsplot} function to choose a different theme.

Usage

```
theme_ggstatsplot()
```

Value

A {ggplot2} theme object.

Examples

```
library(ggplot2)

ggplot(mtcars, aes(wt, mpg)) +
  geom_point() +
  theme_ggstatsplot()
```

Titanic_full *Titanic dataset*

Description

Titanic dataset

Usage

```
Titanic_full
```

Format

A data frame with 2201 rows and 5 variables

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

Details

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

This is a modified dataset from `{datasets}` package.

Examples

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

Index

∗ **datasets**
bugs_long, 3
iris_long, 80
movies_long, 81
Titanic_full, 82

bugs_long, 3, *46*

combine_plots, 4
combine_plots(), *54*, *57*, *60, 61*, *63, 64*, *66, 67*, *69, 70*, *72, 73*, *75, 76*, *79*
correlation::correlation(), *6*, *24*, *27*

dplyr::group_by(), *24*
droplevels(), *8*, *37*, *55*, *70*

extract_caption *(*extract_stats*)*, 5
extract_stats, 5
extract_stats(), *9*, *15*, *22*, *27*, *31*, *35*, *39*, *44*, *49*, *57*, *60*, *66*, *69*, *72*, *75*, *79*
extract_subtitle *(*extract_stats*)*, 5

ggbarstats, 7, *55*
ggbarstats(), *8*, *38*, *40*, *54, 55*, *57*, *70*, *72*
ggbetweenstats, 12, *58*
ggbetweenstats(), *9*, *15*, *22*, *26*, *31*, *34*, *39*, *44*, *49*, *53*, *56, 57*, *59, 60*, *63*, *66*, *69*, *71*, *75*, *79*
ggcoefstats, 20
ggcorrmat, 24, *61*
ggcorrmat(), *45*, *61*, *63*, *75*
ggcorrplot::ggcorrplot(), *26*, *62*
ggdotplotstats, 28, *64*
ggdotplotstats(), *36*, *64*, *66*, *69*
gghistostats, 32, *68*
gghistostats(), *32*, *66, 67*, *69*
ggpiestats, 36, *70*
ggpiestats(), *11*, *57*, *70*, *72*
ggplot2::geom_boxplot(), *14*, *49*, *59*, *78*
ggplot2::geom_errorbar(), *22*, *30*, *66*
ggplot2::geom_label(), *8*, *38*, *55*, *71*
ggplot2::geom_path(), *49*, *77*
ggplot2::geom_point(), *14*, *22*, *30*, *43*, *49*, *59*, *66*, *75*, *78*
ggplot2::geom_smooth(), *43*, *73*
ggplot2::geom_text(), *8*, *55*
ggplot2::geom_violin(), *15*, *49*, *59*, *78*
ggplot2::geom_vline(), *22*, *30*, *35*, *64*, *68*
ggplot2::position_jitter(), *43*, *74*
ggplot2::stat_bin(), *34*, *68*
ggplot2::theme_bw(), *9*, *15*, *22*, *26*, *30*, *34*, *39*, *44*, *49*, *56*, *59*, *63*, *66*, *69*, *71*, *75*, *78*, *82*
ggrepel::geom_label_repel(), *14*, *22*, *38*, *43*, *45*, *49*, *59*, *71*, *73*, *78*
ggscatterstats, 41, *73*
ggscatterstats(), *27*, *63*, *73*, *75*
ggside::geom_xsidehistogram(), *43*, *74*
ggside::geom_ysidehistogram(), *43*, *74*
ggsignif::geom_signif(), *15*, *49*, *59*, *78*
ggwithinstats, 46, *77*
ggwithinstats(), *19*, *60*, *76*, *79*
grouped_ggbarstats, 54
grouped_ggbarstats(), *11*, *40*, *72*
grouped_ggbetweenstats, 57
grouped_ggbetweenstats(), *19*, *53*, *79*
grouped_ggcorrmat, 61
grouped_ggcorrmat(), *27*, *45*, *75*
grouped_ggdotplotstats, 64
grouped_ggdotplotstats(), *32*, *36*, *69*
grouped_gghistostats, 67
grouped_gghistostats(), *32*, *36*, *66*
grouped_ggpiestats, 70
grouped_ggpiestats(), *11*, *40*, *57*
grouped_ggscatterstats, 73
grouped_ggscatterstats(), *27*, *45*, *63*
grouped_ggwithinstats, 76
grouped_ggwithinstats(), *19*, *53*, *60*

iris_long, 80

metaBMA::meta_random(), *21*
metafor::rma(), *21*
metaplus::metaplus(), *21*
movies_long, 81

parameters::model_parameters(), *6*, *21*, *22*
patchwork::plot_annotation(), *4*, *57*, *60*, *63*, *66*, *67*, *72*, *75*, *79*
patchwork::wrap_plots(), *4*, *56*, *60*, *63*, *66*, *67*, *72*, *75*, *79*
performance::model_performance(), *6*
purrr::pmap(), *43*, *73*

signif(), *8*, *14*, *21*, *26*, *30*, *34*, *38*, *43*, *48*, *56*, *59*, *62*, *65*, *68*, *72*, *74*, *79*
statsExpressions::complete_subjects(), *47*, *77*
statsExpressions::pairwise_contingency_table(), *11*, *39*

theme(legend.position=...), *4*
theme_ggstatsplot, 82
theme_ggstatsplot(), *9*, *15*, *22*, *26*, *30*, *34*, *39*, *44*, *49*, *56*, *59*, *63*, *66*, *69*, *71*, *75*, *78*
Titanic_full, 82