Package 'ggbrace'

July 22, 2025

Type Package

Title Curly Braces for 'ggplot2'

Depends R (>= 4.3)

Imports ggplot2 (>= 3.4.2), stats (>= 4.3.1)

Version 0.1.2

Description Provides curly braces and square brackets in 'ggplot2' plus matching text.
stat_brace() plots braces/brackets to embrace data.
stat_bracetext() plots corresponding text, fitting to the braces from stat_brace().

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

URL https://github.com/NicolasH2/ggbrace

Suggests knitr, rmarkdown

NeedsCompilation no

Author Nicolas Huber [aut, cre, cph]

Maintainer Nicolas Huber <info.huber@aol.de>

Repository CRAN

Date/Publication 2025-07-09 03:40:02 UTC

Contents

`.coordCorrection` *Imports: stats*

Description

Imports: stats

Usage

```
.coordCorrection(
  x,
  y,
  rotate,
  mid,
  textdistance = NULL,
  distance,
  outerstart,
  width,
  outside,
  bending,
  discreteAxis = FALSE
)
```

Arguments

`x`	vector, x values of all data points
`y`	vector, y value of all data points
`rotate`	number in degrees. Defines where the brace points to: 0=up (default), 90=right, 180=down, 270=left
`mid`	number from 0.25 to 0.75. Position of the pointer within the brace space. If NULL (default), will be determined by data.
`textdistance`	number. Distance of the label to the brace pointer
`distance`	number. Space between the brace and the nearest data point. If NULL (default), will be determined by data.
`outerstart`	number. If not NULL, overwrites distance and sets all braces to the same origin
`width`	number. Distance from the brace's start to its tip. If NULL (default), will be determined by data.
`outside`	boolean. If TRUE (default), brace is next to the data area. If FALSE, brace is inside the data area
`bending`	number from 0 to 0.5. Determines bend of the brace curves (0=rectangular). If NULL (default), will be determined by data. If too high, values will result in zick-zack lines
`discreteAxis`	boolean. Set to TRUE if the axis along which the brace expands is discrete (often true for bar graphs)

`.seekBrace` *Imports: stats*

Description

Imports: stats

Usage

```
.seekBrace(x, y, rotate, bending, npoints, bracketType)
```

Arguments

`x`	vector, x values of all data points
`y`	vector, y value of all data points
`rotate`	number in degrees. Defines where the brace points to: 0=up (default), 90=right, 180=down, 270=left
`bending`	number from 0 to 0.5. Determines bend of the brace curves (0=rectangular). If NULL (default), will be determined by data. If too high, values will result in zick-zack lines
`npoints`	integer. Number of points generated for the brace curves. Will be rounded to be a multiple of 4 for calculation purposes.
`bracketType`	text choice. Either "curly" (default) or "square"

`stat_brace` *create curly braces as a layer in ggplot*

Description

Imports: ggplot2

Usage

```
stat_brace(
  mapping = NULL,
  data = NULL,
  ...,
  rotate = 0,
  width = NULL,
  mid = NULL,
  outside = TRUE,
  distance = NULL,
  outerstart = NULL,
  bending = NULL,
  show.legend = FALSE,
  inherit.aes = TRUE,
  discreteAxis = FALSE,
  bracketType = "curly",
  npoints = 100
)
```

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)`).
`...`	Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `...`. Unknown arguments that are not part of the 4 categories below are ignored. • Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data. • When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept. • Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept. • The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as key glyphs, to change the display of the layer in the legend.
`rotate`	number in degrees. Defines where the brace points to: 0=up (default), 90=right, 180=down, 270=left
`width`	number. Distance from the brace's start to its tip. If NULL (default), will be determined by data.

`mid`	number from 0.25 to 0.75. Position of the pointer within the brace space. If NULL (default), will be determined by data.
`outside`	boolean. If TRUE (default), brace is next to the data area. If FALSE, brace is inside the data area
`distance`	number. Space between the brace and the nearest data point. If NULL (default), will be determined by data.
`outerstart`	number. If not NULL, overwrites distance and sets all braces to the same origin
`bending`	number from 0 to 0.5. Determines bend of the brace curves (0=rectangular). If NULL (default), will be determined by data. If too high, values will result in zick-zack lines
`show.legend`	logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.
`discreteAxis`	boolean. Set to TRUE if the axis along which the brace expands is discrete (often true for bar graphs)
`bracketType`	text choice. Either "curly" (default) or "square"
`npoints`	integer. Number of points generated for the brace curves. Will be rounded to be a multiple of 4 for calculation purposes.

Value

ggplot2 layer object (geom_path) that can directly be added to a ggplot2 object.

Examples

```
library(ggbrace)
library(ggplot2)
data(iris)

# regular braces
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace()

 # rotated braces
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace(rotate = 90)

 # braces inside the given coordinates
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace(outside = FALSE)

 # braces with a defined distance from their data points
```

```
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace(distance = 2)

 # braces starting at a defined point
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace(outerstart = 5)

 # braces starting at a defined point and with defined width
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace(outerstart = 5, width = 1)

 # braces starting at a defined point and with defined width and defined curve bending
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace(outerstart = 5, width = 1, bending = 0.1)

 # braces outside of the plotting area
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace(outerstart = 4.5) +
 coord_cartesian(y=range(iris$Sepal.Width), clip = "off") +
 theme(plot.margin = unit(c(0.25, 0.11, 0.11, 0.11), units="npc"))

 # braces with discrete axes
 df <- iris
 df$Group <- substring(iris$Species,1,1)
 ggplot(df, aes(x=Species, y=Sepal.Length, group=Group)) +
   geom_jitter() +
   stat_brace(discreteAxis=TRUE)
```

`stat_bracetext` *create text for curly braces as a layer in ggplot*

Description

Imports: ggplot2

Usage

```
stat_bracetext(
  mapping = NULL,
  data = NULL,
  geom = "text",
  position = "identity",
  ...,
  rotate = 0,
  width = NULL,
  mid = NULL,
  outside = TRUE,
  distance = NULL,
  outerstart = NULL,
  textdistance = NULL,
  show.legend = FALSE,
  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)`).
`geom`	The geometric object to use to display the data for this layer. When using a `stat_*()` function to construct a layer, the `geom` argument can be used to override the default coupling between stats and geoms. The `geom` argument accepts the following: • A `Geom` ggproto subclass, for example `GeomPoint`. • A string naming the geom. To give the geom as a string, strip the function name of the `geom_` prefix. For example, to use `geom_point()`, give the geom as `"point"`. • For more information and other ways to specify the geom, see the layer geom documentation.
`position`	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following: • The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`. • For more information and other ways to specify the position, see the layer position documentation.

`...`	Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `...`. Unknown arguments that are not part of the 4 categories below are ignored. • Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data. • When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept. • Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept. • The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as key glyphs, to change the display of the layer in the legend.
`rotate`	number in degrees. Defines where the brace points to: 0=up (default), 90=right, 180=down, 270=left
`width`	number. Distance from the brace's start to its tip. If NULL (default), will be determined by data.
`mid`	number from 0.25 to 0.75. Position of the pointer within the brace space. If NULL (default), will be determined by data.
`outside`	boolean. If TRUE (default), brace is next to the data area. If FALSE, brace is inside the data area
`distance`	number. Space between the brace and the nearest data point. If NULL (default), will be determined by data.
`outerstart`	number. If not NULL, overwrites distance and sets all braces to the same origin
`textdistance`	number. Distance of the label to the brace pointer
`show.legend`	logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.
`inherit.aes`	If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

Value

ggplot2 layer object (geom_text or geom_label) that can directly be added to a ggplot2 object.

Examples

```
library(ggbrace)
library(ggplot2)
data(iris)
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species, label=Species)) +
 geom_point() +
 stat_brace() +
 stat_bracetext()
```

Index