Package 'gggibbous'

July 22, 2025

Title Moon Charts, a Pie Chart Alternative

Version 0.1.1

URL https://github.com/mnbram/gggibbous

BugReports https://github.com/mnbram/gggibbous/issues

Description Moon charts are like pie charts except that the proportions are shown as crescent or gibbous portions of a circle, like the lit and unlit portions of the moon. As such, they work best with only one or two groups. 'gggibbous' extends 'ggplot2' to allow for plotting multiple moon charts in a single panel and does not require a square coordinate system.

Depends ggplot2 (>= 3.2.1), R (>= 3.6.0)

Imports grid, scales (>= 1.0.0)

Suggests knitr, mapproj, maps, rmarkdown, testthat (>= 2.1.0), vdiffr (>= 0.3.1)

License GPL-3

LazyData true

RoxygenNote 7.1.1

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation no

Author Michael Bramson [aut, cre]

Maintainer Michael Bramson <mnbramson@gmail.com>

Repository CRAN

Date/Publication 2021-01-06 17:40:10 UTC

Contents

`dmeladh` Adh *allele frequencies in Australasian* Drosophila melanogaster

---

Description

This data set contains allele frequencies for the "fast" and "slow" variants of the enzyme alcohol dehydrogenase in Australasian (mostly Australian) populations of *Drosophila melanogaster*. The data are taken from Oakeshott, J.G., et al. 1982. Alcohol dehydrogenase and glycerol-3-phosphate dehydrogenase clines in *Drosophila melanogaster* on different continents. Evolution, 36(1): 86-96.

Usage

```
dmeladh
```

Format

A data frame with 34 rows and 6 variables:

Locality location of population sample

Latitude latitude of population sample

Longitude longitude of population sample

N number of samples

AdhF percent of samples with *Adh^F* allele, as an integer

AdhS percent of samples with *Adh^S* allele, as an integer

`draw_key_moon` *Moon key glyph for legends*

Description

Draws the legend key glyphs used in `geom_moon`.

Usage

```
draw_key_moon(data, params, size)

draw_key_moon_left(data, params, size)

draw_key_full_moon(data, params, size)
```

Arguments

data	A single row data frame containing the scaled aesthetics to display in this key
params	A list of additional parameters supplied to the geom.
size	Width and height of key in mm.

Details

draw_key_moon (the default in geom_moon) draws a gibbous moon filled from the right. draw_key_moon_left draws a crescent moon from the right. draw_key_full_moon draws a circle, which is very similar to draw_key_point in ggplot2, but the size is calculated slightly differently and the default aesthetics differ.

Value

A grid grob.

---

geom_moon *Moon charts*

Description

The moon geom is used to create moon charts, which are like pie charts except that the proportions are shown as crescent or gibbous portions of a circle, like the lit and unlit portions of the moon. As such, they work best with only one or two groups.

Usage

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

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)).
stat	The statistical transformation to use on the data for this layer, as a string.
position	Position adjustment, either as a string, or the result of a call to a position adjustment function.
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.
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().
...	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.

Details

geom_moon acts like geom_point in that multiple moons can be plotted on the same panel with x and y in the plot's coordinate system, but size determined independently of the coordinate system. This behavior also means that the moons will always be circular even if the coordinate system is not square.

In order to get a full circle with two complementary sections (a crescent and a gibbous moon), you need to plot two shapes: one with right = TRUE and one with right = FALSE, with ratio on the second one equal to 1 - ratio on the first.

Aesthetics

x and y are required aesthetics. size, fill, colo(u)r, alpha, stroke, and group aesthetics are understood as in other geoms. Two new aesthetics are also introduced: ratio and right. ratio controls the proportion of the moon to be plotted, from 0 to 1. right takes a boolean value to indicate whether the moon should be filled from the right or the left.

Examples

```
ggplot(
  data.frame(x = 1:5, y = 1, size = 1:5, ratio = 1:5 * 0.2),
  aes(x = x, y = y, size = size, ratio = ratio)
) +
```

```
  geom_moon()

# To make full moon charts, you need two calls to geom_moon(), one with
# right = TRUE and one with right = FALSE and ratio equal to 1 - ratio
# from the first one
ggplot(dmeladh) +
  geom_moon(
    x = 0.5, y = 0.5, fill = "forestgreen", color = "forestgreen",
    aes(ratio = AdhF / 100)
  ) +
  geom_moon(
    x = 0.5, y = 0.5, fill = "gold", color = "gold",
    aes(ratio = AdhS / 100), right = FALSE
  ) +
  facet_wrap(~Locality, ncol = 7)

# The same thing can be accomplished with a single call to geom_moon()
# using a "long" data frame with both frequencies if you set a grouping
# variable and set the `right` variable to a boolean column
dmeladh_long <- reshape(
  dmeladh,
  varying = c("AdhF", "AdhS"),
  v.names = "freq",
  timevar = "allele",
  times = c("AdhF", "AdhS"),
  idvar = c("Locality", "Latitude", "Longitude", "N"),
  direction = "long"
)
dmeladh_long$right <- rep(c(TRUE, FALSE), each = nrow(dmeladh))
ggplot(dmeladh_long) +
  geom_moon(
    x = 0.5, y = 0.5, key_glyph = draw_key_rect,
    aes(ratio = freq / 100, fill = allele, color = allele, right = right),
  ) +
  facet_wrap(~Locality, ncol = 7)

# Moon charts (and pie charts) are sometimes useful on maps when x and y
# cannot be used as aesthetic dimensions because they are already spatial
# dimensions. Overplotting needs to be considered carefully, however.
ggplot(
  subset(dmeladh, N > 200),
  aes(Longitude, Latitude)
) +
  geom_moon(aes(ratio = AdhF / 100), fill = "black") +
  geom_moon(aes(ratio = AdhS / 100), right = FALSE) +
  coord_fixed()
```

---

gggibbous *gggibbous: Moon charts, a pie chart alternative for two groups*

Description

Moon charts are like pie charts except that the proportions are shown as crescent or gibbous portions of a circle, like the lit and unlit portions of the moon. As such, they work best with only one or two groups. `gggibbous` extends `ggplot2` to allow for plotting multiple moon charts in a single panel and does not require a square coordinate system.

Details

The workhorse function is `geom_moon`, which adds a moon chart layer to a ggplot2 plot. The `draw_key_moon`, `draw_key_moon_left`, and `draw_key_full_moon` functions provides legend key glyphs for plots that use `geom_moon`. There are also functions for the raw `grid` grobs: `grid.moon` and `moonGrob`.

For more information, see the `gggibbous` vignette.

`lunardist` *Lunar distances and principal phases for 2019*

Description

This data set contains the distance from the Earth to the Moon for each day in 2019, as well as the dates (in UTC) of each occurrence of the four principal phases of the moon. The data are adapted from NASA.

Usage

```
lunardist
```

Format

A data frame with 365 rows and 3 variables:

date Date

distance Distance from the Earth to the Moon in kilometers

phase Principal phase of the moon (full, new, first quarter, third quarter) or NA if no principal phase on that date

Source

https://svs.gsfc.nasa.gov/4442

moonGrob *Draw a moon*

Description

Functions to create and draw crescent or gibbous moon shapes.

Usage

```
moonGrob(
  x,
  y,
  ratio = 0.25,
  right = TRUE,
  r = 10,
  angle = 0,
  default.units = "npc",
  size.units = "mm",
  ...
)

grid.moon(..., draw = TRUE)
```

Arguments

x	A numeric vector or unit object specifying x-locations.
y	A numeric vector or unit object specifying y-locations.
ratio	A numeric vector with values from 0 to 1 specifying the proportion of the moons to draw.
right	A boolean vector specifying whether the moon should be filled from the right (TRUE) or left (FALSE).
r	A numeric vector specifying the radii of the circles describing the moons.
angle	Not used.
default.units	A string indicating the default units to use if x, y, width, or height are only given as numeric vectors.
size.units	A string indicating the units to use for the radii of the moons.
...	Arguments passed on to grid::polygonGrob.
draw	A logical value indicating whether graphics output should be produced.

Details

Both functions create a moon grob (a graphical object describing a crescent or gibbous moon), but only grid.moon draws the moon (and then only if draw is TRUE).

These functions calculate a set of points describing the perimeters of the moons and pass these points on to grid::polygonGrob.

The units in `default.units` and `size.units` can be different; `grid` will add them together appropriately before drawing.

Value

A grob object.

Examples

```
grid::grid.newpage()
grid.moon(x = 1:3 * 0.25, y = rep(0.5, 3), ratio = 1:3 * 0.25, r = 10)
```

Index