Package 'gghourglass'

June 20, 2025

Title Plot Records per Time of Day

Version 0.0.3

Author Pepijn de Vries [aut, cre] (ORCID:
<https://orcid.org/0000-0002-7961-6646>),
Sander Lagerveld [dtc] (ORCID: <https://orcid.org/0000-0003-1291-4021>)

Description Splits date and time of day components from
continuous 'datetime' objects, then plots them using grammar
of graphics ('ggplot2'). Plots can also be decorated with
solar cycle information (e.g., sunset, sunrise, etc.).
This is useful for visualising data that are associated with
the solar cycle.

Depends R (>= 4.1.0)

Imports dplyr (>= 1.1.4), ggplot2 (>= 3.4.4), grid (>= 4.1.0),
lubridate (>= 1.9.3), rlang (>= 1.1.2), suncalc (>= 0.5.1),
tidyr (>= 1.3.0)

Suggests knitr, readr (>= 2.1.4), rmarkdown, testthat (>= 3.0.0),
usethis (>= 2.2.2), vdiffr

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

LazyData true

VignetteBuilder knitr

Config/testthat/edition 3

URL https://pepijn-devries.github.io/gghourglass/,
https://github.com/pepijn-devries/gghourglass

BugReports https://github.com/pepijn-devries/gghourglass/issues

NeedsCompilation no

Maintainer Pepijn de Vries <pepijn.devries@outlook.com>

Repository CRAN

Date/Publication 2025-06-20 21:30:07 UTC

Contents

`AnnotateDaylight` *Annotate ggplot with a band indicating solar events*

Description

Annotate a ggplot (currently only plots using `coord_hourglass()` is supported) with a coloured band indicating solar events, such as sunset and sunrise.

Usage

```
AnnotateDaylight

annotate_daylight(
  longitude = 0,
  latitude = 60,
  sun_prop = c("sunrise", "sunset"),
  ...
)
```

Arguments

`longitude`, `latitude`	Geographical location that will be used to calculate sunlight times.
`sun_prop`	A vector of two solar events that should be captured by the annotation. It will be shown as a coloured band between these two events. Default is `c("sunrise", "sunset")`, but could also be `c("dusk", "dawn")`. See `suncalc::getSunlightTimes()` for all allowed solar events.
`...`	Passed to the list of layer parameters.

Format

An object of class `AnnotateDaylight` (inherits from `GeomPolygon`, `Geom`, `ggproto`, `gg`) of length 6.

Value

Returns a ggplot2::layer() which can be added to a ggplot2::ggplot()

Author(s)

Pepijn de Vries

Examples

```
library(ggplot2)
data(bats)

monitoring <- attr(bats, "monitoring")

ggplot(subset(bats, format(RECDATETIME, "%Y") == "2018"),
       aes(x = RECDATETIME, col = SPECDESCSCI)) +
 annotate_daylight(monitoring$longitude[1], monitoring$latitude[1], c("sunset", "sunrise")) +
 annotate_daylight(monitoring$longitude[1], monitoring$latitude[1], c("dusk", "dawn")) +
 geom_hourglass()
```

AnnotateLunarphase *Annotate ggplot with lunar phases*

Description

This function uses the suncalc package to calculate the lunar phase and uses it to annotate your plot. If your plot has an axis with a continuous datetime scale, lunar phases are plot along this axis. Otherwise you have to specify the date of the lunar phase.

Usage

```
AnnotateLunarphase

annotate_lunarphase(
  date = NULL,
  longitude = NULL,
  latitude = NULL,
  breaks = ggplot2::waiver(),
  placement = 0.9,
  radius = grid::unit(5, "mm"),
  n = 26,
  ...
)
```

Arguments

`date`	A datetime object used to calculate the illuminated fraction of the moon
`longitude, latitude`	Used to calculate zenith angle. This will result in a more accurate shape of the moon as observed at the specified location.
`breaks`	One of: • `NULL` for no breaks • `waiver()` for the breaks specified by `date_breaks` • A `Date`/`POSIXct` vector giving positions of breaks • A function that takes the limits as input and returns breaks as output
`placement`	Relative placement of the lunar annotation in the plotting panel. It should be between 0 and 1. Default is 0.9.
`radius`	Size of the lunar pictogram. It is best to use an absolute unit from the `grid` package. Default is a radius of 5 mm (`grid::unit(5, "mm")`)
`n`	Number of coordinates in the returned polygon shape (should be even).
`...`	Passed to the list of layer parameters.

Format

An object of class `AnnotateLunarphase` (inherits from `GeomPolygon`, `Geom`, `ggproto`, gg) of length 6.

Value

Returns a `ggplot2::layer()` which can be added to a `ggplot2::ggplot()`

Author(s)

Pepijn de Vries

Examples

```
library(ggplot2)
library(dplyr)
library(lubridate)
data(bats)

monitoring <- attr(bats, "monitoring")

## A lunar annotation can be added to a geom_hourglass layer
ggplot(mutate(bats, YEAR = year(RECDATETIME), MONTH = month(RECDATETIME)) |>
         filter(YEAR == 2018, MONTH == 5),
       aes(x = RECDATETIME, col = SPECDESCSCI)) +

  geom_hourglass() +

  annotate_lunarphase(
    longitude = monitoring$longitude[[1]],
```

```
    latitude  = monitoring$latitude[[1]],
    placement = 0.8) +

  scale_x_datetime(limits = as_datetime(c("2018-04-27", "2018-05-31")))

## In fact, it can be added to any plot with a continuous datetime scale

ggplot(data.frame(stamp = seq(as_datetime("2025-04-01 UTC"),
                              as_datetime("2025-04-30 UTC"),
                              length.out = 20),
                  value = 1:20), aes(x = stamp, y = value)) +
  geom_point() +
  annotate_lunarphase()

## Moreover, you can add it to an arbitrary plot without such scales,
## but then you need to specify the date

ggplot(data.frame(stamp = 1:20,
                  value = 1:20), aes(x = stamp, y = value)) +
  geom_point() +
  annotate_lunarphase(date = "2020-01-01", placement = c(0.1, 0.9))
```

`annotate_periodstates` *Annotate a period in an hourglass plot*

Description

Adds rectangles to a `geom_hourglass()` plot layer. It can be used to mark specific periods. The example shows how this annotation can be used to mark the periods when detector (used for the observations) was active. Note that this may not work correctly when displaying data that uses datetime objects with daylight saving time. In those cases you could split the periods into parts with and without daylight saving. Or convert your data to a timezone without daylight saving time (e.g. UTC).

Usage

```
annotate_periodstates(mapping, data, hour_center = 0, ...)
```

Arguments

`mapping`	A `ggplot2::aes()` object that maps the periods. It needs `x`, `y`, `xend` and `yend`, which mark the conrners of the rectangles (i.e. periods)
`data`	A `data.frame` containing information about the periods.
`hour_center`	The hour at which the time of day is centred. Default is 0, meaning midnight. -12 centres around noon of the preceding day, +12 centres around noon of the next day.
`...`	Passed to layer parameters.

Author(s)

Pepijn de Vries

Examples

```
library(ggplot2)
library(dplyr)
library(lubridate)

## Extract monitoring periods from 'bats' data
monitoring_periods <-
  attr(bats, "monitoring") |>
  mutate(time_on  = as_datetime(time_on),
         time_off = as_datetime(time_off))

ggplot(bats, aes(x = RECDATETIME, col = SPECDESCSCI)) +

  ## Set background to transparent red to contrast with
  ## monitoring periods
  theme(panel.background = element_rect(fill = "#FF000044")) +

  ## Annotate periods in which the detector was active with
  ## white rectangles
  annotate_periodstates(
    aes(x    = start,   xend = end,
        y    = time_on, yend = time_off),
    monitoring_periods,
    fill = "white") +

 ## plot observations
  geom_hourglass(hour_center = -6)
```

`bats` *Observations from a bat detector*

Description

A dataset containing detections of audio call sequences from bats. It is a small subset of the data published by Lagerveld et al. (2023)

Format

A `data.frame` with 1,037 rows and 2 columns:

- `RECDATETIME`: datetime of the recorded bat call sequence
- `SPECDESCSCI`: scientific species name

It also contains an attribute named `monitoring` which is a `data.frame` containing monitoring periods at which the bat detector was active. Each row is a monitoring period, and it holds the following columns:

- `start`: start datetime of the monitoring period
- `end`: end datetime of the monitoring period
- `time_on`: time of day at which the detector is activated during the monitoring period
- `time_off`: time of day at which the detector is deactivated during the monitoring period
- `longitude` and `latitude`: coordinates of the detector's location
- `altitude`: altitude in meters above sea level of the detector.

References

Lagerveld, S., Wilkes, T., van Puijenbroek, M.E.B., Noort, B.C.A., Geelhoed, S.C.V. Acoustic monitoring reveals spatiotemporal occurrence of Nathusius' pipistrelle at the southern North Sea during autumn migration. Environ Monit Assess 195, 1016 (2023) doi:10.1007/s10661-023-11590-2

Examples

```
data("bats")
```

`CoordHourglass` *Hourglass coordinates for a ggplot*

Description

A Cartesian coordinate system that adds sensible guides to axes in a `geom_hourglass()` layer. It is added automatically to `geom_hourglass()`. There is no need to explicitly add it to a ggplot, unless you wish to tweak the coordinate system.

Usage

```
CoordHourglass

coord_hourglass(
  xlim = NULL,
  ylim = NULL,
  expand = TRUE,
  default = FALSE,
  clip = "on",
  date_labels = "%H:%M",
  layer = NULL,
  ...
)
```

Arguments

xlim, ylim	Limits for the x and y axes.
expand	If TRUE, the default, adds a small expansion factor to the limits to ensure that data and axes don't overlap. If FALSE, limits are taken exactly from the data or xlim/ylim.
default	Is this the default coordinate system? If FALSE (the default), then replacing this coordinate system with another one creates a message alerting the user that the coordinate system is being replaced. If TRUE, that warning is suppressed.
clip	Should drawing be clipped to the extent of the plot panel? A setting of "on" (the default) means yes, and a setting of "off" means no. In most cases, the default of "on" should not be changed, as setting clip = "off" can cause unexpected results. It allows drawing of data points anywhere on the plot, including in the plot margins. If limits are set via xlim and ylim and some data points fall outside those limits, then those data points may show up in places such as the axes, the legend, the plot title, or the plot margins.
date_labels	Formating string for formatting the time labels on the axis. By default it is "%H:%M".
layer	This argument allows you to pass the hourglass layer to which the coordinate system should be applied. It is used to derive the orientation of the plot. By default it is NULL and an attempt is made to derive the orientation of the plot from its axes scales.
...	Arguments passed as extra params to ggplot2::layer()

Format

An object of class CoordHourglass (inherits from CoordCartesian, Coord, ggproto, gg) of length 3.

Value

Returns a ggproto object inheriting from coord_cartesian().

Author(s)

Pepijn de Vries

Examples

```
coord_hourglass()
```

GeomHourglass *Add an 'hourglass' layer to a ggplot*

Description

geom_hourglass() takes a continuous datetime object, splits in in discrete dates and time of day with stat_hourglass(). This geometry is a wrapper to add it as a layer to a ggplot. GeomHourglass is a ggproto object inheriting from ?ggplot2::GeomPoint. It should not be used directly. Instead call geom_hourglass().

Usage

```
GeomHourglass

geom_hourglass(
  mapping = NULL,
  data = NULL,
  stat = "hourglass",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  hour_center = 0,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::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. The hourglass stat and geom requires either the x axis or the y axis to be mapped. The mapped aesthetic will show the date of the variable, whereas the opposite axis will show the time of day.
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 ggplot2::ggplot(). Otherwise, a data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See ggplot2::fortify() for which variables will be created. The data should contain a column with datetime values (e.g., ?POSIXct)
stat	Can be used to overwrite the default connection between geom_hourglass and stat_hourglass().
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.

`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.

`hour_center` The hour at which the time of day is centred. Default is 0, meaning midnight. -12 centres around noon of the preceding day, +12 centres around noon of the next day.

`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()`.

`...` Arguments passed to geometry.

Format

An object of class `GeomHourglass` (inherits from `GeomPoint`, `Geom`, `ggproto`, `gg`) of length 3.

Value

Returns a `ggplot2::layer()` which can be added to a `ggplot2::ggplot()`

Author(s)

Pepijn de Vries

Examples

```
library(ggplot2)
data(bats)

monitoring <- attr(bats, "monitoring")

ggplot(subset(bats, format(RECDATETIME, "%Y") == "2019"),
       aes(x = RECDATETIME, col = SPECDESCSCI)) +
  geom_hourglass()

ggplot(dplyr::mutate(bats, YEAR = format(RECDATETIME, "%Y")),
       aes(x = RECDATETIME, col = SPECDESCSCI)) +
  geom_hourglass() +
  facet_wrap(~YEAR, scales = "free_x")
```

get_hour *Functions to split datetime into date and time of day*

Description

Split a datetime object in a date component (`get_date()`) and time of day (`get_hour()`) component, centred around a specific hour of the day. They are used by `stat_hourglass()`.

Usage

```
get_hour(x, hour_center = 0, ...)

get_date(x, hour_center = 0, ...)
```

Arguments

`x`	A datetime object (e.g., `as.POSIXct()`) to extract day of time from
`hour_center`	The hour at which the time of day is centred. Default is 0, meaning midnight. -12 centres around noon of the preceding day, +12 centres around noon of the next day.
`...`	Ignored

Value

Returns a `period` (`lubridate::as.period()`) in case of `get_hour()`. Returns a datetime object in case of `get_date()`

Author(s)

Pepijn de Vries

Examples

```
my_datetime <- as.POSIXct("2020-02-02 02:20:02 UTC", tz = "UTC")
get_hour(my_datetime)
get_hour(my_datetime, -12)

get_date(my_datetime)
get_date(my_datetime, -12)

## This will return the original `my_date`
get_date(my_datetime) + get_hour(my_datetime)

## This will too
get_date(my_datetime, -12) + get_hour(my_datetime, -12)
```

`lunar_phase_polygon` *Get the shape of the illuminated part of the moon*

Description

Function that calculates coordinates of a polygon representing the shape of the illuminated fraction of the moon, as observed from Earth. The shape has a radius of 1 and is centred around (0, 0). It does not consider lunar eclipses.

Usage

```
lunar_phase_polygon(date, longitude, latitude, n = 100)
```

Arguments

`date`	A datetime object used to calculate the illuminated fraction of the moon
`longitude, latitude`	Used to calculate zenith angle. This will result in a more accurate shape of the moon as observed at the specified location.
`n`	Number of coordinates in the returned polygon shape (should be even).

Value

Returns a `data.frame` with coordinates of a polygon representing the shape of the illuminated fraction of the moon.

Author(s)

Pepijn de Vries

Examples

```
disc_illum <- lunar_phase_polygon(as.POSIXct("2025-04-01"))
plot(NA, NA, xlim = c(-1,1), ylim = c(-1, 1), asp = 1,
     xlab = "x coord", ylab = "y coord")
polygon(disc_illum$x, disc_illum$y, col = "white")

disc_illum <- lunar_phase_polygon(as.POSIXct("2025-04-01"), 5, 50)
plot(NA, NA, xlim = c(-1,1), ylim = c(-1, 1), asp = 1,
     xlab = "x coord", ylab = "y coord")
polygon(disc_illum$x, disc_illum$y, col = "white")
```

StatHourglass *A ggplot2* stat *function to wrangle data for* geom_hourglass.

Description

Splits mapped x or y aesthetic from a continuous datetime into discrete date values on the mapped axis. The hour of day is mapped to the opposite axis.

Usage

```
StatHourglass

stat_hourglass(
  mapping = NULL,
  data = NULL,
  geom = "hourglass",
  position = "identity",
  show.legend = NA,
  inherit.aes = TRUE,
  hour_center = 0,
  na.rm = FALSE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by ggplot2::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. The hourglass stat and geom requires either the x axis or the y axis to be mapped. The mapped aesthetic will show the date of the variable, whereas the opposite axis will show the time of day.
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 ggplot2::ggplot(). Otherwise, a data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See ggplot2::fortify() for which variables will be created. The data should contain a column with datetime values (e.g., ?POSIXct)
geom	Can be used to overwrite the default connection between stat_hourglass and [geom_hourglass].
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.

`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()`.

`hour_center` The hour at which the time of day is centred. Default is 0, meaning midnight. -12 centres around noon of the preceding day, +12 centres around noon of the next day.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`...` Arguments passed as extra `params` to `ggplot2::layer()`

Format

An object of class `StatHourglass` (inherits from `Stat`, `ggproto`, `gg`) of length 7.

Value

Returns a `ggplot2::layer()` which can be added to a `ggplot2::ggplot()`

Author(s)

Pepijn de Vries

Examples

```
stat_hourglass()
```

`uses_dst` *Test if datetime object potentially uses daylight saving time*

Description

Function to check if a datetime object potentially uses daylight saving time. It is not the same as `lubridate::dst()`, which will determine if daylight saving time is set for the requested date.

Usage

```
uses_dst(x)
```

Arguments

x	A datetime object.

Value

Returns a `logical` value indicating of the time zone used by the datetime object potentially uses daylight saving time.

Author(s)

Pepijn de Vries

Examples

```
uses_dst(as.POSIXct("2020-03-29 02:00:00 CET", tz = "CET"))
uses_dst(as.POSIXct("2020-03-29 02:00:00 UTC", tz = "UTC"))
```

Index