Package 'qualpalr'

May 9, 2026

Title Automatic Generation of Qualitative Color Palettes

Version 2.0.0

Description Automatic generation of maximally distinct qualitative color palettes, optionally tailored to color deficiency. A set of colors or a subspace of a color space is used as input and a final palette of specified size is generated by picking colors that maximize the minimum pairwise difference among the chosen colors. Adaptations to color vision deficiency, background colors, and white points are supported.

License GPL-3

BugReports https://github.com/jolars/qualpalr/issues

URL https://jolars.github.io/qualpalr/

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

SystemRequirements C++17

Depends R (>= 4.0.0)

Imports graphics, stats, utils, Rcpp

Suggests randtoolbox (>= 1.17), testthat, knitr, rmarkdown, maps, rgl, spelling, covr

LinkingTo Rcpp (>= 0.12.9)

Language en-US

NeedsCompilation yes

Author Johan Larsson [aut, cre] (ORCID: <https://orcid.org/0000-0002-4029-5945>)

Maintainer Johan Larsson <johan@jolars.co>

Repository CRAN

Date/Publication 2026-01-29 17:10:07 UTC

Contents

`analyze_palette` *Analyze a categorical color palette*

Description

Analyze a categorical color palette with respect to the differences between the colors in the palette.

Usage

```
analyze_palette(
  palette,
  cvd = c(protan = 0, deutan = 0, tritan = 0),
  bg = NULL,
  metric = c("ciede2000", "din99d", "cie76")
)
```

Arguments

`palette`	Either a matrix of RGB values (with values between 0 and 1), a data frame with RGB values, or a character vector of hex colors.
`cvd`	Color vision deficiency adaptation. This must be a named vector with names `protan`, `deutan`, and `tritan` and values between 0 and 1, where 0 means no adaptation and 1 means full adaptation.
`bg`	Background color to consider (but not include) when generating the palette. This is useful to avoid colors that are too close to the background/canvas color. If `NULL` (the default), the background color is not considered at all. Any color that is convertable via `col2rgb` is acceptable, including hex colors.
`metric`	The color metric to use for the color distance matrix.

Value

A list of lists, one for each type of color vision deficiency plus normal vision. Each list contains `difference_matrix`, `min_distances`, and `bg_min_distance`

See Also

qualpal()

Examples

```
pal <- qualpal(5)
analyze_palette(pal$hex, cvd = c(protan = 1))
```

---

convert_colors *Convert colors between colorspaces*

Description

Convert colors between colorspaces

Usage

```
convert_colors(colors, from, to)
```

Arguments

colors	A matrix of colors
from	The colorspace of the input colors, one of "rgb", "hsl", "din99d", "lab", "xyz"
to	The colorspace of the output colors, one of "rgb", "hsl", "lab", "xyz"

Value

The colors converted to the new colorspace

---

get_palette *Retrieve one of the built-in color palettes*

Description

This function retrieves a color palette from the built-in palettes. To see the available palettes, use list_palettes().

Usage

```
get_palette(palette = "ColorBrewer:Accent")
```

Arguments

palette	A character string specifying the name of the color palette to

Value

A character vector of colors in hex format.

See Also

`list_palettes()`

Examples

```
get_palette("Vermeer:LittleStreet")
```

`list_palettes` *List available color palettes*

Description

List available color palettes

Usage

```
list_palettes()
```

Value

A list of available color palettes. Each palette is a named list with a character vector.

See Also

`qualpal()`

Examples

```
list_palettes()
```

pairs.qualpal *Scatterplot matrix of qualitative color palette*

Description

Plots the colors in an object of class "qualpal" as a scatterplot matrix on either the DIN99d (the default) or HSL color space.

Usage

```
## S3 method for class 'qualpal'
pairs(x, colorspace = c("DIN99d", "HSL", "RGB"), ...)
```

Arguments

x	A list object of class "qualpal" generated from qualpal.
colorspace	The color space in which to plot the colors ("DIN99d", "HSL", or "RGB").
...	Arguments to pass on to pairs.

See Also

qualpal, plot.qualpal, pairs

Examples

```
col_pal <- qualpal(3)
pairs(col_pal)
pairs(col_pal, colorspace = "HSL")
```

plot.qualpal *Multidimensional scaling map of qualitative color palette*

Description

Uses the colors in a qualpal object to compute and plot a multidimensional scaling (MDS) map using cmdscale on the Delta E DIN99d distance matrix.

Usage

```
## S3 method for class 'qualpal'
plot(x, ...)
```

Arguments

x	An object of class "qualpal" generated from qualpal.
...	Arguments to pass on to plot.

See Also

qualpal, pairs.qualpal, plot

Examples

```
col_pal <- qualpal(3)
plot(col_pal)
```

---

print.qualpal *Print qualpal palette*

Description

Print the result from a call to qualpal.

Usage

```
## S3 method for class 'qualpal'
print(x, colorspace = c("HSL", "DIN99d", "RGB"), digits = 2, ...)
```

Arguments

x	An object of class "qualpal".
colorspace	Color space to print colors in.
digits	Number of significant digits for the output. (See print.default.) Setting it to NULL uses getOption("digits").
...	Arguments to pass to print.default.

Value

Prints the colors as a matrix in the specified color space as well as a distance matrix of the color differences. Invisibly returns x.

Examples

```
f <- qualpal(3)
print(f, colorspace = "DIN99d", digits = 3)
```

`qualpal` *Generate qualitative color palettes*

Description

Given a collection of colors, `qualpal()` algorithmically tries to select to `n` most distinct colors from the provided input colors, optionally taking color vision deficiency into account.

Usage

```
qualpal(
  n,
  colorspace = list(h = c(0, 360), s = c(0.2, 0.5), l = c(0.6, 0.85)),
  cvd = c(protan = 0, deutan = 0, tritan = 0),
  cvd_severity,
  bg = NULL,
  metric = c("ciede2000", "din99d", "cie76"),
  extend = NULL,
  white_point = c("D65", "D50", "D55", "A", "E")
)
```

Arguments

`n` The number of colors to generate.

`colorspace` A color space to generate colors from. Can be any of the following:

- A `list` that describes a color space in either HSL or LCHab color space. In the first case (HSL), the list must contain the following *named* vectors, each of length two, giving a range for each item.
  - `h` Hue, in range from -360 to 360
  - `s` Saturation, in the range from 0 to 1
  - `l` Lightness, in the range from 0 to 1

  In the second case (LCHab), the list must contain the following *named* vectors, each of length two, giving a range for each item.
  - `h` Hue, in range from -360 to 360
  - `c` Chroma, in the range from 0 to infinity
  - `l` Lightness, in the range from 0 to 100

  In these cases, `qualpal()` will generate
- A `character` vector of length one in the form of "Source:Palette", where *Domain* is the name of a source that provides a color palette, and *Palette* is the name of a color palette from that source. See `list_palettes()` for available palettes.
- A matrix or data frame of RGB values (with values between 0 and 1).

`cvd` Color vision deficiency adaptation. This must be a named vector with names `protan`, `deutan`, and `tritan` and values between 0 and 1, where 0 means no adaptation and 1 means full adaptation.

cvd_severity	DEPRECATED. Use a named cvd vector instead, e.g. c(protan = 0.5, deutan = 0.2, tritan = 0).
bg	Background color to consider (but not include) when generating the palette. This is useful to avoid colors that are too close to the background/canvas color. If NULL (the default), the background color is not considered at all. Any color that is convertable via col2rgb is acceptable, including hex colors.
metric	The color metric to use for the color distance matrix.
extend	A palette of colors to use as a fixed set of initial colors in the palette, which can be either a matrix or data frame of RGB values (with values between 0 and 1) or a character vector of hex colors (or any other format that's acceptable in grDevices::col2rgb()).
white_point	The white point to use for color space conversions. Can be one of "D65" (default, daylight at 6500K), "D50" (daylight at 5000K), "D55" (daylight at 5500K), "A" (incandescent tungsten at 2856K), or "E" (equal energy).

Details

The main idea is to compute a distance matrix from all the input colors, and then try to select the most distinct colors based on the color differences between them. It does this iteratively by first selecting the first n colors from the input colors, then iterates over the palette, putting colors back into the total set and replaces it with a new color until it has gone through the whole range without changing any of the colors.

Optionally, qualpal can adapt palettes to cater to color vision deficiency (CVD). This is accomplished by taking the colors provided by the user and transforming them to colors that someone with CVD would see, that is, simulating CVD qualpal then chooses colors from these new colors.

Value

A list of class qualpal with the following components.

HSL	A matrix of the colors in the HSL color space.
RGB	A matrix of the colors in the sRGB color space.
hex	A character vector of the colors in hex notation.
de_DIN99d	A distance matrix of color differences according to the metric used. The name is misleading, bu kept for backwards compatibility.
hex	A character vector of the colors in hex notation.
min_de_DIN99d	The minimum pairwise DIN99d color difference among all colors in the palette.

See Also

plot.qualpal(), pairs.qualpal(), list_palettes()

Examples

```
# Generate 3 distinct colors from the default color space
qualpal(3)

# Provide a custom color space
qualpal(n = 3, list(h = c(35, 360), s = c(0.5, 0.7), l = c(0, 0.45)))

qualpal(3, "ColorBrewer:Set2")

# Adapt palette to deuteranopia
qualpal(5, "ColorBrewer:Dark2", cvd = c(deutan = 1))

# Adapt palette to protanomaly with severity 0.4
qualpal(8, cvd = c(protan = 0.4))

# Generate and extend a palette with 3 colors, using the DIN99d
# metric
pal <- qualpal(3)
qualpal(5, extend = pal$hex, metric = "din99d")

# Use a different white point (D50, common in printing)
qualpal(5, white_point = "D50")

## Not run:
# The range of hue cannot exceed 360
qualpal(3, list(h = c(-20, 360), s = c(0.5, 0.7), l = c(0, 0.45)))

## End(Not run)
```

Index