Package: ggChernoff (via r-universe)

October 14, 2026

Title Chernoff Faces for 'ggplot2'

Version 0.3.0

Description Provides a Chernoff face geom for 'ggplot2'. Maps multivariate data to human-like faces. Inspired by Chernoff (1973) <doi:10.1080/01621459.1973.10482434>.

Depends R (>= 3.2.5)

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports ggplot2 (>= 2.2.0), grid, scales

RoxygenNote 7.2.1

URL https://github.com/Selbosh/ggChernoff

BugReports https://github.com/Selbosh/ggChernoff/issues

Repository https://selbosh.r-universe.dev

Date/Publication 2022-11-16 15:28:00 UTC

RemoteUrl https://github.com/selbosh/ggchernoff

RemoteRef HEAD

RemoteSha 3fa9bcec46a10a8621c68dda43c6e467efba8585

Contents

chernoffGrob 2
geom_chernoff 3
scale_brow_continuous 5
scale_eyes_continuous 6
scale_smile_continuous 7

Index **9**

chernoffGrob *Draw a smiley face*

Description

Uses Grid graphics to draw a face.

Usage

```
chernoffGrob(
  x = 0.5,
  y = 0.5,
  size = 1,
  colour = "black",
  fill = NA,
  alpha = 1,
  smile = 1,
  brow = NA,
  nose = FALSE,
  eyes = 1
)
```

Arguments

x	horizontal position
y	vertical position
size	area of the face
colour	colour of outlines and features
fill	fill colour
alpha	transparency, where 0 is transparent and 1 is opaque
smile	amount of smiling/frowning
brow	eyebrow angle, to represent anger or concern
nose	logical. Adds a nose to the face
eyes	distance between the eyes

Value

A grobTree object.

See Also

geom_chernoff

Examples

```
face <- chernoffGrob(.5, .5, size = 1e3, smile = -1, brow = 1, colour = 'navy', fill = 'lightblue')
grid::grid.newpage()
grid::grid.draw(face)
```

---

geom_chernoff *Chernoff faces in ggplot2*

Description

The Chernoff geom is used to create data visualisations in the shape of human-like faces. By mapping to the relevant aesthetics, faces can appear to vary in happiness, anger, size, colour and so on.

Usage

```
geom_chernoff(
  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), is combined with the default mapping at the top level of the plot. You only need to supply mapping if there isn't a mapping defined for the plot.
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.
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.

Value

A Geom layer object for use with ggplot2.

Aesthetics

geom_chernoff understands the following aesthetics (required aesthetics are in bold):

- x
- y
- colour
- fill
- size

The following aesthetics are unique to geom_chernoff:

- smile
- brow
- nose
- eyes

For details, see chernoffGrob.

References

Chernoff, H. (1973). The use of faces to represent points in *k*-dimensional space graphically. *Journal of the American Statistical Association, 68*(342), 361–368.

See Also

chernoffGrob

Examples

```
library(ggplot2)
ggplot(iris, aes(Sepal.Width, Sepal.Length, smile = Petal.Length, fill = Species)) +
  geom_chernoff()

ggplot(data.frame(x = 1:4,
                  y = c(3:1, 2.5),
                  z = factor(1:4),
                  w = rnorm(4),
                  n = c(rep(FALSE, 3), TRUE)
                  )) +
    aes(x, y, fill = z, size = x, nose = n, smile = w) +
    geom_chernoff()
```

---

scale_brow_continuous *Scales for angry eyebrows*

Description

scale_brow lets you customise how eyebrows are generated from your data. It also lets you tweak the appearance of legends and so on. By default, brow is set to NA, in which case no eyebrows will appear (see Examples).

Usage

```
scale_brow_continuous(..., range = c(-1, 1), midpoint = mean)

scale_brow(..., range = c(-1, 1), midpoint = mean)
```

Arguments

...	Other arguments passed onto continuous_scale to control name, limits, breaks, labels and so forth.
range	Output range of eyebrow angles. +1 corresponds to very angry and -1 corresponds to a worried look.
midpoint	A value or function of your data that will return level eyebrows, i.e. `|:-)`

Details

Use range to vary how angrily your maximum/minimum values are represented. Minima smaller than -1 and maxima greater than +1 are possible but might look odd! You can use midpoint to set a specific 'zero' value in your data or to have eyebrow angles represented as relative to average.

The function scale_brow is an alias of scale_brow_continuous. At some point we might also want to design a scale_brow_discrete, scale_brow_manual and so on.

Legends are a work in progress. In particular, size mappings might produce odd results.

Value

A Scale layer object for use with ggplot2.

See Also

geom_chernoff, scale_smile

Examples

```
library(ggplot2)
p <- ggplot(iris) +
    aes(Sepal.Width, Sepal.Length, fill = Species, brow = Sepal.Length) +
    geom_chernoff()
p
p + scale_brow_continuous(midpoint = min)
p + scale_brow_continuous(range = c(-.5, 2))

# Only show eyebrows if 'sad', otherwise hide them
usa <- data.frame(date = c(time(presidents)), rating = c(presidents))
ggplot(subset(usa, complete.cases(usa))) +
    aes(date, rating, smile = rating, fill = rating,
        brow = ifelse(rating < 50, rating, NA)) +
    geom_line() +
    geom_chernoff(show.legend = FALSE) +
    scale_brow(range = -1:0) +
    scale_fill_gradient(low = 'skyblue1', high = 'goldenrod1')
```

---

scale_eyes_continuous *Scales for eye separation*

Description

scale_eyes lets you customise how eye separation is determined from your data. It also lets you tweak the appearance of legends and so on.

Usage

```
scale_eyes_continuous(..., range = c(0.1, 2), midpoint = mean)

scale_eyes(..., range = c(0.1, 2), midpoint = mean)
```

Arguments

...	Other arguments passed onto continuous_scale to control name, limits, breaks, labels and so forth.
range	Output range of eye distances. 0 corresponds to a cyclops and +1 to a 'normal' distance.
midpoint	A value or function of your data that will return a 'normal' separation

Details

Use range to vary how happily/sadly your maximum/minimum values are represented. Minima smaller than -1 and maxima greater than +1 are possible but might look odd! You can use midpoint to set a specific 'zero' value in your data or to have eye width represented as relative to average.

The function scale_eyes is an alias of scale_eyes_continuous.

Legends are a work in progress. In particular, size mappings might produce odd results.

Value

A Scale layer object for use with ggplot2.

See Also

geom_chernoff, scale_brow, scale_smile

Examples

```
library(ggplot2)
p <- ggplot(iris) +
    aes(Sepal.Width, Sepal.Length, fill = Species, eyes = Sepal.Length) +
    geom_chernoff()
p
p + scale_eyes_continuous(midpoint = min)
p + scale_eyes_continuous(range = c(0, 2))
```

---

scale_smile_continuous

Scales for smiling and frowning

Description

scale_smile lets you customise how smiles are generated from your data. It also lets you tweak the appearance of legends and so on.

Usage

```
scale_smile_continuous(..., range = c(-1, 1), midpoint = mean)

scale_smile(..., range = c(-1, 1), midpoint = mean)
```

Arguments

...	Other arguments passed onto continuous_scale to control name, limits, breaks, labels and so forth.
range	Output range of smiles. +1 corresponds to a full smile and -1 corresponds to a full frown.
midpoint	A value or function of your data that will return a neutral/straight face, i.e. :-|

Details

Use range to vary how happily/sadly your maximum/minimum values are represented. Minima smaller than -1 and maxima greater than +1 are possible but might look odd! You can use midpoint to set a specific 'zero' value in your data or to have smiles represented as relative to average.

The function scale_smile is an alias of scale_smile_continuous. At some point we might also want to design a scale_smile_discrete, scale_smile_manual and so on.

Legends are a work in progress. In particular, size mappings might produce odd results.

Value

A Scale layer object for use with ggplot2.

See Also

geom_chernoff, scale_brow

Examples

```
library(ggplot2)
p <- ggplot(iris) +
    aes(Sepal.Width, Sepal.Length, fill = Species, smile = Sepal.Length) +
    geom_chernoff()
p
p + scale_smile_continuous(midpoint = min)
p + scale_smile_continuous(range = c(-.5, 2))
```

Index

aes, *3*
aes_, *3*

borders(), *4*

chernoffGrob, 2, *4*
continuous_scale, *5–7*

fortify(), *3*

Geom, *4*
geom_chernoff, *2*, 3, *6–8*
ggplot(), *3*
Grid, *2*
grobTree, *2*

layer(), *4*

Scale, *6–8*
scale_brow, *7, 8*
scale_brow *(*scale_brow_continuous*)*, 5
scale_brow_continuous, 5
scale_eyes *(*scale_eyes_continuous*)*, 6
scale_eyes_continuous, 6
scale_smile, *6, 7*
scale_smile *(*scale_smile_continuous*)*, 7
scale_smile_continuous, 7