Helpful ggplot2 Extensions: **ggbump CHEAT SHEET**

ggbump

ggbump Introduction

The R package `ggbump` creates varies bump charts in ggplot. Bump charts are good to plot the path between nodes that have no statistical significance. For example, ranking or classification. `ggbump` also includes functions to create custom smooth lines called sigmoid curves.

Installation

Install `ggbump` from CRAN with:

```
install.packages("ggbump")
```

or directly from github

```
devtools::install_github("davidsjoberg/ggbump")
```

Four Essential Components

geom_bump	Creates a ggplot that makes a smooth rank over time.
geom_sigmoid	Creates a ggplot that makes a smooth rank over time with endpoints.
rank_sigmoid	Creates a longer dataframe with coordinates for a smoothed line.
sigmoid	Creates a longer dataframe with coordinates for a smoothed line with endpoints

Basic ggbump format

geom_bump	ggplot(df, aes()) + geom_bump()
geom_sigmoid	ggplot(df, aes()) + geom_sigmoid()
rank_sigmoid	rank_sigmoid(x, y, smooth = 5, direction = "x")
sigmoid	sigmoid(x_start, x_end, y_start, y_end, smooth = 5, n = 100, direction = "x")

Arguments

sigmoid

x_start	start vector x value
x_end	end vector x value
y_start	start vector y value
y_end	end vector y values
smooth	smooth parameter. Smaller = more smooth
n	number of point to be smoothed
direction	x or y depending on direction of smoothing

rank_sigmoid

x	vector x
y	vector y
smooth	smooth parameter. Smaller = more smooth
direction	x or y depending on direction of smoothing

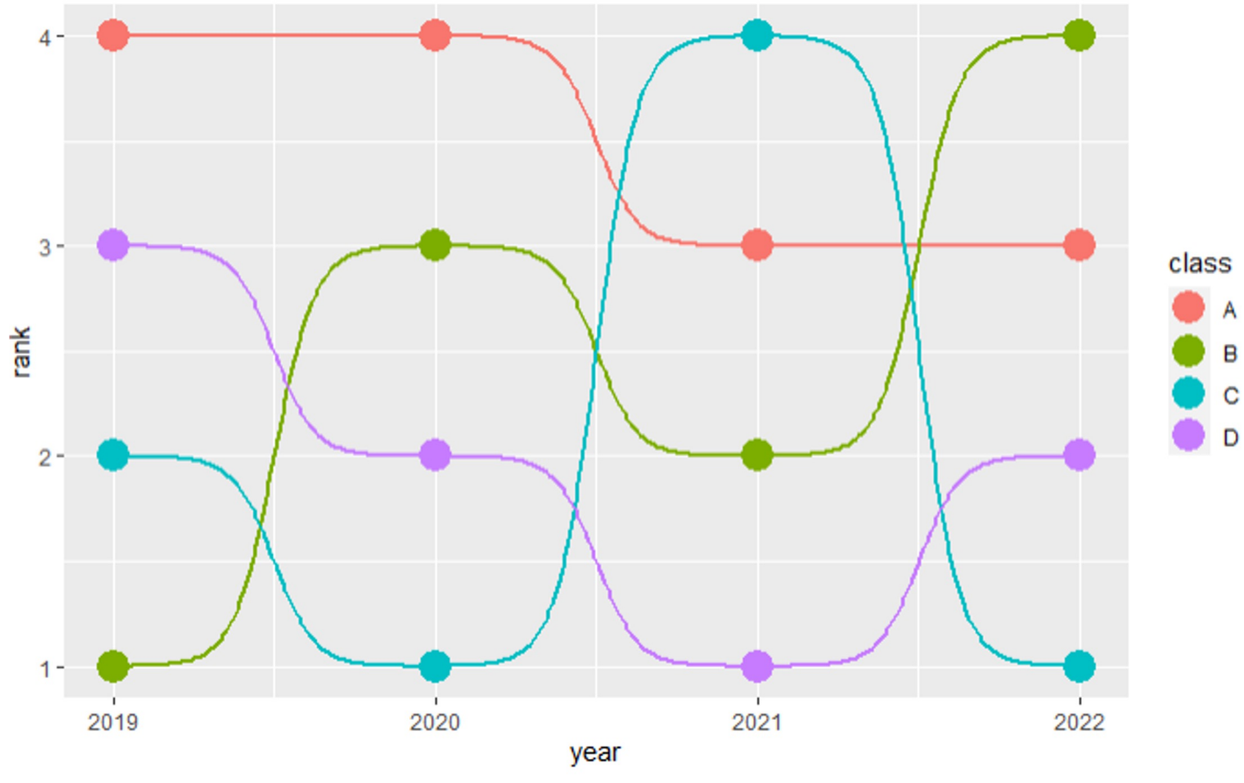
geom_sigmoid & geom_bump

mapping	own numeric mapping
data	own data
geom	change of geom
position	change of position
smooth	smooth parameter. Smaller = more smooth
direction	x or y depending on direction of smoothing
na.rm	remove missing values
show.legend	show legend in plot
inherit.aes	should the geom inherits aesthetics

Example

geom_bump

```
library(ggplot2)
library(ggbump)
library(dplyr)
df <- data.frame(class = c(
  "A", "A", "A","A",
  "B", "B", "B","B",
  "C", "C", "C","C",
  "D", "D", "D","D"),
year = c(2019, 2020, 2021, 2022,
         2019, 2020, 2021, 2022,
         2019, 2020, 2021, 2022,
         2019, 2020, 2021, 2022),
rank = c(4, 4, 3, 3, 1, 3, 2, 4,  2, 1, 4, 1, 3 ,2 ,1 ,2))
ggplot(df, aes(year, rank, color = class)) +
       geom_point(size = 6) +
       geom_bump(size = 1,position = "identity",smooth = 8)
```



Frank Li • zl3204@columbia.edu • Liang Zhuang • lz2862@columbia.edu

Helpful ggplot2 Extensions : **ggradar CHEAT SHEET**

ggradar

ggradar Introduction

The R package `ggradar` enables users to build radar charts. Radar charts are useful for data values with multiple common variables and is widely used for performance analysis. Using Radar charts, users can easily make comparisons among different entries based on the difference of the common variables. `ggradar` is best for only a few data values/groups. Otherwise, it is hard to distinguish lines for different groups and hard to make comparisons.

Installation

Install `ggradar` directly from GitHub:

```
install.packages("devtools")
devtools::install_github(
          "ricardo-bion/ggrader",
          dependencies = TRUE)
```

Basic Syntax

rescale the data frame

```
library(dplyr)
library(scales)
library(tibble)
df <- df %>%
   as_tibble(rownames = "group") %>%
   mutate_at(vars(-group), rescale)
```

plot radar chart

```
library(ggradar)
ggradar(plot.data = df)
```

NOTE: Rescaling is important in order to plot a radar chart. Otherwise, the following error will occur:
Error: 'plot.data' contains value(s) > grid.max

Arguments

plot

plot.data	data frame of the plot
plot.legend	a boolean value indicating whether to include legend
plot.title	title of of the plot
plot.extent.x.sf	relative size of the plot on the x axis
plot.extent.y.sf	relative size of the plot on the y axis

axis

axis.labels	change the labels of the axes, the default value is the column name of the data frame
axis.label.size	size of axis label
axis.line.colour	axis color

grid & gridline

NOTE: Replace **LINE** with min, mid or max. min corresponds to the minimum grid line, mid corresponds to the middle grid line, and max corresponds to the maximum grid line.

grid.**LINE**	plot certain grid line at certain value
grid.line.width	width of grid lines
label.gridline.**LINE**	a boolean value indicating whether to include the label for certain grid line
grid.label.size	size of grid line labels
gridline.**LINE**.linetype	line type of certain grid line
gridline.**LINE**.colour	line color of certain grid line

fill

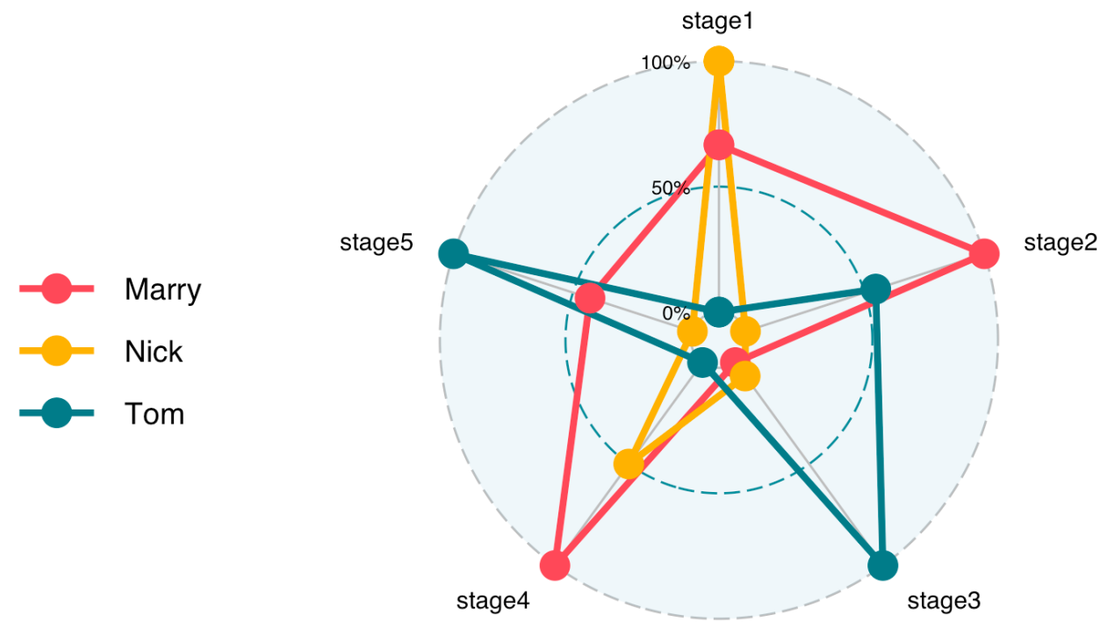
fill	a boolean value indicating whether to fill different polygons
fill.alpha	transparency of polygons

background

background.circle.colour	set color of background radar
background.circle.transparency	transparency of background radar

Example

```
library(dplyr)
library(scales)
library(tibble)
library(ggradar)
df <- data.frame(stage1 = c(4,8,10),stage2 = c(19,24,13),
      stage3 = c(84,69,70),stage4 = c(2,4,3), stage5 = c(15,11,8))
row.names(df) <- c("Tom", "Marry", "Nick")
df <- df %>% as_tibble(rownames = "group") %>%
      mutate_at(vars(-group), rescale)
ggradar(plot.data = df, grid.label.size = 4, grid.mid = 0.5,
        axis.label.size = 4,
        background.circle.colour = "#ADD8E6")
```



Source: https://github.com/ricardo-bion/ggradar

Frank Li • zl3204@columbia.edu • Liang Zhuang • lz2862@columbia.edu

Helpful ggplot2 Extensions : **Other ggplot2 Extensions**

ggpol

ggpol Introduction

The R package `ggpol` adds additional features to ggplot2 including `GeomArcbar`, `GeomParliament`, `GeomCircle`, `GeomTshignlight`, `FacetShare`, `GeomBartext`, and `GeomBoxjitter`. For detailed information of all those features, see this link: https://erocoar.github.io/ggpol/. This section includes one of the interesting features: `GeomParliament`. It combines half pie chart with dot plot, which is a great way to visualize the proportion of values with certain characteristics when the total sample size is not too large.

Installation

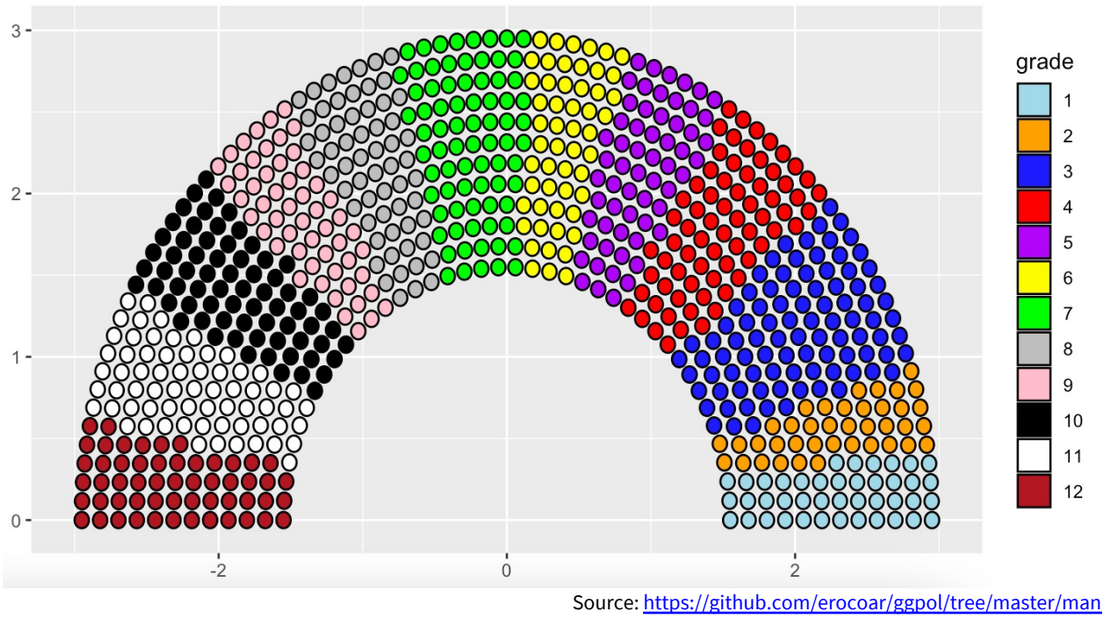
```
install.packages("ggbump") or
devtools::install_github("erocoar/ggpol")
```

Syntax

```
geom_parliament(data = DATAFRAME,
                aes(seats = COUNTS, fill = CATEGORIES))
```

Example

```
library(ggpol)
df_enrollment <- data.frame(grade = c(1,2,3,4,5,6,7,8,9,10,11,12),
  enrollment = c(42, 39, 92, 71, 60, 53, 77, 61, 54, 70, 66, 55),
  colors  = c("light blue", "orange", "blue", "red", "purple", "yellow",
              "green", "grey", "pink", "black", "white", "brown"))
ggplot() + geom_parliament(data = df_enrollment,
                           aes(seats = enrollment, fill = grade)) +
scale_fill_manual(values = df_enrollment$colors)
```



Source: https://github.com/erocoar/ggpol/tree/master/man

treemapify

treemapify Introduction

The R package `ggpol` enables users to draw a tree map in ggplot2. It is useful for data with hierarchy, such as country GDP and company's stock market share.

Installation

```
install.packages("treemapify") or
devtools::install_github("wilkox/treemapify")
```

Syntax

Treemap with no subgroup (such as continent)

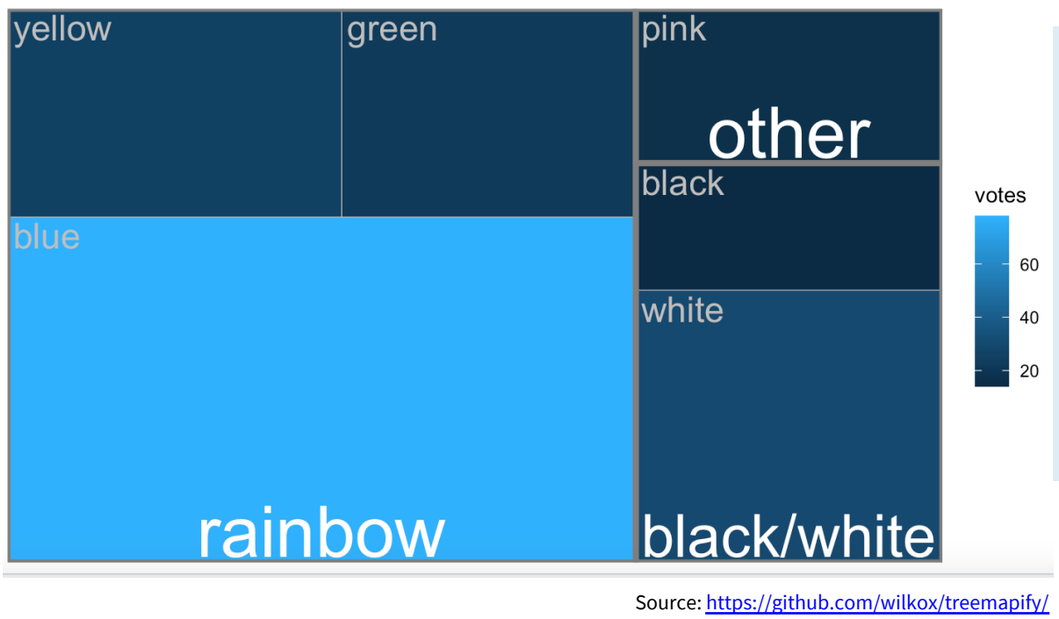
```
ggplot(data = DATAFRAME, aes(area = VALUE1, fill = VALUE2, label = NAME)) +
   geom_treemap() + geom_treemap_text()
```

Treemap with subgroup

```
ggplot(data = DATAFRAME, aes(area = VALUE1, fill = VALUE2,
           label = NAME, subgroup = SUBGROUP)) +
   geom_treemap() + geom_treemap_text() +
   geom_treemap_subgroup_text() + geom_treemap_subgroup_border()
```

Example

```
library(ggplot2)
library(treemapify)
df <- data.frame(color = c("blue","yellow","white","black","green","pink"),
   group=c("rainbow","rainbow","black/white","black/white","rainbow","other"),
   votes = c(78, 25, 30, 14, 22, 17))
ggplot(df, aes(area = votes, fill = votes, label = color, subgroup = group)) +
  geom_treemap() + geom_treemap_text(colour = "grey") +
  geom_treemap_subgroup_text(colour = "white") +geom_treemap_subgroup_border()
```



Source: https://github.com/wilkox/treemapify/

Other ggplot2 Extensions: https://exts.ggplot2.tidyverse.org/gallery/

Frank Li • zl3204@columbia.edu • Liang Zhuang • lz2862@columbia.edu