Package ‘ggHoriPlot’

October 13, 2022

Type Package

Title Horizon Plots for 'ggplot2'

Version 1.0.1

Maintainer Iker Rivas-González `<ikerrivas96@gmail.com>`

Description A user-friendly, highly customizable R package for building horizon plots in the 'ggplot2' environment.

License GPL-3

URL `https://rivasiker.github.io/ggHoriPlot/`,
`https://github.com/rivasiker/ggHoriPlot`,
`https://CRAN.R-project.org/package=ggHoriPlot`

BugReports `https://github.com/rivasiker/ggHoriPlot/issues`

Depends ggplot2, R (>= 2.10)

Imports dplyr, glue, magrittr, stringr, tidyr

Suggests ggthemes, knitr, patchwork, rmarkdown, testthat (>= 3.0.0), tidyverse, utils

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

NeedsCompilation no

Author Iker Rivas-González [aut, cre] (<`https://orcid.org/0000-0002-0515-0628`>)

Repository CRAN

Date/Publication 2022-10-11 16:22:33 UTC

R topics documented:

`climate_CPH` *Average temperature in Copenhagen*

Description

A dataset containing the average temperature in degrees Celsius for Copenhagen between the year 1995 and 2019.

Usage

```
climate_CPH
```

Format

A data frame with 9,132 rows and 9 variables:

Region geographic region

Country country

State state or territory

City city or town

Month month

Day day

Year year

AvgTemperature average temperature in Fahrenheit

date_mine data in yyyy-mm-dd format

Source

`https://www.kaggle.com/sudalairajkumar/daily-temperature-of-major-cities`

`climate_US` *Average temperature in major cities of the US*

Description

A dataset containing the average temperature in Fahrenheit for major cities in the US for the year 2000.

Usage

```
climate_US
```

Format

A data frame with 57,828 rows and 9 variables:

Region geographic region

Country country

State state or territory

City city or town

Month month

Day day

Year year

AvgTemperature average temperature in Fahrenheit

date_mine data in yyyy-mm-dd format

Source

https://www.kaggle.com/sudalairajkumar/daily-temperature-of-major-cities

https://benschmidt.org/2014/06/05/optimally-ordering-geographical-entities-in-linear-space/

`COVID` *Distribution of COVID-19 cases in Asia*

Description

A dataset containing the geographic distribution of COVID-19 cases in Asia during 2020.

Usage

```
COVID
```

Format

A data frame with 12,695 rows and 3 variables:

date_mine date of the measurement in yyyy-mm-dd format

y standardized number of cases

countriesAndTerritories countries and territories in Asia

Source

https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-covid-

---

`geom_horizon` *Horizon plots*

Description

This function builds horizon plots in ggplot2. It allows for the customization of the origin and the horizon scale.

Usage

```
geom_horizon(
  mapping = NULL,
  data = NULL,
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = TRUE,
  origin = "midpoint",
  horizonscale = 6,
  rm.outliers = FALSE,
  reverse = FALSE,
  mirror = FALSE,
  inherit.aes = TRUE
)

stat_horizon(
  mapping = NULL,
  data = NULL,
  geom = "ribbon",
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = TRUE,
  inherit.aes = TRUE,
  origin = "midpoint",
```

```
    horizonscale = 6,
    rm.outliers = FALSE,
    reverse = FALSE,
    mirror = FALSE
  )
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()` or `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`	Position adjustment, either as a string, or the result of a call to a position adjustment function.
`...`	Other arguments passed on to `layer()`. These are often aesthetics, used to set an aesthetic to a fixed value, like `colour = "red"` or `size = 3`. They may also be parameters to the paired geom/stat.
`na.rm`	If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.
`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.
`origin`	Origin of the horizon plot. It can either be a string, namely `"midpoint"` (the default), `"median"`, `"mean"`, `"min"` or `"quantiles"`, or a user-specified number.
`horizonscale`	Cutpoints of the horizon plot. It can either be an integer specifying the number of ranges (default is `6`), or a user-supplied numeric vector with the cutpoints defining the different ranges.
`rm.outliers`	If `TRUE`, all the values below `quantile(y, 0.25)-1.5*IQR(y)` and above `quantile(y, 0.75)+1.5*IQR(y)` are excluded from the origin and cutpoint calculations (default is `FALSE`). @param reverse If `TRUE`, the horizon peaks for the values below the origin are reversed (default is `FALSE`).
`reverse`	IF `TRUE`, the horizon peaks for all the values below the origin are reversed (default is `FALSE`).
`mirror`	If `TRUE`, the horizon peaks for all the values are reversed (default is `FALSE`).
`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()`.
`geom`	Used geom, default to 'ribbon'

Details

A horizon plot is a special type of area plot in which the original data is transformed based on an origin and a horizon scale. The data is cut in different intervals, and the further the data is from the origin, the deeper its color usually is. All the intervals above the origin are then stacked on top of one another, keeping the intervals closest to the origin in the bottom and the furthest away ones on top. Likewise, the intervals below the origin are normally given a different color palette and they are stacked in a similar manner in the same area as the intervals above the origin. You can learn more about how horizon plots are built in `vignette('ggHoriPlot')` or at https://bernatgel.github.io/karyoploter_tutorial/Tutorial/PlotHorizon/PlotHorizon.html.

Value

'ggplot2' layer for building a horizon plot.

Orientation

This geom treats each axis differently and, thus, can thus have two orientations. Often the orientation is easy to deduce from a combination of the given mappings and the types of positional scales in use. Thus, ggplot2 will by default try to guess which orientation the layer should have. Under rare circumstances, the orientation is ambiguous and guessing may fail. In that case the orientation can be specified directly using the `orientation` parameter, which can be either `"x"` or `"y"`. The value gives the axis that the geom should run along, `"x"` being the default orientation you would expect for the geom.

Aesthetics

`geom_horizon()` understands the following aesthetics (required aesthetics are in bold):

- `x`
- `y`
- `xend`
- `alpha`
- `colour`
- `fill`

Examples

```
# Generate data
huron <- data.frame(year = 1875:1972, level = as.vector(LakeHuron))

# Basic plot with default colors
h <- ggplot(huron) + geom_horizon(aes(year, level))

# Add color scheme
h + theme_void() + scale_fill_hcl()

# Add cupoints
ggplot(huron) +
  geom_horizon(aes(year, level, fill = ..Cutpoints..)) +
```

```
  theme_void() +
  scale_fill_hcl()
```

`rmsk` *Distribution of repeats along the human genome*

Description

A dataset containing the percentage of simple repeats in 100 kb windows along the human genome (hg38).

Usage

```
rmsk
```

Format

A data frame with 30,885 rows and 4 variables:

genoName chromosome name

bin starting coordinate of window

bin_2 end coordinate of window

p_repeat percentage of repeats

Source

https://genome.ucsc.edu/cgi-bin/hgTrackUi?g=rmsk

`scale_fill_hcl` *Create your own discrete scale*

Description

These functions allow you to specify your own set of mappings from levels in the data to aesthetic values.

Usage

```
scale_fill_hcl(..., palette = "RdYlBu", reverse = FALSE)
```

Arguments

`...`	Arguments passed on to `ggplot2::discrete_scale` `scale_name` The name of the scale that should be used for error messages associated with this scale. `name` The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted. `breaks` One of: • `NULL` for no breaks • `waiver()` for the default breaks (the scale limits) • A character vector of breaks • A function that takes the limits as input and returns breaks as output. Also accepts rlang lambda function notation. `labels` One of: • `NULL` for no labels • `waiver()` for the default labels computed by the transformation object • A character vector giving labels (must be same length as `breaks`) • A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation. `limits` One of: • `NULL` to use the default scale values • A character vector that defines possible values of the scale and their order • A function that accepts the existing (automatic) values and returns new ones. Also accepts rlang lambda function notation. `na.translate` Unlike continuous scales, discrete scales can easily show missing values, and do so by default. If you want to remove missing values from a discrete scale, specify `na.translate = FALSE`. `na.value` If `na.translate = TRUE`, what aesthetic value should the missing values be displayed as? Does not apply to position scales where `NA` is always placed at the far right. `drop` Should unused factor levels be omitted from the scale? The default, `TRUE`, uses the levels that appear in the data; `FALSE` uses all the levels in the factor. `guide` A function used to create a guide or its name. See `guides()` for more information. `super` The super class to use for the constructed scale
`palette`	the name of the palette to generate colors from. A list of all available palettes can be found by running `colorspace::hcl_palettes()` (default is "RdYlBu").
`reverse`	If `TRUE`, the order of the colors is reversed (default is `FALSE`)’

Details

The functions ‘scale_colour_manual()‘, ‘scale_fill_manual()‘, ‘scale_size_manual()‘, etc. work on the aesthetics specified in the scale name: ‘colour‘, ‘fill‘, ‘size‘, etc. However, the functions ‘scale_colour_manual()‘ and ‘scale_fill_manual()‘ also have an optional ‘aesthetics‘ argument that can be used to define both ‘colour‘ and ‘fill‘ aesthetic mappings via a single function call (see examples). The function ‘scale_discrete_manual()‘ is a generic scale that can work with any aesthetic or set of aesthetics provided via the ‘aesthetics‘ argument.

Value

Scale layer for the fill aesthetic.

`sports_time` *Peaks times for sports and leisure activities*

Description

A dataset containing the peak time for doing 29 sports and leisure activities.

Usage

```
sports_time
```

Format

A data frame with 8,092 rows and 3 variables:

activity name of the activity

time time of the day, in hhmm format

p standardized peak

Source

`https://github.com/halhen/viz-pub/blob/master/sports-time-of-day/activity.tsv`

`https://eagereyes.org/blog/2017/joy-plots`

Index