

Package ‘colorrepel’

July 22, 2025

Title Repel Visually Similar Colors for Colorblind Users in Various Plots

Version 0.4.3

Description Iterate and repel visually similar colors away in various 'ggplot2' plots. When many groups are plotted at the same time on multiple axes, for instance stacked bars or scatter plots, effectively ordering colors becomes difficult. This tool iterates through color combinations to find the best solution to maximize visual distinctness of nearby groups, so plots are more friendly toward color-blind users. This is achieved by two distance measurements, distance between groups within the plot, and CIELAB color space distances between colors as described in Carter et al., (2018) <[doi:10.25039/TR.015.2018](https://doi.org/10.25039/TR.015.2018)>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

Depends R (>= 3.5.0)

Imports grDevices, matrixStats, Matrix, distances, stats, dqrng, gtools, purrr, dplyr, plyr, stringr, ggplot2, ggrepel, plotly, knitr, png, Polychrome

Suggests colorspace

URL https://github.com/raysinensis/color_repel

ByteCompile true

NeedsCompilation no

Author Rui Fu [cre, aut, cph] (ORCID: <<https://orcid.org/0000-0001-8183-4549>>)

Maintainer Rui Fu <raysinensis@gmail.com>

Repository CRAN

Date/Publication 2025-07-21 13:52:02 UTC

Contents

average_clusters 2

average_clusters_rowwise	3
by_cluster_sampling	4
calc_distance	5
color_repel	6
get_labs	7
ggplotly_background	8
gg_color_repel	9
label_repel	11
matrix2_score	12
matrix2_score_n	12

Index	14
--------------	-----------

average_clusters	<i>Average expression values per cluster</i>
------------------	--

Description

Average expression values per cluster

Usage

```
average_clusters(  
  mat,  
  metadata,  
  cluster_col = "cluster",  
  if_log = TRUE,  
  cell_col = NULL,  
  low_threshold = 0,  
  method = "mean",  
  output_log = TRUE,  
  cut_n = NULL  
)
```

Arguments

mat	expression matrix
metadata	data.frame or vector containing cluster assignments per cell. Order must match column order in supplied matrix. If a data.frame provide the cluster_col parameters.
cluster_col	column in metadata with cluster number
if_log	input data is natural log, averaging will be done on unlogged data
cell_col	if provided, will reorder matrix first
low_threshold	option to remove clusters with too few cells
method	whether to take mean (default), median, 10% truncated mean, or trimean, max, min, sum

output_log	whether to report log results
cut_n	set on a limit of genes as expressed, lower ranked genes are set to 0, considered unexpressed

Value

average or other desired calculation by group/cluster matrix

Examples

```
mat <- average_clusters(data.frame(
  z = c(1, 2, 3, 4, 5, 6),
  y = c(1, 2, 3, 4, 5, 6),
  x = c(1, 2, 3, 4, 5, 6)
), metadata = c(1, 1, 2), method = "sum")
```

average_clusters_rowwise

Rowwise math from matrix/data.frame per cluster based on another vector/metadata, similar to clustifyr::average_clusters but ids as rows

Description

Rowwise math from matrix/data.frame per cluster based on another vector/metadata, similar to clustifyr::average_clusters but ids as rows

Usage

```
average_clusters_rowwise(
  mat,
  metadata,
  cluster_col = "cluster",
  if_log = FALSE,
  cell_col = NULL,
  low_threshold = 0,
  method = "mean",
  output_log = FALSE,
  cut_n = NULL,
  trim = FALSE
)
```

Arguments

mat	expression matrix
metadata	data.frame or vector containing cluster assignments per cell. Order must match column order in supplied matrix. If a data.frame provide the cluster_col parameters.

cluster_col	column in metadata with cluster number
if_log	input data is natural log, averaging will be done on unlogged data
cell_col	if provided, will reorder matrix first
low_threshold	option to remove clusters with too few cells
method	whether to take mean (default), median, 10% truncated mean, or trimean, max, min, sum
output_log	whether to report log results
cut_n	set on a limit of genes as expressed, lower ranked genes are set to 0, considered unexpressed
trim	whether to remove 1 percentile when doing min calculation

Value

average expression matrix, with genes for row names, and clusters for column names

Examples

```
mat <- average_clusters_rowwise(data.frame(
  y = c(1, 2, 3, 4, 5, 6),
  x = c(1, 2, 3, 4, 5, 6)
), metadata = c(1, 2, 1, 2, 1, 2), method = "min")
```

by_cluster_sampling	<i>Balanced downsampling of matrix/data.frame based on cluster assignment vector</i>
---------------------	--

Description

Balanced downsampling of matrix/data.frame based on cluster assignment vector

Usage

```
by_cluster_sampling(df, vec, frac, seed = 34)
```

Arguments

df	expression matrix or data.frame
vec	vector of ids
frac	fraction 0-1 to downsample to
seed	sampling randomization seed

Value

list with new downsampled matrix/data.frame and id vector

Examples

```
res <- by_cluster_sampling(data.frame(y = c(1, 2, 3, 4, 5, 6)),
  vec = c(1, 2, 1, 2, 1, 2), frac = 0.5
)
```

calc_distance

*Distance calculations for spatial coord***Description**

Distance calculations for spatial coord

Usage

```
calc_distance(
  coord,
  metadata,
  cluster_col = "cluster",
  collapse_to_cluster = FALSE
)
```

Arguments

coord	dataframe or matrix of spatial coordinates, cell barcode as rownames
metadata	data.frame or vector containing cluster assignments per cell. Order must match column order in supplied matrix. If a data.frame provide the cluster_col parameters.
cluster_col	column in metadata with cluster number
collapse_to_cluster	instead of reporting min distance to cluster per cell, summarize to cluster level

Value

min distance matrix

color_repel

Reorder ggplot colors to maximize color differences in space

Description

Reorder ggplot colors to maximize color differences in space

Usage

```
color_repel(
  g,
  coord = NULL,
  groups = NULL,
  nsamp = 50000,
  sim = NULL,
  severity = 0.5,
  verbose = FALSE,
  downsample = 5000,
  polychrome_recolor = FALSE,
  seed = 34,
  col = "colour",
  autoswitch = TRUE,
  layer = 1,
  out_orig = FALSE,
  out_worst = FALSE,
  ggbuild = NULL
)
```

Arguments

g	ggplot plot object
coord	coordinates, default is inferred
groups	groups corresponding to color/fill, default is inferred
nsamp	how many random sampling color combinations to test, default 50000
sim	passing a colorbind simulation function if needed
severity	severity of the color vision defect, between 0 and 1
verbose	whether to print messages
downsample	downsample when too many datapoints are present, or use chull
polychrome_recolor	whether to replace the original colors with polychrome creation
seed	sampling randomization seed
col	colour or fill in ggplot
autoswitch	try to switch between colour and fill automatically

layer	layer to detect color, defaults to first
out_orig	output the original colors as named vector
out_worst	output the worst combination instead of best
ggbuild	already built ggplot_built object if available

Value

vector of reordered colors

Examples

```
a <- ggplot2::ggplot(ggplot2::mpg, ggplot2::aes(displ, hwy)) +  
  ggplot2::geom_point(ggplot2::aes(color = as.factor(cyl)))  
new_colors <- color_repel(a)  
b <- a + ggplot2::scale_color_manual(values = new_colors)
```

get_labs

Extract custom labels from ggplot object

Description

Extract custom labels from ggplot object

Usage

```
get_labs(g, ggbuild = NULL)
```

Arguments

g	ggplot object
ggbuild	already built ggplot_built object if available

Value

named vector of labels

Examples

```
a <- ggplot2::ggplot(ggplot2::mpg, ggplot2::aes(displ, hwy)) +  
  ggplot2::geom_point(ggplot2::aes(color = as.factor(cyl))) +  
  ggplot2::geom_text(ggplot2::aes(label = model))  
get_labs(a)
```

ggplotly_background *Prepare ggplot object to ggplotly-compatible layer and image layer*

Description

Prepare ggplot object to ggplotly-compatible layer and image layer

Usage

```
ggplotly_background(
  g,
  repel_color = TRUE,
  repel_label = TRUE,
  encircle = FALSE,
  mascarade = FALSE,
  width = 5,
  height = 5,
  filename = "temp.png",
  draw_box = NULL,
  background = NULL,
  background_alpha = 1,
  use_cairo = FALSE,
  label_lim = 0.05,
  ggbuild = NULL,
  crop = TRUE,
  size_nudge = 0,
  tooltip = NULL,
  ...
)
```

Arguments

<code>g</code>	ggplot plot object
<code>repel_color</code>	whether to rearrange colors
<code>repel_label</code>	whether to add centroid labels with <code>ggrepel</code>
<code>encircle</code>	whether to draw <code>geom_encircle</code> by cluster
<code>mascarade</code>	use mascarade package to outline clusters
<code>width</code>	plot width
<code>height</code>	plot height
<code>filename</code>	temp file location for saving image
<code>draw_box</code>	if a colored background should be included
<code>background</code>	if specified, use this ggplot object or file as background instead
<code>background_alpha</code>	alpha value of background image

use_cairo	whether to use cairo for saving plots, maybe needed for certain ggplot extensions
label_lim	whether to limit labels to avoid edge fraction
ggbuild	already built ggplot_built object if available
crop	whether to call cropping of the background image to remove whitespace
size_nudge	slight image size adjustment, default to none
tooltip	tooltip vector for ggplotly
...	arguments passed to gg_color_repel

Value

plotly object with background image of layers unsupported by plotly

Examples

```
a <- ggplot2::ggplot(ggplot2::mpg, ggplot2::aes(displ, hwy)) +
  ggplot2::geom_point(ggplot2::aes(color = as.factor(cyl)))
new_colors <- color_repel(a)
```

gg_color_repel	<i>Wrapper to reorder ggplot colors to maximize color differences in space</i>
----------------	--

Description

Wrapper to reorder ggplot colors to maximize color differences in space

Usage

```
gg_color_repel(
  g = ggplot2::last_plot(),
  col = "colour",
  sim = NULL,
  severity = 0.5,
  verbose = FALSE,
  downsample = 5000,
  nsamp = 50000,
  polychrome_recolor = FALSE,
  seed = 34,
  autoswitch = TRUE,
  layer = 1,
  out_orig = FALSE,
  out_worst = FALSE,
  repel_label = FALSE,
  encircle = FALSE,
  encircle_alpha = 0.25,
  encircle_expand = 0.02,
```

```

    encircle_shape = 0.5,
    encircle_threshold = 0.01,
    encircle_nmin = 0.01,
    mascarade = FALSE,
    ggbuild = NULL,
    ...
  )

```

Arguments

<code>g</code>	ggplot plot object
<code>col</code>	colour or fill in ggplot
<code>sim</code>	passing a colorbind simulation function if needed
<code>severity</code>	severity of the color vision defect, between 0 and 1
<code>verbose</code>	whether to print messages
<code>downsample</code>	downsample when too many datapoints are present
<code>nsamp</code>	how many random sampling color combinations to test, default 50000
<code>polychrome_recolor</code>	whether to replace the original colors with polychrome creation
<code>seed</code>	sampling randomization seed
<code>autoswitch</code>	try to switch between colour and fill automatically
<code>layer</code>	layer to detect color, defaults to first
<code>out_orig</code>	output the original colors as named vector
<code>out_worst</code>	output the worst combination instead of best
<code>repel_label</code>	whether to add centroid labels with ggrepel
<code>encircle</code>	whether to draw geom_encircle by cluster
<code>encircle_alpha</code>	alpha argument passed to geom_encircle
<code>encircle_expand</code>	expand argument passed to geom_encircle
<code>encircle_shape</code>	shape/smoothing argument passed to geom_encircle
<code>encircle_threshold</code>	threshold for removing outliers
<code>encircle_nmin</code>	number of near neighbors for removing outliers
<code>mascarade</code>	use mascarade package to outline clusters
<code>ggbuild</code>	already built ggplot_built object if available
<code>...</code>	passed to <code>repel_label</code>

Value

new ggplot object

Examples

```

a <- ggplot2::ggplot(ggplot2::mpg, ggplot2::aes(displ, hwy)) +
  ggplot2::geom_point(ggplot2::aes(color = as.factor(cyl)))
b <- gg_color_repel(a, col = "colour")

```

label_repel	<i>ggrepel labeling of clusters</i>
-------------	-------------------------------------

Description

ggrepel labeling of clusters

Usage

```
label_repel(  
  g,  
  group_col = "auto",  
  x = "x",  
  y = "y",  
  txt_pt = 3,  
  remove_current = "auto",  
  layer = "auto",  
  ggbuild = NULL,  
  ...  
)
```

Arguments

<code>g</code>	ggplot object or data.frame
<code>group_col</code>	column name in data.frame, default to "label" or "group" in ggplot data
<code>x</code>	column name in data.frame for x
<code>y</code>	column name in data.frame for y
<code>txt_pt</code>	text size
<code>remove_current</code>	whether to remove current text
<code>layer</code>	text layer to remove, defaults to last
<code>ggbuild</code>	already built ggplot_built object if available
<code>...</code>	arguments passed to <code>geom_text_repel</code>

Value

function, if data.frame input, or new ggplot object

Examples

```
g <- label_repel(ggplot2::ggplot(mtcars, ggplot2::aes(x = hp, y = wt, color = as.character(cyl))) +  
  ggplot2::geom_point(), remove_current = FALSE)
```

matrix2_score	<i>Score matrix distances</i>
---------------	-------------------------------

Description

Score matrix distances

Usage

```
matrix2_score(dist1, dist2)
```

Arguments

dist1	distanct matrix 1
dist2	distanct matrix 2

Value

numeric score

matrix2_score_n	<i>Score matrix distances in multiple combinations</i>
-----------------	--

Description

Score matrix distances in multiple combinations

Usage

```
matrix2_score_n(
  dist1,
  dist2,
  n = min(factorial(ncol(dist2)) * 10, 20000),
  verbose = FALSE,
  seed = 34,
  out_worst = FALSE
)
```

Arguments

dist1	distanct matrix 1
dist2	distanct matrix 2
n	number of iterations
verbose	whether to output more messages
seed	random seed
out_worst	instead of default output of best combination, output worst instead

matrix2_score_n

13

Value

reordered vector

Index

average_clusters, [2](#)
average_clusters_rowwise, [3](#)

by_cluster_sampling, [4](#)

calc_distance, [5](#)
color_repel, [6](#)

get_labs, [7](#)
gg_color_repel, [9](#)
ggplotly_background, [8](#)

label_repel, [11](#)

matrix2_score, [12](#)
matrix2_score_n, [12](#)