

Package ‘plutor’

July 23, 2025

Title Useful Functions for Visualization

Version 0.1.0

Description In ancient Roman mythology, 'Pluto' was the ruler of the underworld and presides over the afterlife. 'Pluto' was frequently conflated with 'Plutus', the god of wealth, because mineral wealth was found underground. When plotting with R, you try once, twice, practice again and again, and finally you get a pretty figure you want. It's a 'plot tour', a tour about repetition and reward. Hope 'plutor' helps you on the tour!

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Imports baizer, cli, dplyr, ggh4x, ggplot2, ggsci, grDevices, grid, magrittr, methods, purrr, RColorBrewer, repr, rlang, scales, stats, stringr, tibble, tidyr, utils

Suggests covr, svglite, testthat (>= 3.0.0), vdiff

Config/testthat/edition 3

Depends R (>= 2.10)

LazyData true

URL <https://github.com/william-swl/plutor>

BugReports <https://github.com/william-swl/plutor/issues>

NeedsCompilation no

Author William Song [aut, cre]

Maintainer William Song <william_swl@163.com>

Repository CRAN

Date/Publication 2023-10-27 08:00:02 UTC

Contents

assign_colors	3
bioletter_colors	3
brewer_colors	4
canvas_size	4
cm2inch	5
cm2pt	5
extract_compare	6
geom2trace.GeomCompare	6
geom2trace.GeomDescribe	7
GeomCompare	7
GeomDescribe	8
geom_compare	8
geom_describe	10
gradient_colors	12
inch2cm	13
inch2mm	13
lpt	14
mini_diamond	14
mm2inch	15
mm2pt	15
plot_colors	16
pl_init	16
pl_save	17
pl_size	18
PositionFloatxPL	19
PositionFloatyPL	19
position_floatxPL	19
position_floatyPL	20
pt2cm	20
pt2mm	21
revert_pos_scale	21
scale_ele	22
scale_x_continuous_pl	22
scale_x_log10_pl	24
scale_y_continuous_pl	26
scale_y_log10_pl	29
sci_colors	31
StatCompare	31
StatCountPL	32
StatDescribe	32
StatFuncPL	32
StatMeanPL	33
theme_pl	33
theme_pl0	34
tpt	35
trans_pos_scale	35

Index**36**

assign_colors	<i>assign colors by a column in a tibble, for the convenience to use scale_color_identity()</i>
---------------	---

Description

assign colors by a column in a tibble, for the convenience to use scale_color_identity()

Usage

```
assign_colors(df, by, colors = sci_colors("npg", 10), na = "#F5F5F5")
```

Arguments

df	tibble
by	assign colors according to this column
colors	a vector of color values
na	if colors are not enough, fill na values

Value

tibble

Examples

```
assign_colors(mini_diamond, cut, colors = sci_colors("nejm", 8))
```

bioletter_colors	<i>colors of nucleotides and amino acids</i>
------------------	--

Description

colors of nucleotides and amino acids

Usage

```
bioletter_colors
```

Format

```
bioletter_colors:  
colors for biological letters like amino acids or nucleotides
```

Source

according to the print format

brewer_colors	<i>select colors from RColorBrewer package presets</i>
---------------	--

Description

select colors from RColorBrewer package presets

Usage

```
brewer_colors(name, n = 3, ...)
```

Arguments

name	presets name
n	number of colors
...	other arguments of RColorBrewer::brewer.pal

Value

colors

Examples

```
brewer_colors("Blues", 5)
```

canvas_size	<i>width and height of built-in canvas</i>
-------------	--

Description

width and height of built-in canvas

Usage

```
canvas_size
```

Format

```
canvas_size:  
canvas sizes list
```

Source

according to the print format

cm2inch	<i>trans cm to inch</i>
---------	-------------------------

Description

trans cm to inch

Usage

cm2inch(x)

Arguments

x cm value

Value

inch value

Examples

cm2inch(1)

cm2pt	<i>trans cm to pt</i>
-------	-----------------------

Description

trans cm to pt

Usage

cm2pt(x)

Arguments

x cm value

Value

pt value

Examples

cm2pt(1)

extract_compare	<i>extract the result of geom_compare from a ggplot object</i>
-----------------	--

Description

extract the result of geom_compare from a ggplot object

Usage

extract_compare(p)

Arguments

p ggplot object

Value

compare tibble

geom2trace.GeoCompare	<i>geom2trace.GeoCompare</i>
-----------------------	------------------------------

Description

geom2trace.GeoCompare

Usage

geom2trace.GeoCompare(data, params, plot)

Arguments

data, params, plot
 params

Value

no return value

geom2trace.GeoDescribe	<i>geom2trace.GeoDescribe</i>
------------------------	-------------------------------

Description

geom2trace.GeoDescribe

Usage

geom2trace.GeoDescribe(data, params, plot)

Arguments

data, params, plot
 params

Value

no return value

GeomCompare	<i>GeomCompare</i>
-------------	--------------------

Description

GeomCompare

Usage

GeomCompare

Format

An object of class GeomCompare (inherits from Geom, ggproto, gg) of length 6.

GeomDescribe	<i>GeomDescribe</i>
--------------	---------------------

Description

GeomDescribe

Usage

GeomDescribe

Format

An object of class GeomDescribe (inherits from Geom, ggproto, gg) of length 5.

geom_compare	<i>add p value and fold change on a plot</i>
--------------	--

Description

add p value and fold change on a plot

Usage

```
geom_compare(  
  mapping = NULL,  
  data = NULL,  
  stat = "compare",  
  position = "identity",  
  ...,  
  na.rm = FALSE,  
  show.legend = NA,  
  inherit.aes = TRUE,  
  lab_pos = NULL,  
  step_increase = 0.1,  
  tip_length = 0.02,  
  lineend = "round",  
  cp_label = c("psymbol"),  
  ns_lineheight_just = 0.2,  
  ignore_ns = FALSE,  
  fc_method = NULL,  
  comparisons = NULL,  
  paired = FALSE,  
  alternative = "two.sided",  
  test_method = "wilcoxon",
```

```

    ns_symbol = "NS",
    cp_ref = NULL,
    cp_inline = FALSE,
    brackets_widen = 0,
    fc_digits = 2,
    cp_result = NULL,
    cp_manual = NULL
  )

```

Arguments

mapping	Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot(). A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
stat	The statistical transformation to use on the data for this layer, either as a <code>ggproto</code> Geom subclass or as a string naming the stat stripped of the <code>stat_</code> prefix (e.g. "count" rather than "stat_count")
position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use position_jitter), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
...	Other arguments passed on to <code>ggplot2::geom_segment()</code> .
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If <code>FALSE</code> , overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. borders() .
lab_pos	position of the label brackets
step_increase	the increase height for next bracket, a ratio according to the whole panel height
tip_length	the length for tips at the ends of the brackets, a ratio according to the whole panel height
lineend	Line end style (round, butt, square).
cp_label	which values will be add on the plot, a character vector with some of <code>psymbol</code> , <code>p</code> , <code>right_deno_fc</code> , <code>left</code> in it. If <code>comparisons</code> is assigned, you can also include <code>fc1</code> , <code>fc2</code>

ns_lineheight_just	if show psymbol in the label, justify the NS labels to make the lineheights look balanced
ignore_ns	if TRUE will ignore all label items if $p \geq 0.05$, or you can assign a character vector like cp_label to ignore some items of the label
fc_method	fold change method, default is mean. If you use log10 or log2 axis, default is geom_mean.
comparisons	a list of two-element vector, to assign the comparisons should be performed
paired	paired test or not, FALSE as default. If TRUE, you should use mapping=aes(paired_by=col) to indicate pairs by an extra column
alternative	one of two.sided, greater, less
test_method	wilcoxon as default, one of wilcoxon, t
ns_symbol	the symbol of non-significant, NS as default
cp_ref	reference item, the others will be compared with it
cp_inline	draw in line or not, default is FALSE
brackets_widen	widen the brackets, can be a negative value
fc_digits	fold change digits
cp_result	comparation result tibble
cp_manual	manual comparisons table, please refer to extract_compare()

Value

ggplot object

geom_describe	<i>Description values plot</i>
---------------	--------------------------------

Description

The describe geom is used to create description values plot, including center symbol and error symbol. The center symbol can be mean, median or other custom functions, the error symbol can be sd, quantile or other custom functions.

Usage

```
geom_describe(
  mapping = NULL,
  data = NULL,
  stat = "describe",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
```

```

    lineend = "round",
    show_error = TRUE,
    center_symbol = "bar",
    center_width = 0.3,
    error_width = 0.2,
    center_func = mean,
    low_func = function(x, na.rm) {
      mean(x, na.rm = na.rm) - sd(x, na.rm = na.rm)
    },
    high_func = function(x, na.rm) {
      mean(x, na.rm = na.rm) + sd(x, na.rm = na.rm)
    },
    ...
  )

```

Arguments

mapping	Set of aesthetic mappings created by aes() . If specified and <code>inherit.aes = TRUE</code> (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If <code>NULL</code>, the default, the data is inherited from the plot data as specified in the call to ggplot(). A <code>data.frame</code>, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a <code>data.frame</code>, and will be used as the layer data. A function can be created from a formula (e.g. <code>~ head(.x, 10)</code>).
stat	The statistical transformation to use on the data for this layer, either as a ggproto <code>Geom</code> subclass or as a string naming the stat stripped of the <code>stat_</code> prefix (e.g. "count" rather than "stat_count")
position	Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use <code>position_jitter</code>), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.
na.rm	If <code>FALSE</code> , the default, missing values are removed with a warning. If <code>TRUE</code> , missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? <code>NA</code> , the default, includes if any aesthetics are mapped. <code>FALSE</code> never includes, and <code>TRUE</code> always includes. It can also be a named logical vector to finely select the aesthetics to display.
inherit.aes	If <code>FALSE</code> , overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. borders() .
lineend	Line end style (round, butt, square).
show_error	show error symbol
center_symbol	one of point, bar

<code>center_width</code>	if <code>center_symbol='bar'</code> , the width of the bar
<code>error_width</code>	the width of the error bar
<code>center_func</code>	the center function, mean as default
<code>low_func</code>	the low error function, mean minus sd as default
<code>high_func</code>	the high error function, mean plus sd as default
<code>...</code>	Other arguments passed on to <code>ggplot2::point()</code> or <code>ggplot2::geom_segment</code> .

Value

ggplot object

<code>gradient_colors</code>	<i>generate gradient colors</i>
------------------------------	---------------------------------

Description

generate gradient colors

Usage

`gradient_colors(x, n)`

Arguments

<code>x</code>	colors
<code>n</code>	number of colors to output

Value

gradient colors

Examples

`gradient_colors(c("blue", "red"), 10)`

inch2cm	<i>trans inch to cm</i>
---------	-------------------------

Description

trans inch to cm

Usage

inch2cm(x)

in2cm(x)

cm2in(x)

Arguments

x inch value

Value

cm value

Examples

inch2cm(1)

inch2mm	<i>trans inch to mm</i>
---------	-------------------------

Description

trans inch to mm

Usage

inch2mm(x)

in2mm(x)

Arguments

x inch value

Value

mm value

Examples

inch2mm(1)

lpt	<i>trans geom line point and theme line point to the real point</i>
-----	---

Description

trans geom line point and theme line point to the real point

Usage

lpt(x)

Arguments

x line point in geom or theme

Value

real point

Examples

lpt(1)

mini_diamond	<i>Minimal tibble dataset adjusted from diamond</i>
--------------	---

Description

Minimal tibble dataset adjusted from diamond

Usage

mini_diamond

Format

mini_diamond:
A data frame with 100 rows and 7 columns:
id unique id
cut, clarity 2 category variables
carat, price, x, y 4 continuous variables ...

Source

adjusted from ggplot2

mm2inch	<i>trans mm to inch</i>
---------	-------------------------

Description

trans mm to inch

Usage

mm2inch(x)

mm2in(x)

Arguments

x mm value

Value

inch value

Examples

mm2inch(1)

mm2pt	<i>trans mm to pt</i>
-------	-----------------------

Description

trans mm to pt

Usage

mm2pt(x)

Arguments

x mm value

Value

pt value

Examples

mm2pt(1)

plot_colors	<i>plot colors</i>
-------------	--------------------

Description

plot colors

Usage

```
plot_colors(x, ncol = 10, show_name = TRUE)
```

Arguments

- x color values
- ncol color number of each row
- show_name use vector names as label, FALSE to show the color value

Value

ggplot object

Examples

```
plot_colors(gradient_colors(c("blue", "red"), 10))
```

pl_init	<i>set size, resolution and default theme</i>
---------	---

Description

set size, resolution and default theme

Usage

```
pl_init(  
  width = 4,  
  height = 3,  
  res = 300,  
  w = NULL,  
  h = NULL,  
  theme = theme_pl()  
)
```

Arguments

width	width
height	height
res	resolution, 300 as default
w	alias of width
h	alias of height
theme	default theme

Value

no return value

Examples

```
pl_init()
```

pl_save	<i>save plot, support save into a blank canvas</i>
---------	--

Description

save plot, support save into a blank canvas

Usage

```
pl_save(  
  plot,  
  filename,  
  width,  
  height,  
  units = "in",  
  canvas = NULL,  
  canvas_pos_x = 0.5,  
  canvas_pos_y = 0.1,  
  ...  
)
```

Arguments

plot	ggplot object
filename	filename
width	plot width
height	plot height
units	units, 'in' for inch as default. Can be 'in', 'cm'

canvas	NULL as default, pass character to use built-in canvas ('A4', 'A4v'), or pass a numeric vector in 'c(width, heigh)' form
canvas_pos_x	from 0 to 1, the horizontal position of plot in canvas
canvas_pos_y	from 0 to 1, the vertical position of plot in canvas
...	other arguments from ggsave

Value

no return value

<i>pl_size</i>	<i>set repr size and resolution</i>
----------------	-------------------------------------

Description

set repr size and resolution

Usage

`pl_size(width = 4, height = 3, res = 300, w = NULL, h = NULL)`

Arguments

width	width
height	height
res	resolution, 300 as default
w	alias of width
h	alias of height

Value

no return value

Examples

`pl_size(width = 3, height = 2)`

PositionFloatxPL	<i>PositionFloatxPL</i>
------------------	-------------------------

Description

PositionFloatxPL

Usage

PositionFloatxPL

Format

An object of class PositionFloatxPL (inherits from Position, ggproto, gg) of length 4.

PositionFloatyPL	<i>PositionFloatyPL</i>
------------------	-------------------------

Description

PositionFloatyPL

Usage

PositionFloatyPL

Format

An object of class PositionFloatyPL (inherits from Position, ggproto, gg) of length 4.

position_floatxPL	<i>a new Position object to create float x position</i>
-------------------	---

Description

a new Position object to create float x position

Usage

position_floatxPL(float = -0.05, cycle = 2)

Arguments

- | | |
|-------|--|
| float | float range, a ratio according to the whole panel height |
| cycle | float cycle |

Value

Position object

position_floatyPL	<i>a new Position object to create float y position</i>
-------------------	---

Description

a new Position object to create float y position

Usage

```
position_floatyPL(float = -0.05, cycle = 2)
```

Arguments

float	float range, a ratio according to the whole panel height
cycle	float cycle

Value

Position object

pt2cm	<i>trans pt to cm</i>
-------	-----------------------

Description

trans pt to cm

Usage

```
pt2cm(x)
```

Arguments

x	pt value
---	----------

Value

cm value

Examples

```
pt2cm(1)
```

pt2mm	<i>trans pt to mm</i>
-------	-----------------------

Description

trans pt to mm

Usage

```
pt2mm(x)
```

Arguments

x	pt value
---	----------

Value

mm value

Examples

```
pt2mm(1)
```

revert_pos_scale	<i>revert the position scale transformation</i>
------------------	---

Description

revert the position scale transformation

Usage

```
revert_pos_scale(s)
```

Arguments

s	ScaleContinuousPosition object, e.g. scales\$y in compute_group()
---	---

Value

function

scale_ele	<i>scale element according to a vector of element scales</i>
-----------	--

Description

scale element according to a vector of element scales

Usage

```
scale_ele(level, base, ele_scales)
```

Arguments

level	output level
base	value of base level
ele_scales	vector of element scales

Value

value of output level

Examples

```
scale_ele(level = 2, base = 5, ele_scales = c(1, 2))
```

scale_x_continuous_pl	<i>A variant of scale_x_continuous() to show axis minor breaks</i>
-----------------------	--

Description

A variant of scale_x_continuous() to show axis minor breaks

Usage

```
scale_x_continuous_pl(
  name = waiver(),
  breaks = waiver(),
  minor_breaks = NULL,
  n.breaks = NULL,
  labels = waiver(),
  limits = NULL,
  expand = ggplot2::expansion(),
  oob = scales::oob_keep,
  na.value = NA_real_,
  trans = "identity",
```

```

    guide = ggh4x::guide_axis_minor(),
    position = "bottom",
    sec.axis = waiver(),
    show_minor_breaks = TRUE,
    minor_break_step = NULL
  )

```

Arguments

name	The name of the scale. Used as the axis or legend title. If <code>waiver()</code> , the default, the name of the scale is taken from the first mapping used for that aesthetic. If <code>NULL</code> , the legend title will be omitted.
breaks	One of: • <code>NULL</code> for no breaks • <code>waiver()</code> for the default breaks computed by the transformation object • A numeric vector of positions • A function that takes the limits as input and returns breaks as output (e.g., a function returned by <code>scales::extended_breaks()</code>). Also accepts <code>rlang</code> lambda function notation.
minor_breaks	One of: • <code>NULL</code> for no minor breaks • <code>waiver()</code> for the default breaks (one minor break between each major break) • A numeric vector of positions • A function that given the limits returns a vector of minor breaks. Also accepts <code>rlang</code> lambda function notation.
n.breaks	An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if <code>breaks = waiver()</code> . Use <code>NULL</code> to use the default number of breaks given by the transformation.
labels	One of: • <code>NULL</code> for no labels • <code>waiver()</code> for the default labels computed by the transformation object • A character vector giving labels (must be same length as breaks) • An expression vector (must be the same length as breaks). See <code>?plotmath</code> for details. • A function that takes the breaks as input and returns labels as output. Also accepts <code>rlang</code> lambda function notation.
limits	One of: • <code>NULL</code> to use the default scale range • A numeric vector of length two providing limits of the scale. Use <code>NA</code> to refer to the existing minimum or maximum

	• A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will remove data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see coord_cartesian()).
expand	use <code>expansion()</code> to dismiss the blank between x axis low limit and y axis
oob	use <code>scales::oob_keep</code> instead of <code>scales::oob_censor</code> , which will always consider the data points out of the limits
na.value	Missing values will be replaced with this value.
trans	For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time". A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called <code><name>_trans</code> (e.g., <code>scales::boxcox_trans()</code>). You can create your own transformation with <code>scales::trans_new()</code>.
guide	A function used to create a guide or its name. See guides() for more information.
position	For position scales, The position of the axis. left or right for y axes, top or bottom for x axes.
sec.axis	sec_axis() is used to specify a secondary axis.
show_minor_breaks	show minor breaks or not
minor_break_step	the step of minor breaks

Value

scale object

scale_x_log10_pl	<i>A variant of <code>scale_x_log10()</code> to show axis minor breaks and better axis labels</i>
------------------	---

Description

A variant of `scale_x_log10()` to show axis minor breaks and better axis labels

Usage

```
scale_x_log10_pl(
  name = waiver(),
  breaks = NULL,
  minor_breaks = NULL,
  n.breaks = NULL,
  labels = NULL,
  limits = NULL,
  expand = ggplot2::expansion(),
  oob = scales::oob_keep,
  na.value = NA_real_,
  trans = scales::log10_trans(),
  guide = ggh4x::guide_axis_minor(),
  position = "bottom",
  sec.axis = waiver(),
  show_minor_breaks = TRUE
)
```

Arguments

name	The name of the scale. Used as the axis or legend title. If <code>waiver()</code> , the default, the name of the scale is taken from the first mapping used for that aesthetic. If <code>NULL</code> , the legend title will be omitted.
breaks	One of: • <code>NULL</code> for no breaks • <code>waiver()</code> for the default breaks computed by the transformation object • A numeric vector of positions • A function that takes the limits as input and returns breaks as output (e.g., a function returned by <code>scales::extended_breaks()</code>). Also accepts <code>rlang</code> lambda function notation.
minor_breaks	One of: • <code>NULL</code> for no minor breaks • <code>waiver()</code> for the default breaks (one minor break between each major break) • A numeric vector of positions • A function that given the limits returns a vector of minor breaks. Also accepts <code>rlang</code> lambda function notation.
n.breaks	An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if <code>breaks = waiver()</code> . Use <code>NULL</code> to use the default number of breaks given by the transformation.
labels	One of: • <code>NULL</code> for no labels • <code>waiver()</code> for the default labels computed by the transformation object • A character vector giving labels (must be same length as breaks)

	• An expression vector (must be the same length as breaks). See <code>?plotmath</code> for details. • A function that takes the breaks as input and returns labels as output. Also accepts rlang <code>lambda</code> function notation.
limits	One of: • NULL to use the default scale range • A numeric vector of length two providing limits of the scale. Use NA to refer to the existing minimum or maximum • A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang <code>lambda</code> function notation. Note that setting limits on positional scales will remove data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see <code>coord_cartesian()</code>).
expand	use <code>expansion()</code> to dismiss the blank between x axis low limit and y axis
oob	use <code>scales::oob_keep</code> instead of <code>scales::oob_censor</code> , which will always consider the data points out of the limits
na.value	Missing values will be replaced with this value.
trans	For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time". A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called <code><name>_trans</code> (e.g., <code>scales::boxcox_trans()</code>). You can create your own transformation with <code>scales::trans_new()</code>.
guide	A function used to create a guide or its name. See <code>guides()</code> for more information.
position	For position scales, The position of the axis. left or right for y axes, top or bottom for x axes.
sec.axis	<code>sec_axis()</code> is used to specify a secondary axis.
show_minor_breaks	show minor breaks or not

Value

scale object

`scale_y_continuous_pl` A variant of `scale_y_continuous()` to show axis minor breaks

Description

A variant of `scale_y_continuous()` to show axis minor breaks

Usage

```
scale_y_continuous_pl(
  name = waiver(),
  breaks = waiver(),
  minor_breaks = NULL,
  n.breaks = NULL,
  labels = waiver(),
  limits = NULL,
  expand = ggplot2::expansion(),
  oob = scales::oob_keep,
  na.value = NA_real_,
  trans = "identity",
  guide = ggh4x::guide_axis_minor(),
  position = "left",
  sec.axis = waiver(),
  show_minor_breaks = TRUE,
  minor_break_step = NULL
)
```

Arguments

name	The name of the scale. Used as the axis or legend title. If <code>waiver()</code> , the default, the name of the scale is taken from the first mapping used for that aesthetic. If <code>NULL</code> , the legend title will be omitted.
breaks	One of: • <code>NULL</code> for no breaks • <code>waiver()</code> for the default breaks computed by the transformation object • A numeric vector of positions • A function that takes the limits as input and returns breaks as output (e.g., a function returned by <code>scales::extended_breaks()</code>). Also accepts <code>rlang</code> lambda function notation.
minor_breaks	One of: • <code>NULL</code> for no minor breaks • <code>waiver()</code> for the default breaks (one minor break between each major break) • A numeric vector of positions • A function that given the limits returns a vector of minor breaks. Also accepts <code>rlang</code> lambda function notation.
n.breaks	An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if <code>breaks = waiver()</code> . Use <code>NULL</code> to use the default number of breaks given by the transformation.
labels	One of: • <code>NULL</code> for no labels • <code>waiver()</code> for the default labels computed by the transformation object

	• A character vector giving labels (must be same length as breaks) • An expression vector (must be the same length as breaks). See <code>?plotmath</code> for details. • A function that takes the breaks as input and returns labels as output. Also accepts <code>rlang::lambda</code> function notation.
<code>limits</code>	One of: • <code>NULL</code> to use the default scale range • A numeric vector of length two providing limits of the scale. Use <code>NA</code> to refer to the existing minimum or maximum • A function that accepts the existing (automatic) limits and returns new limits. Also accepts <code>rlang::lambda</code> function notation. Note that setting limits on positional scales will remove data outside of the limits. If the purpose is to zoom, use the <code>limit</code> argument in the coordinate system (see <code>coord_cartesian()</code>).
<code>expand</code>	use <code>expansion()</code> to dismiss the blank between y axis low limit and x axis
<code>oob</code>	use <code>scales::oob_keep</code> instead of <code>scales::oob_censor</code> , which will always consider the data points out of the limits
<code>na.value</code>	Missing values will be replaced with this value.
<code>trans</code>	For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time". A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the <code>scales</code> package, and are called <code><name>_trans</code> (e.g., <code>scales::boxcox_trans()</code>). You can create your own transformation with <code>scales::trans_new()</code>.
<code>guide</code>	A function used to create a guide or its name. See <code>guides()</code> for more information.
<code>position</code>	For position scales, The position of the axis. <code>left</code> or <code>right</code> for y axes, <code>top</code> or <code>bottom</code> for x axes.
<code>sec.axis</code>	<code>sec_axis()</code> is used to specify a secondary axis.
<code>show_minor_breaks</code>	show minor breaks or not
<code>minor_break_step</code>	the step of minor breaks

Value

scale object

scale_y_log10_pl	<i>A variant of scale_y_log10() to show axis minor breaks and better axis labels</i>
------------------	--

Description

A variant of scale_y_log10() to show axis minor breaks and better axis labels

Usage

```
scale_y_log10_pl(
  name = waiver(),
  breaks = NULL,
  minor_breaks = NULL,
  n.breaks = NULL,
  labels = NULL,
  limits = NULL,
  expand = ggplot2::expansion(),
  oob = scales::oob_keep,
  na.value = NA_real_,
  trans = scales::log10_trans(),
  guide = ggh4x::guide_axis_minor(),
  position = "left",
  sec.axis = waiver(),
  show_minor_breaks = TRUE
)
```

Arguments

name	The name of the scale. Used as the axis or legend title. If waiver(), the default, the name of the scale is taken from the first mapping used for that aesthetic. If NULL, the legend title will be omitted.
breaks	One of: • NULL for no breaks • waiver() for the default breaks computed by the transformation object • A numeric vector of positions • A function that takes the limits as input and returns breaks as output (e.g., a function returned by scales::extended_breaks()). Also accepts rlang lambda function notation.
minor_breaks	One of: • NULL for no minor breaks • waiver() for the default breaks (one minor break between each major break) • A numeric vector of positions

	• A function that given the limits returns a vector of minor breaks. Also accepts rlang lambda function notation.
n.breaks	An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if <code>breaks = waiver()</code> . Use NULL to use the default number of breaks given by the transformation.
labels	One of: • NULL for no labels • <code>waiver()</code> for the default labels computed by the transformation object • A character vector giving labels (must be same length as breaks) • An expression vector (must be the same length as breaks). See <code>?plotmath</code> for details. • A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation.
limits	One of: • NULL to use the default scale range • A numeric vector of length two providing limits of the scale. Use NA to refer to the existing minimum or maximum • A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will remove data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see coord_cartesian()).
expand	use <code>expansion()</code> to dismiss the blank between y axis low limit and x axis
oob	use <code>scales::oob_keep</code> instead of <code>scales::oob_censor</code> , which will always consider the data points out of the limits
na.value	Missing values will be replaced with this value.
trans	For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time". A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called <code><name>_trans</code> (e.g., <code>scales::boxcox_trans()</code>). You can create your own transformation with <code>scales::trans_new()</code>.
guide	A function used to create a guide or its name. See guides() for more information.
position	For position scales, The position of the axis. left or right for y axes, top or bottom for x axes.
sec.axis	sec_axis() is used to specify a secondary axis.
show_minor_breaks	show minor breaks or not

Value

scale object

sci_colors	<i>select colors from ggsci package presets</i>
------------	---

Description

select colors from ggsci package presets

Usage

```
sci_colors(name, n = 3, alpha = 1)
```

Arguments

name	presets name
n	number of colors
alpha	alpha

Value

colors

Examples

```
sci_colors("npg", 5)
```

StatCompare	<i>StatCompare</i>
-------------	--------------------

Description

StatCompare

Usage

StatCompare

Format

An object of class StatCompare (inherits from Stat, ggproto, gg) of length 6.

StatCountPL	<i>StatCountPL</i>
-------------	--------------------

Description

StatCountPL

Usage

StatCountPL

Format

An object of class StatCountPL (inherits from Stat, ggproto, gg) of length 4.

StatDescribe	<i>StatDescribe</i>
--------------	---------------------

Description

StatDescribe

Usage

StatDescribe

Format

An object of class StatDescribe (inherits from Stat, ggproto, gg) of length 4.

StatFuncPL	<i>StatFuncPL</i>
------------	-------------------

Description

StatFuncPL

Usage

StatFuncPL

Format

An object of class StatFuncPL (inherits from Stat, ggproto, gg) of length 4.

StatMeanPL	<i>StatMeanPL</i>
------------	-------------------

Description

StatMeanPL

Usage

StatMeanPL

Format

An object of class StatMeanPL (inherits from Stat, ggproto, gg) of length 4.

theme_pl	<i>a new extensible theme</i>
----------	-------------------------------

Description

a new extensible theme

Usage

```
theme_pl(  
  base_size = 10,  
  base_line_size = lpt(base_size/10),  
  base_rect_size = lpt(1),  
  size_scales = c(5, 6, 7),  
  margin_factor = 0.25,  
  plot_margin_factor = 0.5,  
  legend_spacing_factor = 1.2,  
  font_family = "",  
  ...  
)
```

Arguments

- base_size base size of fonts and margins
- base_line_size base linewidth
- base_rect_size base linewidth of the rectangles
- size_scales a vector of element size scales, namely:
 - 1. base size, used by legend text, axis text, caption
 - 2. used by legend title, axis title, strip text (facet title), subtitle

- 3. used by title, tag
- margin_factor factor to adjust the element margins according to size_scales
- plot_margin_factor factor to adjust the plot margins according to size_scales[3]
- legend_spacing_factor factor to adjust the space of legend items according to size_scales[2]
- font_family font family
- ... arguments from ggplot2::theme()

Value

theme object of ggplot

Examples

theme_pl()

theme_pl0	<i>a blank theme</i>
-----------	----------------------

Description

a blank theme

Usage

theme_pl0(...)

Arguments

... arguments of theme_pl()

Value

theme object of ggplot

Examples

theme_pl0()

tpt	<i>trans geom text or point to the real point</i>
-----	---

Description

trans geom text or point to the real point

Usage

tpt(x)

ppt(x)

Arguments

x text point in geom

Value

real point

Examples

tpt(1)

trans_pos_scale	<i>perform the position scale transformation</i>
-----------------	--

Description

perform the position scale transformation

Usage

trans_pos_scale(s)

Arguments

s ScaleContinuousPosition object, e.g. scales\$y in compute_group()

Value

function

Index

* datasets

- bioletter_colors, 3
 - canvas_size, 4
 - GeomCompare, 7
 - GeomDescribe, 8
 - mini_diamond, 14
 - PositionFloatxPL, 19
 - PositionFloatyPL, 19
 - StatCompare, 31
 - StatCountPL, 32
 - StatDescribe, 32
 - StatFuncPL, 32
 - StatMeanPL, 33
- aes(), 9, 11
- assign_colors, 3
- bioletter_colors, 3
- borders(), 9, 11
- brewer_colors, 4
- canvas_size, 4
- cm2in (inch2cm), 13
- cm2inch, 5
- cm2pt, 5
- coord_cartesian(), 24, 26, 28, 30
- extract_compare, 6
- fortify(), 9, 11
- geom2trace.GeomCompare, 6
- geom2trace.GeomDescribe, 7
- geom_compare, 8
- geom_describe, 10
- GeomCompare, 7
- GeomDescribe, 8
- ggplot(), 9, 11
- gradient_colors, 12
- guides(), 24, 26, 28, 30
- in2cm (inch2cm), 13
- in2mm (inch2mm), 13
- inch2cm, 13
- inch2mm, 13
- lambda, 23–30
- lpt, 14
- mini_diamond, 14
- mm2in (mm2inch), 15
- mm2inch, 15
- mm2pt, 15
- pl_init, 16
- pl_save, 17
- pl_size, 18
- plot_colors, 16
- position_floatxPL, 19
- position_floatyPL, 20
- PositionFloatxPL, 19
- PositionFloatyPL, 19
- ppt (tpt), 35
- pt2cm, 20
- pt2mm, 21
- revert_pos_scale, 21
- scale_ele, 22
- scale_x_continuous_pl, 22
- scale_x_log10_pl, 24
- scale_y_continuous_pl, 26
- scale_y_log10_pl, 29
- scales::boxcox_trans(), 24, 26, 28, 30
- scales::extended_breaks(), 23, 25, 27, 29
- scales::trans_new(), 24, 26, 28, 30
- sci_colors, 31
- sec_axis(), 24, 26, 28, 30
- StatCompare, 31
- StatCountPL, 32
- StatDescribe, 32
- StatFuncPL, 32

StatMeanPL, [33](#)

theme_pl, [33](#)

theme_pl0, [34](#)

tpt, [35](#)

trans_pos_scale, [35](#)

transformation object, [23](#), [25](#), [27](#), [29](#)