Package: NHSRplotthedots (via r-universe)

August 21, 2024

Type Package

Title Draw XmR Charts for NHSE/I 'Making Data Count' Programme

Version 0.1.0.9000

Maintainer Christopher Reading <christopher.reading1@nhs.net>

Description Provides tools for drawing Statistical Process Control (SPC) charts. This package supports the NHSE/I programme 'Making Data Count', and allows users to draw XmR charts, use change points and apply rules with summary indicators for when rules are breached.

URL https://nhs-r-community.github.io/NHSRplotthedots, https://github.com/nhs-r-community/NHSRplotthedots

BugReports https://nhs-r-community.github.io/NHSRplotthedots/issues

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

Imports assertthat, base64enc, crayon, dplyr, ggplot2, grid, magrittr, plotly, rlang, rsvg, scales, stringr, tidyselect (>= 1.2.0)

Suggests covr, hexSticker, knitr, lintr (>= 3.0.0), mockery, NHSRdatasets, pillar, rmarkdown, spelling, testthat (>= 3.0.0), tibble, utils, withr

Depends R (>= 4.1.0)

VignetteBuilder knitr

Config/testthat/edition 3

Language en-GB

Repository https://nhs-r-community.r-universe.dev

RemoteUrl https://github.com/nhs-r-community/NHSRplotthedots

RemoteRef main

RemoteSha e74d1f3ed292060a7177bae0254ca3260d3a52a9

Contents

`GeomPTDIcon` *GeomPTDIcon*

Description

The Geom for the PTD icons. See `geom_ptd_icon()`.

Usage

```
GeomPTDIcon
```

Format

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

`geom_ptd_icon` *PTD Icons*

Description

Inserts the Making Data Count icons for variation/assurance to a plot

Usage

```
geom_ptd_icon(
  data = NULL,
  icons_size = 8L,
  icons_position = c("top right", "bottom right", "bottom left", "top left"),
  ...
)
```

Arguments

data	the dataframe to plot. Must be of type ptd_spc_df
icons_size	the size of the icons
icons_position	the position of the icons in the plot
...	currently unused

`ptd_create_ggplot` *Create ggplot2*

Description

Creates a ggplot2 object using the parameters passed in.

Usage

```
ptd_create_ggplot(
  x,
  point_size = 4,
  percentage_y_axis = FALSE,
  main_title,
  x_axis_label,
  y_axis_label,
  fixed_x_axis_multiple = TRUE,
  fixed_y_axis_multiple = TRUE,
  x_axis_date_format = "%d/%m/%y",
  x_axis_breaks = NULL,
  y_axis_breaks = NULL,
  label_limits = FALSE,
  icons_size = 8L,
 icons_position = c("top right", "bottom right", "bottom left", "top left", "none"),
  colours = ptd_spc_colours(),
  theme_override = NULL,
  break_lines = c("both", "limits", "process", "none"),
  ...
)
```

Arguments

x	An object created by `ptd_spc()`
point_size	Specify the plotting point size for the ggplot2 output. The default is 2.5.
percentage_y_axis	Specify whether the y axis values are percentages. Accepted values are `TRUE` for a percentage y axis, or `FALSE` for an integer y axis. Defaults to `FALSE`.
main_title	Specify a character string value for the plot title.
x_axis_label	Specify a character string value for the x axis title.

y_axis_label	Specify a character string value for the y axis title.
fixed_x_axis_multiple	Specify whether, if producing a faceted spc, the x axis should be fixed for all facet plots. Accepted values are TRUE for fixed x axes or FALSE for individual x axes.
fixed_y_axis_multiple	Specify whether, if producing a faceted spc, the y axis should be fixed for all facet plots. Accepted values are TRUE for fixed y axes or FALSE for individual y axes.
x_axis_date_format	Specify how dates on the x axis should be displayed. The format should be provided as a character string using 'd m Y' -type syntax.
x_axis_breaks	Specify an interval value for breaks on the x axis. The value should be a character string expressing interval length and type, e.g. "3 months", "7 days".
y_axis_breaks	Specify an interval value for breaks on the y axis. The value should be a numeric vector of length 1, either an integer for integer scales or a decimal value for percentage scales. This option is ignored if faceting is in use.
label_limits	Whether to add a secondary y axis that just provides labels for the values of the UCL, LCL and mean. The default is FALSE.
icons_size	The size of the icons, defined in terms of font size. Defaults to 8.
icons_position	Where to show the icons, either "top right" (default), "bottom right", "bottom left", "top left", or "none".
colours	Specify the colours to use in the plot. Use the ptd_spc_colours() function to change defaults.
theme_override	Specify a list containing ggplot2 theme elements that can be used to override the default appearance of the plot.
break_lines	Whether to break lines when a rebase happens. Defaults to "both", but can break just "limits" lines, "process" lines, or "none".
...	Currently ignored

Value

A ggplot2 object

`ptd_create_plotly` *Create plotly*

Description

Creates a plotly object using the parameters passed in.

Usage

```
ptd_create_plotly(
  x,
  point_size = 4,
  percentage_y_axis = FALSE,
  main_title,
  x_axis_label,
  y_axis_label,
  fixed_x_axis_multiple = TRUE,
  fixed_y_axis_multiple = TRUE,
  x_axis_date_format = "%d/%m/%y",
  x_axis_breaks = NULL,
  y_axis_breaks = NULL,
  icons_size = 0.15,
 icons_position = c("top right", "bottom right", "bottom left", "top left", "none"),
  colours = ptd_spc_colours(),
  theme_override = NULL,
  break_lines = c("both", "limits", "process", "none"),
  ...
)
```

Arguments

x	an object created by ptd_spc()
point_size	Specify the plotting point size for the ggplot output. Default is 2.5.
percentage_y_axis	Specify whether the y axis values are percentages. Accepted values are TRUE for percentage y axis, FALSE for integer y axis. Defaults to FALSE.
main_title	Specify a character string value for the ggplot title.
x_axis_label	Specify a character string value for the x axis title.
y_axis_label	Specify a character string value for the y axis title.
fixed_x_axis_multiple	Specify whether, if producing a faceted spc, x axis should be fixed for all facet plots. Accepted values are TRUE for fixed x axes or FALSE for individual x axes.
fixed_y_axis_multiple	Specify whether, if producing a faceted spc, y axis should be fixed for all facet plots. Accepted values are TRUE for fixed y axes or FALSE for individual y axes.
x_axis_date_format	Specify how dates on the x axis should be displayed. Format should be provided as a character string using 'd m Y' etc syntax.
x_axis_breaks	Specify an interval value for breaks on the x axis. Value should be a character string expressing interval length and type, e.g. "3 months", "7 days".
y_axis_breaks	Specify an interval value for breaks on the y axis. Value should be a numeric vector of length 1, either an integer for integer scales or a decimal value for percentage scales. This option is ignored if faceting is in use.

icons_size	The size of the icons, defined in terms of font size. Defaults to 0.15.
icons_position	Where to show the icons, either "top right" (default), "bottom right", "bottom left", "top left", or "none".
colours	Specify the colours to use in the plot, use the ptd_spc_colours() function to change defaults.
theme_override	Specify a list containing ggplot theme elements which can be used to override the default appearance of the plot.
break_lines	whether to break lines when a rebase happens. Defaults to "both", but can break just "limits" lines, "process" lines, or "none".
...	currently ignored

Value

The plotly object

---

ptd_rebase *Rebase*

Description

Produces an object that can be used for rebasing an SPC chart. This method provides two different ways to rebase:

1. You can either provide a single vector of dates, which will rebase every facet of an SPC with the same dates
2. You can provide named vectors of dates, where the names correspond to the names of the facets, in order to rebase a faceted chart.

Usage

```
ptd_rebase(...)
```

Arguments

...	Either a single vector of dates, or named vectors of dates. See examples.

Value

A list

Examples

```
# If you aren't using a faceted chart, or you want to rebase each facet at
# the same dates, then you can simply call this method with a vector of dates.
# For example, to rebase on the 1^st^ January 2020 and 22^nd^ March 2020:

ptd_rebase(as.Date(c("2020-01-01", "2020-03-22")))

# If you are using a faceted chart, and wish to rebase each facet with
# different dates, then you can call this method, naming each vector of dates
# with the name of the facet. If there are facets that you don't need to rebase
# you can simply ignore them. For example, if you had a chart with facets "a",
# "b", and "c", and you wanted to rebase "a" on the 1^st^ January 2020, and
# "b" on the 22^nd^ March 2020:

ptd_rebase("a" = as.Date("2020-01-01"), "b" = as.Date("2020-03-22"))
```

---

ptd_spc *SPC Plotting Function*

Description

ptd_spc returns a plot object or data table with SPC values using NHSI 'plot the dots' logic.

Usage

```
ptd_spc(
  .data,
  value_field,
  date_field,
  facet_field,
  rebase = ptd_rebase(),
  fix_after_n_points = NULL,
  improvement_direction = "increase",
  target = ptd_target(),
  trajectory,
  screen_outliers = TRUE
)
```

Arguments

.data	A data frame containing a value field, a date field, and a category field (if for faceting). There should be no gaps in the time series for each category.
value_field	Specify the field name which contains the value data, to be plotted on y axis. The field name can be specified using non-standard evaluation (i.e. without quotation marks).
date_field	Specify the field name which contains the date data, to be plotted on x axis. The field name can be specified using non-standard evaluation (i.e. without quotation marks).

`facet_field` Optional: Specify field name which contains a grouping/ faceting variable. SPC logic will be applied to each group separately, with outputs combined. Currently accepts 1 variable only. The field name can be specified using non-standard evaluation (i.e. without quotation marks).

`rebase` Specify a date vector of dates when to rebase, or, if `facet_field` is set, a named list of date vectors of when to rebase. Each item in the list should be named after the facet you wish to rebase. See `ptd_rebase()`.

`fix_after_n_points` Specify a number points after which to fix SPC calculations.

`improvement_direction` Specify whether process improvement is represented by an increase or decrease in measured variable, or is neutral. Accepted values are 'increase' for increase as improvement, 'decrease' for decrease as improvement, and 'neutral' where neither direction represents an improvement. Defaults to 'increase'.

`target` Specify a single value, which will apply the same target to every facet of an SPC chart, or named values of targets, where the names correspond to the names of the facets, in order to have different targets for each facet. See `ptd_target()`.

`trajectory` Specify a field name which contains a trajectory value. The field name can be specified using non-standard evaluation (i.e. without quotation marks).

`screen_outliers` Whether to screen for outliers when calculating the control limits. Defaults to `TRUE`.

Details

This function is designed to produce consistent SPC charts across Information Department reporting, according to the 'plot the dots' logic produced by NHSI. The function can return either a plot or data frame.

Value

An object of type `ptd_spc_df`. This is a data.frame which can be further manipulated like any other data.frame. The default print() method for `ptd_spc_df` is to call `ptd_create_ggplot()`, displaying the plot. If you would like to get the data.frame, call as_tibble() or as.data.frame() on the object.

Examples

```
library(NHSRdatasets)
library(dplyr)
data("ae_attendances")

# Pick a trust at random to look at their data for two years
trust1 <- subset(ae_attendances, org_code == "RJZ" & type == 1)

# Basic chart with improvement direction decreasing
ptd_spc(trust1,
  value_field = breaches, date_field = period,
  improvement_direction = "decrease"
```

```
)

# Pick a few trust, and plot individually using facet
# Also set the x-axis scale to vary for each and date groups to 3 months
orgs <- c("RAS", "RJZ", "RR1", "RJC", "RQ1")
trusts4 <- filter(ae_attendances, org_code %in% orgs, type == 1)

s <- ptd_spc(trusts4,
  value_field = breaches, date_field = period, facet_field = org_code,
  improvement_direction = "decrease"
)
plot(s, fixed_y_axis_multiple = FALSE, x_axis_breaks = "3 months")

# Save the first chart as an object this time then alter the ggplot theme
my_spc <- ptd_spc(trust1,
  value_field = "breaches", date_field = "period",
  improvement_direction = "decrease"
)

plot(my_spc) + ggplot2::theme_classic()
```

---

ptd_spc_colours *SPC Colours*

Description

Produces a list of colours that controls the geoms in the plot

Usage

```
ptd_spc_colours(
  common_cause = "#a6a6a6",
  special_cause_improvement = "#00b0f0",
  special_cause_neutral = "#490092",
  special_cause_concern = "#e46c0a",
  value_line = "#a6a6a6",
  mean_line = "#000000",
  lpl = "#a6a6a6",
  upl = "#a6a6a6",
  target = "#de1b1b",
  trajectory = "#490092"
)
```

Arguments

common_cause, special_cause_improvement, special_cause_neutral, special_cause_concern	the colour of the points
value_line	the colour of the line joining the points

`mean_line`	the colour of the "mean" (average) line
`lpl`, `upl`	the colour the the lower and upper process limit lines
`target`	the colour of the target line
`trajectory`	the colour of the trajectory line

Value

A list of colours

`ptd_target` *Target*

Description

Produces an object that can be used for adding Targets to an SPC chart. This method provides two different ways to add a target:

1. You can either provide a single value, which will apply the same target to every facet of an SPC
2. You can provide named values of targets, where the names correspond to the names of the facets, in order to have different targets for each facet.

Usage

```
ptd_target(...)
```

Arguments

...	Either a single value, or named values, of the target(s). See examples.

Details

This function is a helper to provide data in the correct format for use with `ptd_spc()`. See **Value** section for details of return type. If you are trying to do something like `ptd_spc(list_of_values)` then you can skip using the function and just use `list_of_values`, so long as the list meets the requirements as listed above.

Value

Either:

- a single numeric value, in this case all facets in the plot will use this target value
- a named list of single numeric values, where each item is named as for one of the facets in the plot. If a facet isn't specified then it will not have a target.

Examples

```
# If you aren't using a faceted chart, or you want to use the same target for
# each facet, you can simply call this method with a single value. For
# example, to use a target of 90%:

ptd_target(0.9)

# If you are using a faceted chart, and wish to use a different target for
# each facet, then you can call this method, naming each value with the name
# of the facet. Any facet that isn't listed will not have a target applied to
# it.

# For example, to apply a target of 25 to the "a" facet and 10 to the "b"
# facet:

ptd_target(
  "a" = 25,
  "b" = 10
)

# If you already have your data in a list, you do not need to use
# ptd_target(). But, if you wanted to check that your values are valid, you
# could call it like so:

my_targets <- list("a" = 25, "b" = 10)
do.call(ptd_target, my_targets)

# or, if your targets are in a numeric vector:
my_targets <- c("a" = 25, "b" = 10)
do.call(ptd_target, as.list(my_targets))
```

Index