Package 'ggdistribute'

November 15, 2018

Title A 'ggplot2' Extension for Plotting Unimodal Distributions

Version 1.0.3

Date 2018-11-15

Description The 'ggdistribute' package is an extension for plotting posterior or other types of unimodal distributions that require overlaying information about a distribution's intervals. It makes use of the 'ggproto' system to extend 'ggplot2', providing additional ``geoms'', ``stats'', and ``positions.'' The extensions integrate with existing 'ggplot2' layer elements.

URL https://github.com/iamamutt/ggdistribute

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports data.table, ggplot2 (>= 3.0.0), tibble, magrittr, grDevices, dplyr

Suggests knitr, testthat, rmarkdown, viridisLite, extrafont

VignetteBuilder knitr

RoxygenNote 6.1.1

NeedsCompilation no

Author Joseph M. Burling [aut, cre]

Maintainer Joseph M. Burling <josephburling@gmail.com>

Repository CRAN

Date/Publication 2018-11-15 19:10:04 UTC

R topics documented:

`ggdistribute-package` *A 'ggplot2' Extension for Plotting Unimodal Distributions*

Description

The `ggdistribute` package is an extension for plotting posterior or other types of unimodal distributions that require overlaying information about a distribution's intervals. It makes use of the `ggproto` system to extend `ggplot2`, providing additional "geoms", "stats", and "positions." The extensions integrate with existing `ggplot2` layer elements.

Details

Displaying the distributions relies heavily on stacking distributions using position_spread, which may not align with other geoms.

See Also

ggplot2::geom_density, ggplot2::position_dodge

See `help(package = "ggdistribute")` for a list of functions.

View vignettes with `browseVignettes(package ="ggdistribute")`.

annotate_corner *Write text to one of four corners of a plot*

Description

Write text to one of four corners of a plot

Usage

```
annotate_corner(text, pos = "tl", geom = c("text", "label"), ...)
```

Arguments

text	character string
pos	character of "tl", "tr", "bl", "br" to indicate position (you may also use the full names, "bottomleft", etc...)
geom	"text" or "label" geoms
...	additional options passed to ggplot2::geom_label

Details

If multiple panels, will write to all panels.

Value

ggplot layer

Examples

```
posterior_plot() + annotate_corner("Hi.", "bottomright")
```

cmode *Mode from counting frequency*

Description

Finds the most frequent value from a vector of discrete values

Usage

```
cmode(x)
```

Arguments

x	an integer vector

Value

scalar integer value

Examples

```
cmode(rpois(1000, 20))
```

`data_normal_sample` *Testing dataset of grouped Normal distributions*

Description

Testing dataset of grouped Normal distributions

Usage

```
data_normal_sample(mu = c(-0.5, 4), n = 500L, sd_range = c(0.6, 1.4),
  seed = 19850519)
```

Arguments

`mu`	Means for each group. A numeric vector with the length corresponding to the number of groups.
`n`	Number of observations for each group. Length 1 integer.
`sd_range`	The min and max to use for standard deviations. Length 2
`seed`	A seed value to generate the same sample. numeric vector.

Value

A data.frame with the following variables: `Group`, `Condition`, `value`.

Examples

```
data_normal_sample(0, 100)
```

dmode *Mode from density estimation*

Description

Finds the mode using the density function and then obtains the maximum value.

Usage

```
dmode(x, adjust = 1.5)
```

Arguments

x	Value vector. Numeric or integers.
adjust	Bandwidth adjustment. See density.

Examples

```
x <- rchisq(1000, 3)
hist(x, br=50)
abline(v = dmode(x), col = "red")
abline(v = median(x), col = "green")
abline(v = mean(x), col = "blue")
```

example_plot *Print an example of the package functions*

Description

Print an example of the package functions

Usage

```
example_plot()
```

geom_posterior *Geom for plotting posterior distributions*

Description

This geom may be used to plot the density of any type of numeric variable but the displayed intervals may not be informative if the distribution deviates too much from a unimodal, symmetric distribution.

Usage

```
geom_posterior(mapping = NULL, data = NULL, stat = "DensityCI",
  position = "spread", ..., draw_ci = TRUE, draw_sd = TRUE,
  midline = "#767698", brighten = TRUE, mirror = FALSE,
  interp_thresh = NULL, na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE)

stat_density_ci(mapping = NULL, data = NULL, geom = "Posterior",
  position = "spread", ..., center_stat = "median", ci_width = 0.9,
  interval_type = "ci", bw = "nrd0", adjust = 1,
  kernel = "gaussian", cut = 1, n = 1024, trim = 0.01,
  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.
stat	Used to override the default connection between geom_posterior and stat_density_ci.
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.
draw_ci	*geom*. Toggles drawing of the confidence interval lines and segments.
draw_sd	*geom*. Toggles drawing of the standard deviation interval lines and segments.

midline	*geom*. Color of the vertical, center line. Set to NA to omit the line.
brighten	*geom*. Numeric adjustments to the fill color. A value above 1 increases brightness, below decreases. Should be of length 1 or 5, otherwise values are recycled
mirror	*geom*. Show standard densities (mirror=FALSE) or horizontal violin densities (mirror=TRUE).
interp_thresh	*geom*. If the number of samples used to estimate the density is low, this will result in gaps between segments. This argument decides to interpolate points based on gap proportion for a segment.
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().
geom	Use to override the default connection between geom_posterior and stat_density_ci
center_stat	*stat*. character string of method to compute the distribution's central tendency, such as "median", "mean", or "mode".
ci_width	*stat*. Width of the distribution's confidence/highest density interval, e.g., 0.95
interval_type	*stat*. method of computing the interval, either "hdi" or "ci"
bw	The smoothing bandwidth to be used. If numeric, the standard deviation of the smoothing kernel. If character, a rule to choose the bandwidth, as listed in stats::bandwidth. If the bandwidth character starts wit a "." (e.g., ".nrd0"), then the average bandwidth will be calculated among all groups in a panel and used for each density estimate.
adjust	A multiplicate bandwidth adjustment. This makes it possible to adjust the bandwidth while still using the a bandwidth estimator. For example, adjust = 1/2 means use half of the default bandwidth.
kernel	Kernel. See list of available kernels in density().
cut	The values to use for the start and end of the density estimation are cut bandwidths (e.g., 0.5*bw) *beyond* the extremes of the data. This allows the estimated density to drop to approximately zero at the extremes.
n	number of equally spaced points at which the density is to be estimated, should be a power of two, see density() for details
trim	If a value between 0 and 1 is given, trim the tails of x by some proportion according to trim. If NULL or NA, don't trim the tails. See trim_ends().

Functions

- geom_posterior: geom_posterior Posterior Geom
- stat_density_ci: stat_density_ci Computes a distribution density and confidence intervals for each group

Aesthetics

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

- **x**
- **y**
- xmin
- xmax
- alpha - colour
- fill
- group
- linetype
- size
- weight

Computed Variables

stat_density_ci:

- density: density estimate from stats::density
- scaled: Normalized density values: `density / max(density)`
- count: Number of samples at density level: `(density / sum(density)) * n`
- xmin: minimum value of `x` from the data
- cil: cil cutoff value based on `ci_width`
- sdl: central value minus 1 sd of `x`
- mid: value of central tendency
- sdu: central value plus 1 sd of `x`
- ciu: ciu cutoff value based on `ci_width`
- xmax: maximum value of `x` from the data

position_spread

- ymin: minimum value of `y` for each group in a panel.
- ymax: maximum value of `y` for each group in a panel.

Examples

```
library(ggplot2)

x <- data_normal_sample(mu = c(-1, 0, 1), n = 500)

p <- ggplot(x, aes(x = value))

p + geom_posterior()
```

```
p + geom_posterior(aes(y = Condition))

p + geom_posterior(aes(y = GroupScore, fill = Condition))

p + geom_posterior(aes(y = GroupScore, fill = Group),
brighten = c(1.3, 0, -1.3),
position = position_spread(
height=0.5,
padding = 0))
```

---

hdi *Highest density interval*

Description

This is a function that will calculate the highest density interval from a posterior sample.

Usage

```
hdi(x, prob = 0.95, warn = TRUE)
```

Arguments

x	Numeric vector of a distribution of data, typically a posterior sample
prob	Width of the interval from some distribution. Defaults to 0.95.
warn	Option to turn off multiple sample warning message Must be in the range of [0,1].

Details

The default is to calcualte the highest 95 percent interval. It can be used with any numeric vector instead of having to use one of the specific MCMC classes. This function has been adapted from John K. Kruschke (2011). Doing Bayesian Data Analaysis: A Tutorial with R and BUGS.

Value

Numeric range

Examples

```
x <- qnorm(seq(1e-04, .9999, length.out=1001))
hdi_95 <- hdi(x, .95)
hdi_50 <- hdi(x, .50)

hist(x, br=50)
abline(v=hdi_95, col="red")
abline(v=hdi_50, col="green")
```

```
x <- exp(seq(pi * (1 - (1/16)), pi, len = 1000))
x <- c(x, rev(x)[-1])
x <- c(-x, x)
plot(sort(x), type="l")
plot(density(x, adjust=0.25))
abline(v=hdi(x, p=.49), col=2)
abline(v=hdi(x, p=.50), col=3)
```

`label_plot` *Add labels to existing plot*

Description

Uses a normalized coordinate system to add text anywhere on the current plot.

Usage

```
label_plot(labels, x, y, g = list(fontsize = 14, fontface = "bold"), ...)
```

Arguments

`labels`	character vector of labels to use
`x`	horz positions of items in `labels`
`y`	vert positions of items in `labels`
`g`	list of options passed to grid::gpar
`...`	optional args passed to grid::grid.text

Value

NULL. prints to current graphics device.

See Also

grid::grid.text, grid::gpar

Examples

```
example_plot()
label_plot(c('a label', 'another one'), c(.1, .9), c(.95, .1))

# use extra options from grid::grid.text
label_plot('last one', 0.5, 0.5, just='center')
```

`mejr_geom_defaults` *Setup defaults for specific geoms*

Description

Setup defaults for specific geoms

Usage

```
mejr_geom_defaults(base_size = 11, black = 51, gray = 214, lty = 3,
  lwd = base_size/20, cex = base_size/9, stroke = base_size * 0.05,
  alpha = 0.5, pch = 21, txt = base_size/4, reset = FALSE)
```

Arguments

`base_size`	base font size
`black`	Values from 0 to 255, indicating the darkest line and text colors (255).
`gray`	gray color value (0-255)
`lty`	linetype
`lwd`	linewidth
`cex`	point size
`stroke`	stroke width
`alpha`	alpha
`pch`	point shape
`txt`	text size
`reset`	reset all back to default

Examples

```
# This will change the point size and shape for
#  all geoms in which GeomPoint inherits from.
mejr_geom_defaults(cex = 1.1, pch = 19)

# Reset defaults back to their original state.
mejr_geom_defaults(reset=TRUE)
```

mejr_palette *theme_mejr color mejr_palette*

Description

theme_mejr color mejr_palette

Usage

```
mejr_palette()
```

Value

list

Examples

```
mejr_palette()
```

position_spread *Spread Overlapping Grobs Spread overlapping groups by shrinking them to fit within the data's* y *range.*

Description

Spread Overlapping Grobs Spread overlapping groups by shrinking them to fit within the data's y range.

Usage

```
position_spread(height = NULL, reverse = FALSE, padding = 0.2)
```

Arguments

height	One of total (default), panel, single, a numeric scalar value to give all geoms equal space, or a numeric vector the length of panels*groups for manually specifying the height of each group.
reverse	Reverse the order of segments within overlapping y ranges.
padding	Multiple of height. Will shrink/enlarge groups to fit within a region.

Examples

```
library(ggplot2)

x <- data.frame(y = rnorm(1000), x="", myGroup=sample(1:3, 1000, TRUE))

ggplot(x, aes(x, y))+
geom_point(aes(group=myGroup), position=position_spread(height = 0.5))
```

posterior_plot *Print a small example plot with geom_posterior*

Description

Print a small example plot with geom_posterior

Usage

```
posterior_plot(data, x, y = "..count..", ...)
```

Arguments

data	A dataset to use, called at the top layer within ggplot.
x	A character string of the x axis variable name (e.g., values making up the distribution).
y	A grouping variable for generating groups of distributions. Defaults to ..count.. for no groups and displays the density as counts of the number of samples for the value of x.
...	Additional arguments passed to geom_posterior().

Value

Object of class gg, ggplot.

Examples

```
# Generate a basic example plot if no data is specified.
posterior_plot()
```

post_int *Posterior intervals*

Description

Returns cutoff points from a posterior distribution

Usage

```
post_int(x, mid = c("median", "mean", "mode"), int = c("hdi", "ci"),
  widths = c(0.5, 0.95), adj = 1.5, rope = NULL, warn = FALSE)
```

Arguments

x	Vector of numeric values. Typically a posterior sample.
mid	Central tendency estimator. Defaults to "median". Other options include "mean" and "mode".
int	interval type, either "hdi" or "ci"
widths	interval widths
adj	Bandwidth adjustment used only with the "mode" estimator. See dmode.
rope	Region of practical equivalence. Check how much of the distribution is within rope value.
warn	Turn off warning for flat intervals found (multiple possible values)

Value

data.table

Examples

```
x <- rpois(5000, 15)
ints <- post_int(x, warn = FALSE)
hist(x, br=50)
abline(v=ints$c, col="cyan")
abline(v=ints[, c("l.wide", "r.wide")], col="magenta")

post_int(x, "median", warn = FALSE)
post_int(x, "mean", warn = FALSE)
post_int(x, "mode", adj=2, rope = c(14, 16), warn = FALSE)
```

---

scale_add *scale and add*

Description

scale and add

Usage

```
scale_add(base_size, amount = 1, adj = 0)
```

Arguments

base_size	start value
amount	multiple by
adj	add after

Value

numeric

show_colors *plot and show hex values of colors*

Description

plot and show hex values of colors

Usage

```
show_colors(colors, show.legend = TRUE, ncols = NULL, alpha = NA)
```

Arguments

colors	character vector of hex value colors
show.legend	show the legend with hex values (logical)
ncols	number of columns in the plot
alpha	set alpha level for all colors

Value

A plot with the index of the color in the tile

Examples

```
show_colors(mejr_palette())
show_colors(topo.colors(25))
show_colors(cm.colors(64), FALSE)
show_colors(viridisLite::viridis(15), alpha = .8)
```

sre *Raw SRE dataset*

Description

Raw SRE dataset

Usage

```
sre

sre_data(n = 1000, seed = 19850519)
```

Arguments

n	number of samplers per effect and contrast
seed	set.seed number

Format

Dataset is an object of class tibble::tibble.

Value

A dataset of tibble::tibble.

Functions

- `sre_data`: create samples from sre data

See Also

`sre_data()`

Examples

```
sre

sre_data()
```

---

test_mejr_theme *Test theme by printing plots to pdf and viewport*

Description

Test theme by printing plots to pdf and viewport

Usage

```
test_mejr_theme(w = 6.875, h = 4.5, eplot = list(), mejr = list(),
  gg = theme(), print = TRUE, with_test_theme = FALSE, dir)
```

Arguments

`w`	pdf width (inches)
`h`	pdf height (inches)
`eplot`	list of options for `example_plot`
`mejr`	list of options for `theme_mejr(...)`
`gg`	further theme customization with ggplot::theme(...)
`print`	show the `eplot` plot
`with_test_theme`	also print the same plot using `ggplot2::theme_test`
`dir`	directory where to save temp pdf files, such as tempdir()

theme_mejr *Custom ggplot2 theme*

Description

A complete, minimal theme to be used with the ggplot2 package

Usage

```
theme_mejr(base_size = 11,
  base_family = getOption("ggdistribute.font"), black = 67,
  margin_add = 2, debug = FALSE, FUN = NULL, ...)
```

Arguments

base_size	base font size
base_family	base font family
black	Values from 0 to 255, indicating the darkest line and text colors (255).
margin_add	additive adjustment of margin spacing and tick length (in "pt" units). May be positive or negative.
debug	Add debug info to text.
FUN	Call a function before returning the theme elements.
...	Arguments passed to FUN

Details

You can use theme_update to change some aspect of this theme after using theme_set.

See Also

mejr_geom_defaults, ggplot2::theme_update, ggplot2::theme_set

Examples

```
library(ggplot2)

theme_set(theme_mejr(debug = TRUE))
example_plot()

theme_set(theme_mejr())
theme_update(axis.text = element_blank()) # any updates can go here
example_plot()
```

trim_ends *Trim extreme values at each end of a vector.*

Description

Trim extreme values at each end of a vector.

Usage

```
trim_ends(x, trim = 0.05, na.rm = TRUE)
```

Arguments

x	A numeric vector
trim	Proportion of vector length to trim. Must be between 0 and 1. E.g., a value 0.05 (default) trims 2.5% off each end of a sorted vector.
na.rm	omit NA values. May result in different size vector.

Value

A numeric vector in the original order of x, but with trimmed values as NA if na.rm=TRUE or with these values removed if FALSE (which will result in a different sized vector from the input).

Examples

```
x <- rgamma(10000, 1, 1)
range(x)
length(x)     # <- 10000
sum(is.na(x)) # <- 0

t <- trim_ends(x, trim = 0.1)
range(t)
length(t)     # <- 9000
sum(is.na(t)) # <- 0

t <- trim_ends(x, 0.1, na.rm = FALSE)
range(t, na.rm = TRUE)
length(t)     # <- 10000
sum(is.na(t)) # <- 1000
```

Index