Package 'ggstatsplot'

January 15, 2022

Type Package

Title 'ggplot2' Based Plots with Statistical Details

Version 0.9.1

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.

License GPL-3 | file LICENSE

URL https://indrajeetpatil.github.io/ggstatsplot/, https://github.com/IndrajeetPatil/ggstatsplot

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

Depends R (>= 4.0.0)

Imports correlation, dplyr, ggplot2, ggrepel, ggsignif, glue, insight (>= 0.15.0), paletteer, parameters (>= 0.16.0), patchwork, performance, purrr, rlang, stats, statsExpressions (>= 1.3.0), tidyr, utils, WRS2

Suggests afex, BayesFactor (>= 0.9.12-4.3), forcats, gapminder, ggcorrplot, gginnards, ggside, knitr, lme4, MASS, metaBMA, metafor, PMCMRplus, psych, rmarkdown, spelling, survival, testthat (>= 3.1.0), tibble, vdiffr

VignetteBuilder knitr

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.1.2

Config/testthat/edition 3

Config/testthat/parallel true

NeedsCompilation no

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

Repository CRAN

Date/Publication 2022-01-14 23:42:42 UTC

R topics documented:

ggstatsplot-package *ggstatsplot: 'ggplot2' Based Plots with Statistical Details*

Description

{ggstatsplot} is an extension of {ggplot2} package. It creates graphics with details from statistical tests included in the plots themselves. It provides an easier API 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 tests: parametric, nonparametric, robust, and Bayesian versions of *t*-test/ANOVA, correlation analyses, contingency table analysis, meta-analysis, and regression analyses.

Details

ggstatsplot

The main functions are:

- ggbetweenstats function to produce information-rich comparison plot *between* different groups or conditions with {ggplot2} and details from the statistical tests in the subtitle.
- ggwithinstats function to produce information-rich comparison plot *within* different groups or conditions with {ggplot2} and details from the statistical tests in the subtitle.
- ggscatterstats function to produce {ggplot2} scatterplots along with a marginal distribution plots from ggside package and details from the statistical tests in the subtitle.
- ggpiestats function to produce pie chart with details from the statistical tests in the subtitle.
- ggbarstats function to produce stacked bar chart with details from the statistical tests in the subtitle.
- gghistostats function to produce histogram for a single variable with results from one sample test displayed in the subtitle.
- ggdotplotstats function to produce Cleveland-style dot plots/charts for a single variable with labels and results from one sample test displayed in the subtitle.
- ggcorrmat function to visualize the correlation matrix.
- ggcoefstats function to visualize results from regression analyses.
- combine_plots helper function to combine multiple {ggstatsplot} plots using patchwork::wrap_plots().

For more documentation, see the dedicated Website.

Author(s)

Maintainer: Indrajeet Patil <patilindrajeet.science@gmail.com> (ORCID) (@patilindrajeets) [copyright holder]

See Also

Useful links:

- https://indrajeetpatil.github.io/ggstatsplot/
- https://github.com/IndrajeetPatil/ggstatsplot
- Report bugs at https://github.com/IndrajeetPatil/ggstatsplot/issues

---

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

Description

Tidy version of the "Bugs" dataset.

Usage

```
bugs_long
```

Format

A data frame with 372 rows and 6 variables

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

Details

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

Source

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

Examples

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

bugs_wide *Wide-format version of the "Bugs" dataset.*

Description

Wide-format version of the "Bugs" dataset.

Usage

```
bugs_wide
```

Format

A data frame with 93 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.
- ldlf,ldhf,hdlf,hdhf.The desire to kill an arthropod was indicated on a scale from 0 to 10 in each condition of the experiment (**LDLF**: low freighteningness and low disgustingness; **LFHD**: low freighteningness and high disgustingness; **HFHD**: high freighteningness and low disgustingness; **HFHD**: high freighteningness and high disgustingness).

Details

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

Source

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

Examples

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

`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

Combined plot with annotation labels

Examples

```
# loading the necessary libraries
library(ggplot2)

# preparing the first plot
p1 <- ggplot(
```

```
  data = subset(iris, iris$Species == "setosa"),
  aes(x = Sepal.Length, y = Sepal.Width)
) +
  geom_point() +
  labs(title = "setosa")

# preparing the 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 dataframes with statistical details from* `{ggstatsplot}`

Description

Extracting dataframes with statistical details from `{ggstatsplot}`

Usage

```
extract_stats(p, ...)
```

Arguments

`p`	A plot from `{ggstatsplot}` package
`...`	Ignored

Details

This is a convenience function to extract dataframes with statistical details that are used to create expressions displayed in {ggstatsplot} plots as subtitle and/or as caption. Note that all of this analysis is carried out by the {statsExpressions} package.

For more details about underlying tests and effect size estimates, see the following vignette: https://indrajeetpatil.github.io/sta

Value

A list of tibbles containing statistical analysis summaries.

Examples

```
if (require("PMCMRplus")) {
  set.seed(123)
  library(ggstatsplot)

  # in case of group comparisons
  p <- ggbetweenstats(mtcars, cyl, mpg)
  extract_stats(p)

  # the exact details depend on the function
  extract_stats(ggbarstats(mtcars, cyl, am))
}
```

---

ggbarstats *Bar (column) 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,
  counts = NULL,
  type = "parametric",
  paired = FALSE,
  results.subtitle = TRUE,
  label = "percentage",
  label.args = list(alpha = 1, fill = "white"),
  k = 2L,
  proportion.test = results.subtitle,
  perc.k = 0L,
```

```
  bf.message = TRUE,
  ratio = NULL,
  conf.level = 0.95,
  sampling.plan = "indepMulti",
  fixed.margin = "rows",
  prior.concentration = 1,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  legend.title = NULL,
  xlab = NULL,
  ylab = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  package = "RColorBrewer",
  palette = "Dark2",
  ggplot.component = NULL,
  output = "plot",
  ...
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`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. This argument can not be `NULL` for `ggbarstats` function.
`counts`	A string naming a 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`). If `TRUE`, McNemar's test expression will be returned. If `FALSE`, Pearson's chi-square test will be returned.
`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 geom_label.
k	Number of digits after decimal point (should be an integer) (Default: k = 2L).
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.
perc.k	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. This means if there are two levels this will be ratio = c(0.5,0.5) or if there are four levels this will be ratio = c(0.25,0.25,0.25,0.25), etc.
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% confidence/credible intervals (0.95).
sampling.plan	Character describing the sampling plan. Possible options are "indepMulti" (independent multinomial; default), "poisson", "jointMulti" (joint multinomial), "hypergeom" (hypergeometric). For more, see ?BayesFactor::contingencyTableBF().
fixed.margin	For the independent multinomial sampling plan, which margin is fixed ("rows" or "cols"). Defaults to "rows".
prior.concentration	Specifies the prior concentration parameter, set to 1 by default. It indexes the expected deviation from the null hypothesis under the alternative, and corresponds to Gunel and Dickey's (1974) "a" parameter.
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.
legend.title	Title text for the legend.
xlab	Custom text for the x axis label (Default: NULL, which will cause the x axis label to be the x variable).
ylab	Custom text for the y axis label (Default: NULL).
ggtheme	A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.).
package	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).
palette	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).

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.

output
: Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.

...
: Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggpiestats.html

See Also

grouped_ggbarstats, ggpiestats, grouped_ggpiestats

Examples

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

# association test (or contingency table analysis)
ggbarstats(mtcars, x = vs, y = cyl)
```

---

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,
  plot.type = "boxviolin",
```

```
  type = "parametric",
  pairwise.comparisons = TRUE,
  pairwise.display = "significant",
  p.adjust.method = "holm",
  effsize.type = "unbiased",
  bf.prior = 0.707,
  bf.message = TRUE,
  results.subtitle = TRUE,
  xlab = NULL,
  ylab = NULL,
  caption = NULL,
  title = NULL,
  subtitle = NULL,
  k = 2L,
  var.equal = FALSE,
  conf.level = 0.95,
  nboot = 100L,
  tr = 0.2,
  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),
  outlier.tagging = FALSE,
  outlier.label = NULL,
  outlier.coef = 1.5,
  outlier.shape = 19,
  outlier.color = "black",
  outlier.label.args = list(size = 3),
 point.args = list(position = ggplot2::position_jitterdodge(dodge.width = 0.6), alpha
    = 0.4, size = 3, stroke = 0),
  violin.args = list(width = 0.5, alpha = 0.2),
  ggsignif.args = list(textsize = 3, tip_length = 0.01),
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  package = "RColorBrewer",
  palette = "Dark2",
  ggplot.component = NULL,
  output = "plot",
  ...
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`x`	The grouping (or independent) variable from the dataframe `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 al-

ready 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 the dataframe `data`.

`plot.type` Character describing the *type* of plot. Currently supported plots are `"box"` (for only boxplots), `"violin"` (for only violin plots), and `"boxviolin"` (for a combination of box and violin plots; default).

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

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

You can specify just the initial letter.

`pairwise.comparisons`

Logical that decides whether pairwise comparisons are to be displayed (default: `TRUE`). Please note that only **significant** comparisons will be shown by default. To change this behavior, select appropriate option with `pairwise.display` argument. The pairwise comparison dataframes are prepared using the `pairwise_comparisons` function. For more details about pairwise comparisons, see the documentation for that function.

`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.

`p.adjust.method`

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

`effsize.type` Type of effect size needed for *parametric* tests. The argument can be `"eta"` (partial eta-squared) or `"omega"` (partial omega-squared).

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors.

`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, ylab` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

`caption` The text for the plot caption.

`title` The text for the plot title.

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

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

`var.equal` a logical variable indicating whether to treat the two variances as being equal. If `TRUE` then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.

`conf.level` Scalar between `0` and `1`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

`nboot` Number of bootstrap samples for computing confidence interval for the effect size (Default: `100L`).

`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.

`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:

- `"parameteric"` (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 `geom_point` and `ggrepel::geom_label_repel` geoms, which are involved in mean plotting.

`outlier.tagging`

Decides whether outliers should be tagged (Default: `FALSE`).

`outlier.label` Label to put on the outliers that have been tagged. This **can't** be the same as `x` argument.

`outlier.coef` Coefficient for outlier detection using Tukey's method. With Tukey's method, outliers are below (1st Quartile) or above (3rd Quartile) `outlier.coef` times the Inter-Quartile Range (IQR) (Default: `1.5`).

`outlier.shape` Hiding the outliers can be achieved by setting `outlier.shape = NA`. Importantly, this does not remove the outliers, it only hides them, so the range calculated for the y-axis will be the same with outliers shown and outliers hidden.

`outlier.color` Default aesthetics for outliers (Default: `"black"`).

`outlier.label.args`
A list of additional aesthetic arguments to be passed to `ggrepel::geom_label_repel` for outlier label plotting.

`point.args` A list of additional aesthetic arguments to be passed to the `geom_point` displaying the raw data.

`violin.args` A list of additional aesthetic arguments to be passed to the `geom_violin`.

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

`ggtheme` A {ggplot2} theme. Default value is `ggstatsplot::theme_ggstatsplot()`. Any of the {ggplot2} themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.).

`package, palette`
Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_names)`.

`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.

`output` Character that describes what is to be returned: can be `"plot"` (default) or `"subtitle"` or `"caption"`. Setting this to `"subtitle"` will return the expression containing statistical results. If you have set `results.subtitle = FALSE`, then this will return a `NULL`. Setting this to `"caption"` will return the expression containing details about Bayes Factor analysis, but valid only when `type = "parametric"` and `bf.message = TRUE`, otherwise this will return a `NULL`.

`...` Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggbetweenstats.html

See Also

grouped_ggbetweenstats, ggwithinstats, grouped_ggwithinstats

Examples

```
if (require("PMCMRplus")) {
  # to get reproducible results from bootstrapping
  set.seed(123)
  library(ggstatsplot)
```

```
  # simple function call with the defaults
  ggbetweenstats(mtcars, am, mpg)

  # more detailed function call
  ggbetweenstats(
    data = morley,
    x = Expt,
    y = Speed,
    type = "robust",
    xlab = "The experiment number",
    ylab = "Speed-of-light measurement",
    pairwise.comparisons = TRUE,
    p.adjust.method = "fdr",
    outlier.tagging = TRUE,
    outlier.label = Run
  )
}
```

---

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.

Usage

```
ggcoefstats(
  x,
  output = "plot",
  statistic = NULL,
  conf.int = TRUE,
  conf.level = 0.95,
  k = 2L,
  exclude.intercept = FALSE,
  effsize = "eta",
  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"),
  errorbar.args = list(height = 0),
  vline = TRUE,
  vline.args = list(size = 1, linetype = "dashed"),
  stats.labels = TRUE,
  stats.label.color = NULL,
  stats.label.args = list(size = 3, direction = "y", min.segment.length = 0),
  package = "RColorBrewer",
  palette = "Dark2",
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  ...
)
```

Arguments

x	A model object to be tidied, or a tidy data frame containing results from a regression model. Function internally uses `parameters::model_parameters()` to get a tidy dataframe. If a dataframe is entered, it *must* contain at the minimum two columns named `term` (names of predictors) and `estimate` (corresponding estimates of coefficients or other quantities of interest).
output	Character describing the expected output from this function: `"plot"` (visualization of regression coefficients) or `"tidy"` (tidy dataframe of results `parameters::model_parameters`) or `"glance"` (object from `performance::model_performance`).
statistic	Which statistic is to be displayed (either `"t"` or `"f"`or `"z"` or `"chi"`) in the label. This is relevant if the `x` argument is a *dataframe*.
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`).
k	Number of digits after decimal point (should be an integer) (Default: `k = 2L`).
exclude.intercept	Logical that decides whether the intercept should be excluded from the plot (Default: `FALSE`).
effsize	Character describing the effect size to be displayed: `"eta"` (default) or `"omega"`. This argument is relevant only for models objects with *F*-statistic.
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` function will be used. If `"robust"`, `metaplus::metaplus` function will be used. If `"bayes"`, `metaBMA::meta_random` function 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	Labels for x and y axis variables. If NULL (default), variable names for x and y will be used.
ylab	Labels for x and y axis variables. If NULL (default), variable names for x and 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.
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. Relevant only when the output is a plot.
point.args	Additional arguments that will be passed to geom_point geom. Please see documentation for that function to know more about these arguments.
errorbar.args	Additional arguments that will be passed to geom_errorbarh 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 geom. Please see documentation for that function to know more about these arguments.
package	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).
palette	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).
ggtheme	A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.).
...	Additional arguments to tidying method. For more, see parameters::model_parameters.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggcoefstats.html

Note

1. In case you want to carry out meta-analysis, you will be asked to install the needed packages (`{metafor}`, `{metaplus}`, or `{metaBMA}`) for meta-analysis (if 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 the GitHub versions of `{parameters}` and `{performance}` in order to make most of this function.

Examples

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

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

# to get a plot
ggcoefstats(mod, output = "plot")

# to get a tidy dataframe
ggcoefstats(mod, output = "tidy")

# to get a glance summary
ggcoefstats(mod, output = "glance")
```

---

ggcorrmat *Visualization of a correlation matrix*

Description

Correlation matrix or a dataframe containing results from pairwise correlation tests. The package internally uses `ggcorrplot::ggcorrplot` for creating the visualization matrix, while the correlation analysis is carried out using the `correlation::correlation` function.

Usage

```
ggcorrmat(
  data,
  cor.vars = NULL,
  cor.vars.names = NULL,
  output = "plot",
  matrix.type = "upper",
  type = "parametric",
```

```
  tr = 0.2,
  partial = FALSE,
  k = 2L,
  sig.level = 0.05,
  conf.level = 0.95,
  bf.prior = 0.707,
  p.adjust.method = "holm",
  pch = "cross",
 ggcorrplot.args = list(method = "square", outline.color = "black", pch.cex = 14),
  package = "RColorBrewer",
  palette = "Dark2",
  colors = c("#E69F00", "white", "#009E73"),
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  ggplot.component = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  ...
)
```

Arguments

`data`	Dataframe from which variables specified are preferentially 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.
`output`	Character that decides expected output from this function. If `"plot"`, the visualization matrix will be returned. If `"dataframe"` (or literally anything other than `"plot"`), a dataframe containing all details from statistical analyses (e.g., correlation coefficients, statistic values, *p*-values, no. of observations, etc.) will be returned.
`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.

k	Number of digits after decimal point (should be an integer) (Default: k = 2L).
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. Relevant only when output = "plot".
conf.level	Scalar between 0 and 1. If unspecified, the defaults return 95% confidence/credible intervals (0.95).
bf.prior	A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.
p.adjust.method	Adjustment method for *p*-values for multiple comparisons. Possible methods are: "holm" (default), "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none".
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.
package	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).
palette	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).
colors	A vector of 3 colors for low, mid, and high correlation values. If set to NULL, manual specification of colors will be turned off and 3 colors from the specified palette from package will be selected.
ggtheme	A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.).
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.
...	Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggcorrmat.html

See Also

grouped_ggcorrmat ggscatterstats grouped_ggscatterstats

Examples

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

# to get a plot (assumes that `ggcorrplot` is installed)
if (require("ggcorrplot")) ggcorrmat(iris)

# to get a dataframe
ggcorrmat(
  data = ggplot2::msleep,
  cor.vars = sleep_total:bodywt,
  partial = TRUE,
  output = "dataframe"
)
```

---

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 details.

Usage

```
ggdotplotstats(
  data,
  x,
  y,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  type = "parametric",
  test.value = 0,
  bf.prior = 0.707,
  bf.message = TRUE,
```

```
  effsize.type = "g",
  conf.level = 0.95,
  tr = 0.2,
  k = 2L,
  results.subtitle = TRUE,
  point.args = list(color = "black", size = 3, shape = 16),
  centrality.plotting = TRUE,
  centrality.type = type,
  centrality.line.args = list(color = "blue", size = 1, linetype = "dashed"),
  ggplot.component = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  output = "plot",
  ...
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`x`	A numeric variable from the dataframe `data`.
`y`	Label or grouping variable.
`xlab`	Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.
`ylab`	Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `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.
`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`).
`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`).
`effsize.type`	Type of effect size needed for *parametric* tests. The argument can be `"d"` (for Cohen's *d*) or `"g"` (for Hedge's *g*).

`conf.level` Scalar between `0` and `1`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

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

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

`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 passed to `geom_point`.

`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:

- `"parameteric"` (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 `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 `ggstatsplot::theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.).

`output` Character that describes what is to be returned: can be `"plot"` (default) or `"subtitle"` or `"caption"`. Setting this to `"subtitle"` will return the expression containing statistical results. If you have set `results.subtitle = FALSE`,

then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.

... Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggdotplotstats.html

See Also

grouped_gghistostats, gghistostats, grouped_ggdotplotstats

Examples

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

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

---

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,
  bf.prior = 0.707,
  bf.message = TRUE,
  effsize.type = "g",
  conf.level = 0.95,
  tr = 0.2,
  k = 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", size = 1, linetype = "dashed"),
  normal.curve = FALSE,
  normal.curve.args = list(size = 2),
  ggplot.component = NULL,
  output = "plot",
  ...
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`x`	A numeric variable from the dataframe `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`	Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `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.
`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`).
`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`).

`effsize.type` Type of effect size needed for *parametric* tests. The argument can be `"d"` (for Cohen's *d*) or `"g"` (for Hedge's *g*).

`conf.level` Scalar between `0` and `1`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

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

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

`ggtheme` A `{ggplot2}` theme. Default value is `ggstatsplot::theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.).

`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:

- `"parameteric"` (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 `geom_line` used to display the lines corresponding to the centrality parameter.

`normal.curve` A logical value that decides whether to super-impose a normal curve using `stats::dnorm(mean(x),sd(x))`. Default is `FALSE`.

normal.curve.args
: A list of additional aesthetic arguments to be passed to the normal curve.

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.

output
: Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.

...
: Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/gghistostats.html

See Also

grouped_gghistostats, ggdotplotstats, grouped_ggdotplotstats

Examples

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

# using defaults, but modifying which centrality parameter is to be shown
gghistostats(
  data            = ToothGrowth,
  x               = len,
  xlab            = "Tooth length",
  centrality.type = "np"
)
```

---

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,
  k = 2L,
  proportion.test = results.subtitle,
  perc.k = 0L,
  bf.message = TRUE,
  ratio = NULL,
  conf.level = 0.95,
  sampling.plan = "indepMulti",
  fixed.margin = "rows",
  prior.concentration = 1,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  legend.title = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  package = "RColorBrewer",
  palette = "Dark2",
  ggplot.component = NULL,
  output = "plot",
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
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. This argument can not be NULL for ggbarstats function.
counts	A string naming a 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). If TRUE, McNemar's test expression will be returned. If FALSE, Pearson's chi-square test will be returned.

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 geom_label.

label.repel — Whether labels should be repelled using ggrepel package. This can be helpful in case the labels are overlapping.

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

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.

perc.k — 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. This means if there are two levels this will be ratio = c(0.5,0.5) or if there are four levels this will be ratio = c(0.25,0.25,0.25,0.25), etc.

conf.level — Scalar between 0 and 1. If unspecified, the defaults return 95% confidence/credible intervals (0.95).

sampling.plan — Character describing the sampling plan. Possible options are "indepMulti" (independent multinomial; default), "poisson", "jointMulti" (joint multinomial), "hypergeom" (hypergeometric). For more, see ?BayesFactor::contingencyTableBF().

fixed.margin — For the independent multinomial sampling plan, which margin is fixed ("rows" or "cols"). Defaults to "rows".

prior.concentration — Specifies the prior concentration parameter, set to 1 by default. It indexes the expected deviation from the null hypothesis under the alternative, and corresponds to Gunel and Dickey's (1974) "a" parameter.

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.
legend.title	Title text for the legend.
ggtheme	A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps() etc.).
package	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).
palette	Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_names).
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.
output	Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.
...	Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggpiestats.html

See Also

grouped_ggpiestats, ggbarstats, grouped_ggbarstats

Examples

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

# one sample goodness of fit proportion test
ggpiestats(mtcars, x = vs)

# association test (or contingency table analysis)
ggpiestats(mtcars, x = vs, y = cyl)
```

ggscatterstats *Scatterplot with marginal distributions and statistical results*

Description

Scatterplots from {ggplot2} combined with marginal densigram (density + histogram) 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,
  k = 2L,
  results.subtitle = TRUE,
  label.var = NULL,
  label.expression = NULL,
  marginal = TRUE,
  xfill = "#009E73",
  yfill = "#D55E00",
  point.args = list(size = 3, alpha = 0.4, stroke = 0, na.rm = TRUE),
  point.width.jitter = 0,
  point.height.jitter = 0,
  point.label.args = list(size = 3, max.overlaps = 1e+06),
 smooth.line.args = list(size = 1.5, color = "blue", method = "lm", formula = y ~ x,
    na.rm = TRUE),
  xsidehistogram.args = list(fill = xfill, color = "black", na.rm = TRUE),
  ysidehistogram.args = list(fill = yfill, color = "black", na.rm = TRUE),
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  ggplot.component = NULL,
  output = "plot",
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
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. If unspecified, the defaults return 95% confidence/credible intervals (0.95).
bf.prior	A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors and posterior estimates. In addition to numeric arguments, several named values are also recognized: "medium", "wide", and "ultrawide", corresponding to *r* scale values of 1/2, sqrt(2)/2, and 1, respectively. In case of an ANOVA, this value corresponds to scale for fixed effects.
bf.message	Logical that decides whether to display 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.
k	Number of digits after decimal point (should be an integer) (Default: k = 2L).
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.
xfill, yfill	Character describing color fill for x and y axes marginal distributions (default: "#009E73" (for x) and "#D55E00" (for y)). Note that the defaults are colorblind-friendly.
point.args	A list of additional aesthetic arguments to be passed to geom_point geom used to display the raw data points.

`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 `ggside` package to change the marginal distribution histograms plots.

`xlab`
: Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

`ylab`
: Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `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.

`ggtheme`
: A `{ggplot2}` theme. Default value is `ggstatsplot::theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.).

`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.

`output`
: Character that describes what is to be returned: can be `"plot"` (default) or `"subtitle"` or `"caption"`. Setting this to `"subtitle"` will return the expression containing statistical results. If you have set `results.subtitle = FALSE`, then this will return a `NULL`. Setting this to `"caption"` will return the expression containing details about Bayes Factor analysis, but valid only when `type = "parametric"` and `bf.message = TRUE`, otherwise this will return a `NULL`.

`...`
: Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggscatterstats.html

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

```
# to get reproducible results from bootstrapping
set.seed(123)
library(ggstatsplot)
library(dplyr, warn.conflicts = FALSE)

# creating dataframe with rownames converted to a new column
mtcars_new <- as_tibble(mtcars, rownames = "car")

# simple function call with the defaults
if (require("ggside")) {
  ggscatterstats(
    data = mtcars_new,
    x = wt,
    y = mpg,
    label.var = car,
    label.expression = wt < 4 & mpg < 20
  ) + # making further customization with `{ggplot2}` functions
    geom_rug(sides = "b")
}
```

---

ggwithinstats *Box/Violin plots for within-subjects (or 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",
  pairwise.comparisons = TRUE,
  pairwise.display = "significant",
  p.adjust.method = "holm",
  effsize.type = "unbiased",
  bf.prior = 0.707,
  bf.message = TRUE,
  results.subtitle = TRUE,
```

```
  xlab = NULL,
  ylab = NULL,
  caption = NULL,
  title = NULL,
  subtitle = NULL,
  k = 2L,
  conf.level = 0.95,
  nboot = 100L,
  tr = 0.2,
  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(size = 1, color = "red", alpha = 0.5),
  point.args = list(size = 3, alpha = 0.5),
  point.path = TRUE,
  point.path.args = list(alpha = 0.5, linetype = "dashed"),
  outlier.tagging = FALSE,
  outlier.label = NULL,
  outlier.coef = 1.5,
  outlier.label.args = list(size = 3),
  boxplot.args = list(width = 0.2, alpha = 0.5),
  violin.args = list(width = 0.5, alpha = 0.2),
  ggsignif.args = list(textsize = 3, tip_length = 0.01),
  ggtheme = ggstatsplot::theme_ggstatsplot(),
  package = "RColorBrewer",
  palette = "Dark2",
  ggplot.component = NULL,
  output = "plot",
  ...
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`x`	The grouping (or independent) variable from the dataframe `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 the dataframe `data`.
`type`	A character specifying the type of statistical approach: • `"parametric"` • `"nonparametric"`

- "robust"
- "bayes"

You can specify just the initial letter.

pairwise.comparisons
: Logical that decides whether pairwise comparisons are to be displayed (default: TRUE). Please note that only **significant** comparisons will be shown by default. To change this behavior, select appropriate option with pairwise.display argument. The pairwise comparison dataframes are prepared using the pairwise_comparisons function. For more details about pairwise comparisons, see the documentation for that function.

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.

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

effsize.type
: Type of effect size needed for *parametric* tests. The argument can be "eta" (partial eta-squared) or "omega" (partial omega-squared).

bf.prior
: A number between 0.5 and 2 (default 0.707), the prior width to use in calculating Bayes factors.

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
: Labels for x and y axis variables. If NULL (default), variable names for x and y will be used.

ylab
: Labels for x and y axis variables. If NULL (default), variable names for x and y will be used.

caption
: The text for the plot caption.

title
: The text for the plot title.

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

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

conf.level
: Scalar between 0 and 1. If unspecified, the defaults return 95% confidence/credible intervals (0.95).

nboot
: Number of bootstrap samples for computing confidence interval for the effect size (Default: 100L).

`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.

`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:

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

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

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

`centrality.label.args`
A list of additional aesthetic arguments to be passed to `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 `geom_path` connecting raw data points and mean points.

`point.args` A list of additional aesthetic arguments to be passed to the `geom_point` displaying the raw data.

`point.path, centrality.path`
Logical that decides whether individual data points and means, respectively, should be connected using `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.

`outlier.tagging`
Decides whether outliers should be tagged (Default: `FALSE`).

`outlier.label` Label to put on the outliers that have been tagged. This **can't** be the same as `x` argument.

`outlier.coef` Coefficient for outlier detection using Tukey's method. With Tukey's method, outliers are below (1st Quartile) or above (3rd Quartile) `outlier.coef` times the Inter-Quartile Range (IQR) (Default: `1.5`).

outlier.label.args
: A list of additional aesthetic arguments to be passed to `ggrepel::geom_label_repel` for outlier label plotting.

boxplot.args
: A list of additional aesthetic arguments passed on to `geom_boxplot`.

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

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

ggtheme
: A {ggplot2} theme. Default value is `ggstatsplot::theme_ggstatsplot()`. Any of the {ggplot2} themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()` etc.).

package
: Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_names)`.

palette
: Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_names)`.

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.

output
: Character that describes what is to be returned: can be `"plot"` (default) or `"subtitle"` or `"caption"`. Setting this to `"subtitle"` will return the expression containing statistical results. If you have set `results.subtitle = FALSE`, then this will return a `NULL`. Setting this to `"caption"` will return the expression containing details about Bayes Factor analysis, but valid only when `type = "parametric"` and `bf.message = TRUE`, otherwise this will return a `NULL`.

...
: Currently ignored.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggwithinstats.html

See Also

grouped_ggbetweenstats, ggbetweenstats, grouped_ggwithinstats

Examples

```
if (require("PMCMRplus")) {
  # setup
  set.seed(123)
  library(ggstatsplot)
  library(dplyr, warn.conflicts = FALSE)

  # two groups (*t*-test)
  ggwithinstats(
```

```
  data = filter(bugs_long, condition %in% c("HDHF", "HDLF")),
  x    = condition,
  y    = desire
)

# more than two groups (anova)
library(WRS2)

ggwithinstats(
  data            = WineTasting,
  x               = Wine,
  y               = Taste,
  type            = "r",
  outlier.tagging = TRUE,
  outlier.label   = Taster
)
}
```

`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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`...`	Arguments passed on to `ggbarstats` `xlab` Custom text for the `x` axis label (Default: `NULL`, which will cause the `x` axis label to be the `x` variable). `ylab` Custom text for the `y` axis label (Default: `NULL`). `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. This argument can not be NULL for ggbarstats function.

counts A string naming a 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). If TRUE, McNemar's test expression will be returned. If FALSE, Pearson's chi-square test will be returned.

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 geom_label.

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

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.

perc.k 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. This means if there are two levels this will be ratio = c(0.5,0.5) or if there are four levels this will be ratio = c(0.25,0.25,0.25,0.25), etc.

conf.level Scalar between 0 and 1. If unspecified, the defaults return 95% confidence/credible intervals (0.95).

sampling.plan Character describing the sampling plan. Possible options are "indepMulti" (independent multinomial; default), "poisson", "jointMulti" (joint multinomial), "hypergeom" (hypergeometric). For more, see ?BayesFactor::contingencyTa

fixed.margin For the independent multinomial sampling plan, which margin is fixed ("rows" or "cols"). Defaults to "rows".

prior.concentration Specifies the prior concentration parameter, set to 1 by default. It indexes the expected deviation from the null hypothesis under the alternative, and corresponds to Gunel and Dickey's (1974) "a" parameter.

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

caption The text for the plot caption.

legend.title Title text for the legend.

ggtheme A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps(), etc.).

package Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_nam

palette Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_nam

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.

grouping.var A single grouping variable.

output Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.

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://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggpiestats.html

See Also

ggbarstats, ggpiestats, grouped_ggpiestats

Examples

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

```
# let's create a smaller dataframe
diamonds_short <- ggplot2::diamonds %>%
  filter(cut %in% c("Very Good", "Ideal")) %>%
  filter(clarity %in% c("SI1", "SI2", "VS1", "VS2")) %>%
  sample_frac(size = 0.05)

# plot
grouped_ggbarstats(
  data          = diamonds_short,
  x             = color,
  y             = clarity,
  grouping.var  = cut,
  plotgrid.args = list(nrow = 2)
)
```

`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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`...`	Arguments passed on to `ggbetweenstats`
	`plot.type` Character describing the *type* of plot. Currently supported plots are `"box"` (for only boxplots), `"violin"` (for only violin plots), and `"boxviolin"` (for a combination of box and violin plots; default).
	`xlab` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

`ylab` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

`pairwise.comparisons` Logical that decides whether pairwise comparisons are to be displayed (default: `TRUE`). Please note that only **significant** comparisons will be shown by default. To change this behavior, select appropriate option with `pairwise.display` argument. The pairwise comparison dataframes are prepared using the `pairwise_comparisons` function. For more details about pairwise comparisons, see the documentation for that function.

`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.

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors.

`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.

`outlier.color` Default aesthetics for outliers (Default: `"black"`).

`outlier.tagging` Decides whether outliers should be tagged (Default: `FALSE`).

`outlier.label` Label to put on the outliers that have been tagged. This **can't** be the same as `x` argument.

`outlier.shape` Hiding the outliers can be achieved by setting `outlier.shape = NA`. Importantly, this does not remove the outliers, it only hides them, so the range calculated for the y-axis will be the same with outliers shown and outliers hidden.

`outlier.label.args` A list of additional aesthetic arguments to be passed to `ggrepel::geom_label_repel` for outlier label plotting.

`outlier.coef` Coefficient for outlier detection using Tukey's method. With Tukey's method, outliers are below (1st Quartile) or above (3rd Quartile) `outlier.coef` times the Inter-Quartile Range (IQR) (Default: `1.5`).

`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:

- `"parameteric"` (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 `geom_point` displaying the raw data.

`violin.args` A list of additional aesthetic arguments to be passed to the `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.

`package` Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_nam`

`palette` Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_nam`

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

`centrality.label.args` A list of additional aesthetic arguments to be passed to `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 `ggstatsplot::theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.).

`x` The grouping (or independent) variable from the dataframe `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 the dataframe `data`.

type A character specifying the type of statistical approach:

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

You can specify just the initial letter.

effsize.type Type of effect size needed for *parametric* tests. The argument can be "eta" (partial eta-squared) or "omega" (partial omega-squared).

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

var.equal a logical variable indicating whether to treat the two variances as being equal. If TRUE then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.

conf.level Scalar between 0 and 1. If unspecified, the defaults return 95% confidence/credible intervals (0.95).

nboot Number of bootstrap samples for computing confidence interval for the effect size (Default: 100L).

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.

grouping.var A single grouping variable.

output Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.

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

```
if (require("PMCMRplus")) {
  # to get reproducible results from bootstrapping
  set.seed(123)
  library(ggstatsplot)
  library(dplyr, warn.conflicts = FALSE)
  library(ggplot2)

  # the most basic function call
```

```
  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))
    )
  )
}
```

`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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	Dataframe from which variables specified are preferentially 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. Relevant only when `output = "plot"`.

`colors` A vector of 3 colors for low, mid, and high correlation values. If set to `NULL`, manual specification of colors will be turned off and 3 colors from the specified `palette` from `package` will be selected.

`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`.

`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.

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

`conf.level` Scalar between `0` and `1`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

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

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

package Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_nam

palette Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running View(paletteer::palettes_d_nam

ggtheme A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps(), etc.).

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.

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

caption The text for the plot caption.

grouping.var A single grouping variable.

output Character that decides expected output from this function. If "plot", the visualization matrix will be returned. If "dataframe" (or literally anything other than "plot"), a dataframe containing all details from statistical analyses (e.g., correlation coefficients, statistic values, *p*-values, no. of observations, etc.) will be returned.

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://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggcorrmat.html

See Also

ggcorrmat, ggscatterstats, grouped_ggscatterstats

Examples

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

# for plot
if (require("ggcorrplot")) {
  grouped_ggcorrmat(
    data = iris,
    grouping.var = Species,
    type = "robust",
```

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

# for dataframe
grouped_ggcorrmat(
  data = ggplot2::msleep,
  grouping.var = vore,
  type = "bayes",
  output = "dataframe"
)
```

`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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`...`	Arguments passed on to `ggdotplotstats` `y` Label or grouping variable. `point.args` A list of additional aesthetic arguments passed to `geom_point`. `x` A numeric variable from the dataframe `data`. `xlab` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used. `subtitle` The text for the plot subtitle. Will work only if `results.subtitle = FALSE`.

`caption` The text for the plot caption.

`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`).

`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`).

`effsize.type` Type of effect size needed for *parametric* tests. The argument can be `"d"` (for Cohen's *d*) or `"g"` (for Hedge's *g*).

`conf.level` Scalar between `0` and `1`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

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

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

`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.

`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:

- `"parameteric"` (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 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 ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps(), etc.).

ylab Labels for x and y axis variables. If NULL (default), variable names for x and y will be used.

grouping.var A single grouping variable.

output Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.

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://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggdotplotstats.html

See Also

grouped_gghistostats, ggdotplotstats, gghistostats

Examples

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

---

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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list(),
  ...
)
```

Arguments

data	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
x	A numeric variable from the dataframe 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.
output	Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.
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

normal.curve
: A logical value that decides whether to super-impose a normal curve using stats::dnorm(mean(x),sd(x)). Default is FALSE.

normal.curve.args
: A list of additional aesthetic arguments to be passed to the normal curve.

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 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).

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.

effsize.type
: Type of effect size needed for *parametric* tests. The argument can be "d" (for Cohen's *d*) or "g" (for Hedge's *g*).

conf.level
: Scalar between 0 and 1. If unspecified, the defaults return 95% confidence/credible intervals (0.95).

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

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

xlab
: Labels for x and y axis variables. If NULL (default), variable names for x and y will be used.

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

caption
: The text for the plot caption.

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).

ggtheme
: A {ggplot2} theme. Default value is ggstatsplot::theme_ggstatsplot(). Any of the {ggplot2} themes (e.g., theme_bw()), or themes from extension packages are allowed (e.g., ggthemes::theme_fivethirtyeight(), hrbrthemes::theme_ipsum_ps(), etc.).

`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.

`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:

- `"parameteric"` (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.

Details

For details, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/gghistostats.html

See Also

gghistostats, ggdotplotstats, grouped_ggdotplotstats

Examples

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

# 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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`...`	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. This argument can not be `NULL` for `ggbarstats` function. `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. `perc.k` 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 `geom_label`. `label.repel` Whether labels should be repelled using `ggrepel` package. This can be helpful in case the labels are overlapping.

`legend.title` Title text for the legend.

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

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

You can specify just the initial letter.

`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.

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

`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`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

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

`caption` The text for the plot caption.

`ggtheme` A `{ggplot2}` theme. Default value is `ggstatsplot::theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.).

`package` Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_nam`

`palette` Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_nam`

`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.

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

`paired` Logical indicating whether data came from a within-subjects or repeated measures design study (Default: `FALSE`). If `TRUE`, McNemar's test expression will be returned. If `FALSE`, Pearson's chi-square test will be returned.

`ratio` A vector of proportions: the expected proportions for the proportion test (should sum to 1). Default is `NULL`, which means the null is equal theoretical proportions across the levels of the nominal variable. This means if there are two levels this will be `ratio = c(0.5,0.5)` or if there are four levels this will be `ratio = c(0.25,0.25,0.25,0.25)`, etc.

`sampling.plan` Character describing the sampling plan. Possible options are `"indepMulti"` (independent multinomial; default), `"poisson"`, `"jointMulti"` (joint multinomial), `"hypergeom"` (hypergeometric). For more, see `?BayesFactor::contingencyTa`

`fixed.margin` For the independent multinomial sampling plan, which margin is fixed (`"rows"` or `"cols"`). Defaults to `"rows"`.

`prior.concentration` Specifies the prior concentration parameter, set to 1 by default. It indexes the expected deviation from the null hypothesis under the alternative, and corresponds to Gunel and Dickey's (1974) `"a"` parameter.

`grouping.var` A single grouping variable.

`output` Character that describes what is to be returned: can be `"plot"` (default) or `"subtitle"` or `"caption"`. Setting this to `"subtitle"` will return the expression containing statistical results. If you have set `results.subtitle = FALSE`, then this will return a `NULL`. Setting this to `"caption"` will return the expression containing details about Bayes Factor analysis, but valid only when `type = "parametric"` and `bf.message = TRUE`, otherwise this will return a `NULL`.

`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://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggpiestats.html

See Also

ggbarstats, ggpiestats, grouped_ggbarstats

Examples

```
set.seed(123)
library(ggstatsplot)

# grouped one-sample proportion test
grouped_ggpiestats(mtcars, x = cyl, grouping.var = am)
```

---

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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`...`	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. `point.args` A list of additional aesthetic arguments to be passed to `geom_point` geom used to display the raw data points. `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` 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.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. `xfill` Character describing color fill for `x` and `y` axes marginal distributions (default: `"#009E73"` (for `x`) and `"#D55E00"` (for `y`)). Note that the defaults are colorblind-friendly. `yfill` Character describing color fill for `x` and `y` axes marginal distributions (default: `"#009E73"` (for `x`) and `"#D55E00"` (for `y`)). Note that the defaults are colorblind-friendly. `xsidehistogram.args` A list of arguments passed to respective `geom_s` from `ggside` package to change the marginal distribution histograms plots.

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

`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`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

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

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

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

`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` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

`ylab` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

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

`caption` The text for the plot caption.

`ggtheme` A `{ggplot2}` theme. Default value is `ggstatsplot::theme_ggstatsplot()`. Any of the `{ggplot2}` themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.).

`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.

grouping.var	A single grouping variable.
output	Character that describes what is to be returned: can be "plot" (default) or "subtitle" or "caption". Setting this to "subtitle" will return the expression containing statistical results. If you have set results.subtitle = FALSE, then this will return a NULL. Setting this to "caption" will return the expression containing details about Bayes Factor analysis, but valid only when type = "parametric" and bf.message = TRUE, otherwise this will return a NULL.
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://indrajeetpatil.github.io/ggstatsplot/articles/web_only/ggscatterstats.html

See Also

ggscatterstats, ggcorrmat, grouped_ggcorrmat

Examples

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

# basic function call
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")
)
```

`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,
  output = "plot",
  plotgrid.args = list(),
  annotation.args = list()
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`...`	Arguments passed on to `ggwithinstats` `point.path` Logical that decides whether individual data points and means, respectively, should be connected using `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` Logical that decides whether individual data points and means, respectively, should be connected using `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` A list of additional aesthetic arguments passed on to `geom_path` connecting raw data points and mean points.

`point.path.args` A list of additional aesthetic arguments passed on to `geom_path` connecting raw data points and mean points.

`boxplot.args` A list of additional aesthetic arguments passed on to `geom_boxplot`.

`x` The grouping (or independent) variable from the dataframe `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 the dataframe `data`.

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

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

You can specify just the initial letter.

`pairwise.comparisons` Logical that decides whether pairwise comparisons are to be displayed (default: `TRUE`). Please note that only **significant** comparisons will be shown by default. To change this behavior, select appropriate option with `pairwise.display` argument. The pairwise comparison dataframes are prepared using the `pairwise_comparisons` function. For more details about pairwise comparisons, see the documentation for that function.

`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.

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

`effsize.type` Type of effect size needed for *parametric* tests. The argument can be `"eta"` (partial eta-squared) or `"omega"` (partial omega-squared).

`bf.prior` A number between `0.5` and `2` (default `0.707`), the prior width to use in calculating Bayes factors.

`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` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

`ylab` Labels for `x` and `y` axis variables. If `NULL` (default), variable names for `x` and `y` will be used.

`caption` The text for the plot caption.

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

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

`conf.level` Scalar between `0` and `1`. If unspecified, the defaults return 95% confidence/credible intervals (`0.95`).

`nboot` Number of bootstrap samples for computing confidence interval for the effect size (Default: `100L`).

`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.

`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:

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

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

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

`centrality.label.args` A list of additional aesthetic arguments to be passed to `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 `geom_point` displaying the raw data.

`outlier.tagging` Decides whether outliers should be tagged (Default: `FALSE`).

`outlier.label` Label to put on the outliers that have been tagged. This **can't** be the same as `x` argument.

`outlier.coef` Coefficient for outlier detection using Tukey's method. With Tukey's method, outliers are below (1st Quartile) or above (3rd Quartile) `outlier.coef` times the Inter-Quartile Range (IQR) (Default: `1.5`).

`outlier.label.args` A list of additional aesthetic arguments to be passed to `ggrepel::geom_label_repel` for outlier label plotting.

`violin.args` A list of additional aesthetic arguments to be passed to the `geom_violin`.

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

`ggtheme` A {ggplot2} theme. Default value is `ggstatsplot::theme_ggstatsplot()`. Any of the {ggplot2} themes (e.g., `theme_bw()`), or themes from extension packages are allowed (e.g., `ggthemes::theme_fivethirtyeight()`, `hrbrthemes::theme_ipsum_ps()`, etc.).

`package` Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_nam`

`palette` Name of the package from which the given palette is to be extracted. The available palettes and packages can be checked by running `View(paletteer::palettes_d_nam`

`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.

`grouping.var` A single grouping variable.

`output` Character that describes what is to be returned: can be `"plot"` (default) or `"subtitle"` or `"caption"`. Setting this to `"subtitle"` will return the expression containing statistical results. If you have set `results.subtitle = FALSE`, then this will return a `NULL`. Setting this to `"caption"` will return the expression containing details about Bayes Factor analysis, but valid only when `type = "parametric"` and `bf.message = TRUE`, otherwise this will return a `NULL`.

`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

```
if (require("PMCMRplus")) {
  # to get reproducible results from bootstrapping
  set.seed(123)
  library(ggstatsplot)
  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,
    grouping.var     = gender,
    type             = "np", # non-parametric test
    # additional modifications for **each** plot using `{ggplot2}` functions
    ggplot.component = scale_y_continuous(breaks = seq(0, 10, 1), limits = c(0, 10))
  )
}
```

`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, https://imdb.com/, is a website devoted to collecting movie data supplied by studios and fans. It claims to be the biggest movie database on the web and is run by amazon.

Source

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

Examples

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

`pairwise_comparisons` *Multiple pairwise comparison tests with tidy data*

Description

Calculate parametric, non-parametric, robust, and Bayes Factor pairwise comparisons between group levels with corrections for multiple testing.

Usage

```
pairwise_comparisons(
  data,
  x,
  y,
  subject.id = NULL,
  type = "parametric",
  paired = FALSE,
  var.equal = FALSE,
  tr = 0.2,
  bf.prior = 0.707,
  p.adjust.method = "holm",
  k = 2L,
  ...
)
```

Arguments

`data`	A dataframe (or a tibble) from which variables specified are to be taken. Other data types (e.g., matrix,table, array, etc.) will **not** be accepted.
`x`	The grouping (or independent) variable from the dataframe `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 the dataframe `data`.
`subject.id`	Relevant in case of a repeated measures or within-subjects design (`paired = TRUE`, i.e.), it specifies the subject or repeated measures identifier. **Important**: Note that if this argument is `NULL` (which is the default), 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 and you leave this argument unspecified, the results *can* be inaccurate when there are more than two levels in `x` and there are `NA`s present.
`type`	A character specifying the type of statistical approach:

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

You can specify just the initial letter.

paired — Logical that decides whether the experimental design is repeated measures/within-subjects or between-subjects. The default is FALSE.

var.equal — a logical variable indicating whether to treat the two variances as being equal. If TRUE then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used.

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".

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

... — Additional arguments passed to other methods.

Value

A tibble dataframe containing two columns corresponding to group levels being compared with each other (group1 and group2) and p.value column corresponding to this comparison. The dataframe will also contain a p.value.label column containing a *label* for this *p*-value, in case this needs to be displayed in ggsignif::geom_ggsignif. In addition to these common columns across the different types of statistics, there will be additional columns specific to the type of test being run.

This function provides a unified syntax to carry out pairwise comparison tests and internally relies on other packages to carry out these tests. For more details about the included tests, see the documentation for the respective functions:

- *parametric* : stats::pairwise.t.test() (paired) and PMCMRplus::gamesHowellTest() (unpaired)
- *non-parametric* : PMCMRplus::durbinAllPairsTest() (paired) and PMCMRplus::kwAllPairsDunnTest() (unpaired)
- *robust* : WRS2::rmmcp() (paired) and WRS2::lincon() (unpaired)
- *Bayes Factor* : BayesFactor::ttestBF()

References

For more, see: https://indrajeetpatil.github.io/ggstatsplot/articles/web_only/pairwise.html

Examples

```
if (require("PMCMRplus")) {
  # for reproducibility
  set.seed(123)
  library(ggstatsplot)
  library(statsExpressions) # for data

  # show all columns and make the column titles bold
  # as a user, you don't need to do this; this is just for the package website
 options(tibble.width = Inf, pillar.bold = TRUE, pillar.neg = TRUE, pillar.subtle_num = TRUE)

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

  # parametric
  # if `var.equal = TRUE`, then Student's t-test will be run
  pairwise_comparisons(
    data            = mtcars,
    x               = cyl,
    y               = wt,
    type            = "parametric",
    var.equal       = TRUE,
    paired          = FALSE,
    p.adjust.method = "none"
  )

  # if `var.equal = FALSE`, then Games-Howell test will be run
  pairwise_comparisons(
    data            = mtcars,
    x               = cyl,
    y               = wt,
    type            = "parametric",
    var.equal       = FALSE,
    paired          = FALSE,
    p.adjust.method = "bonferroni"
  )

  # non-parametric (Dunn test)
  pairwise_comparisons(
    data            = mtcars,
    x               = cyl,
    y               = wt,
    type            = "nonparametric",
    paired          = FALSE,
    p.adjust.method = "none"
  )

  # robust (Yuen's trimmed means *t*-test)
  pairwise_comparisons(
    data            = mtcars,
    x               = cyl,
    y               = wt,
```

```
  type            = "robust",
  paired          = FALSE,
  p.adjust.method = "fdr"
)

# Bayes Factor (Student's *t*-test)
pairwise_comparisons(
  data   = mtcars,
  x      = cyl,
  y      = wt,
  type   = "bayes",
  paired = FALSE
)

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

# parametric (Student's *t*-test)
pairwise_comparisons(
  data            = bugs_long,
  x               = condition,
  y               = desire,
  subject.id      = subject,
  type            = "parametric",
  paired          = TRUE,
  p.adjust.method = "BH"
)

# non-parametric (Durbin-Conover test)
pairwise_comparisons(
  data            = bugs_long,
  x               = condition,
  y               = desire,
  subject.id      = subject,
  type            = "nonparametric",
  paired          = TRUE,
  p.adjust.method = "BY"
)

# robust (Yuen's trimmed means t-test)
pairwise_comparisons(
  data            = bugs_long,
  x               = condition,
  y               = desire,
  subject.id      = subject,
  type            = "robust",
  paired          = TRUE,
  p.adjust.method = "hommel"
)

# Bayes Factor (Student's *t*-test)
pairwise_comparisons(
  data       = bugs_long,
  x          = condition,
```

```
    y          = desire,
    subject.id = subject,
    type       = "bayes",
    paired     = TRUE
  )
}
```

`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/reader. The theme is a wrapper around `theme_bw()`.

Usage

```
theme_ggstatsplot()
```

Value

A `ggplot` object with the `theme_ggstatsplot` theme overlaid.

Examples

```
library(ggplot2)
library(ggstatsplot)

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