Package 'ggrain'

March 3, 2023

Title A Raincloud Geom for 'ggplot2'

Version 0.0.3

Description
The 'geom_rain()' function adds different geoms together using 'ggplot2' to create raincloud plots.

License MIT + file LICENSE

Encoding UTF-8

Depends ggplot2 (>= 3.4.0), R (>= 3.4.0)

Imports grid, gghalves, ggpp, rlang, vctrs (>= 0.5.0), cli

RoxygenNote 7.2.1

URL https://github.com/njudd/ggrain

BugReports https://github.com/njudd/ggrain/issues

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Nicholas Judd [aut, cre] (<https://orcid.org/0000-0002-0196-9871>),
Jordy van Langen [aut] (<https://orcid.org/0000-0003-2504-2381>),
Micah Allen [ctb] (<https://orcid.org/0000-0001-9399-4179>),
Rogier Kievit [aut] (<https://orcid.org/0000-0003-0700-4568>)

Maintainer Nicholas Judd <nickkjudd@gmail.com>

Repository CRAN

Date/Publication 2023-03-03 12:50:02 UTC

R topics documented:

geom_paired_raincloud 2
geom_rain 3

Index 7

`geom_paired_raincloud` *Paired raincloud plot*

Description

Taking from https://raw.githubusercontent.com/yjunechoe/geom_paired_raincloud/master/geom_paired_raincloud.R on 30-10-22 attribution to https://yjunechoe.github.io/

Usage

```
geom_paired_raincloud(
  mapping = NULL,
  data = NULL,
  stat = "ydensity",
  position = "dodge",
  trim = TRUE,
  scale = "area",
  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.
`trim`	If `TRUE` (default), trim the tails of the violins to the range of the data. If `FALSE`, don't trim the tails.

scale	if "area" (default), all violins have the same area (before trimming the tails). If "count", areas are scaled proportionally to the number of observations. If "width", all violins have the same maximum width.
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

Create a paired raincloud plot (useful for visualizing difference between experimental conditions tested on the same subjects or items).

Adopted from the geom_violinhalf() source code from the see package

See Also

https://github.com/easystats/see/blob/master/R/geom_violinhalf.R

Examples

```
library(ggplot2)
```

---

geom_rain *Raincloud Plots*

Description

This function displays individual data points, a boxplot and half a violin plot. It also has the option to connect data points with lines across groups by specifying an id to connect by. Lastly, if desired one can color the dots based of another variable.

Usage

```
geom_rain(
  mapping = NULL,
  data = NULL,
  inherit.aes = TRUE,
  id.long.var = NULL,
  cov = NULL,
  rain.side = NULL,
  likert = FALSE,
```

```
  seed = 42,
  ...,
  point.args = rlang::list2(...),
 point.args.pos = rlang::list2(position = position_jitter(width = 0.04, height = 0, seed
    = seed)),
  line.args = rlang::list2(alpha = 0.2, ...),
 line.args.pos = rlang::list2(position = position_jitter(width = 0.04, height = 0, seed
    = seed), ),
  boxplot.args = rlang::list2(outlier.shape = NA, ...),
 boxplot.args.pos = rlang::list2(width = 0.05, position = position_nudge(x = 0.1), ),
  violin.args = rlang::list2(...),
 violin.args.pos = rlang::list2(side = "r", width = 0.7, position = position_nudge(x =
    0.15), )
)
```

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)).
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().
id.long.var	A group to connect the lines by - must be a string (e.g., "id").
cov	A covariate to color the dots by - must be as a string (e.g., "cov")
rain.side	How you want the raincloud displayed, right ("r"), left ("l") or flanking ("f"), for a 1-by-1 flanking raincloud use ("f1x1") and for a 2-by-2 use ("f2x2").
likert	Currently developing, right now just addes y-jitter.
seed	For the jittering in point & line to match.
...	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.
point.args	A list of args for the dots
point.args.pos	A list of positional args for the points
line.args	A list of args for the lines, you need to specify a group to connect them with id.long.var

`line.args.pos` A list of positional args for the lines

`boxplot.args` A list of args for the boxplot

`boxplot.args.pos` A list of positional args for the boxplot

`violin.args` A list of args for the violin

`violin.args.pos` A list of positional args for the violin

Value

Returns a list of three environments to be used with the 'ggplot()' function in the 'ggplot2' package.

If the id.long.var argument is used the output will be a list of 4 environments.

These 4 environments have a similar structure to 'geom_boxplot()', 'geom_violin()', 'geom_point()' and 'geom_line()' from 'ggplot2'. need library(rlang) need library(ggplot2) depends = ggplot2

References

Allen, M., Poggiali, D., Whitaker, K., Marshall, T. R., van Langen, J., & Kievit, R. A. Raincloud plots: a multi-platform tool for robust data visualization Wellcome Open Research 2021, 4:63. https://doi.org/10.12688/wellcomeopenres.15191.2

Examples

```
e1 <- ggplot(iris, aes(Species, Sepal.Width, fill = Species))
e1 + geom_rain()

# x must be the discrete variable
# orinetation can be changed with coord_flip()
e1 + geom_rain(alpha = .5) + coord_flip()

# we can color the dots by a covariate
e1 + geom_rain(cov = "Sepal.Length")

# we can edit elements individually
e1 + geom_rain(violin.args = list(alpha = .3, color = NA))

# we can flip them
e1 + geom_rain(rain.side = 'l')
# and move them
e1 +
geom_rain(boxplot.args.pos = list(width = .1, position = position_nudge(x = -.2)))

# they also work longitudinally
e2 <- ggplot(sleep, aes(group, extra, fill = group))
e2 + geom_rain(id.long.var = "ID")

# we can add groups
sleep_dat <- cbind(sleep, data.frame(sex = c(rep("male", 5),
rep("female", 5), rep("male", 5), rep("female", 5))))
e3 <- ggplot(sleep_dat, aes(group, extra, fill = sex))
```

```
e3 + geom_rain(alpha = .6)

# add likert example
e4 <- ggplot(mpg, aes(1, hwy, fill = manufacturer))
e4 + geom_rain(likert= TRUE)

# lets make it look nicer
e4 + geom_rain(likert= TRUE,
 boxplot.args.pos = list(position = ggpp::position_dodgenudge(x = .095), width = .1),
 violin.args = list(color = NA, alpha = .5))
```

Index

aes(), *2*, *4*

borders(), *3*, *4*

fortify(), *2*, *4*

geom_paired_raincloud, 2
geom_rain, 3
ggplot(), *2*, *4*

layer(), *3*, *4*