Data Visualization with GGPlot

Matt Steele

Resources

- Tidyverse Documentation
- O'Reilly Learning Platform
  - R for Data Science, 2nd Edition
  - R Programming for Statistics and Data Science

Recap

- Functions and Arguments
- Objects
- Run Code

Working Directory

The working directory in R is the folder where you are working. Hence, it's the place (the environment) where you have to store your files of your project in order to load them or where your R objects will be saved.

> *Session > Set Working Directory > Choose Directory*

```
getwd() # show current directory that you are in


setwd("path/to/your/directory") # sets the working directory
```

GGPlot

visualize your data with ggplot

- ggplot package

```
1  library(tidyverse)
```

Grammar of Graphics

the grammar of graphics is implemented through a layered approach to building plots.

Required

1. Data: The raw data that you want to visualize.
2. Aesthetic Mapping (aes()): Defines how variables in the data map to visual properties like position, color, size, shape, etc.
3. Geometric Objects (geom_*): Represent the actual geometric shapes on the plot (e.g., points, lines, bars).

Grammar of Graphics

the grammar of graphics is implemented through a layered approach to building plots.

Optional

4. Coordinate System (coord_*):Defines the coordinate system for the plot (e.g., Cartesian, polar).
5. Faceting (facet_*): Divides the plot into subplots based on one or more categorical variables.
6. Themes (theme_*): Customize the appearance of the plot, including titles, labels, and overall aesthetics.

Line graph

lets create a line graph that tracks approval ratings for the Supreme Court of the United States over time.

Data

start by loading the data set we are going to work on

```
scotus <- read_csv("scotus_approval.csv")

scotus


# let's just view the pollster YouGov using the filter function

scotus_yg <- scotus |>
  filter(pollster == "YouGov")

scotus_yg
```

Aesthetic Mapping (aes())

next we are going to set the variables that will use by using the ggplot function along with the aes function

```
1  scotus.line <- ggplot(scotus_yg, aes(date, per_yes))
```

Geometric Objects (geom_*)

finally we are going to choose the plot that we want to use for our visualization using the geom_* element

- Geom Cheatsheet

```
scotus.line +
  geom_line()
```

- in ggplot we combine elements of our plot using +

Theme: Color, Geom Size, Transparancy

fill = or **color =**	change geom color
size =	change geom size
alpha =	change geom transparency

```
# add color

scotus.line +
  geom_line(color = "coral")

# change the size

scotus.line +
  geom_line(color = "coral", size = 2)
```

Theme: Labels

the labs element will allow you to add or change labels in the plot

```
scotus.line +
  geom_line(color = "coral", size = 2) +
  labs(
    title = "SCOTUS Approval",
    subtitle = "2023",
    caption = "polls from YouGov",
    y = "Approval",
    x = NULL
  )
```

Themes: Built-in

ggplot has built-in themes with pre-set settings for you

```
scotus.line +
  geom_line(color = "coral", size = 2) +
  labs(
    title = "SCOTUS Approval",
    subtitle = "2023",
    caption = "polls from YouGov",
    y = "Approval",
    x = NULL
  ) +
  theme_minimal()
```

Themes: Customize

the theme element will allow you to customize the appearance of axes, legends, and labels

```
scotus.line +
  geom_line(color = "coral", size = 2) +
  labs(
    title = "SCOTUS Approval",
    subtitle = "2023",
    caption = "polls from YouGov",
    y = "Approval",
    x = NULL
  ) +
  theme_minimal() +
  theme(plot.title = element_text(size = 20, color = "navy"))
```

Theme: Scales

the scales element allows you to fin

- the **scale_x_date** element allows
- Date Formats - strptime

```
1  scotus.line +
2    geom_line(color = "coral", size = 2) +
3    labs(
4      title = "SCOTUS Approval",
```

Smoothed Lines

You can reduce overplotting using **loess** or **linear regression lines** with the geom_smooth or stat_smooth element

```
scotus.line +
  geom_smooth(color = "coral", size = 2) +
  labs(
    title = "SCOTUS Approval",
    subtitle = "2023",
    caption = "polls from YouGov",
    y = "Approval",
    x = NULL
  ) +
  theme_minimal() +
  theme(plot.title = element_text(size = 20, color = "coral")) +
  scale_x_date( date_breaks = "6 weeks",
                date_labels = "%b %d")
```

Export your plot

The ggsave function will export the most recent plot called in a file type specified by the user

- Additionally you can use the export options in RStudio's Plot tab in the Misc Pane

```
ggsave("scotus_approval.png", width = 6, height = 4, dpi = 300)
```

Histogram Graph

the histogram geom allows you to see the distribution of a continuous (dbl or num) variable

- set data, aesthetics, and geom

```
# load demographics data frame

demo <- read_csv("demographics.csv")

# let's look at the distribution of the age variable by creating a histogra

demo.hist <- ggplot(demo, aes(age))

demo.hist +
  geom_histogram()
```

Binning

the binning argument allows you to group continuous data into discrete intervals or bins

```
# number of bins to use

demo.hist +
  geom_histogram(bins = 10)

# length of a bins

demo.hist +
  geom_histogram(binwidth = 15)
```

Theme: Color, Geom Size, Transparancy

fill = or **color =**	change geom color
size =	change geom size
alpha =	change geom transparency

```
demo.hist +
  geom_histogram(bins = 25,
                 color = "coral",
                 fill = "skyblue",
                 alpha = .5) +
  theme_light()
```

Order of Elements

The order that the elements appear on the plot is dictated by its position in your code.

- The first elements in the code appear at the bottom of the plot and the last elements appear on the top of you plot

Mutliple Geoms

We can add multiple geoms into a plot by adding theme as their own element

- Example: the geom_vline/geom_hline element allows you to add a reference line to your plot

```
# add a reference line

demo.hist +
  geom_vline(xintercept = 40, color = "navy", size = 3) +
  geom_histogram(bins = 25, color = "coral", fill = "skyblue", alpha = .5)
  theme_light()
```

Faceting

the facet_grid or facet_wrap element will allow you to break your plot out by categorical variables

```
demo.hist +
  geom_vline(xintercept = 40, color = "navy", size = 3) +
  geom_histogram(bins = 25, color = "coral", fill = "skyblue", alpha = .5)
  theme_light() +
  facet_wrap(facets = vars(inccat), nrow = 3)
```

Bar Graph

the geom_bar element allows you create a bar chart uses the number of cases of each group in a categorical variable

```
demo.bar <- ggplot(demo, aes(carcat))

demo.bar +
  geom_bar()
```

Bar Graph

the geom_col element allows you to create a bar chart using a categorical and continuous variable

```
demo.col <- ggplot(demo, aes(carcat, income))

demo.col +
  geom_col()
```

Reorder Plot

you can order the bar graph using the fct_reorder function from Forcats

```
1 demo.col <- ggplot(demo, aes(fct_reorder(carcat, income), income))
2
3 demo.col +
4   geom_col()
```

Add Additional Variable

you can use the fill argument in aes to map an additional variable onto individual bars

```
demo.col +
  geom_col(aes(fill = ed))
```

Theme: Scale Color Palette

The scale_fill_brewer function will allow you to add pre-built palettes to your plot

- Color Brewer

```
demo.col +
  geom_col(aes(fill = ed)) +
  scale_fill_brewer(palette = "Pastel1")
```

Theme: Scale Numeric Axis

the scales element allows you to fine-tune and adjust the mapping/scale of labels, breaks, and legends

- the **scale_y_continuous** or **scale_x_continuous** along with label_number elements allows you to adjust a numeric axis

```
demo.col +
  geom_col(aes(fill = ed)) +
  scale_fill_brewer(palette = "Pastel1") +
  scale_y_continuous(labels = scales::label_number_si())
```

RMarkdown

RMarkdown is a user-friendly markup language that combines text, code, and output in a single document, making it easy to create dynamic reports and documents with integrated data analysis using R.

- RMarkdown Gallery
- Quarto Gallery

Why Create a Markdown Document

- reproducibility
- collaboration
- seamless integration of code and narrative.
- HTML, PDF, or Word Export

Structure: YAML Front Matter

The YAML (YAML Ain’t Markup Language) front matter is an optional section at the beginning of the document, enclosed by three dashes (—).

- It is used to set document metadata and configuration options.
- Common YAML parameters include *title*, *author*, *date*, and *output format*.

Structure: YAML Front Matter

```
---

title: "A Pandoc Markdown Article Starter and Template"
thanks: "Replication files are available on the author's Github account (ht
author:
  name: Steven V. Miller
  affiliation: Clemson University
  abstract: "This document provides an introduction to R Markdown, argues f
  keywords: "pandoc, r markdown, knitr"
date: "November 13, 2023"
output: html_document

---
```

Structure: YAML Front Matter

A Pandoc Markdown Article Starter and Template *

Steven V. Miller *Clemson University*

This document provides an introduction to R Markdown, argues for its benefits, and presents a sample manuscript template intended for an academic audience. I include basic syntax to R Markdown and a minimal working example of how the analysis itself can be conducted within R with the `knitr` package.

Keywords: pandoc, r markdown, knitr

Structure: YAML Front Matter: Themes

- RMarkdown
- Quarto HTML
- Quarto Revealjs for Presentations
- Quarto Dashboard

Text (Markdown) Chunks

The main body of the document consists of text written in Markdown, a lightweight markup language. Markdown allows you to format text using simple syntax, including headers, lists, emphasis, and more.

Code Chunks

- Code chunks allow you to integrate executable code into your document.
- Code chunks start and end with three backticks (```). You can specify the language (e.g., r for R) after the opening backticks.
- R code within code chunks is executed when the document is rendered.

Render Options

- Include: allows you to include or not include the chunk code in the final product when rendered.
- Eval: run or not run a code chunk when the document is rendered
- Echo: show the code chunk when the document is rendered
- Message: include messages when the document is rendered