Package 'ezplot'

November 26, 2022

Type Package

Title Functions for Common Chart Types

Version 0.7.5

Author Wojtek Kostelecki

Maintainer Wojtek Kostelecki <wojtek.kostelecki@gmail.com>

Description Wrapper for the 'ggplot2' package that creates a variety of common charts (e.g. bar, line, area, ROC, waterfall, pie) while aiming to reduce typing.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.3)

RoxygenNote 7.2.2

Imports dplyr, forcats, ggplot2, lubridate, rlang

Suggests covr, DT, e1071, knitr, methods, miniUI, rmarkdown, ROCR, shiny, stats, testthat, tibble, tidyr, tsibble, tsibbledata

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2022-11-26 22:10:02 UTC

R topics documented:

`agg_data` *Aggregates data*

Description

Aggregates data

Usage

```
agg_data(
  data,
  cols = names(data),
  group_by = NULL,
  agg_fun = function(x) sum(x, na.rm = TRUE),
  group_by2 = NULL,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
cols	Named character vector of column names.
group_by	Vector of grouping columns.
agg_fun	Function to use for aggregating.
group_by2	Vector of grouping column names to use for delayed (post aggregation) calculation.
env	Environment for extra variables.

Value

An aggregated data.frame.

Examples

```
suppressPackageStartupMessages(library(tsibble))
library(tsibbledata)
agg_data(ansett, c("Passengers", count = "1"))
agg_data(ansett["Class"])
agg_data(ansett[c("Class", "Passengers")])
agg_data(ansett, "Passengers", "Class")
agg_data(ansett, "Passengers", c("Class", "Airports"))
agg_data(ansett, c(x = "Airports", y = "Passengers"), c(x = "Airports"))
agg_data(ansett, c(x = "Class", y = "1", group = "Airports"), c(x = "Class", group = "Airports"))
```

---

area_plot *area_plot*

Description

Aggregates a data.frame and creates a stacked area chart.

Usage

```
area_plot(
  data,
  x,
  y = "1",
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  size = 11,
  reorder = c("group", "facet_x", "facet_y"),
  palette = ez_col,
  labels_y = if (position == "fill") {
     function(x) ez_labels(100 * x, append =
    "%")
 } else {
     ez_labels
 },
  labels_x = NULL,
  use_theme = theme_ez,
  position = c("stack", "fill"),
  facet_scales = "fixed",
  facet_ncol = NULL,
  legend_ncol = NULL,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
size	theme size for `use_theme()`. Default is 14.
reorder	A character vector specifying the group variables to reorder. Default is `c("group", "facet_x", "facet_y")`.
palette	Colour function.
labels_y	label formatting function
labels_x	label formatting function
use_theme	ggplot theme function
position	Either `"stack"` (default) or `"fill"`
facet_scales	Option passed to scales argument in `facet_wrap` or `facet_grid`. Default is `"fixed"`.

facet_ncol	Option passed to ncol argument in facet_wrap or facet_grid. Default is NULL.
legend_ncol	Number of columns in legend.
env	environment for evaluating expressions.

Value

A ggplot object.

Examples

```
library(tsibble)
library(tsibbledata)
area_plot(ansett, x = "as.Date(Week)", y = "Passengers")
area_plot(ansett,
          x = "as.Date(Week)", y = c("Weekly Passengers" = "Passengers"), "Class")
area_plot(ansett, "as.Date(Week)",
          y = c("Weekly Passengers" = "Passengers"),
          group = "substr(Airports, 5, 7)",
          facet_x = "substr(Airports, 1, 3)",
          facet_y = "Class",
          facet_scales = "free_y")
```

---

bar_plot *bar_plot*

Description

bar_plot

Usage

```
bar_plot(
  data,
  x,
  y = "1",
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  size = 11,
  width = NULL,
  reorder = c("group", "facet_x", "facet_y"),
  palette = ez_col,
  labels_y = if (position == "fill") {
     function(x) ez_labels(100 * x, append =
    "%")
 } else {
```

```
    ez_labels
 },
  labels_x = identity,
  label_pos = c("auto", "inside", "top", "both", "none"),
  rescale_y = 1.1,
  label_cutoff = 0.12,
  use_theme = theme_ez,
  position = "stack",
  facet_scales = "fixed",
  legend_ncol = NULL,
  coord_flip = FALSE
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
size	theme size for `use_theme()`. Default is 14.
width	Width of bar.
reorder	A character vector specifying the group variables to reorder. Default is `c("group", "facet_x", "facet_y")`.
palette	Colour function.
labels_y	label formatting function
labels_x	label formatting function
label_pos	Position of labels. Can be "auto", "inside", "top", "both" or "none".
rescale_y	Rescaling factor for y-axis limits
label_cutoff	Cutoff size (proportion of y data range) for excluding labels
use_theme	ggplot theme function
position	Either `"stack"` (default) or `"fill"`
facet_scales	Option passed to scales argument in `facet_wrap` or `facet_grid`. Default is `"fixed"`.
legend_ncol	Number of columns in legend.
coord_flip	logical (default is FALSE). If TRUE, flips the x and y coordinate using ggplot2::coord_flip()

Value

A ggplot object.

Examples

```
library(tsibble)
library(tsibbledata)
library(lubridate)
bar_plot(ansett, "year(Week)", "Passengers", size = 16)
bar_plot(ansett, "year(Week)", "Passengers", "Class")
bar_plot(ansett, "Airports", c("Share of Passengers" = "Passengers"), "Class", position = "fill")
bar_plot(ansett, "Airports", "Passengers", "Class", reorder = NULL, label_pos = "both")
bar_plot(ansett, "Airports",
         c(Passengers = "ifelse(Class == 'Economy', Passengers, -Passengers)"),
         "Class", label_pos = "both")
bar_plot(ansett, "year(Week)", "Passengers", "Class", label_pos = "both", coord_flip = TRUE)
```

`calendar_plot` *calendar_plot*

Description

calendar_plot

Usage

```
calendar_plot(data, x, y, ...)
```

Arguments

`data`	A data.frame.
`x`	date column
`y`	A named character value. Evaluates to a column.
`...`	additional arguments for tile_plot

Examples

```
library(tsibbledata)
calendar_plot(vic_elec, "Time", "Demand", zlim = c(NA, NA))
```

density_plot *density_plot*

Description

creates a density plot

Usage

```
density_plot(
  data,
  x,
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  palette = ez_col,
  adjust = 1,
  alpha = 0.5,
  facet_scales = "fixed",
  facet_ncol = NULL,
  legend_ncol = NULL,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
palette	Colour function.
adjust	multiplicate bandwidth adjustment
alpha	alpha
facet_scales	Option passed to scales argument in facet_wrap or facet_grid. Default is "fixed".
facet_ncol	Option passed to ncol argument in facet_wrap or facet_grid. Default is NULL.
legend_ncol	Number of columns in legend.
env	environment for evaluating expressions.

Examples

```
library(tsibbledata)
density_plot(mtcars, "wt", "cyl")
density_plot(subset(tsibbledata::olympic_running, Length == 100 & Year >= 1980),
         "Time", "Year - Year %% 10", "Sex", facet_scales = "free", facet_ncol = 1, adjust = 2)
```

distribution_plot *distribution_plot*

Description

distribution_plot

Usage

```
distribution_plot(
  data,
  x,
  facet_x = NULL,
  nbins = 20,
  use_theme = theme_ez,
  size = 11,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
nbins	Number of bins for histogram. Default is 20.
use_theme	ggplot theme function
size	theme size for use_theme(). Default is 14.
env	environment for evaluating expressions.

Examples

```
n = 100
df = data.frame(residuals = rnorm(n),
                group1 = sample(c("a", "b"), n, replace = TRUE))
distribution_plot(df, "residuals")
distribution_plot(df, "residuals", "group1")
```

ez_app *ez_app*

Description

ez_app

Usage

```
ez_app(data = NULL)
```

Arguments

data	A data frame

Examples

```
## Not run:
library(tsibble)
library(tsibbledata)
ez_app(ansett)

## End(Not run)
```

ez_col *Color palette interpolation*

Description

Color palette interpolation

Usage

```
ez_col(n = 50, palette = NULL)
```

Arguments

n	number of colours
palette	palette to interpolate from

Value

rgb

Examples

```
ez_col(15)
ez_col(2, c("blue", "red"))
ez_col(3, c("blue", "red"))
```

ez_jet *ez_jet*

Description

color palette for

Usage

```
ez_jet(
  n = 100,
 palette = c("dodgerblue4", "steelblue2", "olivedrab3", "darkgoldenrod1", "brown")
)
```

Arguments

n	Number of colours to return.
palette	Vector of colours.

Examples

```
ez_jet(100)
ez_jet(1)
```

ez_labels *Function for formatting numeric labels*

Description

Function for formatting numeric labels

Usage

```
ez_labels(
  x,
  prepend = "",
  append = "",
  as_factor = FALSE,
  round = Inf,
  signif = Inf
)
```

Arguments

x	numeric
prepend	character
append	character
as_factor	logical
round	numeric passed to round()
signif	numeric passed to signif()

Value

y

Examples

```
ez_labels(10^(0:10))
ez_labels(2000, append = " apples")
ez_labels(0:10, append = " apples", as_factor = TRUE)
ez_labels(c(0, 0.1, 0.01, 0.001, 0.0001))
```

---

ez_png *ez_png*

Description

Saves ggplot or ezplot objects to png (with useful defaults).

Usage

```
ez_png(
  g,
  file,
  width = 1200,
  height = 600,
  res = 72,
  resx = 1,
  ...,
  vp = NULL,
  dir.create = FALSE,
  check = TRUE
)
```

Arguments

g	A ggplot or ezplot object.
file	A png file path.
width	Image width (in pixels). Default is 1200.
height	Image height (in pixels). Default is 600.
res	Resolution (PPI) of output image. Default is 72.
resx	Resolution multiplier. Default is 1.
...	Further arguments to pass to png().
vp	A viewport object created with grid::viewport.
dir.create	Logical. If TRUE, creates the directory to save into. Default is FALSE.
check	Logical. If TRUE, opens png file after saving. Default is TRUE.

---

ez_server *ez_server*

Description

ez_server

Usage

```
ez_server(data)
```

Arguments

data	A data frame

---

ez_ui *ez_ui*

Description

ez_ui

Usage

```
ez_ui(data)
```

Arguments

data	A data frame

get_incr *get_incr*

Description

returns the minimum increment between sorted unique values of a vector

Usage

```
get_incr(x)
```

Arguments

x	A numeric or date vector

histogram_plot *histogram_plot*

Description

creates a histogram plot

Usage

```
histogram_plot(
  data,
  x,
  y = "count",
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  palette = ez_col,
  position = "stack",
  bins = 30,
  alpha = 0.5,
  facet_scales = "fixed",
  facet_ncol = NULL,
  legend_ncol = NULL,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
palette	Colour function.
position	Either "stack" (default) or "fill"
bins	number of bins
alpha	fill alpha
facet_scales	Option passed to scales argument in facet_wrap or facet_grid. Default is "fixed".
facet_ncol	Option passed to ncol argument in facet_wrap or facet_grid. Default is NULL.
legend_ncol	Number of columns in legend.
env	environment for evaluating expressions.

Examples

```
histogram_plot(airquality, "Wind", group = "Month")
histogram_plot(airquality, "Wind", "density", facet_x = "Month")
```

---

ks_plot *ks_plot*

Description

ks plot

Usage

```
ks_plot(
  data,
  fitted,
  actual,
  palette = ez_col,
  size_line = 1,
  size = 11,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
fitted	Vector of fitted values
actual	Vector of actual values
palette	Colour function.
size_line	width of line for geom_line(). Default is 1.
size	theme size for use_theme(). Default is 14.
env	environment for evaluating expressions.

Examples

```
ks_plot(mtcars, "-disp", "am")
x = c(rnorm(100), rnorm(100) + 2)
label = c(rep('low', 100), rep('high', 100))
ks_plot(data.frame(x, label), "x", "label")
ks_plot(data.frame(x, label = factor(label, c('low', 'high'))), "x", "label")
```

---

lift_plot *lift_plot*

Description

precision-recall plot

Usage

```
lift_plot(
  data,
  fitted,
  actual,
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  size_line = 1,
  size = 11,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
fitted	Vector of fitted values
actual	Vector of actual values
group	A character value. Evaluates to a column.

facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
size_line	width of line for geom_line(). Default is 1.
size	theme size for use_theme(). Default is 14.
env	environment for evaluating expressions.

Examples

```
library(ggplot2)
n = 1000
df = data.frame(actual = sample(c(FALSE, TRUE), n, replace = TRUE),
                runif = runif(n))
df[["fitted"]] = runif(n) ^ ifelse(df[["actual"]] == 1, 0.5, 2)

density_plot(df, "fitted", "actual")

lift_plot(df, "fitted", "actual")
lift_plot(df, "fitted", "actual") + scale_y_log10()
lift_plot(df, "runif", "actual", size_line = 0.5)


library(dplyr, warn.conflicts = FALSE)
lift_plot(df, "fitted", "actual", "sample(c(1, 2), n(), TRUE)")

lift_plot(df, "fitted", "actual",
        "sample(c(1, 2), n(), TRUE)",
        "sample(c(3, 4), n(), TRUE)")

lift_plot(df, "fitted", "actual",
        "sample(c(1, 2), n(), TRUE)",
        "sample(c(3, 4), n(), TRUE)",
        "sample(c(5, 6), n(), TRUE)")
```

---

line_plot *line_plot*

Description

Creates line plots.

Usage

```
line_plot(
  data,
  x,
  y = "1",
  group = NULL,
```

```
  facet_x = NULL,
  facet_y = NULL,
  yoy = FALSE,
  size_line = 1,
  points = FALSE,
  size = 11,
  reorder = c("group", "facet_x", "facet_y"),
  palette = ez_col,
  labels_y = ez_labels,
  limits_y = c(NA, NA),
  use_theme = theme_ez,
  facet_scales = "fixed",
  na.rm = FALSE,
  legend_ncol = NULL
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
yoy	Logical used to indicate whether a YOY grouping should be created. Default is FALSE.
size_line	width of line for geom_line(). Default is 1.
points	logical. Option to include points
size	theme size for use_theme(). Default is 14.
reorder	A character vector specifying the group variables to reorder. Default is c("group", "facet_x", "facet_y").
palette	Colour function.
labels_y	label formatting function
limits_y	vector of c(min, max) y-axis limits
use_theme	ggplot theme function
facet_scales	Option passed to scales argument in facet_wrap or facet_grid. Default is "fixed".
na.rm	logical. Option to exclude NAs
legend_ncol	Number of columns in legend.

Value

A ggplot object.

Examples

```
suppressPackageStartupMessages(library(tsibble))
library(tsibbledata)
line_plot(ansett, x = "Week", y = "Passengers")
line_plot(ansett, x = "Week", y = "Passengers", "Class")
line_plot(pelt, "Year", "Hare", limits_y = c(0, NA))
line_plot(pelt, "Year", c("Hare", "Lynx"))
line_plot(pelt, "Year", "Hare", use_theme = ggplot2::theme_bw)
line_plot(pelt, "Year", c("Hare Population" = "Hare"))
```

`model_plot` *model_plot*

Description

model_plot

Usage

```
model_plot(
  data,
  x,
  actual,
  fitted,
  facet_x = NULL,
  point_size = 2,
  res_bins = NA_real_,
  size = 11
)
```

Arguments

`data`	A data.frame.
`x`	A named character value. Evaluates to a column.
`actual`	A character value. Evaluates to a logical or binary column.
`fitted`	A character value. Evaluates to a numeric column.
`facet_x`	A character value. Evaluates to a column.
`point_size`	Numeric. Default is 2.
`res_bins`	Number of bins in the residual distribution. Default value (NA) doesn't show the distribution.
`size`	theme size for `use_theme()`. Default is 14.

Value

A ggplot object.

Examples

```
y = rnorm(26)
df = data.frame(ID = 1:26, actual = y + rnorm(26), fitted = y, id = letters)
model_plot(df, "ID", "actual", "fitted")
model_plot(df, "id", "actual", "fitted")
model_plot(df, "ID", "actual", "fitted", res_bins = 10)
model_plot(df, "id", "actual", "fitted", res_bins = 10)
```

`nameifnot` *nameifnot*

Description

Names unnamed elements of a character vector.

Usage

```
nameifnot(x, make.names = FALSE)
```

Arguments

`x`	A character vector.
`make.names`	Logical. Whether to force names of x to be valid variable names. Default is FALSE.

Value

A named vector.

`na_plot` *na_plot*

Description

Visual representation of the NAs in a data.frame

Usage

```
na_plot(data, palette = ez_col)
```

Arguments

`data`	A data.frame.
`palette`	Colour function.

Value

A ggplot object.

Examples

```
na_plot(airquality)
```

---

not_numeric *not_numeric*

Description

Returns names of non-numeric columns.

Usage

```
not_numeric(x)
```

Arguments

x	A data.frame.

Value

A character vector.

---

no_null *no_null*

Description

Converts "NULL" character to NULL.

Usage

```
no_null(x)
```

Arguments

x	A character vector.

Value

y

Examples

```
no_null(NULL)
no_null("NULL")
no_null("NOPE")
```

`perf` *perf*

Description

Precision recall calculation

Usage

```
perf(fitted, actual, x_measure, y_measure)
```

Arguments

`fitted`	Vector with values between 0 and 1
`actual`	Vector with two levels
`x_measure`	metric for ROCR::performance
`y_measure`	metric for ROCR::performance

Examples

```
ezplot:::perf(runif(1), sample(c(TRUE, FALSE), 1, replace = TRUE), "rpp", "lift")
ezplot:::perf(runif(10), sample(c(TRUE, FALSE), 10, replace = TRUE), "rpp", "lift")
ezplot:::perf(runif(5), sample(c(TRUE, FALSE), 5, replace = TRUE), "rec", "prec")
ezplot:::perf(runif(5), sample(c(TRUE, FALSE), 5, replace = TRUE), "fpr", "tpr")
ezplot:::perf(runif(5), sample(c(TRUE, FALSE), 5, replace = TRUE), "cutoff", "tpr")
```

`performance_plot` *performance_plot*

Description

plots binary classification performance metrics

Usage

```
performance_plot(
  data,
  fitted,
  actual,
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  x = "fpr",
  y = "tpr",
  auc = c("title", "group"),
  size_line = 1,
  size = 11,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
fitted	A character value. Evaluates to a numeric column.
actual	A character value. Evaluates to a logical or binary column.
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
x	ROCR::performance() measure
y	ROCR::performance() measure
auc	character vector indicating which AUC values should be displayed. Options are 'title' and 'group'
size_line	width of line for `geom_line()`. Default is 1.
size	theme size for `use_theme()`. Default is 14.
env	environment for evaluating expressions.

Examples

```
performance_plot(mtcars, "-disp", "am")
performance_plot(mtcars, "-disp", "am", "cyl")
performance_plot(mtcars, "-disp", "am", "cyl", x = "rec", y = "prec")
performance_plot(mtcars, "-disp", "am", x = "rpp", y = "gain")
performance_plot(mtcars, "-disp", "am", x = "rpp", y = "lift")
performance_plot(mtcars, "-disp", "am", x = "cutoff", y = "tpr")
```

perf_df *perf_df*

Description

shows classification performance statistics as a table

Usage

```
perf_df(fitted, actual, quantiles = NULL)
```

Arguments

fitted	A character value. Evaluates to a numeric column.
actual	A character value. Evaluates to a logical or binary column.
quantiles	Number of quantiles to show. If NULL, uses distinct values of fitted for the cutoffs rather than showing quantiles.

Examples

```
perf_df(mtcars$mpg, mtcars$am)
perf_df(mtcars$mpg, mtcars$am, quantiles = 4)
perf_df(mtcars$mpg, mtcars$am, quantiles = 10)
perf_df(mtcars$wt, mtcars$am==0)
```

---

pie_plot *pie_plot*

Description

Creates pie charts.

Usage

```
pie_plot(
  data,
  x,
  y = "1",
  facet_x = NULL,
  facet_y = NULL,
  labels_y = function(x) ez_labels(x * 100, append = "%", round = round, signif =
    signif),
  size = 11,
  label_cutoff = 0.04,
  round = Inf,
```

```
  signif = 3,
  palette = ez_col,
  reorder = c("x", "facet_x", "facet_y"),
  label_x = 0.8,
  legend_ncol = NULL
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
labels_y	label formatting function
size	theme size for `use_theme()`. Default is 14.
label_cutoff	Cutoff size (proportion of y data range) for excluding labels
round	Option for rounding label.
signif	Option for retaining significant figures in label.
palette	Colour function.
reorder	A character vector specifying the group variables to reorder. Default is `c("group", "facet_x", "facet_y")`.
label_x	Position of label from centre of pie. 0 is the centre of the pie and 1 is the outer edge.
legend_ncol	Number of columns in legend.

Value

ggplot object

Examples

```
suppressPackageStartupMessages(library(tsibble))
library(tsibbledata)
pie_plot(ansett, "Class", "Passengers")
pie_plot(ansett, "Class", "Passengers", reorder = NULL, label_x = 0.5)
pie_plot(ansett, "Class", "Passengers", "Airports", reorder = NULL, label_x = 0.5)
```

prec_rec *prec_rec*

Description

Precision recall calculation

Usage

```
prec_rec(fitted, actual)
```

Arguments

fitted	Vector with values between 0 and 1
actual	Vector with two levels

Examples

```
ezplot:::prec_rec(runif(1), sample(c(TRUE, FALSE), 1, replace = TRUE))
ezplot:::prec_rec(runif(5), sample(c(TRUE, FALSE), 5, replace = TRUE))
```

pr_plot *pr_plot*

Description

precision-recall plot

Usage

```
pr_plot(
  data,
  fitted,
  actual,
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  palette = ez_col,
  size_line = 1,
  size = 11,
  labs = "short",
  env = parent.frame()
)
```

Arguments

data	A data.frame.
fitted	Vector of fitted values
actual	Vector of actual values
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
palette	Colour function.
size_line	width of line for geom_line(). Default is 1.
size	theme size for use_theme(). Default is 14.
labs	'short' or 'long'
env	environment for evaluating expressions.

Examples

```
library(ggplot2)
n = 1000
df = data.frame(actual = sample(c(FALSE, TRUE), n, replace = TRUE),
                runif = runif(n))
df[["fitted"]] = runif(n) ^ ifelse(df[["actual"]] == 1, 0.5, 2)

density_plot(df, "fitted", "actual")

pr_plot(df, "fitted", "actual")
pr_plot(df, "runif", "actual", size_line = 0.5)


library(dplyr, warn.conflicts = FALSE)
pr_plot(df, "fitted", "actual", "sample(c(1, 2), n(), TRUE)")

pr_plot(df, "fitted", "actual",
        "sample(c(1, 2), n(), TRUE)",
        "sample(c(3, 4), n(), TRUE)")

pr_plot(df, "fitted", "actual",
        "sample(c(1, 2), n(), TRUE)",
        "sample(c(3, 4), n(), TRUE)",
        "sample(c(5, 6), n(), TRUE)")
```

---

quick_facet *Quick facet*

Description

Applies faceting to ggplot objects when g[["data"]] has a facet_x or facet_y column.

Usage

```
quick_facet(g, ncol = NULL, ...)
```

Arguments

g	A ggplot object.
ncol	Number of facet columns.
...	Arguments to pass to facet_grid or facet_wrap.

---

reorder_levels *Order levels of factor columns using fct_reorder*

Description

Order levels of factor columns using fct_reorder

Usage

```
reorder_levels(
  data,
  cols = c("group", "facet_x", "facet_y"),
  y = "y",
  .desc = rep(TRUE, length(cols))
)
```

Arguments

data	A data.frame.
cols	Names of columns to reorder.
y	Numeric column for order priority.
.desc	A logical vector of length 1 or ncol(data). Default is TRUE for all columns in cols.

Value

A data.frame.

Examples

```
str(ezplot:::reorder_levels(mtcars, "cyl", "1"))
str(ezplot:::reorder_levels(mtcars, "cyl", "1", FALSE))
str(ezplot:::reorder_levels(mtcars, "cyl", "mpg"))
```

roc *roc*

Description

Calculates ROC and AUC

Usage

```
roc(fitted, actual)
```

Arguments

fitted	Vector with values between 0 and 1
actual	Vector with two levels

Examples

```
ezplot:::roc(runif(1), sample(c(TRUE, FALSE), 1, replace = TRUE))
ezplot:::roc(runif(3), sample(c(TRUE, FALSE), 3, replace = TRUE))
```

roc_plot *roc_plot*

Description

roc_plot

Usage

```
roc_plot(
  data,
  fitted,
  actual,
  group = NULL,
  facet_x = NULL,
  facet_y = NULL,
  palette = ez_col,
  size_line = 1,
  size = 11,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
fitted	Vector of fitted values
actual	Vector of actual values
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
palette	Colour function.
size_line	width of line for geom_line(). Default is 1.
size	theme size for use_theme(). Default is 14.
env	environment for evaluating expressions.

Examples

```
library(ggplot2)
n = 1000
df = data.frame(actual = sample(c(FALSE, TRUE), n, replace = TRUE),
                runif = runif(n))
df[["fitted"]] = runif(n) ^ ifelse(df[["actual"]] == 1, 0.5, 2)

ggplot(df) +
  geom_density(aes(fitted, fill = actual), alpha = 0.5)

roc_plot(df, "actual", "actual")
roc_plot(df, "fitted", "actual")
roc_plot(df, "runif", "actual", size_line = 0.5)


library(dplyr, warn.conflicts = FALSE)
roc_plot(df, "fitted", "actual", "sample(c(1, 2), n(), TRUE)")

roc_plot(df, "fitted", "actual",
         "sample(c(1, 2), n(), TRUE)",
         "sample(c(3, 4), n(), TRUE)")

roc_plot(df, "fitted", "actual",
         "sample(c(1, 2), n(), TRUE)",
         "sample(c(3, 4), n(), TRUE)",
         "sample(c(5, 6), n(), TRUE)")
```

---

save_png *save_png*

Description

Saves ggplot or ezplot objects to png.

Usage

```
save_png(g, file, width, height, res, ..., vp = NULL)
```

Arguments

g	A ggplot or ezplot object.
file	A png file path.
width	Width of output image.
height	Height or output image.
res	Resolution of output image.
...	Further arguments to pass to png().
vp	A viewport object created with grid::viewport.

---

scatter_plot *scatter plot*

Description

create a scatter plot

Usage

```
scatter_plot(
  data,
  x,
  y,
  group = NULL,
  palette = ez_col,
  size = 11,
  point_size = 2.5,
  env = parent.frame()
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
group	A character value. Evaluates to a column.
palette	Colour function.
size	theme size for use_theme(). Default is 14.
point_size	Numeric. Default is 2.
env	environment for evaluating expressions.

Examples

```
scatter_plot(mtcars, "wt", "hp")
scatter_plot(mtcars, "wt", "hp", "factor(cyl)")
scatter_plot(mtcars, "factor(cyl)", "hp")
```

`secondary_plot` *secondary_plot creates a plot with a secondary y-axis*

Description

secondary_plot creates a plot with a secondary y-axis

Usage

```
secondary_plot(
  data,
  x,
  y1 = "1",
  y2 = "1",
  facet_x = NULL,
  facet_y = NULL,
  palette = ez_col,
  size_line = 1,
  labels_y1 = ez_labels,
  labels_y2 = ez_labels,
  ylim1 = NULL,
  ylim2 = NULL,
  reorder = c("facet_x", "facet_y"),
  size = 11
)
```

Arguments

`data`	A data.frame.
`x`	A named character value. Evaluates to a column.
`y1`	Variable to plot on the left-hand axis
`y2`	Variable to plot on the right-hand axis
`facet_x`	A character value. Evaluates to a column.
`facet_y`	A character. Evaluates to a column.
`palette`	Colour function.
`size_line`	line size
`labels_y1`	label formatting function
`labels_y2`	label formatting function
`ylim1`	(optional) left axis limits

ylim2	(optional) right axis limits
reorder	A character vector specifying the group variables to reorder. Default is c("group", "facet_x", "facet_y").
size	theme size for use_theme(). Default is 14.

Value

A ggplot object.

Examples

```
suppressPackageStartupMessages(library(tsibble))
library(tsibbledata)
secondary_plot(pelt, "Year", "Hare", "Lynx")
secondary_plot(pelt, "Year", c("Hare Population" = "Hare"), c("Lynx Population" = "Lynx"))
secondary_plot(aus_production, "Quarter",
               c("Quarterly Beer Production (megalitres)" = "Beer"),
               c("Quarterly Cement Production (tonnes)" = "Cement"),
               "lubridate::quarter(Quarter)",
               ylim1 = c(0, 600), ylim2 = c(0, 3000),
               size = 10)
```

---

side_plot *side_plot*

Description

side_plot

Usage

```
side_plot(
  data,
  x,
  y = "1",
  labels_y = ez_labels,
  size = 11,
  palette = ez_col,
  signif = 3,
  reorder = TRUE,
  rescale_y = 1.25
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
labels_y	label formatting function
size	theme size for use_theme(). Default is 14.
palette	Colour function.
signif	Number of significant digits.
reorder	A character vector specifying the group variables to reorder. Default is c("group", "facet_x", "facet_y").
rescale_y	Rescaling factor for y-axis limits

Examples

```
side_plot(mtcars, "gear", "1", rescale_y = 4/3)
side_plot(mtcars, "cyl", c("Cars with <120 HP" = "hp < 120"))
side_plot(mtcars, "cyl", c(count = "ifelse(cyl == 4, 1, -1)", "hp <= 120"))
side_plot(mtcars, "cyl", c("hp <= 120", "~ - wt / cyl"), rescale_y = 1.5)
side_plot(mtcars, "cyl", c("1", "-1"))
```

---

text_contrast *text_contrast*

Description

text_contrast

Usage

```
text_contrast(x)
```

Arguments

x	Vector of colours.

Value

Vector indicating whether black or white should be used for text overlayed on x.

Examples

```
text_contrast("#000000")
text_contrast("black")
```

theme_ez *Default theme*

Description

Default theme

Usage

```
theme_ez(base_size = 11, base_family = "")
```

Arguments

base_size	base font size
base_family	base fond family

Value

theme

Examples

```
library(ggplot2)
ggplot(mtcars) + geom_point(aes(cyl, mpg)) + theme_ez()
```

tile_plot *tile_plot*

Description

Creates tile plots.

Usage

```
tile_plot(
  data,
  x,
  y,
  z = c(Count = "1"),
  facet_x = NULL,
  facet_y = NULL,
  size = 11,
  facet_ncol = NULL,
  labels_x = NULL,
  labels_y = NULL,
  labels_z = ez_labels,
```

```
  zlim = function(x) c(pmin(0, x[1]), pmax(0, x[2])),
  palette = ez_jet,
  reorder = c("facet_x", "facet_y")
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
z	A named character. Evaluates to a column and is mapped to the fill colour of the tiles.
facet_x	A character value. Evaluates to a column.
facet_y	A character. Evaluates to a column.
size	theme size for `use_theme()`. Default is 14.
facet_ncol	Option passed to ncol argument in `facet_wrap` or `facet_grid`. Default is `NULL`.
labels_x	label formatting function
labels_y	label formatting function
labels_z	label formatting function
zlim	argument for `scale_fill_grandientn(limits = zlim)`
palette	Colour function.
reorder	A character vector specifying the group variables to reorder. Default is `c("group", "facet_x", "facet_y")`.

Examples

```
## Not run:
library(tsibbledata)
library(dplyr)
nyc_bikes %>%
  mutate(duration = as.numeric(stop_time - start_time)) %>%
  filter(between(duration, 0, 16)) %>%
  tile_plot(c("Hour of Day" = "lubridate::hour(start_time) + 0.5"),
            c("Ride Duration (min)" = "duration - duration %% 2 + 1"))

## End(Not run)
```

unpack_cols *Unpack cols argument to agg_data*

Description

Unpack cols argument to agg_data

Usage

```
unpack_cols(x)
```

Arguments

x	cols

Value

list

Examples

```
ezplot:::unpack_cols("x")
ezplot:::unpack_cols(c(x = "x", y = "x + y", expr = "~ x + y"))
```

variable_plot *variable_plot*

Description

Plots variables (multiple "y" values) broken out as vertical facets.

Usage

```
variable_plot(
  data,
  x,
  y,
  group = NULL,
  facet_x = NULL,
  palette = ez_col,
  size = 14,
  labels_y = ez_labels,
  geom = "line",
  size_line = 1,
  legend_ncol = NULL,
  ylab = NULL,
```

```
  yoy = FALSE,
  switch = "y",
  rescale_y = 1
)
```

Arguments

data	A data.frame.
x	A named character value. Evaluates to a column.
y	A named character value. Evaluates to a column.
group	A character value. Evaluates to a column.
facet_x	A character value. Evaluates to a column.
palette	Colour function.
size	theme size for use_theme(). Default is 14.
labels_y	label formatting function
geom	Either "line", "col" or "bar". Default is "line"
size_line	width of line for geom_line(). Default is 1.
legend_ncol	Number of columns in legend.
ylab	y label text
yoy	Logical used to indicate whether a YOY grouping should be created. Default is FALSE.
switch	Option to switch location of variable (facet) labels. Default is 'y' (yes) which shows facet strips on left side of panels.
rescale_y	Rescaling factor for y-axis limits

Examples

```
suppressPackageStartupMessages(library(tsibble))
library(tsibbledata)
variable_plot(ansett, "Week", "Passengers", facet_x = "Class")
variable_plot(ansett, "Week", "Passengers", facet_x = "Class", yoy = TRUE)
variable_plot(pelt, "Year", c("Lynx", "Hare"), "round(Year, -1)")
variable_plot(hh_budget, "Year", c("Debt", "Expenditure"), "Country")
variable_plot(PBS, "Type", "Scripts", "Concession", switch = "y", geom = "col")
## Not run:
variable_plot(subset(hh_budget, Year > 2013), "Year",
              c("Debt\n(% of disposable income)" = "Debt",
                "Expenditure\nGrowth (%)" = "Expenditure",
                "Unemployment (%)" = "Unemployment"),
               facet_x = "Country", geom = "bar")
variable_plot(subset(hh_budget, Year > 2013), "Year",
              c("Debt\n(% of disposable income)" = "Debt",
                "Expenditure\nGrowth (%)" = "Expenditure",
                "Unemployment (%)" = "Unemployment"),
               group = "Country", geom = "bar")

## End(Not run)
```

`waterfall_plot` *waterfall_plot*

Description

function for creating waterfall charts

Usage

```
waterfall_plot(
  data,
  x,
  y,
  group,
  size = 11,
  labels = ez_labels,
  label_rescale = 1,
  y_min = "auto",
  rescale_y = 1.1,
  n_signif = 3,
  rotate_xlabel = FALSE,
  bottom_label = TRUE,
  ingroup_label = FALSE,
  n_x = 2,
  env = parent.frame()
)
```

Arguments

`data`	A data.frame.
`x`	A named character value. Evaluates to a column.
`y`	A named character value. Evaluates to a column.
`group`	A character value. Evaluates to a column.
`size`	theme size for `use_theme()`. Default is 14.
`labels`	Function for formatting labels.
`label_rescale`	Scaling factor for chart labels (relative to axis labels).
`y_min`	Minimum limit of y axis.
`rescale_y`	Rescaling factor for y-axis limits
`n_signif`	Number of significant figures in labels.
`rotate_xlabel`	Logical.
`bottom_label`	Logical.
`ingroup_label`	Logical. Shows in-group percentage change.
`n_x`	Number of x levels to show in chart.
`env`	environment for evaluating expressions.

Examples

```
library(tsibbledata)
waterfall_plot(aus_retail,
               "lubridate::year(Month)",
               "Turnover",
               "sub(' Territory', '\nTerritory', State)",
               rotate_xlabel = TRUE)
waterfall_plot(aus_retail,
               "lubridate::year(Month)",
               "Turnover",
               "sub(' Territory', '\nTerritory', State)",
               rotate_xlabel = TRUE,
               label_rescale = 0.5,
               ingroup_label = TRUE,
               bottom_label = FALSE,
               n_x = 3,
               size = 20,
               y_min = 0)
```

Index