# Package 'ezplot'

July 20, 2019

**Type** Package

**Title** Functions for Common Chart Types

**Version** 0.3.1

**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

**LazyData** true

**Depends** R (>= 2.10)

**Imports** dplyr, forcats, ggplot2, rlang

**RoxygenNote** 6.1.1

**Suggests** covr, knitr, lubridate, methods, psych, rmarkdown, ROCR, stats, testthat, tibble, tidyr, tsibbledata

**VignetteBuilder** knitr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-07-20 21:20:03 UTC

## R topics documented:


agg_data . . . . . 2
area_plot . . . . . 3
bar_plot . . . . . 4
calendar_plot . . . . . 5
distribution_plot . . . . . 6
ez_col . . . . . 6
ez_jet . . . . . 7

ez_labels . . . . . 7
ez_png . . . . . 8
get_incr . . . . . 9
line_plot . . . . . 9
mean_plot . . . . . 10
model_plot . . . . . 11
nameifnot . . . . . 12
na_plot . . . . . 12
not_numeric . . . . . 13
no_null . . . . . 13
pie_plot . . . . . 14
quick_facet . . . . . 15
reorder_levels . . . . . 15
roc . . . . . 16
roc_plot . . . . . 16
save_png . . . . . 17
scatter_plot . . . . . 18
secondary_plot . . . . . 18
side_plot . . . . . 19
text_contrast . . . . . 20
theme_ez . . . . . 21
tile_plot . . . . . 21
unpack_cols . . . . . 22
waterfall_plot . . . . . 23

---

`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

```
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 = "Airpor
```

---

`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 = 14, 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)
```

### 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","fa` |
| `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. |

## Value

A ggplot object.

## Examples

```
library(tsibbledata)
area_plot(ansett, x = "Week", y = "Passengers")
area_plot(ansett,
          x = "Week", y = c("Weekly Passengers" = "Passengers"), "Class")
area_plot(ansett, "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 = 14, 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 }, label_pos = c("auto", "inside", "top", "both"),
  rescale_y = 1.1, label_cutoff = 0.12, use_theme = theme_ez,
  position = "stack", 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. |
| 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","fa |
| palette | Colour function. |
| labels_y | label formatting function |
| label_pos | Position of labels. Can be "auto", "inside", "top" or "both". |
| rescale_y | Rescaling factor for y-axis limits |
| label_cutoff | Cutoff size (proportion of limit 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". |

## Value

A ggplot object.

## Examples

```
library(tsibbledata)
library(lubridate)
bar_plot(ansett, "year(Week)", "Passengers")
bar_plot(ansett, "year(Week)", "Passengers", "Class")
bar_plot(ansett, "Airports", c("Share of Passengers" = "Passengers"), "Class", position = "f
bar_plot(ansett, "Airports", "Passengers", "Class", use_theme = ggplot2::theme_bw)
bar_plot(ansett, "Airports", "Passengers", "Class", reorder = NULL, label_pos = "both")
bar_plot(ansett, "sub('-.*', '', Airports)", c("Total Passengers" = "Passengers"),
         "Class",
         "sub('.*-', '', Airports)")
bar_plot(ansett, "Airports",
         c(Passengers = "ifelse(Class == 'Economy', Passengers, -Passengers)"),
         "Class", label_pos = "both")
```

---

# 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))
```

---

## distribution_plot *distribution_plot*

---

### Description

distribution_plot

### Usage

```
distribution_plot(data, x, facet_x = NULL, nbins = 20,
  use_theme = theme_ez, size = 14)
```

### 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. |

### 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_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

ez_jet

### Usage

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

### Arguments

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

## 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`. |

---

## 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 |

---

## 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 = 14,
  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(tsibbledata)
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"))
```

---

## mean_plot *mean_plot*

---

### Description

Chart to compare the means across groups using a bar chart.

### Usage

```
mean_plot(data, x, y, size = 14, labels = ez_labels)
```

### Arguments

| | |
|---|---|
| data | data.frame |
| x | quoted expression (required) |
| y | quoted expression (required) |
| size | base_size for ggplot2 theme (default is 20) |
| labels | function for formatting labels (default is ez_labels) |

### Examples

```
library(dplyr)
mean_plot(mtcars, c("Number of Cylinders" = "factor(cyl)"), "hp > 110",
          labels = function(x) ez_labels(100 * x, append = "%"))
```

---

## 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 = 14)
```

### Arguments

| | |
|---|---|
| data | A data.frame. |
| x | A named character value. Evaluates to a column. |
| actual | A character value. Evaluates to a column. |
| fitted | A character value. Evaluates to a 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)
```

### Arguments

| | |
|---|---|
| data | A data.frame. |

### 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")
```

---

`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 = 14, 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 limit 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","fa` |
| `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(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)
```

---

## 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(actual, fitted)
```

### Arguments

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

### Examples

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

---

## roc_plot *roc_plot*

---

### Description

roc_plot

### Usage

```
roc_plot(data, actual, fitted, group = NULL, facet_x = NULL,
  facet_y = NULL, size = 14)
```

### Arguments

| | |
|---|---|
| data | A data.frame. |
| actual | Vector of actuals values |
| fitted | Vector of fitted 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 | theme size for use_theme(). Default is 14. |

### Examples

```
library(ggplot2)
n = 10000
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, "actual", "fitted")
roc_plot(df, "actual", "runif")


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

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

roc_plot(df, "actual", "fitted",
         "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, size = 14, point_size = 2.5)
```

### 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. |
| point_size | Numeric. Default is 2. |

### 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, size_line = 1, labels_y1 = ez_labels,
  labels_y2 = ez_labels, ylim1 = NULL, ylim2 = NULL,
  reorder = c("facet_x", "facet_y"), size = 14)
```

## 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. |
| `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","fa` |
| `size` | theme size for `use_theme()`. Default is 14. |

## Value

A ggplot object.

## Examples

```
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 = 14,
  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","fa |
| rescale_y | Rescaling factor for y-axis limits |

### Examples

```
side_plot(mtcars, "gear", "1")
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"))
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 = 14, 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","fa` |

## 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"))
```

## waterfall_plot *waterfall_plot*

### Description

function for creating waterfall charts

### Usage

```
waterfall_plot(data, x, y, group, size = 14, 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)
```

### 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 | Scaling factor to extend y_max. |
| 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. |

## 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)
```