

Package ‘reborn’

July 31, 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. 'reborn' mirrors the 'seaborn' public function API (identical function names, argument names, and defaults) and renders visually indistinguishable plots using 'ggplot2'. Because every 'reborn' 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

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), vdiff

Config/testthat/edition 3

VignetteBuilder knitr

URL <https://reborn.org>, <https://github.com/shawntz/reborn>

BugReports <https://github.com/shawntz/reborn/issues>

NeedsCompilation no

Author Shawn Schwartz [aut, cre] (ORCID:
<<https://orcid.org/0000-0001-6444-8451>>)

Maintainer Shawn Schwartz <shawn.t.schwartz@gmail.com>

Repository CRAN

Date/Publication 2026-07-31 10:40:08 UTC

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

axes_style	<i>Get the parameters that control the general style of the plots</i>
------------	---

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	<i>Show point estimates and errors as bars</i>
---------	--

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

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

Value

A `reborn_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

<code>data</code>	A data frame.
<code>x, y</code>	Variables; the categorical one defines the groups.
<code>hue</code>	Grouping variable for color (dodged).
<code>order, hue_order</code>	Level orderings.
<code>orient</code>	"v", "h", or NULL to infer.
<code>color, palette, saturation, fill</code>	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 `reborn_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	<i>Draw a box plot</i>
---------	------------------------

Description

Port of `seaborn.boxplot`. Returns a [reborn_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

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

Value

A `reborn_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 [reborn_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

<code>data</code>	A data frame.
<code>x, y</code>	Variables; the categorical one defines the groups.
<code>hue</code>	Grouping variable for color (dodged).
<code>row, col, col_wrap, row_order, col_order</code>	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 `reborn_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	<i>Plot a hierarchically-clustered heatmap</i>
------------	--

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 `reborn_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 "hsl".
- 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("hsl", 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

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

Value

A `reborn_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	<i>Decrease the saturation of a color</i>
------------	---

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	<i>Remove spines from a plot</i>
---------	----------------------------------

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	<i>Figure-level interface for distribution plots</i>
---------	--

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

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

Value

A `reborn_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	<i>Who's a good boy?</i>
---------	--------------------------

Description

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

Usage

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

Arguments

... Ignored.

Value

Invisibly NULL.

Examples

```
dogplot()
```

ecdfplot	<i>Plot an empirical cumulative distribution function</i>
----------	---

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	<i>A faceted grid of plots</i>
-----------	--------------------------------

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 `reborn_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 [reborn_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

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

Value

A `reborn_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	<i>Plot a univariate or bivariate histogram</i>
----------	---

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. <code>bw_adjust</code>).
thresh	Bivariate-only. Cells with a value at or below <code>thresh</code> 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; <code>cbar_kws</code> accepts width (the bar width in points).
palette, hue_order, hue_norm, color	Color controls.
cmap	Bivariate-only colormap (name, <code>reborn_cmap</code> , 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 (<code>catplot/displot/relplot</code>). 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 `reborn_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

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

<code>data</code>	A data frame.
<code>x, y</code>	Column name/vector for the histogram variable (use <code>y</code> for a horizontal histogram).
<code>hue</code>	Grouping variable for color.
<code>weights</code>	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 `reborn_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 [reborn_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

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

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

Value

A `reborn_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"
)
```

Implot

Figure-level interface for regression plots

Description

Port of `seaborn.lmplot`. Draws [regplot](#) across a grid of facets and/or hue groups. Returns a [reborn_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

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

Value

A `reborn_plot`.

Examples

```

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

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

```

load_dataset	<i>Load an example dataset from the seaborn-data repository</i>
--------------	---

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	<i>Reposition a plot's legend</i>
-------------	-----------------------------------

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	<i>Plot pairwise relationships in a dataset</i>
----------	---

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 [reborn_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 `reborn_plot`.

Examples

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

plotting_context	<i>Get the parameters that control the scaling of plot elements</i>
------------------	---

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	<i>Show point estimates and errors with markers</i>
-----------	---

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

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

Value

A `reborn_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

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

Value

A `reborn_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 [reborn_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

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

Value

A `reborn_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	<i>Restore matplotlib/ggplot2 defaults</i>
----------------	--

Description

Port of seaborn.reset_defaults / seaborn.reset_orig.

Usage

```
reset_defaults()

reset_orig()
```

Value

Invisibly NULL.

residplot	<i>Plot the residuals of a linear regression</i>
-----------	--

Description

Port of seaborn.residplot. Returns a [reborn_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 `reborn_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)
```

<code>rugplot</code>	<i>Plot marginal rug ticks</i>
----------------------	--------------------------------

Description

Port of `seaborn.rugplot`. Draws small ticks at each observation along the relevant axis. Returns a [reborn_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

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

Value

A `reborn_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

<code>color</code>	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

<code>data</code>	A data frame.
<code>x, y</code>	Column names (strings) or vectors giving the axes.
<code>hue, size, style</code>	Column names/vectors for color, size, and marker-style semantics.
<code>palette, hue_order, hue_norm</code>	Control the color mapping.
<code>sizes, size_order, size_norm</code>	Control the size mapping.
<code>markers, style_order</code>	Control the style (marker) mapping.
<code>legend</code>	"auto", "brief", "full", or FALSE.
<code>...</code>	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	<i>Change how single-letter color codes are interpreted</i>
-----------------	---

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	<i>Independently set the hue, lightness, and/or saturation of a color</i>
----------------	---

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 $[\emptyset, 1]$, or NULL to keep.

Value

A hex color string.

set_palette	<i>Set the matplotlib color cycle / ggplot default discrete palette</i>
-------------	---

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	<i>Set multiple theme parameters in one step</i>
-----------	--

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 unprefix counterpart; see that function for arguments and details.

Value

Each `sns.`-prefixed object is the exact same function as its unprefix 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 unprefix 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

<code>data</code>	A data frame.
<code>x, y</code>	Variables; the categorical one defines the groups.
<code>hue</code>	Grouping variable for color (dodged).
<code>order, hue_order</code>	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 `reborn_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 [reborn_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

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

Value

A `reborn_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	<i>Build a ggplot2 theme replicating a seaborn style + context</i>
---------------	--

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	<i>Draw a violin plot</i>
------------	---------------------------

Description

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

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

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

Value

A `reborn_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](#)

lmpplot, [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