Package ‘ggcompare’

January 31, 2026

Title Mean Comparison in 'ggplot2'

Version 0.0.6

Description Add mean comparison annotations to a 'ggplot'.
This package provides an easy way to indicate if two or more groups are significantly different in a 'ggplot'.
Usually you do not need to specify the test method, you only need to tell stat_compare() whether you want to perform a parametric test or a nonparametric test, and stat_compare() will automatically choose the appropriate test method based on your data.
For comparisons between two groups, the p-value is calculated by t-test (parametric) or Wilcoxon rank sum test (nonparametric). For comparisons among more than two groups, the p-value is calculated by One-way ANOVA (parametric) or Kruskal-Wallis test (nonparametric).

License Artistic-2.0

URL `https://hmu-wh.github.io/ggcompare/`,
`https://github.com/HMU-WH/ggcompare/`

BugReports `https://github.com/HMU-WH/ggcompare/issues/`

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.2.0)

Imports ggplot2

NeedsCompilation no

Author Hao Wang [aut, cre] (ORCID: <`https://orcid.org/0009-0000-9477-9585`>)

Maintainer Hao Wang <`wanghao8772@gmail.com`>

Repository CRAN

Date/Publication 2026-01-31 10:00:02 UTC

Contents

geom_bracket *Add Brackets with Labels to a ggplot*

Description

Add brackets with labeled annotations to a ggplot.

Usage

```
geom_bracket(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  arrow = NULL,
  parse = FALSE,
  bracket = TRUE,
  inherit.aes = TRUE
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).
`stat`	The statistical transformation to use on the data for this layer. When using a `geom_*()` function to construct a layer, the `stat` argument can be used the override the default coupling between geoms and stats. The `stat` argument accepts the following: • A `Stat` ggproto subclass, for example `StatCount`. • A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as `"count"`. • For more information and other ways to specify the stat, see the layer stat documentation.

`position`	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following: • The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`. • For more information and other ways to specify the position, see the layer position documentation.
`...`	additional arguments passed on to layer().
`arrow`	`arrow`, the arrow appear at both ends of the brackets, created by `grid::arrow()`.
`parse`	`logical`, whether to parse the labels as expressions and displayed as described in plotmath.
`bracket`	`logical`, whether to display the bracket. If `FALSE`, only the label will be displayed.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

Value

`LayerInstance`, a layer that can be added to a ggplot.

Aesthetics

- required: `xmin`, `xmax`, `ymin`, `ymax`, `label`
- optional: `angle`, `alpha`, `hjust`, `vjust`, `colour`, `family`, `fontsize`, `fontface`, `linetype`, `linewidth`, `lineheight`

Author(s)

HMU-WH

Examples

```
library(ggplot2)

ggplot(mpg, aes(class, displ)) +
  geom_boxplot() +
  annotate("bracket", xmin = 2, xmax = 4, ymin = 4.5, ymax = 5, label = "label",
           arrow = grid::arrow(type = "closed", length = unit(0.1, "inches")))
```

stat_compare — *Add Mean Comparison for Groups to a ggplot*

Description

Add group mean comparisons to a ggplot. The comparisons can be performed using the t-test, Wilcoxon rank-sum test, one-way ANOVA, or Kruskal-Wallis test.

Usage

```
stat_compare(
  mapping = NULL,
  data = NULL,
  position = "identity",
  ...,
  nudge = 0,
  start = NULL,
  breaks = NULL,
  labels = NULL,
  cutoff = NULL,
  method = NULL,
  overall = FALSE,
  ref_group = NULL,
  tip_length = 0.02,
  parametric = FALSE,
  correction = "none",
  panel_indep = FALSE,
  method_args = NULL,
  comparisons = NULL,
  step_increase = 0.1,
  inherit.aes = TRUE
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).

`position`	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following: • The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`. • For more information and other ways to specify the position, see the layer position documentation.
`...`	additional arguments passed on to `geom_bracket()`.
`nudge`	`numeric`, the nudge of start position in fraction of scale range.
`start`	`numeric`, the bracket start position. Defaults to the maximum value of `y`.
`breaks`	`numeric`, the breaks for p-value labels, like `c(0, 0.001, 0.01, 0.05, 1)`.
`labels`	`character`, the labels for p-value breaks, like `c("***", "**", "*", "ns")`.
`cutoff`	`numeric`, the cutoff for p-value, labels above this value will be removed.
`method`	`function`, the method for the test, it should support both the `x, y` interface for comparing two groups and the `formula` interface for comparing more than two groups. The function should return a `list` with components `p.value` (the test's p-value) and `method` (a character string of the test method name).
`overall`	`logical`, whether to compare each group (on the axis) against the combined mean of all other groups. Applicable only when `ref_group` and `comparisons` are both `NULL`.
`ref_group`	`character`, the reference group for comparison. other groups will be compared to this group.
`tip_length`	`numeric`, the length of the bracket tips in fraction of scale range.
`parametric`	`logical`, whether to use parametric test (t-test, One-way ANOVA) or non-parametric test (Wilcoxon rank sum test, Kruskal-Wallis test). Applicable only when `method` is NULL.
`correction`	`character`, the method for p-value adjustment, options include p.adjust.methods with "none" as the default.
`panel_indep`	`logical`, whether to correct the p-value only at the panel level. If `FALSE`, the p-value will be corrected at the layer level.
`method_args`	`list`, additional arguments to be passed to the test method.
`comparisons`	`list`, a list of comparisons to be made. Each element should contain two groups to be compared.
`step_increase`	`numeric`, the step increase in fraction of scale range for every additional comparison, in order to avoid overlapping brackets.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

Details

Usually you do not need to specify the test method, you only need to tell `stat_compare()` whether you want to perform a parametric test or a nonparametric test, and `stat_compare()` will automatically choose the appropriate test method based on your data. For comparisons between two groups, the p-value is calculated by t-test (parametric) or Wilcoxon rank sum test (nonparametric). For comparisons among more than two groups, the p-value is calculated by One-way ANOVA (parametric) or Kruskal-Wallis test (nonparametric).

Value

`LayerInstance`, a layer that can be added to a ggplot.

Aesthetics

- required: `x`, `y`

Computed variables

- `p`: p-value of the test.
- `q`: adjusted p-value of the test.
- `label`: the label of the bracket.
- `method`: the method name of the test.
- `xmin`, `xmax`, `ymin`, `ymax`: the position of the bracket.

Author(s)

HMU-WH

Examples

```
library(ggplot2)

p <- ggplot(mpg, aes(class, displ, color = class)) +
  geom_boxplot(show.legend = FALSE) +
  theme_test()

# Global comparison: Each x has only one group.
p + stat_compare()
# If you just want to display text, you can set parameters "bracket" to FALSE.
p + stat_compare(bracket = FALSE)
# If you want to display the test method, you can do this.
p + stat_compare(aes(label = after_stat(sprintf("%s: %s", method, label))))

# Comparison between each group and other combined groups.
p + stat_compare(overall = TRUE)

# Comparison between two groups: specify a reference group.
p + stat_compare(ref_group = "minivan")
# If you only want to display the p-value less or equal to 0.01, you can do this.
```

```
p + stat_compare(ref_group = "minivan", cutoff = 0.01)
# If you want to display the significance level, you can do this.
p + stat_compare(ref_group = "minivan", breaks = c(0, 0.001, 0.01, 0.05, 1))

# Comparison between two groups: specify the comparison group.
p + stat_compare(tip_length = 0.05,
                 step_increase = 0,
                 comparisons = list(c("compact", "midsize"), c("pickup", "suv")),
                 arrow = grid::arrow(type = "closed", length = unit(0.1, "inches")))
                 # Yeah, this supports adding arrows.

# Within-group (grouped by the x-axis) population comparison.
ggplot(mpg, aes(drv, displ, fill = class)) +
  geom_boxplot() +
  stat_compare() +
  stat_compare(aes(group = drv), nudge = 0.1, color = "gray") + # add global comparison
  theme_test()

# Better adaptation to faceting.
ggplot(mpg, aes(drv, displ)) +
  geom_boxplot() +
  stat_compare(comparisons = combn(unique(mpg$drv), 2, simplify = FALSE)) +
  facet_grid(cols = vars(class), scales = "free") +
  theme_test()

# P-value correction
p <- ggplot(mpg, aes(class, displ)) +
  geom_boxplot() +
  facet_grid(cols = vars(cyl), scales = "free") +
  theme_test()
# Layer-level P-value correction
p + stat_compare(ref_group = 1, correction = "fdr")
# Panel-level P-value correction
p + stat_compare(ref_group = 1, correction = "fdr", panel_indep = TRUE)
```

Index