Package 'sugarglider'

July 23, 2025

Type Package

Title Create Glyph-Maps of Spatiotemporal Data

Version 1.0.3

Description
Provides 'ggplot2' extensions to construct glyph-maps for visualizing seasonality in spatiotemporal data. See the Journal of Statistical Software reference: Zhang, H. S., Cook, D., Laa, U., Langrené, N., & Menéndez, P. (2024) <doi:10.18637/jss.v110.i07>. The manuscript for this package is currently under preparation and can be found on GitHub at <https://github.com/maliny12/paper-sugarglider>.

License MIT + file LICENSE

URL https://maliny12.github.io/sugarglider/,
https://github.com/maliny12/sugarglider

Encoding UTF-8

LazyData true

Imports dplyr, ggplot2

RoxygenNote 7.3.1

Depends R (>= 3.0.0)

Suggests testthat (>= 3.0.0), ggplotify, tidyr, grid, lubridate, knitr, rmarkdown, gridExtra, ozmaps, sf, tidyverse, viridis, ggthemes, vdiffr, kableExtra, ggiraph, jsonlite, httr, usmap, geosphere, purrr

Config/testthat/edition 3

VignetteBuilder knitr

BugReports https://github.com/maliny12/sugarglider/issues

NeedsCompilation no

Author Maliny Po [aut, cre, cph] (ORCID: <https://orcid.org/0009-0008-4686-6631>), S. Nathan Yang [aut] (ORCID: <https://orcid.org/0009-0002-9985-1042>), H. Sherry Zhang [ctb] (ORCID: <https://orcid.org/0000-0002-7122-1463>), Dianne Cook [ctb] (ORCID: <https://orcid.org/0000-0002-3813-7155>)

Maintainer Maliny Po <malinypo12@gmail.com>

Repository CRAN

Date/Publication 2024-10-24 14:10:02 UTC

Contents

`add_glyph_boxes` *Add Glyph Boxes layer to glyph plot*

Description

This function introduces a custom layer to a ggplot, employing 'glyph boxes' to visually represent individual glyph. Users can specify various aesthetics including alpha, height, width, color, line type, and fill to customize the appearance.

Usage

```
add_glyph_boxes(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  x_major = NULL,
  y_major = NULL,
  height = "default",
  width = "default",
  fill = "white",
  linewidth = 0.1,
  inherit.aes = TRUE,
  show.legend = NA,
  ...
)
```

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. If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot().
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.
x_major, y_major	Aesthetics to map plot coordinates for major and minor glyph components.
height	The height of each glyph. The 'default' is calculated using the ratio (1:1.618) relative to the 'width', to maintain a consistent aspect ratio.
width	The width of each glyph. The 'default' is set to the smallest distance between two consecutive coordinates, converted from meters to degrees of latitude using the Haversine method.
fill	The color used to fill the glyph box.
linewidth	The thickness of the glyph box.
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().
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.
...	Additional arguments passed on to function.

Value

A layer object that can be added to a ggplot.

`add_glyph_legend` *Add Legend Layer to a ggplot*

Description

This function adds a custom legend layer to a ggplot object using the specified aesthetics and parameters.

Usage

```
add_glyph_legend(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  show.legend = NA,
  x_minor = NULL,
  x_scale = identity,
  y_scale = identity,
  fill = "black",
  color = "black",
  linewidth = 0.5,
  alpha = 0.8,
  global_rescale = 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.

`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.

`x_minor` Aesthetics to map plot coordinates for major and minor glyph components.

`x_scale, y_scale`
The scaling function applied to each set of minor values within a grid cell. Defaults to 'identity'.

`fill` The fill color for the geometric object.

`color` The color of the geometric object's border or line.

`linewidth` The width of the geometric object's line.

`alpha` The transparency level of the geometric object, ranging from 0 (fully transparent) to 1 (fully opaque).

`global_rescale` A setting that determines whether to perform rescaling globally or on individual glyphs.

`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()`.

`...` Additional arguments passed on to function.

Value

A ggplot2 layer.

add_ref_lines *Add reference lines to glyph plot*

Description

This function draw reference lines that include both major and minor division markers.

Usage

```
add_ref_lines(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  show.legend = NA,
  x_major = NULL,
  y_major = NULL,
  height = "default",
  width = "default",
  inherit.aes = TRUE,
  linewidth = 0.1,
  ...
)
```

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. If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot().
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.

`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.

`x_major, y_major` Aesthetics to map plot coordinates for major and minor glyph components.

`height` The height of each glyph. The 'default' is calculated using the ratio (1:1.618) relative to the 'width', to maintain a consistent aspect ratio.

`width` The width of each glyph. The 'default' is set to the smallest distance between two consecutive coordinates, converted from meters to degrees of latitude using the Haversine method.

`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()`.

`linewidth` The thickness of the reference line.

`...` Additional arguments passed on to function.

Value

A ggplot2 layer.

`aus_temp` *Australian Weather Data for 2022*

Description

This dataset contains aggregated monthly average temperatures (minimum and maximum) and precipitation for selected Australian weather stations for the year 2022. Stations were selected based on specific criteria such as operational status and completeness of data for the year.

Usage

```
aus_temp
```

Format

A data frame with the following columns:

id Station ID.

long Longitude of the station.

lat Latitude of the station.

month Month for the aggregated data.

tmin Monthly average minimum temperature (in degrees Celsius).

tmax Monthly average maximum temperature (in degrees Celsius).

prcp Monthly average precipitation (in mm).

Source

GHCN Daily data via ‘meteo_pull_monitors‘ from the ‘rnoaa‘ package.

---

flights *Flight Summary from Airports with the Most Cancellations*

Description

This dataset contains information on the minimum and maximum number of flights that originated from the top 10 U.S. airports with the highest number of flight cancellations. The airports included are Denver (DEN), Orlando (MCO), Seattle (SEA), Atlanta (ATL), Dallas/Fort Worth (DFW), Chicago O'Hare (ORD), Las Vegas (LAS), Los Angeles (LAX), and Phoenix (PHX).

Usage

```
flights
```

Format

‘flights‘ A data frame with 120 rows and 6 columns:

origin The origin airport for that flight

month The month of the flight

long Longitude of the airport

lat Latitude of the airport

min_flights The minimum number of flights that originated from the airport

max_flights The maximum number of flights that originated from the airport

geom_glyph_ribbon *Create a Glyph Ribbon plot using ggplot2*

Description

This function creates a ribbon geometry designed to display glyphs based on the combination of ‘x_major‘ and ‘y_major‘. For each ‘x_minor‘ value, ‘geom_glyph_ribbon()‘ displays a y interval defined by ‘ymin_minor‘ and ‘ymax_minor‘.

Usage

```
geom_glyph_ribbon(
  mapping = NULL,
  data = NULL,
  show.legend = NA,
  stat = "identity",
  position = "identity",
  x_major = NULL,
  y_major = NULL,
  x_minor = NULL,
  ymin_minor = NULL,
  ymax_minor = NULL,
  height = "default",
  width = "default",
  x_scale = identity,
  y_scale = identity,
  global_rescale = 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)).

`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.

`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.

`x_major`, `y_major`, `x_minor`, `ymin_minor`, `ymax_minor`
Each combination of 'x_major' and 'y_major' forms a unique grid cell. 'ymin_minor' and 'ymax_minor' define the lower and upper bounds of the geom_ribbon.

`height` The height of each glyph. The 'default' is calculated using the ratio (1:1.618) relative to the 'width', to maintain a consistent aspect ratio.

`width` The width of each glyph. The 'default' is set to the smallest distance between two consecutive coordinates, converted from meters to degrees of latitude using the Haversine method.

`x_scale`, `y_scale`
The scaling function applied to each set of minor values within a grid cell. Defaults to 'identity'.

`global_rescale` A setting that determines whether to perform rescaling globally or on individual glyphs.

`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()`.

`...` Additional arguments passed on to function.

Value

A ggplot object.

Examples

```
library(ggplot2)

# Basic glyph map with base map and custom theme
aus_temp |>
  ggplot(aes(x_major = long, y_major = lat,
         x_minor = month, ymin_minor = tmin, ymax_minor = tmax)) +
  geom_sf(data = ozmaps::abs_ste, fill = "grey95",
          color = "white",inherit.aes = FALSE) +
  geom_glyph_ribbon() +
  ggthemes::theme_map()


# Adjust width and height of the glyph
aus_temp |>
  ggplot(aes(x_major = long, y_major = lat,
         x_minor = month, ymin_minor = tmin, ymax_minor = tmax)) +
  geom_sf(data = ozmaps::abs_ste, fill = "grey95",
          color = "white",inherit.aes = FALSE) +
  geom_glyph_ribbon(width = rel(4.5), height = rel(3)) +
 ggthemes::theme_map()

# Extend glyph map with reference box and line
aus_temp |>
 ggplot(aes(x_major = long, y_major = lat,
         x_minor = month, ymin_minor = tmin, ymax_minor = tmax)) +
  geom_sf(data = ozmaps::abs_ste, fill = "grey95",
          color = "white",inherit.aes = FALSE) +
  add_glyph_boxes() +
  add_ref_lines() +
  geom_glyph_ribbon() +
  ggthemes::theme_map()
```

`geom_glyph_segment` *Create a Glyph Segment plot using ggplot2*

Description

This function enables the creation of segment glyphs by defining major coordinates (longitude and latitude) and minor segment structures within a grid cell. Each glyph's appearance can be customized by specifying its height, width, and scaling, allowing for flexible data representation in a visual context.

Usage

```
geom_glyph_segment(
  mapping = NULL,
  data = NULL,
  stat = "identity",
```

```
  position = "identity",
  ...,
  x_major = NULL,
  x_minor = NULL,
  y_major = NULL,
  y_minor = NULL,
  yend_minor = NULL,
  width = "default",
  x_scale = identity,
  y_scale = identity,
  height = "default",
  global_rescale = TRUE,
  show.legend = NA,
  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, either as a ggproto Geom subclass or as a string naming the stat stripped of the stat_ prefix (e.g. "count" rather than "stat_count")
position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use position_jitter), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
...	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.
x_major, x_minor, y_major, y_minor, yend_minor	The name of the variable (as a string) for the major and minor x and y axes. x_major and y_major specify a longitude and latitude on a map while x_minor, y_minor, and yend_minor provide the structure for glyph.
width	The width of each glyph. The 'default' is set to the smallest distance between two consecutive coordinates, converted from meters to degrees of latitude using the Haversine method.

`y_scale`, `x_scale`
: The scaling function to be applied to each set of minor values within a grid cell. The default is `identity` which produces a result without scaling.

`height`
: The height of each glyph. The 'default' is calculated using the ratio (1:1.618) relative to the 'width', to maintain a consistent aspect ratio.

`global_rescale`
: Determines whether or not the rescaling is performed globally or separately for each individual glyph.

`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

a ggplot object

Examples

```
library(ggplot2)

# Basic glyph map with base map and custom theme
aus_temp |>
  ggplot(aes(x_major = long, y_major = lat,
         x_minor = month, y_minor = tmin, yend_minor = tmax)) +
  geom_sf(data = ozmaps::abs_ste, fill = "grey95",
          color = "white",inherit.aes = FALSE) +
  geom_glyph_segment() +
  ggthemes::theme_map()


# Adjust width and height of the glyph
aus_temp |>
  ggplot(aes(x_major = long, y_major = lat,
         x_minor = month, y_minor = tmin, yend_minor = tmax)) +
  geom_sf(data = ozmaps::abs_ste, fill = "grey95",
          color = "white",inherit.aes = FALSE) +
  geom_glyph_segment(width = rel(4.5), height = rel(3)) +
 ggthemes::theme_map()

# Extend glyph map with reference box and line
aus_temp |>
 ggplot(aes(x_major = long, y_major = lat,
         x_minor = month, y_minor = tmin, yend_minor = tmax)) +
  geom_sf(data = ozmaps::abs_ste, fill = "grey95",
          color = "white",inherit.aes = FALSE) +
  add_glyph_boxes() +
  add_ref_lines() +
  geom_glyph_segment() +
  ggthemes::theme_map()
```

`historical_temp` *Historical Australian Weather Data from 2021-2022*

Description

This dataset contains aggregated monthly average temperatures (minimum and maximum) and precipitation for selected Australian weather stations for the years 2021 and 2022. It provides a broader historical perspective compared to ‘aus_temp‘. Stations were selected based on operational status and data completeness.

Usage

```
historical_temp
```

Format

A data frame with the following columns:

id Station ID.

long Longitude of the station.

lat Latitude of the station.

month Month for the aggregated data.

year Year for the aggregated data, either 2021 or 2022.

tmin Monthly average minimum temperature (in degrees Celsius).

tmax Monthly average maximum temperature (in degrees Celsius).

prcp Monthly average precipitation (in mm).

Source

GHCN Daily data via ‘meteo_pull_monitors‘ from the ‘rnoaa‘ package.

theme_glyph *Customized Theme for Glyph Plots*

Description

'theme_glyph()' provides a customized theme for glyph maps, built on top of 'theme_map()' from 'ggthemes'. It adjusts the plot's appearance, including the legend position, text styles, and background settings, to create a clean, visually consistent layout for glyph visualizations.

Usage

```
theme_glyph(
 control = list(plot.title = rel(1.5), plot.subtitle = rel(1.3), plot.caption = rel(1),
    legend.text = rel(1), legend.title = rel(1))
)
```

Arguments

control	A list specifying the relative font sizes for different plot elements. The list can contain the following components: **plot.title** Font size for the plot title (default: rel(1.5)). **plot.subtitle** Font size for the plot subtitle (default: rel(1.3)). **plot.caption** Font size for the plot caption (default: rel(1)). **legend.text** Font size for the legend text (default: rel(1)). **legend.title** Font size for the legend title (default: rel(1)).

Details

This theme includes:

- Legend positioned inside the plot, at the bottom left corner.
- Horizontal legend direction with 'mono' font for text.
- Centered plot titles with bold, 'mono' font.
- 'mono' fonts for subtitles and captions.
- White background for both the panel and plot.

Value

A ggplot2 theme object with customized settings for glyph plots.

See Also

[ggthemes::theme_map()], [ggplot2::theme()]

Examples

```
library(ggplot2)
library(ozmaps)

aus_temp |>
  ggplot(aes(x_major = long, y_major = lat,
             x_minor = month, ymin_minor = tmin,
             ymax_minor = tmax)) +
 geom_sf(data = abs_ste, fill = "antiquewhite",
         inherit.aes = FALSE, color = "white") +
 add_glyph_boxes() +
 add_ref_lines() +
 geom_glyph_ribbon() +
 theme_glyph()
```

---

train *Hourly Train Station Patronage 2023-2024*

Description

This data set provides a comprehensive hourly summary of patronage at each train station in Victoria for the fiscal year 2023-2024. The number of patronage is definded by the total number of boarding and alighting at each station. This data set includes detailed breakdowns by day types such as weekdays, weekends, and holidays (including school and public holidays). Note the influence of station closures which may skew or omit data on particular days.

Usage

```
data(train)
```

Format

A data frame with each row representing aggregated monthly data per station, containing:

station_name Name of the train station.

hour Operating hour ranging from 5AM to 12PM.

long Longitude of the train station.

lat Latitude of the train station.

services Number of unique services passing through the station.

mode Transportation mode, such as 'Metro', 'VLine', or both.

min_weekday Minimum hourly patronage on a typical weekday.

max_weekday Maximum hourly patronage on a typical weekday.

min_weekend Minimum hourly patronage on weekends.

max_weekend Maximum hourly patronage on weekends.

min_holiday Minimum hourly patronage during school and public holidays.

max_holiday Maximum hourly patronage during school and public holidays.

Note

Stations with incomplete entries due to closure are not included in this dataset.

Source

https://discover.data.vic.gov.au/dataset/train-service-passenger-counts

Index