Package 'ggstatsplot'

May 8, 2026

Type Package

Title 'ggplot2' Based Plots with Statistical Details

Version 1.0.0

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.03236>.

License MIT + file LICENSE

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

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

Depends R (>= 4.3.0)

Imports correlation (>= 0.8.8), datawizard (>= 1.3.0), dplyr (>= 1.2.1), forcats (>= 1.0.1), ggcorrplot (>= 0.1.4.1), ggplot2 (>= 4.0.2), ggrepel (>= 0.9.8), ggside (>= 0.4.1), ggsignif (>= 0.6.4), glue (>= 1.8.1), insight (>= 1.5.0), paletteer (>= 1.7.0), parameters (>= 0.28.3), patchwork (>= 1.3.2), performance (>= 0.16.0), purrr (>= 1.2.2), rlang (>= 1.2.0), statsExpressions (>= 2.0.0), tidyr (>= 1.3.2), utils

Suggests afex, BayesFactor (>= 0.9.12-4.7), bayestestR, gapminder, knitr, lme4 (>= 1.1-37), MASS, metaBMA, metafor, metaplus, patrick, psych, rmarkdown, rstantools, stats, survival, testthat (>= 3.3.2), tibble, vdiffr (>= 1.0.8), withr, WRS2

VignetteBuilder knitr

Config/Needs/check anthonynorth/roxyglobals

Config/roxyglobals/unique TRUE

Config/testthat/edition 3

Config/testthat/parallel true

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.3.3

NeedsCompilation no

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

Repository CRAN

Date/Publication 2026-04-23 16:10:03 UTC

Contents

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

Description

Tidy version of the "Bugs" dataset.

Usage

```
bugs_long
```

Format

A data frame with 372 rows and 6 variables

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

Details

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

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 will return a combined grid of plots with annotations. In case you want to create a grid of plots, it is **highly recommended** that you use `{patchwork}` package directly and not this wrapper around it which is mostly useful with `{ggstatsplot}` plots. It is exported only for backward compatibility.

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` argument which is already separately specified here.
`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 combined plot with annotation labels.

Examples

```
library(ggplot2)

# first plot
p1 <- ggplot(
  data = subset(iris, iris$Species == "setosa"),
```

```
  aes(x = Sepal.Length, y = Sepal.Width)
) +
  geom_point() +
  labs(title = "setosa")

# second plot
p2 <- ggplot(
  data = subset(iris, 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}` package

Details

These are convenience functions to extract data frames or expressions with statistical details that are used to create expressions displayed in {ggstatsplot} plots as subtitle, caption, etc. Note that all of this analysis is carried out by the {statsExpressions} package. And so if you are using these functions only to extract data frames, you are better off using that package.

The only exception is the ggcorrmat() function. But, if a data frame is what you want, you shouldn't be using ggcorrmat() anyway. You can use correlation::correlation() function which provides tidy data frames by default.

Value

A list of tibbles containing summaries of various statistical analyses. The exact details included will depend on the function.

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. Please note that if there are empty factor levels in your variable, they will be dropped.
y	The variable to use as the **columns** in the contingency table. Please note that if there are empty factor levels in your variable, they will be dropped. Default is NULL. If NULL, one-sample proportion test (a goodness of fit test) will be run for the x variable. Otherwise an appropriate association test will be 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.

paired — Logical indicating whether data came from a within-subjects or repeated measures design study (Default: FALSE).

results.subtitle — Decides whether the results of statistical tests are to be displayed as a subtitle (Default: TRUE). If set to FALSE, only the plot will be returned.

label — Character decides what information needs to be displayed on the label in each pie slice. Possible options are "percentage" (default), "counts", "both".

label.args — Additional aesthetic arguments that will be passed to ggplot2::geom_label().

sample.size.label.args — Additional aesthetic arguments that will be 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 — Decides whether proportion test for x variable is to be carried out for each level of y. Defaults to results.subtitle. In ggbarstats(), only *p*-values from this test will be displayed.

digits.perc — Numeric that decides number of decimal places for percentage labels (Default: 0L).

bf.message — Logical that decides whether to display Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (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). If NULL, no confidence intervals will be computed.

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. Will work only if results.subtitle = FALSE.

caption — The text for the plot caption. This argument is relevant only if bf.message = FALSE.

legend.title	Title text for the legend.
xlab	Label for x axis variable. If NULL (default), variable name for x will be used.
ylab	Labels for y axis variable. If NULL (default), variable name for y will be used.
ggtheme	A {ggplot2} theme. Default value is theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., ggplot2::theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.). But note that sometimes these themes will remove some of the details that {ggstatsplot} plots typically contains. For example, if relevant, ggbetweenstats() shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. ggthemes::theme_fivethirtyeight()) will remove the secondary Y-axis and thus the details as well.
palette	Name of the palette in "package::palette" format to be used for coloring. Passed to paletteer::scale_color_paletteer_d(). Run View(paletteer::palettes_d_names) to see all available options.
ggplot.component	A ggplot component to be added to the plot prepared by {ggstatsplot}. This argument is primarily helpful for grouped_ variants of all primary functions. Default is NULL. The argument should be entered as a {ggplot2} function or a list of {ggplot2} functions.
...	Currently ignored.

Details

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

Summary of graphics

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

Contingency table analyses

The table below provides summary about:

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

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

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 and `x` has more than two levels, pairwise contingency table analyses (Fisher's exact tests) are computed using `statsExpressions::pairwise_contingency_table()`. These pairwise results are **not** displayed in the plot because bar and pie charts lack a natural visual representation for pairwise significance annotations (unlike box/violin plots, which use bracket annotations). Additionally, there is no established convention for overlaying pairwise comparisons on pie charts, and both `ggpiestats()` and `ggbarstats()` are designed to remain visually congruent. The pairwise results are available as a data frame via `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`. In case of a repeated measures or within-subjects design, if `subject.id` argument is not available or not explicitly specified, the function assumes that the data has already been sorted by such an id by the user and creates an internal identifier. So if your data is **not** sorted, the results *can* be inaccurate when there are more than two levels in `x` and there are `NA`s present. The data is expected to be sorted by user in subject-1, subject-2, ..., pattern.
`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.
`pairwise.display`	Decides *which* pairwise comparisons to display. Available options are:

- "significant" (abbreviation accepted: "s")
- "non-significant" (abbreviation accepted: "ns")
- "all"

You can use this argument to make sure that your plot is not uber-cluttered when you have multiple groups being compared and scores of pairwise comparisons being displayed. If set to "none", no pairwise comparisons will be displayed.

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).

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 Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: TRUE).

results.subtitle

Decides whether the results of statistical tests are to be displayed as a subtitle (Default: TRUE). If set to FALSE, only the plot will be returned.

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

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

caption The text for the plot caption. This argument is relevant only if bf.message = FALSE.

title The text for the plot title.

subtitle The text for the plot subtitle. Will work 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). If NULL, no confidence intervals will be computed.

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 centrality 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

If you want default centrality parameter, you can specify this using `centrality.type` argument.

`centrality.type`

Decides which centrality parameter is to be displayed. The default is to choose the same as `type` argument. You can specify this to be:

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

Just as `type` argument, abbreviations are also accepted.

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

A list of additional aesthetic arguments to be passed to `ggplot2::geom_point()` and `ggrepel::geom_label_repel()` geoms, which are involved in mean plotting.

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

`boxplot.args` A list of additional aesthetic arguments passed on to `ggplot2::geom_boxplot()`. By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). To customize whisker length, you can use the `coef` parameter, e.g., `boxplot.args = list(coef = 3)` for whiskers extending to 3 * IQR, or `boxplot.args = list(coef = 0)` to show only the range of the data.

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

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

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

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

`ggplot.component`

A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`...` Currently ignored.

Details

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

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:

- **Effect sizes** are always *unbiased* (Hedges' *g* instead of Cohen's *d*, omega-squared instead of eta-squared). Unbiased estimators correct for the positive bias present in their biased counterparts, especially in small samples, and are recommended for meta-analytic work.
- **Welch's *t*-test and one-way test** are used instead of Student's versions (i.e., equal variances are not assumed). Welch's test performs as well as Student's when variances *are* equal and is substantially more accurate when they are not, making it the unconditionally better default.

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

Centrality measures

The table below provides summary about:

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

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

The table below provides summary about:

- statistical test carried out for inferential statistics
- type of effect size estimate and a measure of uncertainty for this estimate
- 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*, Hedge's *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:

Data requirement: Paired tests assume exactly **one observation per subject per condition**. If your data has multiple trials per cell, aggregate first (e.g., take the mean).

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*, Hedge's *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

The table below provides summary about:

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

Data requirement: Repeated measures tests assume a *complete* design with exactly **one observation per subject per condition**. If your data has multiple trials per cell, aggregate first (e.g., take the mean). Verify with `table(data$subject, data$condition)` — every cell should equal 1.

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

The table below provides summary about:

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

Data requirement: Paired pairwise tests assume exactly **one observation per subject per condition**. If your data has multiple trials per cell, aggregate first (e.g., take the mean).

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 few aspects of the plot that will be invariant across models:

- The dot-whisker plot contains a dot representing the **estimate** and their **confidence intervals** (95% is the default). The estimate can either be effect sizes (for tests that depend on the F-statistic) or regression coefficients (for tests with t-, chi^2-, and z-statistic), etc. The function will, by default, display a helpful x-axis label that should clear up what estimates are being displayed. The confidence intervals can sometimes be asymmetric if bootstrapping was used.
- The label attached to dot will provide more details from the statistical test carried out and it will typically contain estimate, statistic, and *p*-value.
- The caption will contain diagnostic information, if available, about models 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 is.
- The output of this function will be a {ggplot2} object and, thus, it can be further modified (e.g. change themes) with {ggplot2}.

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 at the minimum two columns named term (names of predictors) and estimate (corresponding estimates of coefficients or other quantities of interest).
statistic	Relevant statistic for the model ("t", "f", "z", or "chi") in the label. Relevant only if x is a *data frame*.
conf.int	Logical. Decides whether to display confidence intervals as error bars (Default: TRUE).
conf.level	Numeric deciding level of confidence or credible intervals (Default: 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	This is the same as es_type argument of parameters::model_parameters(). Defaults to "omega" (the unbiased estimator), and relevant for ANOVA-like objects.
meta.analytic.effect	Logical that decides whether subtitle for meta-analysis via linear (mixed-effects) models (default: FALSE). If TRUE, input to argument subtitle will be ignored. This will be 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) should be displayed in caption. Defaults to TRUE.
sort	If "none" (default) do not sort, "ascending" sort by increasing coefficient value, or "descending" sort by decreasing coefficient value.
xlab	Label for x axis variable. If NULL (default), variable name for x will be used.
ylab	Labels 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. The input to this argument will be ignored if meta.analytic.effect is set to TRUE.
caption	The text for the plot caption. This argument is relevant only if bf.message = FALSE.

only.significant
: If TRUE, only stats labels for significant effects is 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 to be passed to the ggplot2::geom_point().

errorbar.args
: Additional arguments that will be passed to geom_errorbar() geom. Please see documentation for that function to know more about these arguments.

vline
: Decides whether to display a vertical line (Default: "TRUE").

vline.args
: Additional arguments that will be passed to geom_vline geom. Please see documentation for that function to know more about these arguments.

stats.labels
: Logical. Decides whether the statistic and *p*-values for each coefficient are to be attached to each dot as a text label using {ggrepel} (Default: TRUE).

stats.label.color
: Color for the labels. If set to NULL, colors will be chosen from the specified package (Default: "RColorBrewer") and palette (Default: "Dark2").

stats.label.args
: Additional arguments that will be passed to ggrepel::geom_label_repel().

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

ggtheme
: A {ggplot2} theme. Default value is theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., ggplot2::theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.). But note that sometimes these themes will remove some of the details that {ggstatsplot} plots typically contains. For example, if relevant, ggbetweenstats() shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. ggthemes::theme_fivethirtyeight()) will remove the secondary Y-axis and thus the details as well.

...
: Additional arguments to tidying method. For more, see parameters::model_parameters().

Details

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

Summary of graphics

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

Random-effects meta-analysis

The table below provides summary about:

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

1. In case you want to carry out meta-analysis, you will be asked to install the needed packages (`{metafor}`, `{metaplus}`, or `{metaBMA}`) if they are unavailable.
2. All rows of regression estimates where either of the following quantities is `NA` will be removed if labels are requested: `estimate`, `statistic`, `p.value`.
3. Given the rapid pace at which new methods are added to these packages, it is recommended that you install development versions of `{easystats}` packages using the `install_latest()` function from `{easystats}`.

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`	List of variables for which the correlation matrix is to be computed and visualized. 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"`, display full matrix, lower triangular or upper triangular matrix.
`type`	A character specifying the type of statistical approach: • `"parametric"` • `"nonparametric"` • `"robust"` • `"bayes"` You can specify just the initial letter.
`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 *p*-value in *p*-value matrix is bigger than `sig.level`, then the corresponding correlation coefficient is regarded as insignificant and flagged as such in the plot.

`conf.level` Scalar between `0` and `1` (default: `95%` confidence/credible intervals, `0.95`). If `NULL`, no confidence intervals will be computed.

`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 (only valid when `insig = "pch"`). Default: `pch = "cross"`.

`ggcorrplot.args` A list of additional (mostly aesthetic) arguments that will be passed to `ggcorrplot::ggcorrplot()` function. The list should avoid any of the following arguments since they are already internally being used: `corr`, `method`, `p.mat`, `sig.level`, `ggtheme`, `colors`, `lab`, `pch`, `legend.title`, `digits`.

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

`ggplot.component` A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`title` The text for the plot title.

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`...` Currently ignored.

Details

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

Summary of graphics

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

Correlation analyses

The table below provides summary about:

- statistical test carried out for inferential statistics
- type of effect size estimate and a measure of uncertainty for this estimate
- 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)
library(ggcorrplot)
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.
xlab	Label for x axis variable. If NULL (default), variable name for x will be used.
ylab	Labels 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. Will work only if results.subtitle = FALSE.
caption	The text for the plot caption. This argument is relevant only if bf.message = FALSE.
type	A character specifying the type of statistical approach: • "parametric" • "nonparametric" • "robust" • "bayes" You can specify just the initial letter.
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 Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: TRUE).
conf.int	Logical. 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). If NULL, no confidence intervals will be computed.
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`
: Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned.

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

`errorbar.args`
: Additional arguments that will be passed to `geom_errorbar()` geom. Please see documentation for that function to know more about these arguments.

`centrality.plotting`
: Logical that decides whether centrality 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

  If you want default centrality parameter, you can specify this using `centrality.type` argument.

`centrality.type`
: Decides which centrality parameter is to be displayed. The default is to choose the same as `type` argument. You can specify this to be:

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

  Just as `type` argument, abbreviations are also accepted.

`centrality.line.args`
: A list of additional aesthetic arguments to be passed to the `ggplot2::geom_line()` used to display the lines corresponding to the centrality parameter.

`ggplot.component`
: A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`ggtheme`
: A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

`...`
: Currently ignored.

Details

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

Summary of graphics

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

One-sample tests

The table below provides summary about:

- statistical test carried out for inferential statistics
- type of effect size estimate and a measure of uncertainty for this estimate
- 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*, Hedge's *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`, `gghistostats`, `grouped_ggdotplotstats`

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 the max(x) - min(x) / sqrt(N). You should always check this value and explore multiple widths to find the best to illustrate 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. Will work only if results.subtitle = FALSE.
caption	The text for the plot caption. This argument is relevant only if bf.message = FALSE.
type	A character specifying the type of statistical approach: • "parametric" • "nonparametric" • "robust" • "bayes" You can specify just the initial letter.
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 Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: TRUE).
conf.level	Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95). If NULL, no confidence intervals will be computed.
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 value is theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., ggplot2::theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.). But note that sometimes these themes will remove some of the details that {ggstatsplot} plots typically contains. For example, if relevant, ggbetweenstats() shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. ggthemes::theme_fivethirtyeight()) will remove the secondary Y-axis and thus the details as well.

results.subtitle
: Decides whether the results of statistical tests are to be displayed as a subtitle (Default: TRUE). If set to FALSE, only the plot will be returned.

bin.args
: A list of additional aesthetic arguments to be passed to the stat_bin used to display the bins. Do not specify binwidth argument in this list since it has already been specified using the dedicated argument.

centrality.plotting
: Logical that decides whether centrality 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

  If you want default centrality parameter, you can specify this using centrality.type argument.

centrality.type
: Decides which centrality parameter is to be displayed. The default is to choose the same as type argument. You can specify this to be:

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

  Just as type argument, abbreviations are also accepted.

centrality.line.args
: A list of additional aesthetic arguments to be passed to the ggplot2::geom_line() used to display the lines corresponding to the centrality parameter.

ggplot.component
: A ggplot component to be added to the plot prepared by {ggstatsplot}. This argument is primarily helpful for grouped_ variants of all primary functions. Default is NULL. The argument should be entered as a {ggplot2} function or a list of {ggplot2} functions.

...
: Currently ignored.

Details

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

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

The table below provides summary about:

- statistical test carried out for inferential statistics
- type of effect size estimate and a measure of uncertainty for this estimate
- 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*, Hedge's *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. Please note that if there are empty factor levels in your variable, they will be dropped.
y	The variable to use as the **columns** in the contingency table. Please note that if there are empty factor levels in your variable, they will be dropped. Default is NULL. If NULL, one-sample proportion test (a goodness of fit test) will be run for the x variable. Otherwise an appropriate association test will be 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.
paired	Logical indicating whether data came from a within-subjects or repeated measures design study (Default: FALSE).
results.subtitle	Decides whether the results of statistical tests are to be displayed as a subtitle (Default: TRUE). If set to FALSE, only the plot will be returned.
label	Character decides what information needs to be displayed on the label in each pie slice. Possible options are "percentage" (default), "counts", "both".
label.args	Additional aesthetic arguments that will be passed to ggplot2::geom_label().
label.repel	Whether labels should be repelled using {ggrepel} package. This can be helpful in case of overlapping labels.
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	Decides whether proportion test for x variable is to be carried out for each level of y. Defaults to results.subtitle. In ggbarstats(), only *p*-values from this test will be displayed.
digits.perc	Numeric that decides number of decimal places for percentage labels (Default: 0L).
bf.message	Logical that decides whether to display Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (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). If NULL, no confidence intervals will be computed.
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. Will work only if results.subtitle = FALSE.
caption	The text for the plot caption. This argument is relevant only if bf.message = FALSE.
legend.title	Title text for the legend.
ggtheme	A {ggplot2} theme. Default value is theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., ggplot2::theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.). But note that sometimes these themes will remove some of the details that {ggstatsplot} plots typically contains. For example, if relevant, ggbetweenstats() shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. ggthemes::theme_fivethirtyeight()) will remove the secondary Y-axis and thus the details as well.
palette	Name of the palette in "package::palette" format to be used for coloring. Passed to paletteer::scale_color_paletteer_d(). Run View(paletteer::palettes_d_names) to see all available options.
ggplot.component	A ggplot component to be added to the plot prepared by {ggstatsplot}. This argument is primarily helpful for grouped_ variants of all primary functions. Default is NULL. The argument should be entered as a {ggplot2} function or a list of {ggplot2} functions.
...	Currently ignored.

Details

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

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 and `x` has more than two levels, pairwise contingency table analyses (Fisher's exact tests) are computed using `statsExpressions::pairwise_contingency_table()`. These pairwise results are **not** displayed in the plot because bar and pie charts lack a natural visual representation for pairwise significance annotations (unlike box/violin plots, which use bracket annotations). Additionally, there is no established convention for overlaying pairwise comparisons on pie charts, and both `ggpiestats()` and `ggbarstats()` are designed to remain visually congruent. The pairwise results are available as a data frame via `extract_stats(plot)$pairwise_comparisons_data`.

Contingency table analyses

The table below provides summary about:

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

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

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 distributions 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.

conf.level
: Scalar between 0 and 1 (default: 95% confidence/credible intervals, 0.95). If NULL, no confidence intervals will be computed.

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 Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (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
: Decides whether the results of statistical tests are to be displayed as a subtitle (Default: TRUE). If set to FALSE, only the plot will be returned.

label.var
: Variable to use for points 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
: Decides whether marginal distributions will be plotted on axes using {ggside} functions. The default is TRUE. The package {ggside} must already be installed by the user.

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

point.width.jitter, point.height.jitter
: Degree of jitter in x and y direction, respectively. Defaults to 0 (0%) of the resolution of the data. Note that the jitter should not be specified in the point.args because this information will be passed to two different geoms: one displaying the **points** and the other displaying the ***labels** for these points.

point.label.args
: A list of additional aesthetic arguments to be passed to ggrepel::geom_label_repel()geom used to display the labels.

smooth.line.args
: A list of additional aesthetic arguments to be passed to geom_smooth geom used to display the regression line.

`xsidehistogram.args`, `ysidehistogram.args`
: A list of arguments passed to respective `geom_s` from the `{ggside}` package to change the marginal distribution histograms plots.

`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`
: Labels 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. Will work only if `results.subtitle = FALSE`.

`caption`
: The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`ggtheme`
: A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

`ggplot.component`
: A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`...`
: Currently ignored.

Details

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

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

The table below provides summary about:

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

The plot uses `ggrepel::geom_label_repel()` to attempt to keep labels from over-lapping to the largest degree possible. As a consequence plot times will slow down massively (and the plot file will grow in size) if you have a lot of labels that 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.

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 grouping (or independent) variable from `data`. In case of a repeated measures or within-subjects design, if `subject.id` argument is not available or not explicitly specified, the function assumes that the data has already been sorted by such an id by the user and creates an internal identifier. So if your data is **not** sorted, the results *can* be inaccurate when there are more than two levels in `x` and there are `NA`s present. The data is expected to be sorted by user in subject-1, subject-2, ..., pattern.
`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.
`subject.id`	Across repeated measures conditions, each row in the dataset must correspond to a unique unit (e.g., subject or participant). If your data frame is already in such a format, you can ignore the `subject.id` argument (the function will use row number to pair observations). **But if you are not sure, it is always better to specify this argument.** Note that if there are any missing values (i.e., `NA`) in the dependent variable and the `subject.id` is not specified, they will be dropped using a list-wise approach. If you specify `subject.id`, partially observed subjects will still be shown in the plot, but inferential statistics will be computed using only complete repeated-measures pairs.
`pairwise.display`	Decides *which* pairwise comparisons to display. Available options are: • `"significant"` (abbreviation accepted: `"s"`) • `"non-significant"` (abbreviation accepted: `"ns"`) • `"all"`

You can use this argument to make sure that your plot is not uber-cluttered when you have multiple groups being compared and scores of pairwise comparisons being displayed. If set to `"none"`, no pairwise comparisons will be displayed.

`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`).

`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 Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: `TRUE`).

`results.subtitle`

Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned.

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

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`title` The text for the plot title.

`subtitle` The text for the plot subtitle. Will work 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`). If `NULL`, no confidence intervals will be computed.

`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 centrality 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

If you want default centrality parameter, you can specify this using `centrality.type` argument.

`centrality.type`

Decides which centrality parameter is to be displayed. The default is to choose the same as `type` argument. You can specify this to be:

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

Just as `type` argument, abbreviations are also accepted.

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

A list of additional aesthetic arguments to be passed to `ggplot2::geom_point()` and `ggrepel::geom_label_repel()` geoms, which are involved in mean plotting.

`centrality.path.args`, `point.path.args`

A list of additional aesthetic arguments passed on to `ggplot2::geom_path()` connecting raw data points and mean points.

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

`point.path`, `centrality.path`

Logical that decides whether individual data points and means, respectively, should be connected using `ggplot2::geom_path()`. Both default to `TRUE`. Note that `point.path` argument is relevant only when there are two groups (i.e., in case of a *t*-test). In case of large number of data points, it is advisable to set `point.path = FALSE` as these lines can overwhelm the plot.

`boxplot.args` A list of additional aesthetic arguments passed on to `ggplot2::geom_boxplot()`. By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). To customize whisker length, you can use the `coef` parameter, e.g., `boxplot.args = list(coef = 3)` for whiskers extending to 3 * IQR, or `boxplot.args = list(coef = 0)` to show only the range of the data.

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

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

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

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

`ggplot.component`
: A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`...`
: Currently ignored.

Details

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

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

The table below provides summary about:

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

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

The table below provides summary about:

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

- 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*, Hedge's *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:

Data requirement: Paired tests assume exactly **one observation per subject per condition**. If your data has multiple trials per cell, aggregate first (e.g., take the mean).

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*, Hedge's *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

The table below provides summary about:

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

- 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:

Data requirement: Repeated measures tests assume a *complete* design with exactly **one observation per subject per condition**. If your data has multiple trials per cell, aggregate first (e.g., take the mean). Verify with `table(data$subject, data$condition)` — every cell should equal 1.

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

The table below provides summary about:

- statistical test carried out for inferential statistics

- type of effect size estimate and a measure of uncertainty for this estimate
- 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:

Data requirement: Paired pairwise tests assume exactly **one observation per subject per condition**. If your data has multiple trials per cell, aggregate first (e.g., take the mean).

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`, `ggbetweenstats`, `grouped_ggwithinstats`

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

# 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,
  pairwise.display = "none",
  results.subtitle = FALSE
)

# extracting details from statistical tests
extract_stats(p)

# 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

Helper function for `ggstatsplot::ggbarstats()` to apply this function across multiple levels of a given factor and combining the resulting plots using `ggstatsplot::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` `sample.size.label.args` Additional aesthetic arguments that will be passed to `ggplot2::geom_text()`. `x` The variable to use as the **rows** in the contingency table. Please note that if there are empty factor levels in your variable, they will be dropped. `y` The variable to use as the **columns** in the contingency table. Please note that if there are empty factor levels in your variable, they will be dropped. Default is `NULL`. If `NULL`, one-sample proportion test (a goodness of fit test) will be run for the `x` variable. Otherwise an appropriate association test will be run. `proportion.test` Decides whether proportion test for `x` variable is to be carried out for each level of `y`. Defaults to `results.subtitle`. In `ggbarstats()`, only *p*-values from this test will be displayed. `digits.perc` Numeric that decides number of decimal places for percentage labels (Default: `0L`). `label` Character decides what information needs to be displayed on the label in each pie slice. Possible options are `"percentage"` (default), `"counts"`, `"both"`. `label.args` Additional aesthetic arguments that will be passed to `ggplot2::geom_label()`. `legend.title` Title text for the legend. `p.adjust.method` Adjustment method for *p*-values for multiple comparisons. Possible methods are: `"holm"` (default), `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, `"none"`.

`bf.message` Logical that decides whether to display Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: `TRUE`).

`results.subtitle` Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned.

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`ggplot.component` A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`palette` Name of the palette in `"package::palette"` format to be used for coloring. Passed to `paletteer::scale_color_paletteer_d()`. Run `View(paletteer::palettes` to see all available options.

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

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

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

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`). If `NULL`, no confidence intervals will be computed.

`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`).

`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.

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

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

`grouping.var` A single grouping variable.

`plotgrid.args` A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides` argument which is already separately specified here.

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

Details

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

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

Violin plots for group or condition comparisons in between-subjects designs repeated across all levels of a grouping variable.

Description

Helper function for `ggstatsplot::ggbetweenstats` to apply this function across multiple levels of a given factor and combining the resulting plots using `ggstatsplot::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` Labels 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"` You can use this argument to make sure that your plot is not uber-cluttered when you have multiple groups being compared and scores of pairwise comparisons being displayed. If set to `"none"`, no pairwise comparisons will be displayed. `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`). `bf.message` Logical that decides whether to display Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: `TRUE`). `results.subtitle` Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned. `subtitle` The text for the plot subtitle. Will work only if `results.subtitle = FALSE`. `caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`centrality.plotting` Logical that decides whether centrality 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

If you want default centrality parameter, you can specify this using `centrality.type` argument.

`centrality.type` Decides which centrality parameter is to be displayed. The default is to choose the same as `type` argument. You can specify this to be:

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

Just as `type` argument, abbreviations are also accepted.

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

`boxplot.args` A list of additional aesthetic arguments passed on to `ggplot2::geom_boxplot()`. By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). To customize whisker length, you can use the `coef` parameter, e.g., `boxplot.args = list(coef = 3)` for whiskers extending to 3 * IQR, or `boxplot.args = list(coef = 0)` to show only the range of the data.

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

`ggplot.component` A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`palette` Name of the palette in `"package::palette"` format to be used for coloring. Passed to `paletteer::scale_color_paletteer_d()`. Run `View(paletteer::palettes` to see all available options.

`centrality.point.args,centrality.label.args` A list of additional aesthetic arguments to be passed to `ggplot2::geom_point()` and `ggrepel::geom_label_repel()` geoms, which are involved in mean plotting.

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

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

x The grouping (or independent) variable from data. In case of a repeated measures or within-subjects design, if subject.id argument is not available or not explicitly specified, the function assumes that the data has already been sorted by such an id by the user and creates an internal identifier. So if your data is **not** sorted, the results *can* be inaccurate when there are more than two levels in x and there are NAs present. The data is expected to be sorted by user in subject-1, subject-2, ..., pattern.

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.

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). If NULL, no confidence intervals will be computed.

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.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides argument which is already separately specified here.

annotation.args

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

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` *Visualization of a correlalogram (or correlation matrix) for all levels of a grouping variable*

Description

Helper function for `ggstatsplot::ggcorrmat()` to apply this function across multiple levels of a given factor and combining the resulting plots using `ggstatsplot::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 List of variables for which the correlation matrix is to be computed and visualized. 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", display full matrix, lower triangular or upper triangular matrix.

sig.level Significance level (Default: 0.05). If the *p*-value in *p*-value matrix is bigger than sig.level, then the corresponding correlation coefficient is regarded as insignificant and flagged as such in the plot.

pch Decides the point shape to be used for insignificant correlation coefficients (only valid when insig = "pch"). Default: pch = "cross".

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 that will be passed to ggcorrplot::ggcorrplot() function. The list should avoid any of the following arguments since they are already internally being used: corr, method, p.mat, sig.level, ggtheme, colors, lab, pch, legend.title, digits.

type A character specifying the type of statistical approach:

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

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). If NULL, no confidence intervals will be computed.

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"`.

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`ggplot.component` A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

`grouping.var` A single grouping variable.

`plotgrid.args` A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides` argument which is already separately specified here.

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

Details

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

See Also

`ggcorrmat`, `ggscatterstats`, `grouped_ggscatterstats`

Examples

```
set.seed(123)

grouped_ggcorrmat(
  data = iris,
  grouping.var = Species,
  type = "robust",
  colors = c("#0072B2", "white", "#D55E00"),
```

```
  p.adjust.method = "holm",
  plotgrid.args = list(ncol = 1L),
  annotation.args = list(tag_levels = "i")
)
```

`grouped_ggdotplotstats` *Grouped histograms for distribution of a labeled numeric variable*

Description

Helper function for `ggstatsplot::ggdotplotstats()` to apply this function across multiple levels of a given factor and combining the resulting plots using `ggstatsplot::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. `centrality.line.args` A list of additional aesthetic arguments to be passed to the `ggplot2::geom_line()` used to display the lines corresponding to the centrality parameter. `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. `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`). If `NULL`, no confidence intervals will be computed.

`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 Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: `TRUE`).

`results.subtitle` Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned.

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`centrality.plotting` Logical that decides whether centrality 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

If you want default centrality parameter, you can specify this using `centrality.type` argument.

`centrality.type` Decides which centrality parameter is to be displayed. The default is to choose the same as `type` argument. You can specify this to be:

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

Just as `type` argument, abbreviations are also accepted.

ggplot.component A ggplot component to be added to the plot prepared by {ggstatsplot}. This argument is primarily helpful for grouped_ variants of all primary functions. Default is NULL. The argument should be entered as a {ggplot2} function or a list of {ggplot2} functions.

ggtheme A {ggplot2} theme. Default value is theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., ggplot2::theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps(), etc.). But note that sometimes these themes will remove some of the details that {ggstatsplot} plots typically contains. For example, if relevant, ggbetweenstats() shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. ggthemes::theme_fivethirtyeight()) will remove the secondary Y-axis and thus the details as well.

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

errorbar.args Additional arguments that will be passed to geom_errorbar() geom. Please see documentation for that function to know more about these arguments.

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

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

grouping.var A single grouping variable.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides argument which is already separately specified here.

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

Details

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

See Also

grouped_gghistostats, ggdotplotstats, gghistostats

Examples

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

# removing factor level with very few no. of observations
df <- filter(ggplot2::mpg, cyl %in% c("4", "6", "8"))

# plot
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

Helper function for `ggstatsplot::gghistostats` to apply this function across multiple levels of a given factor and combining the resulting plots using `ggstatsplot::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.
`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 the `max(x) - min(x) / sqrt(N)`. You should always check this value and explore multiple widths to find the best to illustrate the stories in your data.
`plotgrid.args`	A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides` argument which is already separately specified here.
`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 to be passed to the `stat_bin` used to display the bins. Do not specify `binwidth` argument in this list since it has already been specified using the dedicated argument.

`centrality.line.args` A list of additional aesthetic arguments to be passed to the `ggplot2::geom_line()` used to display the lines corresponding to the centrality parameter.

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

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

You can specify just the initial letter.

`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`). If `NULL`, no confidence intervals will be computed.

`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 Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: `TRUE`).

`results.subtitle` Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned.

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`centrality.plotting` Logical that decides whether centrality 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

If you want default centrality parameter, you can specify this using `centrality.type` argument.

`centrality.type` Decides which centrality parameter is to be displayed. The default is to choose the same as `type` argument. You can specify this to be:

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

Just as `type` argument, abbreviations are also accepted.

`ggplot.component` A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

Details

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

See Also

`gghistostats`, `ggdotplotstats`, `grouped_ggdotplotstats`

Examples

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

# plot
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

Helper function for `ggstatsplot::ggpiestats` to apply this function across multiple levels of a given factor and combining the resulting plots using `ggstatsplot::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. Please note that if there are empty factor levels in your variable, they will be dropped. `y` The variable to use as the **columns** in the contingency table. Please note that if there are empty factor levels in your variable, they will be dropped. Default is `NULL`. If `NULL`, one-sample proportion test (a goodness of fit test) will be run for the `x` variable. Otherwise an appropriate association test will be run. `proportion.test` Decides whether proportion test for `x` variable is to be carried out for each level of `y`. Defaults to `results.subtitle`. In `ggbarstats()`, only *p*-values from this test will be displayed. `digits.perc` Numeric that decides number of decimal places for percentage labels (Default: `0L`). `label` Character decides what information needs to be displayed on the label in each pie slice. Possible options are `"percentage"` (default), `"counts"`, `"both"`. `label.args` Additional aesthetic arguments that will be passed to `ggplot2::geom_label()`.

`label.repel` Whether labels should be repelled using `{ggrepel}` package. This can be helpful in case of overlapping labels.

`legend.title` Title text for the legend.

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

`bf.message` Logical that decides whether to display Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: `TRUE`).

`results.subtitle` Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned.

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`ggplot.component` A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`palette` Name of the palette in `"package::palette"` format to be used for coloring. Passed to `paletteer::scale_color_paletteer_d()`. Run `View(paletteer::palettes` to see all available options.

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes (e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

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

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

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`). If `NULL`, no confidence intervals will be computed.

`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`).

`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.

`plotgrid.args` A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides` argument which is already separately specified here.

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

Details

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

See Also

`ggbarstats`, `ggpiestats`, `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

Scatterplot with marginal distributions for all levels of a grouping variable

---

Description

Grouped scatterplots from `{ggplot2}` combined with marginal distribution plots with statistical details added as a subtitle.

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 points 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 to be passed to ggrepel::geom_label_repel()geom used to display the labels. smooth.line.args A list of additional aesthetic arguments to be passed to geom_smooth geom used to display the regression line. marginal Decides whether marginal distributions will be plotted on axes using {ggside} functions. The default is TRUE. The package {ggside} must already be installed by the user. point.width.jitter,point.height.jitter Degree of jitter in x and y direction, respectively. Defaults to 0 (0%) of the resolution of the data. Note that the jitter should not be specified in the point.args because this information will be passed to two different geoms: one displaying the **points** and the other displaying the ***labels** for these points. xsidehistogram.args,ysidehistogram.args A list of arguments passed to respective geom_s from the {ggside} package to change the marginal distribution histograms plots. 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.

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). If NULL, no confidence intervals will be computed.

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

bf.message Logical that decides whether to display Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: TRUE).

results.subtitle Decides whether the results of statistical tests are to be displayed as a subtitle (Default: TRUE). If set to FALSE, only the plot will be returned.

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

caption The text for the plot caption. This argument is relevant only if bf.message = FALSE.

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

ggplot.component A ggplot component to be added to the plot prepared by {ggstatsplot}. This argument is primarily helpful for grouped_ variants of all primary functions. Default is NULL. The argument should be entered as a {ggplot2} function or a list of {ggplot2} functions.

ggtheme A {ggplot2} theme. Default value is theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., ggplot2::theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps(), etc.). But note that sometimes these themes will remove some of the details that {ggstatsplot} plots typically contains. For example, if relevant, ggbetweenstats() shows details about multiple comparison test as a label on the secondary Y-axis. Some themes

(e.g. `ggthemes::theme_fivethirtyeight()`) will remove the secondary Y-axis and thus the details as well.

`grouping.var` A single grouping variable.

`plotgrid.args` A `list` of additional arguments passed to `patchwork::wrap_plots()`, except for `guides` argument which is already separately specified here.

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

Details

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

See Also

ggscatterstats, ggcorrmat, grouped_ggcorrmat

Examples

```
# to ensure 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"))
)

# using labeling
# (also show how to modify basic plot from within function call)
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())
)

# labeling without expression
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")
)

# customize marginal histogram bins and scales
grouped_ggscatterstats(
  data = filter(movies_long, genre %in% c("Drama", "Comedy")),
  x = rating,
  y = length,
  grouping.var = genre,
  results.subtitle = FALSE,
  xsidehistogram.args = list(fill = "#4285F4", color = "black", na.rm = TRUE, bins = 20),
 ysidehistogram.args = list(fill = "#EA4335", color = "black", na.rm = TRUE, binwidth = 10),
  xsidehistogram.scale = list(breaks = seq(0, 200, 50)),
  ysidehistogram.scale = list(breaks = seq(0, 200, 50))
)
```

`grouped_ggwithinstats` *Violin plots for group or condition comparisons in within-subjects designs repeated across all levels of a grouping variable.*

Description

A combined plot of comparison plot created for levels of a grouping variable.

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`

`point.path,centrality.path` Logical that decides whether individual data points and means, respectively, should be connected using `ggplot2::geom_path()`. Both default to `TRUE`. Note that `point.path` argument is relevant only when there are two groups (i.e., in case of a *t*-test). In case of large number of data points, it is advisable to set `point.path = FALSE` as these lines can overwhelm the plot.

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

`subject.id` Across repeated measures conditions, each row in the dataset must correspond to a unique unit (e.g., subject or participant). If your data frame is already in such a format, you can ignore the `subject.id` argument (the function will use row number to pair observations). **But if you are not sure, it is always better to specify this argument.** Note that if there are any missing values (i.e., `NA`) in the dependent variable and the `subject.id` is not specified, they will be dropped using a list-wise approach. If you specify `subject.id`, partially observed subjects will still be shown in the plot, but inferential statistics will be computed using only complete repeated-measures pairs.

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

`ylab` Labels 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"`

You can use this argument to make sure that your plot is not uber-cluttered when you have multiple groups being compared and scores of pairwise comparisons being displayed. If set to `"none"`, no pairwise comparisons will be displayed.

`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`).

`bf.message` Logical that decides whether to display Bayes Factor in favor of the *null* hypothesis. This argument is relevant only **for parametric test** (Default: `TRUE`).

`results.subtitle` Decides whether the results of statistical tests are to be displayed as a subtitle (Default: `TRUE`). If set to `FALSE`, only the plot will be returned.

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

`caption` The text for the plot caption. This argument is relevant only if `bf.message = FALSE`.

`centrality.plotting` Logical that decides whether centrality 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

If you want default centrality parameter, you can specify this using `centrality.type` argument.

`centrality.type` Decides which centrality parameter is to be displayed. The default is to choose the same as `type` argument. You can specify this to be:

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

Just as `type` argument, abbreviations are also accepted.

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

`boxplot.args` A list of additional aesthetic arguments passed on to `ggplot2::geom_boxplot()`. By default, the whiskers extend to 1.5 times the interquartile range (IQR) from the box (Tukey-style). To customize whisker length, you can use the `coef` parameter, e.g., `boxplot.args = list(coef = 3)` for whiskers extending to 3 * IQR, or `boxplot.args = list(coef = 0)` to show only the range of the data.

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

`ggplot.component` A `ggplot` component to be added to the plot prepared by `{ggstatsplot}`. This argument is primarily helpful for `grouped_` variants of all primary functions. Default is `NULL`. The argument should be entered as a `{ggplot2}` function or a list of `{ggplot2}` functions.

`palette` Name of the palette in `"package::palette"` format to be used for coloring. Passed to `paletteer::scale_color_paletteer_d()`. Run `View(paletteer::palettes` to see all available options.

`centrality.point.args,centrality.label.args` A list of additional aesthetic arguments to be passed to `ggplot2::geom_point()` and `ggrepel::geom_label_repel()` geoms, which are involved in mean plotting.

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

`ggtheme` A `{ggplot2}` theme. Default value is `theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `ggplot2::theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.). But note that sometimes these themes will remove some of the details that `{ggstatsplot}` plots typically contains. For example, if relevant, `ggbetweenstats()` shows details about multiple comparison test as a label on the secondary Y-axis. Some themes

(e.g. ggthemes::theme_fivethirtyeight()) will remove the secondary Y-axis and thus the details as well.

x The grouping (or independent) variable from data. In case of a repeated measures or within-subjects design, if subject.id argument is not available or not explicitly specified, the function assumes that the data has already been sorted by such an id by the user and creates an internal identifier. So if your data is **not** sorted, the results *can* be inaccurate when there are more than two levels in x and there are NAs present. The data is expected to be sorted by user in subject-1, subject-2, ..., pattern.

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.

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). If NULL, no confidence intervals will be computed.

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.

plotgrid.args A list of additional arguments passed to patchwork::wrap_plots(), except for guides argument which is already separately specified here.

annotation.args

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

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 5 variables

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

Details

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

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

Examples

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

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

Description

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

Usage

```
movies_long
```

Format

A data frame with 1,579 rows and 8 variables

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

Details

Modified dataset from `{ggplot2movies}` package.

The internet movie database (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

Common theme used across all plots generated in {ggstatsplot} and *assumed* by the author to be aesthetically pleasing to the user. The theme is a wrapper around ggplot2::theme_bw().

All {ggstatsplot} functions have a ggtheme parameter that let you choose a different theme.

Usage

```
theme_ggstatsplot()
```

Value

A ggplot 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