Package: reaborn (via r-universe)

October 6, 2026

Title Statistical Data Visualization, the 'seaborn' Way

Version 1.0.2

Date 2026-07-30

Description An 'R' port of the 'Python' 'seaborn' library. 'reaborn' mirrors the 'seaborn' public function API (identical function names, argument names, and defaults) and renders visually indistinguishable plots using 'ggplot2'. Because every 'reaborn' plot is a 'ggplot' object, it can be extended with the full 'ggplot2' grammar of graphics.

License BSD_3_clause + file LICENSE

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 8.0.0

Depends R (>= 4.1)

Imports ggplot2 (>= 4.0.0), grDevices, grid, rlang, scales, stats

Suggests colorspace, ggbeeswarm, ggdendro, jsonlite, knitr, MASS, patchwork, ragg, rmarkdown, testthat (>= 3.0.0), vdiffr

Config/testthat/edition 3

VignetteBuilder knitr

URL https://reaborn.org, https://github.com/shawntz/reaborn

BugReports https://github.com/shawntz/reaborn/issues

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

Date/Publication 2026-08-06 15:55:39 UTC

RemoteUrl https://github.com/shawntz/reaborn

RemoteRef HEAD

RemoteSha 3e694740a549411a1ef1654f34f33116a52f052a

Contents

axes_style . . . 3
barplot . . . 3
boxenplot . . . 5
boxplot . . . 6
catplot . . . 8
clustermap . . . 9
color_palette . . . 10
countplot . . . 12
desaturate . . . 14
despine . . . 14
displot . . . 15
dogplot . . . 17
ecdfplot . . . 17
FacetGrid . . . 18
heatmap . . . 19
histplot . . . 21
jointplot . . . 23
kdeplot . . . 24
lineplot . . . 25
lmplot . . . 27
load_dataset . . . 29
move_legend . . . 29
pairplot . . . 30
palplot . . . 31
plotting_context . . . 32
pointplot . . . 32
python-literals . . . 34
regplot . . . 34
relplot . . . 36
reset_defaults . . . 38
residplot . . . 38
rugplot . . . 39
saturate . . . 40
scatterplot . . . 41
set_color_codes . . . 42
set_hls_values . . . 42
set_palette . . . 43
set_theme . . . 43
sns-aliases . . . 44
stripplot . . . 45
swarmplot . . . 46
theme_seaborn . . . 48
violinplot . . . 48

Index **51**

axes_style — *Get the parameters that control the general style of the plots*

Description

Port of `seaborn.axes_style`. Returns the resolved style definition.

Usage

```
axes_style(style = NULL, rc = NULL)

set_style(style = NULL, rc = NULL)
```

Arguments

`style`	One of `"darkgrid"`, `"whitegrid"`, `"dark"`, `"white"`, `"ticks"`.
`rc`	Optional named list of overrides.

Value

A named list describing the style.

barplot — *Show point estimates and errors as bars*

Description

Port of `seaborn.barplot`. Bar heights are an aggregate (default mean) with a bootstrap CI error bar. Returns a reaborn_plot.

Usage

```
barplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  estimator = "mean",
  errorbar = list("ci", 95),
  n_boot = 1000,
  seed = NULL,
  units = NULL,
  weights = NULL,
```

```
  orient = NULL,
  color = NULL,
  palette = NULL,
  saturation = 0.75,
  fill = TRUE,
  width = 0.8,
  dodge = "auto",
  gap = 0,
  capsize = 0,
  err_kws = NULL,
  legend = "auto",
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables; the categorical one defines the groups.
`hue`	Grouping variable for color (dodged).
`order`, `hue_order`	Level orderings.
`estimator`, `errorbar`, `n_boot`, `seed`	Aggregation + error settings.
`units`, `weights`	Bootstrap structure / weights (units reserved).
`orient`	`"v"`, `"h"`, or `NULL` to infer.
`color`, `palette`, `saturation`, `fill`	Color controls (saturation default 0.75).
`width`, `gap`	Box width and gap between dodged boxes.
`dodge`	How to dodge bars by hue (`"auto"`, `TRUE`, or `FALSE`).
`capsize`	Width of the error bar caps.
`err_kws`	Passed to the error bar geom.
`legend`	Legend control.
`.facet_vars`	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
`...`	Passed to the bar geom.

Value

A `reaborn_plot`.

Examples

```
penguins <- load_dataset("penguins")
barplot(data = penguins, x = "island", y = "body_mass_g")
barplot(data = penguins, x = "island", y = "body_mass_g", hue = "sex")
```

boxenplot *Draw an enhanced box plot for larger datasets*

Description

Port of `seaborn.boxenplot` (letter-value plot). Returns a reaborn_plot.

Usage

```
boxenplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  orient = NULL,
  color = NULL,
  palette = NULL,
  saturation = 0.75,
  fill = TRUE,
  width = 0.8,
  gap = 0,
  linewidth = NULL,
  linecolor = NULL,
  width_method = "exponential",
  k_depth = "tukey",
  outlier_prop = 0.007,
  trust_alpha = 0.05,
  showfliers = TRUE,
  legend = "auto",
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables; the categorical one defines the groups.
`hue`	Grouping variable for color (dodged).
`order`, `hue_order`	Level orderings.
`orient`	`"v"`, `"h"`, or `NULL` to infer.
`color`, `palette`, `saturation`, `fill`	Color controls (saturation default 0.75).

width, gap	Box width and gap between dodged boxes.
linewidth	Box outline width.
linecolor	Box outline color.
width_method	"exponential", "linear", or "area".
k_depth	"tukey", "proportion", "trustworthy", "full", or an int.
outlier_prop, trust_alpha	Tail-rule parameters.
showfliers	Draw outlier points.
legend	Legend control.
.facet_vars	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
...	Passed to ggplot2::geom_boxplot.

Value

A reaborn_plot.

Examples

```
tips <- load_dataset("tips")
boxenplot(data = tips, x = "day", y = "total_bill")
boxenplot(data = tips, x = "day", y = "total_bill", hue = "smoker")
```

---

boxplot *Draw a box plot*

Description

Port of seaborn.boxplot. Returns a reaborn_plot.

Usage

```
boxplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  orient = NULL,
  color = NULL,
  palette = NULL,
  saturation = 0.75,
  fill = TRUE,
  dodge = "auto",
```

```
  width = 0.8,
  gap = 0,
  whis = 1.5,
  linecolor = "auto",
  linewidth = NULL,
  fliersize = NULL,
  legend = "auto",
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables; the categorical one defines the groups.
`hue`	Grouping variable for color (dodged).
`order`, `hue_order`	Level orderings.
`orient`	`"v"`, `"h"`, or `NULL` to infer.
`color`, `palette`, `saturation`, `fill`	Color controls (saturation default 0.75).
`dodge`	How to dodge boxes by hue (`"auto"`, `TRUE`, or `FALSE`).
`width`, `gap`	Box width and gap between dodged boxes.
`whis`	Whisker length in IQR units (default 1.5).
`linecolor`, `linewidth`, `fliersize`	Line and outlier styling.
`legend`	Legend control.
`.facet_vars`	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
`...`	Passed to ggplot2::geom_boxplot.

Value

A `reaborn_plot`.

Examples

```
tips <- load_dataset("tips")
boxplot(data = tips, x = "day", y = "total_bill")
boxplot(data = tips, x = "day", y = "total_bill", hue = "smoker")
```

`catplot` *Figure-level interface for categorical plots*

Description

Port of `seaborn.catplot`. Dispatches to stripplot (`kind = "strip"`), boxplot, barplot, pointplot, or countplot and adds row/col faceting. Returns a faceted reaborn_plot.

Usage

```
catplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  row = NULL,
  col = NULL,
  kind = "strip",
  estimator = "mean",
  errorbar = list("ci", 95),
  n_boot = 1000,
  seed = NULL,
  units = NULL,
  weights = NULL,
  order = NULL,
  hue_order = NULL,
  row_order = NULL,
  col_order = NULL,
  col_wrap = NULL,
  height = 5,
  aspect = 1,
  orient = NULL,
  color = NULL,
  palette = NULL,
  legend = "auto",
  facet_kws = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables; the categorical one defines the groups.
`hue`	Grouping variable for color (dodged).
`row`, `col`, `col_wrap`, `row_order`, `col_order`	Faceting controls.

kind	One of "strip", "box", "bar", "point", "count".
estimator, errorbar, n_boot, seed	Aggregation settings (bar/point).
units	Unit grouping for bootstrap (bar/point kinds).
weights	Observation weights (bar/point kinds).
order, hue_order	Level orderings.
height, aspect	Facet sizing (stored as attributes).
orient	"v", "h", or NULL to infer.
color	Single color override.
palette	Palette for the hue mapping.
legend	Legend control.
facet_kws	Reserved for compatibility.
...	Passed to the underlying plotter.

Value

A reaborn_plot.

Examples

```
tips <- load_dataset("tips")
catplot(data = tips, x = "day", y = "total_bill", hue = "smoker",
        kind = "box", col = "time")

catplot(data = tips, x = "day", y = "total_bill", kind = "bar", col = "time")
```

---

clustermap *Plot a hierarchically-clustered heatmap*

Description

Port of seaborn.clustermap. Reorders rows/columns by hierarchical clustering and draws dendrograms alongside the heatmap. Returns a patchwork composition.

Usage

```
clustermap(
  data,
  method = "average",
  metric = "euclidean",
  z_score = NULL,
  standard_scale = NULL,
  row_cluster = TRUE,
  col_cluster = TRUE,
```

```
  cmap = NULL,
  dendrogram_ratio = 0.2,
  ...
)
```

Arguments

`data`	A matrix or data frame.
`method`	Linkage method (default `"average"`).
`metric`	Distance metric (default `"euclidean"`).
`z_score`	Normalize rows (`0`) or columns (`1`) to z-scores.
`standard_scale`	Scale rows (`0`) or columns (`1`) to `[0, 1]`.
`row_cluster, col_cluster`	Whether to cluster rows / columns.
`cmap`	Colormap (default `"rocket"`).
`dendrogram_ratio`	Fraction of the figure used by the dendrograms.
`...`	Passed to heatmap.

Value

A `reaborn_plot` (patchwork).

Examples

```
flights <- load_dataset("flights")
m <- tapply(flights$passengers, list(flights$month, flights$year), sum)
clustermap(m)
clustermap(m, z_score = 0, cmap = "vlag")
```

---

color_palette *Return a list of colors or a continuous colormap defining a palette*

Description

Port of `seaborn.color_palette`. Possible `palette` values include the name of a seaborn palette (`deep`, `muted`, `bright`, `pastel`, `dark`, `colorblind`), a matplotlib colormap name, `"husl"`/`"hls"`, a cubehelix shorthand (`"ch:..."`), `"light:<color>"`, `"dark:<color>"`, `"blend:<c1>,<c2>"`, or a sequence of colors.

Usage

```
color_palette(palette = NULL, n_colors = NULL, desat = NULL, as_cmap = FALSE)

hls_palette(n_colors = 6, h = 0.01, l = 0.6, s = 0.65, as_cmap = FALSE)

husl_palette(n_colors = 6, h = 0.01, s = 0.9, l = 0.65, as_cmap = FALSE)

dark_palette(
  color,
  n_colors = 6,
  reverse = FALSE,
  as_cmap = FALSE,
  input = "rgb"
)

light_palette(
  color,
  n_colors = 6,
  reverse = FALSE,
  as_cmap = FALSE,
  input = "rgb"
)

diverging_palette(
  h_neg,
  h_pos,
  s = 75,
  l = 50,
  sep = 1,
  n = 6,
  center = "light",
  as_cmap = FALSE
)

blend_palette(colors, n_colors = 6, as_cmap = FALSE, input = "rgb")

mpl_palette(name, n_colors = 6, as_cmap = FALSE)

cubehelix_palette(
  n_colors = 6,
  start = 0,
  rot = 0.4,
  gamma = 1,
  hue = 0.8,
  light = 0.85,
  dark = 0.15,
  reverse = FALSE,
  as_cmap = FALSE
```

```
)
```

Arguments

palette	NULL, a string, or a sequence of colors.
n_colors	Number of colors. If NULL, depends on palette.
desat	Proportion to desaturate each color by.
as_cmap	If TRUE, return a continuous colormap (a reaborn_cmap).
h, l, s	Hue, lightness, saturation anchors in [0, 1].
color	Base color for the high end of a sequential palette.
reverse	Reverse the direction of the blend.
input	Color space of the input color: "rgb", "hls", or "husl".
h_neg, h_pos	Anchor hues ([0, 359]) for the negative and positive ends.
sep	Size of the intermediate (center) region.
n	Number of colors (when not returning a cmap).
center	"light" or "dark" center.
colors	A sequence of colors to blend between.
name	Name of a matplotlib colormap.
start, rot, gamma, hue, light, dark	Cubehelix parameters (see seaborn).

Value

A character vector of hex colors, or a reaborn_cmap.

Examples

```
palplot(color_palette("deep"))
palplot(color_palette("husl", 8))
palplot(hls_palette(8))
palplot(cubehelix_palette(8))
palplot(light_palette("seagreen"))
palplot(diverging_palette(220, 20))
```

---

countplot *Show value counts as bars*

Description

Port of seaborn.countplot. Returns a reaborn_plot.

Usage

```
countplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  orient = NULL,
  color = NULL,
  palette = NULL,
  saturation = 0.75,
  fill = TRUE,
  stat = "count",
  width = 0.8,
  dodge = "auto",
  gap = 0,
  legend = "auto",
  .facet_vars = NULL,
  ...
)
```

Arguments

data	A data frame.
x, y	Variables; the categorical one defines the groups.
hue	Grouping variable for color (dodged).
order, hue_order	Level orderings.
orient	"v", "h", or NULL to infer.
color, palette, saturation, fill	Color controls (saturation default 0.75).
stat	"count", "percent", "proportion", or "probability".
width, gap	Box width and gap between dodged boxes.
dodge	How to dodge bars by hue ("auto", TRUE, or FALSE).
legend	Legend control.
.facet_vars	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
...	Passed to the bar geom.

Value

A reaborn_plot.

Examples

```
penguins <- load_dataset("penguins")
countplot(data = penguins, x = "species", hue = "sex")

# Horizontal bars by assigning the categorical variable to y.
tips <- load_dataset("tips")
countplot(data = tips, y = "day")
```

`desaturate` *Decrease the saturation of a color*

Description

Port of `seaborn.desaturate`.

Usage

```
desaturate(color, prop)
```

Arguments

`color`	A matplotlib-compatible color.
`prop`	Proportion (in `[0, 1]`) of the original saturation to keep.

Value

A hex color string.

`despine` *Remove spines from a plot*

Description

Port of `seaborn.despine`. Returns a ggplot2 theme partial that removes the requested plot borders and (for borderless styles) draws explicit axis lines on the kept sides. Add it to a reaborn/ggplot plot: `p + despine()`.

Usage

```
despine(
  fig = NULL,
  ax = NULL,
  top = TRUE,
  right = TRUE,
  left = FALSE,
  bottom = FALSE,
  offset = NULL,
  trim = FALSE
)
```

Arguments

`fig`, `ax`	Ignored (kept for signature compatibility with seaborn).
`top`, `right`, `left`, `bottom`	Logical; whether to remove that spine. Defaults match seaborn: remove top and right, keep left and bottom.
`offset`, `trim`	Not supported by ggplot2; accepted but ignored in v1 with a one-time message. Present for signature compatibility.

Value

A ggplot2 theme object to add to a plot.

`displot` *Figure-level interface for distribution plots*

Description

Port of `seaborn.displot`. Draws histplot (`kind = "hist"`), kdeplot (`kind = "kde"`), or ecdfplot (`kind = "ecdf"`) onto a grid of facets. Returns a faceted reaborn_plot.

Usage

```
displot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  row = NULL,
  col = NULL,
  weights = NULL,
  kind = "hist",
  rug = FALSE,
  rug_kws = NULL,
```

```
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  color = NULL,
  col_wrap = NULL,
  row_order = NULL,
  col_order = NULL,
  legend = TRUE,
  height = 5,
  aspect = 1,
  facet_kws = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Column name/vector for the histogram variable (use `y` for a horizontal histogram).
`hue`	Grouping variable for color.
`row`, `col`, `col_wrap`, `row_order`, `col_order`	Faceting controls.
`weights`	Optional observation weights.
`kind`	`"hist"`, `"kde"`, or `"ecdf"`.
`rug`	Add a marginal rug.
`rug_kws`	Arguments forwarded to the rug layer when `rug = TRUE`.
`palette`, `hue_order`, `hue_norm`, `color`	Color controls.
`legend`	Show the legend.
`height`, `aspect`	Facet size controls (stored as attributes).
`facet_kws`	Reserved for compatibility.
`...`	Passed to the bar geom.

Value

A `reaborn_plot`.

Examples

```
penguins <- load_dataset("penguins")
displot(data = penguins, x = "flipper_length_mm", col = "species")
displot(
  data = penguins, x = "flipper_length_mm",
  hue = "species", col = "sex", kind = "kde"
)
```

dogplot *Who's a good boy?*

Description

Port of `seaborn.dogplot` (an easter egg). Prints an affirmation.

Usage

```
dogplot(...)
```

Arguments

`...`	Ignored.

Value

Invisibly `NULL`.

Examples

```
dogplot()
```

ecdfplot *Plot an empirical cumulative distribution function*

Description

Port of `seaborn.ecdfplot`. Returns a reaborn_plot.

Usage

```
ecdfplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  weights = NULL,
  stat = "proportion",
  complementary = FALSE,
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  legend = TRUE,
  .facet_vars = NULL,
  ...
)
```

Arguments

data	A data frame.
x, y	Column name/vector for the histogram variable (use y for a horizontal histogram).
hue	Grouping variable for color.
weights	Optional observation weights.
stat	"proportion", "count", or "percent".
complementary	Plot the complementary ECDF (1 - F).
palette	Palette for the hue mapping.
hue_order	Order of hue levels.
hue_norm	Normalization for a numeric hue.
legend	Show the legend.
.facet_vars	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
...	Passed to the bar geom.

Value

A reaborn_plot.

Examples

```
penguins <- load_dataset("penguins")
ecdfplot(data = penguins, x = "flipper_length_mm", hue = "species")

# Complementary ECDF with counts
ecdfplot(data = penguins, x = "bill_length_mm", stat = "count", complementary = TRUE)
```

FacetGrid *A faceted grid of plots*

Description

Lightweight port of seaborn.FacetGrid. In reaborn, faceting is usually done by adding + ggplot2::facet_wrap()/facet_ to a plot, or via the figure-level functions (relplot, displot, catplot, lmplot). This constructor returns a base plot you can map geoms onto.

Usage

```
FacetGrid(
  data,
  row = NULL,
  col = NULL,
  hue = NULL,
  col_wrap = NULL,
  height = 3,
  aspect = 1,
  palette = NULL
)
```

Arguments

`data`	A data frame.
`row`, `col`, `hue`	Faceting / hue variables.
`col_wrap`	Wrap columns at this width.
`height`, `aspect`	Facet sizing.
`palette`	Hue palette.

Value

A `reaborn_plot`.

Examples

```
tips <- load_dataset("tips")
FacetGrid(tips, col = "time", hue = "sex") +
  ggplot2::geom_point(ggplot2::aes(x = total_bill, y = tip))
FacetGrid(tips, row = "sex", col = "time") +
  ggplot2::geom_point(ggplot2::aes(x = total_bill, y = tip))
```

`heatmap` *Plot rectangular data as a color-encoded matrix*

Description

Port of `seaborn.heatmap`. Returns a reaborn_plot.

Usage

```
heatmap(
  data,
  vmin = NULL,
  vmax = NULL,
  cmap = NULL,
```

```
  center = NULL,
  robust = FALSE,
  annot = NULL,
  fmt = ".2g",
  annot_kws = NULL,
  linewidths = 0,
  linecolor = "white",
  cbar = TRUE,
  square = FALSE,
  xticklabels = "auto",
  yticklabels = "auto",
  mask = NULL,
  ...
)
```

Arguments

`data`	A matrix or data frame of values.
`vmin`, `vmax`	Color scale limits.
`cmap`	A colormap name (default `"rocket"`, or `"icefire"` with `center`).
`center`	Value at which to center a diverging colormap.
`robust`	Use the 2nd/98th percentiles for the color limits.
`annot`	Annotate each cell with its value (`TRUE`) or a matrix of labels.
`fmt`	Number format for annotations (default `".2g"`).
`annot_kws`	Passed to the text geom.
`linewidths`, `linecolor`	Cell border width and color.
`cbar`	Show the color bar.
`square`	Force square cells.
`xticklabels`, `yticklabels`	Tick label control (`"auto"`, `TRUE`/`FALSE`).
`mask`	Logical matrix of cells to hide.
`...`	Reserved.

Value

A `reaborn_plot`.

Examples

```
flights <- load_dataset("flights")
m <- tapply(flights$passengers, list(flights$month, flights$year), sum)
heatmap(m, annot = TRUE, fmt = "d", cmap = "YlGnBu")

# Center a diverging colormap on a reference value
heatmap(m, center = m["Jan", "1955"], cmap = "icefire")
```

`histplot` *Plot a univariate or bivariate histogram*

Description

Port of `seaborn.histplot`. Returns a reaborn_plot.

Usage

```
histplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  weights = NULL,
  stat = "count",
  bins = "auto",
  binwidth = NULL,
  binrange = NULL,
  discrete = NULL,
  cumulative = FALSE,
  common_bins = TRUE,
  common_norm = TRUE,
  multiple = "layer",
  element = "bars",
  fill = TRUE,
  shrink = 1,
  kde = FALSE,
  kde_kws = NULL,
  thresh = 0,
  cbar = FALSE,
  cbar_kws = NULL,
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  color = NULL,
  cmap = NULL,
  legend = TRUE,
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x, y`	Column name/vector for the histogram variable (use `y` for a horizontal histogram).

hue	Grouping variable for color.
weights	Optional observation weights.
stat	One of "count", "frequency", "density", "probability", "proportion", "percent".
bins, binwidth, binrange, discrete	Binning controls (see rb_hist_bins).
cumulative	Accumulate counts.
common_bins, common_norm	Share bins/normalization across hue groups.
multiple	"layer", "stack", "fill", or "dodge".
element	"bars" or "step".
fill	Whether to fill the bars.
shrink	Shrink bar widths by this factor.
kde	Overlay a KDE curve.
kde_kws	Arguments for the KDE (e.g. bw_adjust).
thresh	Bivariate-only. Cells with a value at or below thresh are left transparent (default 0, so empty cells are blank); NULL fills every cell.
cbar, cbar_kws	Bivariate-only. Draw a color bar for the counts; cbar_kws accepts width (the bar width in points).
palette, hue_order, hue_norm, color	Color controls.
cmap	Bivariate-only colormap (name, reaborn_cmap, or color vector); defaults to a light sequential ramp built from color.
legend	Show the legend.
.facet_vars	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
...	Passed to the bar geom.

Details

When both x and y are supplied, histplot() draws a single 2-D count heatmap (like seaborn.histplot(x, y)). In that bivariate case hue is ignored, with a warning, and the hue-based color controls (palette, hue_order, hue_norm) do not apply; the fill is driven by cmap, which defaults to a light sequential ramp built from color.

Value

A reaborn_plot.

Examples

```
penguins <- load_dataset("penguins")
histplot(data = penguins, x = "flipper_length_mm", hue = "species")

# Stack the hue groups
histplot(data = penguins, x = "flipper_length_mm", hue = "species", multiple = "stack")
```

`jointplot` *Draw a bivariate plot with marginal distributions*

Description

Port of `seaborn.jointplot`. Returns a patchwork composition (printable and saveable like any reaborn plot).

Usage

```
jointplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  kind = "scatter",
  height = 6,
  ratio = 5,
  space = 0.2,
  color = NULL,
  palette = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables.
`hue`	Grouping variable for color.
`kind`	`"scatter"`, `"kde"`, `"reg"`, `"hist"`, or `"hex"`.
`height`	Figure size in inches (stored as an attribute).
`ratio`	Joint-axes-to-marginal size ratio.
`space`	Spacing between joint and marginal axes.
`color`, `palette`	Color controls.
`...`	Passed to the joint plotting function.

Value

A `reaborn_plot` (patchwork).

Examples

```
penguins <- load_dataset("penguins")
jointplot(data = penguins, x = "bill_length_mm", y = "bill_depth_mm", hue = "species")
jointplot(data = penguins, x = "bill_length_mm", y = "bill_depth_mm", kind = "reg")
```

kdeplot *Plot a univariate or bivariate kernel density estimate*

Description

Port of `seaborn.kdeplot`. The KDE matches `scipy.stats.gaussian_kde` exactly. Returns a reaborn_plot.

Usage

```
kdeplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  weights = NULL,
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  color = NULL,
  fill = NULL,
  multiple = "layer",
  common_norm = TRUE,
  common_grid = FALSE,
  cumulative = FALSE,
  bw_method = "scott",
  bw_adjust = 1,
  log_scale = NULL,
  levels = 10,
  thresh = 0.05,
  gridsize = 200,
  cut = 3,
  clip = NULL,
  legend = TRUE,
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x, y`	Column name/vector for the histogram variable (use `y` for a horizontal histogram).
`hue`	Grouping variable for color.
`weights`	Optional observation weights.

palette, hue_order, hue_norm, color
: Color controls.

fill
: Fill under the density curve (default FALSE).

multiple
: "layer", "stack", or "fill".

common_norm, common_grid
: Share normalization / evaluation grid across hue groups.

cumulative
: Plot the cumulative distribution.

bw_method, bw_adjust
: Bandwidth controls (scipy-compatible).

log_scale
: Reserved for compatibility.

levels, thresh
: Bivariate contour levels and density threshold.

gridsize, cut, clip
: KDE grid controls.

legend
: Show the legend.

.facet_vars
: Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.

...
: Passed to the bar geom.

Value

A reaborn_plot.

Examples

```
penguins <- load_dataset("penguins")
kdeplot(data = penguins, x = "flipper_length_mm", hue = "species", fill = TRUE)
kdeplot(data = penguins, x = "flipper_length_mm", hue = "species", multiple = "stack")
```

---

lineplot *Draw a line plot with aggregation and error bands*

Description

Port of seaborn.lineplot. When the data has repeated observations per x value, they are aggregated (default: mean) and an error band (default: 95% bootstrap CI) is drawn. Returns a reaborn_plot.

Usage

```
lineplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  size = NULL,
```

```
  style = NULL,
  units = NULL,
  weights = NULL,
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  sizes = NULL,
  size_order = NULL,
  size_norm = NULL,
  dashes = TRUE,
  markers = NULL,
  style_order = NULL,
  estimator = "mean",
  errorbar = list("ci", 95),
  n_boot = 1000,
  seed = NULL,
  orient = "x",
  sort = TRUE,
  err_style = "band",
  err_kws = NULL,
  legend = "auto",
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Column names (strings) or vectors giving the axes.
`hue`, `size`, `style`	Column names/vectors for color, size, and marker-style semantics.
`units`, `weights`	Column names/vectors for the unit grouping and weights.
`palette`, `hue_order`, `hue_norm`	Control the color mapping.
`sizes`, `size_order`, `size_norm`	Control the size mapping.
`dashes`, `markers`	Style mapping controls.
`style_order`	Order of style levels.
`estimator`	Aggregation function name or callable (default `"mean"`; `NULL` to plot all observations).
`errorbar`	Error representation: a method name or `list(method, level)` (default `list("ci", 95)`).
`n_boot`, `seed`	Bootstrap settings for `errorbar = "ci"`.
`orient`, `sort`, `err_style`, `err_kws`	See seaborn.
`legend`	`"auto"`, `"brief"`, `"full"`, or `FALSE`.

`.facet_vars`	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
`...`	Passed to ggplot2::geom_line.

Value

A `reaborn_plot`.

Examples

```
fmri <- load_dataset("fmri")
# Aggregated mean with a 95% bootstrap CI band across repeated observations
lineplot(data = fmri, x = "timepoint", y = "signal", hue = "event")
# Add a style semantic to distinguish brain regions
lineplot(
  data = fmri, x = "timepoint", y = "signal",
  hue = "event", style = "region"
)
```

`lmplot` *Figure-level interface for regression plots*

Description

Port of `seaborn.lmplot`. Draws regplot across a grid of facets and/or hue groups. Returns a reaborn_plot.

Usage

```
lmplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  col = NULL,
  row = NULL,
  palette = NULL,
  col_wrap = NULL,
  height = 5,
  aspect = 1,
  order = 1,
  logistic = FALSE,
  lowess = FALSE,
  robust = FALSE,
  logx = FALSE,
  ci = 95,
  n_boot = 1000,
  seed = NULL,
```

```
  scatter = TRUE,
  fit_reg = TRUE,
  hue_order = NULL,
  row_order = NULL,
  col_order = NULL,
  legend = TRUE,
  facet_kws = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables.
`hue`, `col`, `row`	Semantic / faceting variables.
`palette`	Hue palette.
`col_wrap`, `row_order`, `col_order`, `hue_order`	Ordering / wrapping.
`height`, `aspect`	Facet sizing.
`order`	Polynomial order for the fit (default 1, linear).
`logistic`, `lowess`, `robust`, `logx`	Alternative fits.
`ci`	Confidence-band width (default 95; `NULL` to omit).
`n_boot`, `seed`	Bootstrap settings.
`scatter`, `fit_reg`	Whether to draw the scatter / the fit.
`legend`, `facet_kws`	Legend / facet options.
`...`	Reserved.

Value

A `reaborn_plot`.

Examples

```
tips <- load_dataset("tips")
lmplot(data = tips, x = "total_bill", y = "tip", hue = "smoker")

# Facet across a second variable with col
lmplot(data = tips, x = "total_bill", y = "tip", hue = "smoker", col = "time")
```

load_dataset *Load an example dataset from the seaborn-data repository*

Description

Port of `seaborn.load_dataset`. Bundled datasets (penguins, tips, iris, flights) load offline; others download from the seaborn-data repo and cache.

Usage

```
load_dataset(name, cache = TRUE, data_home = NULL, ...)

get_dataset_names()
```

Arguments

`name`	Name of the dataset (the stem of a `.csv` in seaborn-data).
`cache`	Whether to use the local cache (and bundled data).
`data_home`	Optional cache directory.
`...`	Reserved for compatibility.

Value

A data.frame, with categorical columns coerced to ordered factors matching seaborn.

For `get_dataset_names`, a character vector of available dataset names.

move_legend *Reposition a plot's legend*

Description

Port of `seaborn.move_legend`. Returns a theme partial controlling legend position. Add it to a plot: `p + move_legend("upper right")`.

Usage

```
move_legend(obj = NULL, loc = "best", ...)
```

Arguments

`obj`	Ignored (signature compatibility).
`loc`	A seaborn/matplotlib location string (e.g. `"upper right"`, `"center left"`), `"best"`, or a length-2 numeric vector of relative coords.
`...`	Additional theme arguments (e.g. `title`).

Value

A ggplot2 theme object to add to a plot.

`pairplot` *Plot pairwise relationships in a dataset*

Description

Port of `seaborn.pairplot`. Returns a patchwork matrix of scatter plots with univariate distributions on the diagonal.

Usage

```
pairplot(
  data,
  vars = NULL,
  hue = NULL,
  kind = "scatter",
  diag_kind = "auto",
  palette = NULL,
  height = 2.5,
  aspect = 1,
  corner = FALSE,
  ...
)
```

Arguments

`data`	A data frame.
`vars`	Columns to include (default all numeric).
`hue`	Grouping variable for color.
`kind`	Off-diagonal kind: `"scatter"` or `"reg"`.
`diag_kind`	Diagonal kind: `"auto"`, `"hist"`, or `"kde"`.
`palette, height, aspect, corner`	Layout controls.
`...`	Reserved.

Value

A `reaborn_plot` (patchwork).

Examples

```
penguins <- load_dataset("penguins")
pairplot(
  data = penguins,
  vars = c("bill_length_mm", "flipper_length_mm", "body_mass_g"),
  hue = "species"
)

# Regression fits off the diagonal
pairplot(
  data = penguins,
  vars = c("bill_length_mm", "flipper_length_mm"),
  hue = "species",
  kind = "reg"
)
```

---

palplot *Plot the values in a color palette as a horizontal array*

Description

Port of `seaborn.palplot`. Returns a reaborn_plot.

Usage

```
palplot(pal, size = 1)
```

Arguments

`pal`	A sequence of colors (e.g. from color_palette).
`size`	Scaling factor for the swatch size.

Value

A `reaborn_plot`.

Examples

```
palplot(color_palette("deep"))
palplot(color_palette("rocket", 8))
```

plotting_context *Get the parameters that control the scaling of plot elements*

Description

Port of `seaborn.plotting_context`. Returns a named list of resolved sizes.

Usage

```
plotting_context(context = NULL, font_scale = 1, rc = NULL)

set_context(context = NULL, font_scale = 1, rc = NULL)
```

Arguments

`context`	One of `"paper"`, `"notebook"`, `"talk"`, `"poster"`, or a list.
`font_scale`	Separate scaling factor applied to the font sizes only.
`rc`	Optional named list of overrides.

Value

A named list of context parameters.

pointplot *Show point estimates and errors with markers*

Description

Port of `seaborn.pointplot`. Returns a reaborn_plot.

Usage

```
pointplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  estimator = "mean",
  errorbar = list("ci", 95),
  n_boot = 1000,
  seed = NULL,
  units = NULL,
  weights = NULL,
```

```
  color = NULL,
  palette = NULL,
  markers = "o",
  linestyles = "-",
  dodge = FALSE,
  orient = NULL,
  capsize = 0,
  legend = "auto",
  err_kws = NULL,
  .facet_vars = NULL,
  ...
)
```

Arguments

data	A data frame.
x, y	Variables; the categorical one defines the groups.
hue	Grouping variable for color (dodged).
order, hue_order	Level orderings.
estimator, errorbar, n_boot, seed	Aggregation + error settings.
units, weights	Bootstrap structure / weights (units reserved).
color	Single color override.
palette	Palette for the hue mapping.
markers, linestyles	Marker and line styling.
dodge	Dodge points by hue.
orient	"v", "h", or NULL to infer.
capsize	Width of the error bar caps.
legend	Legend control.
err_kws	Passed to the error bar geom.
.facet_vars	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
...	Passed to the bar geom.

Value

A reaborn_plot.

Examples

```
tips <- load_dataset("tips")
pointplot(data = tips, x = "time", y = "total_bill", hue = "smoker")
```

```
# Dodge the hue levels and add caps to the error bars
pointplot(
  data = tips, x = "day", y = "total_bill", hue = "sex",
  dodge = TRUE, capsize = 0.1
)
```

`python-literals` *Python literal compatibility values*

Description

`True`, `False`, and `None` are provided so that seaborn Python code pasted into R (e.g. `histplot(data = df, x = "a", kde = True)`) runs unchanged. They are exactly `TRUE`, `FALSE`, and `NULL`.

Usage

```
True

False

None
```

Format

`True` and `False` are length-one logicals; `None` is `NULL`.

Value

These objects are exported constants, not functions. Referencing one yields its stored value: `True` is the length-one logical vector `TRUE`, `False` is the length-one logical vector `FALSE`, and `None` is `NULL`. They exist only so that seaborn code containing Python's `True`, `False`, and `None` literals evaluates in R unchanged.

`regplot` *Plot data and a linear regression model fit*

Description

Port of `seaborn.regplot`. The confidence band is a bootstrap interval, like seaborn. Returns a reaborn_plot.

Usage

```
regplot(
  data = NULL,
  x = NULL,
  y = NULL,
  order = 1,
  logistic = FALSE,
  lowess = FALSE,
  robust = FALSE,
  logx = FALSE,
  ci = 95,
  n_boot = 1000,
  seed = NULL,
  scatter = TRUE,
  fit_reg = TRUE,
  color = NULL,
  marker = "o",
  scatter_kws = NULL,
  line_kws = NULL,
  truncate = TRUE,
  x_jitter = NULL,
  y_jitter = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables.
`order`	Polynomial order for the fit (default 1, linear).
`logistic`, `lowess`, `robust`, `logx`	Alternative fits.
`ci`	Confidence-band width (default 95; `NULL` to omit).
`n_boot`, `seed`	Bootstrap settings.
`scatter`, `fit_reg`	Whether to draw the scatter / the fit.
`color`	Color for points and line (default the first palette color).
`marker`	Marker (accepted for compatibility).
`scatter_kws`, `line_kws`	Lists of extra args for the point / line layers.
`truncate`	Limit the regression line to the data range.
`x_jitter`	Uniform jitter added to x for display only.
`y_jitter`	Uniform jitter added to y for display only.
`...`	Reserved.

Value

A reaborn_plot.

Examples

```
tips <- load_dataset("tips")
regplot(data = tips, x = "total_bill", y = "tip")

# Fit a higher-order polynomial
regplot(data = tips, x = "size", y = "total_bill", order = 2)
```

---

relplot *Figure-level interface for relational plots*

Description

Port of seaborn.relplot. Draws scatterplot (kind = "scatter") or lineplot (kind = "line") onto a grid of facets defined by row/col. Returns a reaborn_plot (a faceted ggplot) with the legend outside, like a seaborn FacetGrid.

Usage

```
relplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  size = NULL,
  style = NULL,
  units = NULL,
  weights = NULL,
  row = NULL,
  col = NULL,
  col_wrap = NULL,
  row_order = NULL,
  col_order = NULL,
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  sizes = NULL,
  size_order = NULL,
  size_norm = NULL,
  markers = NULL,
  dashes = NULL,
  style_order = NULL,
  legend = "auto",
  kind = "scatter",
```

```
  height = 5,
  aspect = 1,
  facet_kws = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x, y`	Column names (strings) or vectors giving the axes.
`hue, size, style`	Column names/vectors for color, size, and marker-style semantics.
`units, weights`	Column names/vectors for the unit grouping and weights.
`row, col`	Column names to facet by.
`col_wrap`	Wrap the column facets at this width.
`row_order, col_order`	Facet orderings.
`palette, hue_order, hue_norm`	Control the color mapping.
`sizes, size_order, size_norm`	Control the size mapping.
`dashes, markers`	Style mapping controls.
`style_order`	Order of style levels.
`legend`	`"auto"`, `"brief"`, `"full"`, or `FALSE`.
`kind`	`"scatter"` or `"line"`.
`height, aspect`	Facet height (inches) and aspect ratio (stored as attributes used as defaults when saving).
`facet_kws`	Reserved for compatibility.
`...`	Passed to ggplot2::geom_line.

Value

A `reaborn_plot`.

Examples

```
fmri <- load_dataset("fmri")
relplot(
  data = fmri, x = "timepoint", y = "signal",
  hue = "event", col = "region", kind = "line"
)

tips <- load_dataset("tips")
relplot(data = tips, x = "total_bill", y = "tip", hue = "day", col = "time")
```

reset_defaults *Restore matplotlib/ggplot2 defaults*

Description

Port of `seaborn.reset_defaults` / `seaborn.reset_orig`.

Usage

```
reset_defaults()

reset_orig()
```

Value

Invisibly `NULL`.

residplot *Plot the residuals of a linear regression*

Description

Port of `seaborn.residplot`. Returns a reaborn_plot.

Usage

```
residplot(
  data = NULL,
  x = NULL,
  y = NULL,
  lowess = FALSE,
  order = 1,
  robust = FALSE,
  color = NULL,
  scatter_kws = NULL,
  line_kws = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x, y`	Variables.
`lowess`	Add a lowess smooth of the residuals.
`order`	Polynomial order for the fit (default 1, linear).

`robust`	Fit a robust regression when computing residuals.
`color`	Color for points and line (default the first palette color).
`scatter_kws, line_kws`	Lists of extra args for the point / line layers.
`...`	Reserved.

Value

A `reaborn_plot`.

Examples

```
tips <- load_dataset("tips")
residplot(data = tips, x = "total_bill", y = "tip")

# Add a lowess smooth to help detect structure in the residuals
residplot(data = tips, x = "total_bill", y = "tip", lowess = TRUE)
```

---

rugplot *Plot marginal rug ticks*

Description

Port of `seaborn.rugplot`. Draws small ticks at each observation along the relevant axis. Returns a reaborn_plot (typically added to another plot, but usable standalone).

Usage

```
rugplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  height = 0.025,
  expand_margins = TRUE,
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  legend = TRUE,
  ...
)
```

Arguments

data	A data frame.
x, y	Column name/vector for the histogram variable (use y for a horizontal histogram).
hue	Grouping variable for color.
height	Tick height as a fraction of the axis (default 0.025).
expand_margins	Reserved for compatibility.
palette	Palette for the hue mapping.
hue_order	Order of hue levels.
hue_norm	Normalization for a numeric hue.
legend	Show the legend.
...	Passed to the bar geom.

Value

A reaborn_plot.

Examples

```
penguins <- load_dataset("penguins")
rugplot(data = penguins, x = "bill_length_mm", y = "bill_depth_mm")

# Add a hue semantic to color ticks by group
rugplot(data = penguins, x = "bill_length_mm", hue = "species")
```

---

saturate *Increase the saturation of a color to its maximum*

Description

Port of seaborn.saturate.

Usage

```
saturate(color)
```

Arguments

color	A matplotlib-compatible color.

Value

A hex color string.

`scatterplot` *Draw a scatter plot with semantic mappings*

Description

Port of `seaborn.scatterplot`. Returns a reaborn_plot (a ggplot), so it can be extended with any ggplot2 component.

Usage

```
scatterplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  size = NULL,
  style = NULL,
  palette = NULL,
  hue_order = NULL,
  hue_norm = NULL,
  sizes = NULL,
  size_order = NULL,
  size_norm = NULL,
  markers = TRUE,
  style_order = NULL,
  legend = "auto",
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Column names (strings) or vectors giving the axes.
`hue`, `size`, `style`	Column names/vectors for color, size, and marker-style semantics.
`palette`, `hue_order`, `hue_norm`	Control the color mapping.
`sizes`, `size_order`, `size_norm`	Control the size mapping.
`markers`, `style_order`	Control the style (marker) mapping.
`legend`	`"auto"`, `"brief"`, `"full"`, or `FALSE`.
`...`	Passed to ggplot2::geom_point.

Value

A `reaborn_plot`.

Examples

```
penguins <- load_dataset("penguins")
scatterplot(data = penguins, x = "bill_length_mm", y = "bill_depth_mm", hue = "species")

# Add size and style semantics
scatterplot(
  data = penguins,
  x = "bill_length_mm",
  y = "bill_depth_mm",
  hue = "species",
  size = "body_mass_g",
  style = "sex"
)
```

---

set_color_codes *Change how single-letter color codes are interpreted*

Description

Port of `seaborn.set_color_codes`. Returns (invisibly) the mapping from the single-letter codes `b g r m y c k` to the colors of the given seaborn palette, so reaborn helpers can resolve them like seaborn does.

Usage

```
set_color_codes(palette = "deep")
```

Arguments

palette	One of `"deep"`, `"muted"`, `"pastel"`, `"bright"`, `"dark"`, `"colorblind"`.

Value

Invisibly, the named character vector of code -> hex mappings.

---

set_hls_values *Independently set the hue, lightness, and/or saturation of a color*

Description

Port of `seaborn.set_hls_values`.

Usage

```
set_hls_values(color, h = NULL, l = NULL, s = NULL)
```

Arguments

color	A matplotlib-compatible color.
h, l, s	New hue, lightness, saturation in [0, 1], or NULL to keep.

Value

A hex color string.

---

set_palette *Set the matplotlib color cycle / ggplot default discrete palette*

Description

Port of seaborn.set_palette.

Usage

```
set_palette(palette, n_colors = NULL, desat = NULL, color_codes = FALSE)
```

Arguments

palette	A palette name or sequence (see color_palette).
n_colors, desat, color_codes	See seaborn.

Value

Invisibly NULL.

---

set_theme *Set multiple theme parameters in one step*

Description

Port of seaborn.set_theme (and its alias set). Sets the global look used by subsequent reaborn (and ggplot2) plots.

Usage

```
set_theme(
  context = "notebook",
  style = "darkgrid",
  palette = "deep",
  font = "sans",
  font_scale = 1,
  color_codes = TRUE,
  rc = NULL
)

set(...)
```

Arguments

context, style, palette, font, font_scale, color_codes, rc	See seaborn.
...	Passed to set_theme.

Value

Invisibly, the applied ggplot2::theme.

---

sns-aliases *seaborn-style* sns. *function aliases*

Description

For copy-paste compatibility with Python, reaborn provides a sns.-prefixed alias for every public plotting, palette, and theming function (R allows dots in identifiers). So pasted seaborn code such as sns.scatterplot(data = df, x = "a", y = "b", hue = "g") runs verbatim. Each alias is identical to its unprefixed counterpart; see that function for arguments and details.

Value

Each sns.-prefixed object is the exact same function as its unprefixed counterpart (sns.scatterplot <- scatterplot, and so on), so calling an alias returns precisely what the counterpart returns. By category: the plotting functions (e.g. sns.scatterplot(), sns.histplot(), sns.heatmap(), sns.pairplot(), sns.FacetGrid(), sns.palplot()) return a reaborn_plot object (a ggplot2/patchwork object that draws when printed), except the easter-egg sns.dogplot(), which prints an affirmation and returns NULL invisibly; the palette constructors (e.g. sns.color_palette(), sns.husl_palette(), sns.cubehelix_palette()) return a character vector of hex colors, or a reaborn_cmap; the color helpers sns.desaturate(), sns.saturate(), and sns.set_hls_values() return a hex color string; sns.axes_style() and sns.plotting_context() return a named list of style/context parameters; sns.despine() and sns.move_legend() return a ggplot2 theme object to add to a plot; sns.load_dataset() returns a data frame and sns.get_dataset_names() a character vector; and the theming setters (sns.set_theme(), sns.set(), sns.set_style(), sns.set_context(),

`sns.set_palette()`, `sns.set_color_codes()`, `sns.reset_defaults()`, `sns.reset_orig()`) are called for their side effect of changing global plot defaults and return their value invisibly. See each unprefixed function's own help page for the precise structure and meaning of its return value.

Examples

```
pen <- load_dataset("penguins")
p <- sns.scatterplot(data = pen, x = "bill_length_mm", y = "bill_depth_mm",
                     hue = "species")
```

`stripplot` *Draw a categorical scatter with jitter*

Description

Port of `seaborn.stripplot`. Returns a reaborn_plot.

Usage

```
stripplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  jitter = TRUE,
  dodge = FALSE,
  orient = NULL,
  color = NULL,
  palette = NULL,
  size = 5,
  edgecolor = "gray",
  linewidth = 0,
  legend = "auto",
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables; the categorical one defines the groups.
`hue`	Grouping variable for color (dodged).
`order`, `hue_order`	Level orderings.

`jitter`	`TRUE`, `FALSE`, or a numeric jitter amount.
`dodge`	Separate hue levels along the categorical axis.
`orient`	`"v"`, `"h"`, or `NULL` to infer.
`color`	Single color override.
`palette`	Palette for the hue mapping.
`size`	Marker size (seaborn default 5).
`edgecolor`, `linewidth`	Marker edge styling.
`legend`	Legend control.
`.facet_vars`	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
`...`	Passed to the point geom.

Value

A `reaborn_plot`.

Examples

```
tips <- load_dataset("tips")
stripplot(data = tips, x = "day", y = "total_bill", hue = "smoker")

# Separate hue levels along the categorical axis with dodge
stripplot(data = tips, x = "day", y = "total_bill", hue = "smoker", dodge = TRUE)
```

`swarmplot` *Draw a categorical scatter with non-overlapping points*

Description

Port of `seaborn.swarmplot`, using a beeswarm layout. Returns a reaborn_plot.

Usage

```
swarmplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  dodge = FALSE,
  orient = NULL,
  color = NULL,
  palette = NULL,
```

```
  size = 5,
  edgecolor = NULL,
  linewidth = 0,
  legend = "auto",
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x, y`	Variables; the categorical one defines the groups.
`hue`	Grouping variable for color (dodged).
`order, hue_order`	Level orderings.
`dodge`	Separate hue levels along the categorical axis.
`orient`	`"v"`, `"h"`, or `NULL` to infer.
`color`	Single color override.
`palette`	Palette for the hue mapping.
`size`	Marker size (seaborn default 5).
`edgecolor, linewidth`	Marker edge styling.
`legend`	Legend control.
`.facet_vars`	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
`...`	Passed to ggbeeswarm::geom_beeswarm.

Value

A `reaborn_plot`.

Examples

```
tips <- load_dataset("tips")
swarmplot(data = tips, x = "day", y = "total_bill")
swarmplot(data = tips, x = "day", y = "total_bill", hue = "sex", dodge = TRUE)
```

theme_seaborn *Build a ggplot2 theme replicating a seaborn style + context*

Description

Build a ggplot2 theme replicating a seaborn style + context

Usage

```
theme_seaborn(
  style = "darkgrid",
  context = "notebook",
  font_scale = 1,
  font = "sans"
)
```

Arguments

style	A seaborn style name (see axes_style).
context	A seaborn context name (see plotting_context).
font_scale	Font scaling factor.
font	Base font family.

Value

A complete ggplot2::theme object.

violinplot *Draw a violin plot*

Description

Port of `seaborn.violinplot`. The kernel density matches `scipy.stats.gaussian_kde`. Returns a reaborn_plot.

Usage

```
violinplot(
  data = NULL,
  x = NULL,
  y = NULL,
  hue = NULL,
  order = NULL,
  hue_order = NULL,
  orient = NULL,
```

```
  color = NULL,
  palette = NULL,
  saturation = 0.75,
  fill = TRUE,
  inner = "box",
  split = FALSE,
  width = 0.8,
  dodge = "auto",
  gap = 0,
  linewidth = NULL,
  linecolor = "auto",
  cut = 2,
  gridsize = 100,
  bw_method = "scott",
  bw_adjust = 1,
  density_norm = "area",
  common_norm = FALSE,
  legend = "auto",
  inner_kws = NULL,
  .facet_vars = NULL,
  ...
)
```

Arguments

`data`	A data frame.
`x`, `y`	Variables; the categorical one defines the groups.
`hue`	Grouping variable for color (dodged).
`order`, `hue_order`	Level orderings.
`orient`	`"v"`, `"h"`, or `NULL` to infer.
`color`, `palette`, `saturation`, `fill`	Color controls (saturation default 0.75).
`inner`	`"box"`, `"quart"`, `"stick"`, `"point"`, or `NULL`.
`split`	Draw split violins for two hue levels.
`width`, `gap`	Box width and gap between dodged boxes.
`dodge`	How to dodge violins by hue (`"auto"`, `TRUE`, or `FALSE`).
`linewidth`	Outline width.
`linecolor`	Outline color (`"auto"` for seaborn's gray).
`cut`, `gridsize`, `bw_method`, `bw_adjust`	KDE controls.
`density_norm`	`"area"`, `"count"`, or `"width"`.
`common_norm`	Normalize densities across all groups together.
`legend`	Legend control.

inner_kws	Passed to the inner annotation geoms.
.facet_vars	Internal; facet columns forwarded by the figure-level dispatchers (catplot/displot/relplot). Not intended for direct use.
...	Passed to ggplot2::geom_boxplot.

Value

A reaborn_plot.

Examples

```
tips <- load_dataset("tips")
violinplot(data = tips, x = "day", y = "total_bill")
violinplot(data = tips, x = "day", y = "total_bill", hue = "sex", split = TRUE)
```

Index

axes_style, 3, *48*

barplot, 3, *8*
blend_palette *(*color_palette*)*, 10
boxenplot, 5
boxplot, 6, *8*

catplot, 8, *18*
clustermap, 9
color_palette, 10, *31*, *43*
countplot, *8*, 12
cubehelix_palette *(*color_palette*)*, 10

dark_palette *(*color_palette*)*, 10
desaturate, 14
despine, 14
displot, 15, *18*
diverging_palette *(*color_palette*)*, 10
dogplot, 17

ecdfplot, *15*, 17

FacetGrid, 18
False *(*python-literals*)*, 34

get_dataset_names *(*load_dataset*)*, 29
ggbeeswarm::geom_beeswarm, *47*
ggplot2::geom_boxplot, *6, 7*, *50*
ggplot2::geom_line, *27*, *37*
ggplot2::geom_point, *41*
ggplot2::theme, *44*, *48*

heatmap, *10*, 19
histplot, *15*, 21
hls_palette *(*color_palette*)*, 10
husl_palette *(*color_palette*)*, 10

jointplot, 23

kdeplot, *15*, 24

light_palette *(*color_palette*)*, 10
lineplot, 25, *36*
lmplot, *18*, 27
load_dataset, 29

move_legend, 29
mpl_palette *(*color_palette*)*, 10

None *(*python-literals*)*, 34

pairplot, 30
palplot, 31
plotting_context, 32, *48*
pointplot, *8*, 32
python-literals, 34

rb_hist_bins, *22*
reaborn_plot, *3*, *5, 6*, *8*, *12*, *15*, *17*, *19*, *21*, *24, 25*, *27*, *31, 32*, *34*, *36*, *38, 39*, *41*, *45, 46*, *48*
regplot, *27*, 34
relplot, *18*, 36
reset_defaults, 38
reset_orig *(*reset_defaults*)*, 38
residplot, 38
rugplot, 39

saturate, 40
scatterplot, *36*, 41
set *(*set_theme*)*, 43
set_color_codes, 42
set_context *(*plotting_context*)*, 32
set_hls_values, 42
set_palette, 43
set_style *(*axes_style*)*, 3
set_theme, 43, *44*
sns-aliases, 44
sns.axes_style *(*sns-aliases*)*, 44
sns.barplot *(*sns-aliases*)*, 44
sns.blend_palette *(*sns-aliases*)*, 44
sns.boxenplot *(*sns-aliases*)*, 44
sns.boxplot *(*sns-aliases*)*, 44

sns.catplot (sns-aliases), 44
sns.clustermap (sns-aliases), 44
sns.color_palette (sns-aliases), 44
sns.countplot (sns-aliases), 44
sns.cubehelix_palette (sns-aliases), 44
sns.dark_palette (sns-aliases), 44
sns.desaturate (sns-aliases), 44
sns.despine (sns-aliases), 44
sns.displot (sns-aliases), 44
sns.diverging_palette (sns-aliases), 44
sns.dogplot (sns-aliases), 44
sns.ecdfplot (sns-aliases), 44
sns.FacetGrid (sns-aliases), 44
sns.get_dataset_names (sns-aliases), 44
sns.heatmap (sns-aliases), 44
sns.histplot (sns-aliases), 44
sns.hls_palette (sns-aliases), 44
sns.husl_palette (sns-aliases), 44
sns.jointplot (sns-aliases), 44
sns.kdeplot (sns-aliases), 44
sns.light_palette (sns-aliases), 44
sns.lineplot (sns-aliases), 44
sns.lmplot (sns-aliases), 44
sns.load_dataset (sns-aliases), 44
sns.move_legend (sns-aliases), 44
sns.mpl_palette (sns-aliases), 44
sns.pairplot (sns-aliases), 44
sns.palplot (sns-aliases), 44
sns.plotting_context (sns-aliases), 44
sns.pointplot (sns-aliases), 44
sns.regplot (sns-aliases), 44
sns.relplot (sns-aliases), 44
sns.reset_defaults (sns-aliases), 44
sns.reset_orig (sns-aliases), 44
sns.residplot (sns-aliases), 44
sns.rugplot (sns-aliases), 44
sns.saturate (sns-aliases), 44
sns.scatterplot (sns-aliases), 44
sns.set (sns-aliases), 44
sns.set_color_codes (sns-aliases), 44
sns.set_context (sns-aliases), 44
sns.set_hls_values (sns-aliases), 44
sns.set_palette (sns-aliases), 44
sns.set_style (sns-aliases), 44
sns.set_theme (sns-aliases), 44
sns.stripplot (sns-aliases), 44
sns.swarmplot (sns-aliases), 44
sns.violinplot (sns-aliases), 44
stripplot, *8*, 45
swarmplot, 46

theme_seaborn, 48
True (python-literals), 34

violinplot, 48