Package ‘colorplaner’

November 1, 2016

Type Package

Title A 'ggplot2' Extension to Visualize Two Variables per Color
Aesthetic Through Color Space Projections

Version 0.1.3

Date 2016-10-31

Description A 'ggplot2' extension to visualize two variables through one color aesthetic via mapping to a color space projection. With this technique for 2-D color mapping, one can create a bivariate choropleth in R as well as other visualizations with multivariate color scales. Includes two new scales and a new guide for 'ggplot2'.

License GPL-2

LazyData TRUE

Imports grid, scales, gtable, digest, ggplot2 (>= 2.0.0)

Depends R (>= 3.1)

Suggests testthat, maps, mapproj, knitr, rmarkdown, covr

RoxygenNote 5.0.1

Copyright Copyright 2016 William Murphy

VignetteBuilder knitr

URL `https://github.com/wmurphyrd/colorplaner`

BugReports `https://github.com/wmurphyrd/colorplaner/issues`

Collate 'other_package_compatibility.R' 'colorplane.R'
'ScaleColorPlane.R' 'colorplaner.R' 'guide_colorplane.R'

NeedsCompilation no

Author William Murphy [aut, cre],
Hadley Wickham [ctb] (Original author for copied and derived code),
Winston Chang [ctb] (Original author for copied and derived code)

Maintainer William Murphy <`william.murphy.rd@gmail.com`>

Repository CRAN

Date/Publication 2016-11-01 11:07:29

R topics documented:

colorplaner — *colorplaner: ggplot2 Extension to Visualize Two Variables Per Color Aesthetic through Color Space Projection*

Description

Add additional dimensionality to visualizations by using the color and/or fill aesthetics to convey the values of two continuous variables each. By projecting variable values onto YUV color space, a scale is created that allows viewers to intuitively determine the values of both variables from the single displayed color. Includes two new scales and a new guide for ggplot2. See `scale_color_colorplane` for usage.

Requirement for Package Attachment

At present, `guide_colorplane` will only function when the colorplaner package is attached to the search list. For scripting or interactive use, use `library(colorplaner)`. For package development, add colorplaner to the Depends list in your DESCRIPTION file.

This requirement exists because ggplot2 guides function through the use of S3 generics and methods, but the generic functions are not exported from the ggplot package. Without access to the generics, the methods for the colorplane guide cannot be properly registered and will only be found by the dispatcher if in the search path.

Check `https://github.com/wmurphyrd/colorplaner/issues/27` for current status and progress towards resolving this issue.

Warning Message About Ignoring Unknown Aesthetics

Layers now produce a warning message when unrecognized aesthetics are found but have no mechanism for notifying them of aesthetics handled by scales. The warning can be avoided by mapping `color2/fill2` at the plot level (i.e. in the initial `ggplot()` statement). If you want to avoid colorplane mapping on all layers, map `color/fill` only on the layers you want, as in the example below.

Examples

```
library(ggplot2)
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width,
  colour2 = Petal.Width)) +
 geom_point(aes(colour = Petal.Length)) +
 geom_line(aes(linetype = Species)) +
scale_color_colorplane()
```

color_projections *Color Space Projections*

Description

Functions to define how variables are mapped into color space. Used for the `color_projection` argument to `scale_color_colorplane` and `scale_fill_colorplane`. Custom functions can also be defined following this signature.

Usage

```
YUV_projection(x, y, Y = 0.3)

red_blue_projection(x, y)

interpolate_projection(x, y, zero_color, horizontal_color, vertical_color)
```

Arguments

`x, y` numeric vectors of equal length containing the values to be mapped to the horizontal and vertical axes of the colorplane.

`Y` numeric value in range 0 to 1 for the fixed luminosity level in YUV projections.

`zero_color, horizontal_color, vertical_color`
Character strings specifying R colors to use in interpolation projections. See Details.

Details

Color space projection functions take two numeric vectors and return a single character vector of the same length that specifies the colors to plot in the form "#rrggbb", as returned by `rgb`. Additional projection functions can be defined following the same signature and passed to the `color_projection` argument of the `scale_*_colorplane` scale constructors. When writing custom projection functions, expect two arguments that are numeric vectors scaled to a range of 0 to 1 and that do not contain missing values. Custom projections can accept additional arguments that are passed through from the `...` of `scale_*_colorplane`.

For `interpolate_projection`, a color space is created via linear RGB-space interpolation for each axis and then blending by averages. `zero_color` is the base color when both `x` and `y` are minimal. `horizontal_color` specified the color to interpolate towards for increasing `x` values and `vertical_color` for `y`. The plotted color will be a simple average of the interpolated `x` and `y` values.

Value

Character vector of colors of the same length as `x` and `y`.

References

YUV conversion matrix from https://en.wikipedia.org/wiki/YUV. UV limits sourced from Deveroux VG. Limiting of YUV Video Signals. British Broadcasting System. 1987 .

See Also

scale_color_colorplane, scale_fill_colorplane

Examples

```
library(ggplot2)
ggplot(mtcars, aes(x = wt, y = mpg, color = disp, colour2 = hp)) +
  geom_point() +
  scale_color_colorplane(color_projection = interpolate_projection,
                         zero_color = "darkorange2",
                         horizontal_color = "mediumspringgreen",
                         vertical_color = "magenta")
```

---

guide_colorplane *Add Guide for Colorplane*

Description

Generates a guide to explain the colors plotted via scale_color_colorplane and scale_fill_colorplane.

Usage

```
guide_colorplane(title = waiver(), title.position = c("top", "bottom"),
  title.theme = NULL, title.hjust = 0.5, title.vjust = NULL,
  axis.title = waiver(), axis.title.position = c("bottom", "top"),
  axis.title.theme = NULL, axis.title.hjust = NULL,
  axis.title.vjust = NULL, axis.title.y = waiver(),
  axis.title.y.position = c("left", "right"), axis.title.y.theme = NULL,
  axis.title.y.hjust = NULL, axis.title.y.vjust = NULL, label = TRUE,
  label.position = c("bottom", "top"), label.theme = NULL,
  label.hjust = NULL, label.vjust = NULL, label.y.position = c("left",
  "right"), label.y.theme = NULL, label.y.hjust = NULL,
  label.y.vjust = NULL, planewidth = NULL, planeheight = NULL,
  nbin = 20, ticks = TRUE, default.unit = "line", order = 0, ...)
```

Arguments

title
: A character string or expression indicating a title of guide. If NULL, the title is not shown. By default (waiver), the name of the scale object or the name specified in labs is used for the title.

title.position
: Character string indicating position for the main title. One of "top" (default) or "bottom".

`title.theme` A theme object for rendering the title text. Usually the object of `element_text` is expected. By default, the theme is specified by `legend.title` in `theme` or theme.

`title.hjust` A number specifying horizontal justification of the title text.

`title.vjust` A number specifying vertical justification of the title text.

`axis.title, axis.title.y`
Character strings or expressions indicating the horizontal and vertical axis titles in the guide, respectively. If `NULL`, the title is not shown. By default (waiver), the name of the scale or the name of the variable mapped to the aesthetic.

`axis.title.position, axis.title.y.position`
Character vectors indicating the position(s) of axis titles. `axis.title.position`: "top" and/or "bottom" (default). `axis.title.y.position`: "left" (default) and/or "right".

`axis.title.theme, axis.title.y.theme`
Theme objects for rendering the axis title text. Typically an `element_text` object. When `NULL`, defaults to settings for `axis.title.x` and `axis.title.y` in the plot theme.

`axis.title.hjust, axis.title.vjust, axis.title.y.vjust, axis.title.y.hjust`
Numerics specifying the horizontal (`hjust`) and vertical (`vjust`) justifications of the horizontal (`axis.title`) and vertical (`axis.title.y`) axis title text.

`label` logical. If `TRUE` then the labels are drawn. If `FALSE` then the labels are invisible.

`label.position, label.y.position`
Character vectors indicating the position(s) of axis labels. For `label.position`, "top" and/or "bottom" (default). For `label.y.position`, "left" (default) and/or "right".

`label.theme, label.y.theme`
Theme objects for rendering axis label text. Usually the object of `element_text` is expected. By default, the theme is specified by `axis.text.*` in the plot theme.

`label.hjust, label.vjust, label.y.hjust, label.y.vjust`
Numerics specifying the horizontal (`hjust`) and vertical (`vjust`) justifications of the horizontal (`label`) and vertical (`label.y`) axis label text.

`planewidth, planeheight`
Numeric or `unit` objects specifying the width and height of the colorplane. Default values are 5 times the `legend.key.width/height` or `legend.key.size` in the plot theme.

`nbin` Number specifying how many color pixels are generated for each dimension of the colorplane. Higher numbers increase guide color accuracy (especially for larger sized guides) at the expense of speed.

`ticks` A logical specifying if tick marks on colorbar should be visible.

`default.unit` A character string indicating unit for `planewidth` and `planeheight`.

`order` positive integer less that 99 that specifies the order of this guide among multiple guides. This controls the order in which multiple guides are displayed, not the contents of the guide itself. If 0 (default), the order is determined by a secret algorithm.

`...` ignored.

Details

The guide is based on guide_colorbar, but extended to be a plane of colors with ticks and labels for both variables in the scale. All *.theme arguments accept two types of arguments: a complete theme object (e.g. the object returned by theme_grey) or an element_text. If a theme is given, the related element will be extracted from the theme and used as-is. If an element is given, any missing parameters will be inherited from the plot's theme before use. If not specified, *.hjust and *.vjust parameters will draw from the corresponding *.theme argument, the plot's theme, or a default of 0.5 (centered). One exception is title.hjust which has been given a default value of 0.5 to override a undesirable default value in the default ggplot theme. Specify title.hjust = NULL to restore normal inheritance if needed.

References

Based on guide_colorbar, modified 2016.

Examples

```
if(requireNamespace("mapproj")) {
  library(ggplot2)
  crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)
  states_map <- map_data("state")
  ggplot(crimes,
         aes(map_id = state, fill = Murder, fill2 = UrbanPop)) +
    geom_map(map = states_map) +
    scale_fill_colorplane() +
    expand_limits(x = states_map$long, y = states_map$lat) +
    coord_map() +
    guides(fill = guide_colorplane("My Title", axis.title = "Murder Rate",
    axis.title.y = "Urban Population %", label.position = c("top", "bottom"),
    label.y.position = c("left", "right")))
 }
```

---

scale_color_colorplane

Bivariate Color Space Projection Scale

Description

Maps two continuous variables into a single display color, using either the color and color2 aesthetics (scale_color_colorplane) or the fill and fill2 aesthetics (scale_fill_colorplane). Variables mapped to color or fill are be mapped to the horizontal component of the colorplane scale and color/fill2 are mapped to the vertical component.

Usage

```
scale_color_colorplane(name = waiver(), axis_title = waiver(),
  axis_title_y = waiver(), breaks = waiver(), breaks_y = waiver(),
  labels = waiver(), labels_y = waiver(), limits = NULL,
  limits_y = NULL, color_projection = "YUV", rescaler = rescale,
  oob = censor, trans = "identity", na.color = "black",
  na.value = NA_real_, guide = "colorplane", ...)

scale_fill_colorplane(name = waiver(), axis_title = waiver(),
  axis_title_y = waiver(), breaks = waiver(), breaks_y = waiver(),
  labels = waiver(), labels_y = waiver(), limits = NULL,
  limits_y = NULL, color_projection = "YUV", rescaler = rescale,
  oob = censor, trans = "identity", na.color = "black",
  na.value = NA_real_, guide = "colorplane", ...)
```

Arguments

`name` Character string or expression to be used as guide title. Defaults to "Color Key" or "Fill Color Key" to match the scale function used.

`axis_title, axis_title_y`
Character strings or expressions indicating the horizontal and vertical axis titles in the guide, respectively. If `NULL`, the title is not shown. By default (waiver), the name of the scale or the name of the variable mapped to the aesthetic.

`breaks` One of:

- `NULL` for no breaks
- `waiver()` for the default breaks computed by the transformation object
- A numeric vector of positions
- A function that takes the limits as input and returns breaks as output

`breaks_y` As `breaks`, but for vertical axis (i.e. `color2` or `fill2`)

`labels` One of:

- `NULL` for no labels
- `waiver()` for the default labels computed by the transformation object
- A character vector giving labels (must be same length as `breaks`)
- A function that takes the breaks as input and returns labels as output

`labels_y` As `labels`, but for vertical axis (i.e. `color2` or `fill2`)

`limits` A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum.

`limits_y` As `limits`, but for vertical axis (i.e. `color2` or `fill2`)

`color_projection`
Projection mapping to use. Either the name of an included projection or a function that performs the projection. See `color_projections`.

`rescaler` Used by diverging and n colour gradients (i.e. `scale_colour_gradient2`, `scale_colour_gradientn`). A function used to scale the input values to the range [0, 1].

`oob` Function that handles limits outside of the scale limits (out of bounds). The default replaces out of bounds values with NA.

`trans` Either the name of a transformation object, or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "exp", "identity", "log", "log10", "log1p", "log2", "logit", "probability", "probit", "reciprocal", "reverse" and "sqrt".

A transformation object bundles together a transform, it's inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called `name_trans`, e.g. `boxcox_trans`. You can create your own transformation with `trans_new`.

`na.color` Character string containing a valid R color to use when plotting missing data or data outside the limits.

`na.value` Missing values will be replaced with this value.

`guide` Name of guide object, or object itself. Defaults to `guide_colorplane` designed for this scale. Behavior of other guides with this scale is not defined.

`...` Additional arguments to pass on to `color_projection` function.

Details

Variable values are projected into color space to create a bivariate gradient. The default projection maps values to the U and V components of YUV color space. In the YUV color space, the full spectrum of chrominance (color difference) is encoded into the U and V components and luminosity (brightness) is encoded in the Y component. For a fixed value of Y, the remaining U-V color space is a plane of all possible colors at that brightness. Therefore, mapping data to this projection utilizes the full color spectrum to provide visual discrimination between differing values.

The YUV projection colorplane scale is visually divided into four quadrants: green when both values are small, fuchsia when both are large, orange when the horizontal variable is small and the vertical is large, and blue when the horizontal variable is large and the vertical is small. Values closer to the extremes are saturated and the center of the scale, representing the mid-point of the ranges for both variables, is grey.

Alternative color projections can be used, but may not be as interpretable. See `color_projections` for information on specifying or creating other color projections.

References

Based on `continuous_scale`, modified 2016.

Examples

```
library(ggplot2)
if(requireNamespace("mapproj")) {
  crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)
  states_map <- map_data("state")
  ggplot(crimes,
         aes(map_id = state, fill = Murder, fill2 = UrbanPop)) +
    geom_map(map = states_map) +
    scale_fill_colorplane() +
    expand_limits(x = states_map$long, y = states_map$lat) +
    coord_map()
 }
# setting upper limit for qsec causes points for higher values to plot
# as na.color (black)
ggplot(mtcars, aes(x = wt, y = mpg, color = qsec, colour2 = hp)) +
  geom_point(size = 4) +
  scale_color_colorplane(limits = c(NA, 18.9))
```

Index