

# Package ‘ezplot’

August 14, 2020

**Type** Package

**Title** Functions for Common Chart Types

**Version** 0.6.2

**Author** Wojtek Kostecki

**Maintainer** Wojtek Kostecki <wojtek.kostecki@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

**LazyData** true

**Depends** R (>= 3.3)

**RoxygenNote** 7.1.1

**Imports** dplyr, forcats, ggplot2, lubridate, rlang, tsibble

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

**VignetteBuilder** knitr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2020-08-14 10:20:02 UTC

## R topics documented:

agg_data . . . . .	2
area_plot . . . . .	3
bar_plot . . . . .	5
calendar_plot . . . . .	7
density_plot . . . . .	7
distribution_plot . . . . .	8
ez_app . . . . .	9

ez_col . . . . .	10
ez_jet . . . . .	10
ez_labels . . . . .	11
ez_png . . . . .	12
ez_server . . . . .	13
ez_ui . . . . .	13
get_incr . . . . .	13
histogram_plot . . . . .	14
ks_plot . . . . .	15
lift_plot . . . . .	16
line_plot . . . . .	17
model_plot . . . . .	18
nameifnot . . . . .	19
na_plot . . . . .	20
not_numeric . . . . .	20
no_null . . . . .	21
perf . . . . .	21
performance_plot . . . . .	22
perf_df . . . . .	23
pie_plot . . . . .	23
prec_rec . . . . .	25
pr_plot . . . . .	25
quick_facet . . . . .	26
reorder_levels . . . . .	27
roc . . . . .	28
roc_plot . . . . .	28
save_png . . . . .	29
scatter_plot . . . . .	30
secondary_plot . . . . .	31
side_plot . . . . .	32
text_contrast . . . . .	33
theme_ez . . . . .	34
tile_plot . . . . .	34
unpack_cols . . . . .	35
variable_plot . . . . .	36
waterfall_plot . . . . .	38
<b>Index</b>	<b>40</b>

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**

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

**Value**

An aggregated data.frame.

**Examples**

```
library(tsibble)
library(tsibbledata)
agg_data(ansest, c("Passengers", count = "1"))
agg_data(ansest["Class"])
agg_data(ansest[c("Class", "Passengers")])
agg_data(ansest, "Passengers", "Class")
agg_data(ansest, "Passengers", c("Class", "Airports"))
agg_data(ansest, c(x = "Airports", y = "Passengers"), c(x = "Airports"))
agg_data(ansest, 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,
  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.
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",
  coord_flip = FALSE
)
```

**Arguments**

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

**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	<i>calendar_plot</i>
---------------	----------------------

---

**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	<i>density_plot</i>
--------------	---------------------

---

**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,
  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.
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	<i>distribution_plot</i>
-------------------	--------------------------

---

**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(ansest)

## End(Not run)
```

---

ez_col	<i>Color palette interpolation</i>
--------	------------------------------------

---

**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	<i>ez_jet</i>
--------	---------------

---

**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**

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

---

ez_server	<i>ez_server</i>
-----------	------------------

---

**Description**

ez\_server

**Usage**

ez\_server(data)

**Arguments**

data                    A data frame

---

ez_ui	<i>ez_ui</i>
-------	--------------

---

**Description**

ez\_ui

**Usage**

ez\_ui(data)

**Arguments**

data                    A data frame

---

get_incr	<i>get_incr</i>
----------	-----------------

---

**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	<i>histogram_plot</i>
----------------	-----------------------

---

### 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,
  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.
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")
```

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,
  size = 11,
  palette = ez_col,
  labels_y = ez_labels,
  use_theme = theme_ez,
  facet_scales = "fixed"
)

```

**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.
size	theme size for use_theme(). Default is 14.
palette	Colour function.
labels_y	label formatting function
use_theme	ggplot theme function
facet_scales	Option passed to scales argument in facet_wrap or facet_grid. Default is "fixed".

**Value**

A ggplot object.

**Examples**

```
library(tsibble)
library(tsibbledata)
line_plot(ansett, x = "Week", y = "Passengers")
line_plot(pelt, "Year", "Hare")
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	<i>model_plot</i>
------------	-------------------

---

**Description**

model\_plot

**Usage**

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

**Arguments**

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

<code>x</code>	A character vector.
<code>make.names</code>	Logical. Whether to force names of <code>x</code> to be valid variable names. Default is FALSE.

**Value**

A named vector.

---

<code>na_plot</code>	<i>na_plot</i>
----------------------	----------------

---

**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)
```

---

<code>not_numeric</code>	<i>not_numeric</i>
--------------------------	--------------------

---

**Description**

Returns names of non-numeric columns.

**Usage**

```
not_numeric(x)
```

**Arguments**

- `x`                A data.frame.

**Value**

A character vector.

---

no_null	<i>no_null</i>
---------	----------------

---

**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	<i>perf</i>
------	-------------

---

**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 ROC::performance
y_measure	metric for ROC::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	<i>performance_plot</i>
------------------	-------------------------

---

**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**

perf\_df

**Usage**

```
perf_df(fitted, actual)
```

**Arguments**

fitted	A character value. Evaluates to a numeric column.
actual	A character value. Evaluates to a logical or binary column.

**Examples**

```

perf_df(mtcars$mpg, mtcars$am)
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
  )

```

### 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.

### Value

ggplot object

### Examples

```

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**

<code>fitted</code>	Vector with values between 0 and 1
<code>actual</code>	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**

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

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

---

<code>reorder_levels</code>	<i>Order levels of factor columns using <code>fct_reorder</code></i>
-----------------------------	--

---

**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**

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

**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	<i>roc</i>
-----	------------

---

**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	<i>roc_plot</i>
----------	-----------------

---

**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**

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

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

---

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**

<code>data</code>	A data.frame.
<code>x</code>	A named character value. Evaluates to a column.
<code>y</code>	A named character value. Evaluates to a column.
<code>group</code>	A character value. Evaluates to a column.
<code>palette</code>	Colour function.
<code>size</code>	theme size for <code>use_theme()</code> . Default is 14.
<code>point_size</code>	Numeric. Default is 2.
<code>env</code>	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	<i>secondary_plot creates a plot with a secondary y-axis</i>
----------------	--

---

**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**

```
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	<i>text_contrast</i>
---------------	----------------------

---

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	<i>Default theme</i>
----------	----------------------

---

**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	<i>tile_plot</i>
-----------	------------------

---

**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_gradientn(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,
  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.
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**

```

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")
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")

```

---

waterfall_plot	<i>waterfall_plot</i>
----------------	-----------------------

---

## 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

agg\_data, [2](#)  
area\_plot, [3](#)  
  
bar\_plot, [5](#)  
  
calendar\_plot, [7](#)  
  
density\_plot, [7](#)  
distribution\_plot, [8](#)  
  
ez\_app, [9](#)  
ez\_col, [10](#)  
ez\_jet, [10](#)  
ez\_labels, [11](#)  
ez\_png, [12](#)  
ez\_server, [13](#)  
ez\_ui, [13](#)  
  
get\_incr, [13](#)  
  
histogram\_plot, [14](#)  
  
ks\_plot, [15](#)  
  
lift\_plot, [16](#)  
line\_plot, [17](#)  
  
model\_plot, [18](#)  
  
na\_plot, [20](#)  
nameifnot, [19](#)  
no\_null, [21](#)  
not\_numeric, [20](#)  
  
perf, [21](#)  
perf\_df, [23](#)  
performance\_plot, [22](#)  
pie\_plot, [23](#)  
pr\_plot, [25](#)  
prec\_rec, [25](#)  
  
quick\_facet, [26](#)  
  
reorder\_levels, [27](#)  
roc, [28](#)  
roc\_plot, [28](#)  
  
save\_png, [29](#)  
scatter\_plot, [30](#)  
secondary\_plot, [31](#)  
side\_plot, [32](#)  
  
text\_contrast, [33](#)  
theme\_ez, [34](#)  
tile\_plot, [34](#)  
  
unpack\_cols, [35](#)  
  
variable\_plot, [36](#)  
  
waterfall\_plot, [38](#)