Creating Graphics with `ggplot2`

R for Reproducible Scientific Analysis
Carpentries workshop 2023-04-24

Chi Zhang
Oslo Center for Biostatistics and Epidemiology, UiO

Outline

13:00-14:00 **Introduction** (scatter plot, grouped lines, colors)

14:00-14:15 Break

14:15-15:15 **Customization for your plots** (facets, text, statistics)

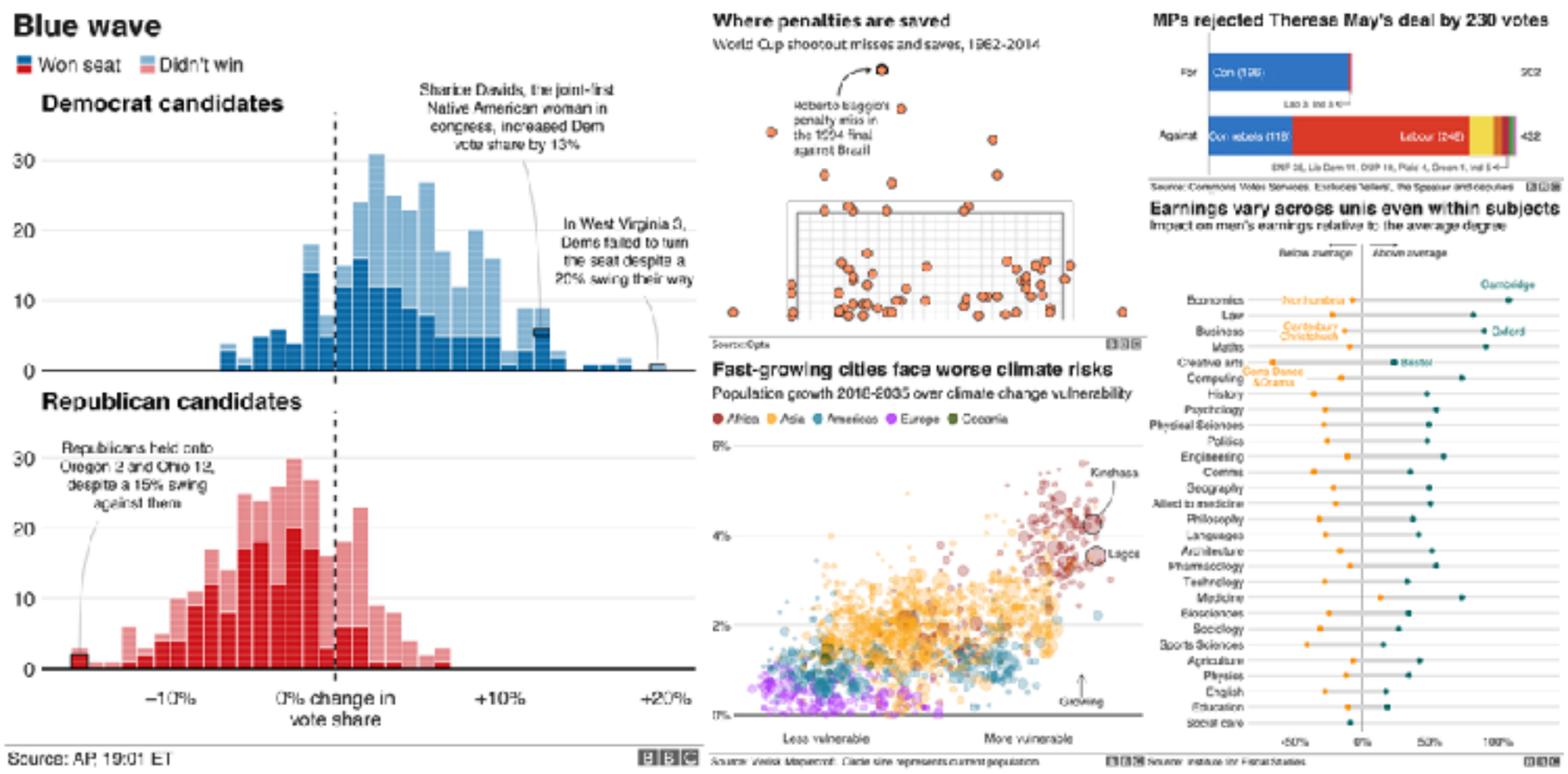
15:15-15:30 **Save data and plots**

15:30 wrap up

Introduction

https://ggplot2-book.org

Data visualisation is a very efficient way to **explore and present** your data
Example: BBC style graphics (https://bbc.github.io/rcookbook/#how_to_create_bbc_style_graphics)



`ggplot2`: an R package for producing graphics, based on the *Grammar of Graphics*

Your first plot

We use the "**gapminder**" data: create a scatter plot between two variables: GDP per capita, and life expectancy.

`gdpPercap, lifeExp`

Open a new R script, load package ggplot2;
Type in and run the following code

```
# install.packages("ggplot2")
library(ggplot2)
ggplot(data = gapminder,
       mapping = aes(x = gdpPercap,
                     y = lifeExp)) + geom_point()
```

Components of a plot Data + mapping

Mapping components

Layer	The geometric elements you see (points, lines); Statistical transformations
Scales	Color, shape, size, legends, axes
Coord (coordinate system)	Usually Cartesian; can be polar coordinates or maps
Facet	Display subsets of data in small multiples
Theme	Background color, font size etc

Components of a plot

Data: gapminder

Mapping:
x axis: gdpPercap;
y axis: lifeExp

Layer:
geom_point(), shows scatter plot

Scales: default
Coord: default
Facet: not applied
Theme: default

```
ggplot(data = gapminder,
       mapping = aes(
           x = gdpPercap,
           y = lifeExp)) +
  geom_point()
```

+: adds layers on top of each other

Tips:
assign a variable to each component,
it helps with debugging

Practice

Modify the code so that it visualizes: **life expectancy** over **time** (year).

`year lifeExp`

What are your x and y axes?

```
ggplot(data = gapminder,
       mapping = aes(x = year,
                     y = lifeExp)) + geom_point()
```

Enrich your plot

The "**gapminder**" data has 6 variables:

```
gdpPercap, lifeExp, country, continent, year, pop
```

A few ways to visualize more than 2 variables with x-y axes:

Colors, shapes, facets, …

Now we try to plot 3 variables into the same plot:

```
lifeExp, continent, year
```

Add color

We wish to visualize the life expectancy for all countries over years, and distinguish countries based on which continent they are in.

`lifeExp, continent, year`

We start with scatter plot, geom_point()

```
plt <- ggplot(data = gapminder,
              mapping = aes(
                  x = gdpPercap,
                  y = lifeExp,
                  color = continent))

plt + geom_point()
```

Add color

There are many color palettes available: to use them, you need to install some R packages. For example, the package **RColorBrewer**

```
# install.packages(“RColorBrewer”)
library(RColorBrewer)

plt <- ggplot(data = gapminder,
       mapping = aes(
           x = gdpPercap,
           y = lifeExp,
           color = continent))
plt <- plt + geom_point()
plt + scale_color_brewer(palette = “Dark2")
```

Visualize trend with line

What if we want to connect the dots with line, to show the trend?

`lifeExp, continent, year`

Replace geom_point() with geom_line()

```
plt <- ggplot(data = gapminder,
              mapping = aes(
                  x = gdpPercap,
                  y = lifeExp,
                  color = continent))

plt + geom_line()
```

Grouped line

What if we want to connect the dots with line, to show the trend?
Use **group** aesthetics

`lifeExp, continent, year, country`

```
plt <- ggplot(data = gapminder,
        mapping = aes(
            x = gdpPercap,
            y = lifeExp,
            group = country, # add group
            color = continent))

plt + geom_line()
```

Grouped line with scatters

It is also possible to add scatters on top of the lines

`lifeExp, continent, year, country`

```
plt <- ggplot(data = gapminder,
        mapping = aes(
            x = gdpPercap,
            y = lifeExp,
            group = country,
            color = continent))

plt + geom_line() + geom_point() # add layer
```

15 minutes break

Customization: facets

We have seen line graphs for countries over time, but it is not easy to spot each country individually: readability is limited.

Facets can help with this, by creating multiple smaller graphs.

Now we focus on countries in the Americas, and try to plot one line per country.

Customization: facets

We focus on countries in the Americas, and try to plot one line per country.

Make a smaller dataset by filtering on the condition: continent == “Americas”.

```
americas <- gapminder[gapminder$continent == “Americas”, ]

plt <- ggplot(data = americas,
       mapping = aes(x = gdpPercap,y = lifeExp))


plt <- plt + geom_line()
plt <- plt + facet_wrap( ~ country)
plt

# to make x axis label more readable
plt + theme(axis.text.x = element_text(angle=45))
```

Customization: texts

We can change the **titles** of the figure, axes labels, and color legends

```
p <- ggplot(data = americas,
    mapping = aes(x = year, y = lifeExp, color=continent))

p <- p + geom_line()
p <- p + facet_wrap( ~ country)
p <- p + theme(axis.text.x = element_text(angle = 90))

p <- p + labs(
        x = "Year",                 # x axis title
        y = "Life expectancy",      # y axis title
        title = "Figure 1",         # main title of figure
        color = "Continent"         # title of legend
)
```

Transformation and statistics

It is possible to use a different scale (e.g. log scale) in your plot, without pre-processing your data.

First, we go back to the scatterplot between gdpPercap and lifeExp

```
p <- ggplot(data = gapminder,
    mapping = aes(x = gdpPercap, y = lifeExp))

p <- p + geom_point()
p
```

Transformation and statistics

We can see some large values (outliers) in the graph.

We do a log transformation (base 10) - this reduces the distance between very large values and the rest.

Essentially plotting `log_10(gdpPercap)` against `lifeExp`

```
p <- ggplot(data = gapminder,
    mapping = aes(x = gdpPercap, y = lifeExp))

p <- p + geom_point()
p <- p + scale_x_log10()
p
```

Transformation and statistics

It is also possible to fit a statistical model, and add the fitted line here.

We fit the following linear model (directly within the ggplot syntax)

`lifeExp = b * log_10(gdpPercap)` (b is regression coefficient)

```
p <- ggplot(data = gapminder,
    mapping = aes(x = gdpPercap, y = lifeExp))

p <- p + geom_point()
p <- p + scale_x_log10()
p <- p + geom_smooth(method = ‘lm’, linewidth = 1.5)
p
```

Save your plots and data

There are multiple ways to save your plot.

Option 1: take a screen shot (personal favorite)

Option 2: click on “Export” button in the Plots panel.

Option 3: this is recommended if you need to automate the process by running the script.

```
ggsave(filename = “folder/plot_lifeExp.png”,
       plot = p,
       width = 12, height = 10)
```

Save your plots and data

To save your data (e.g. a processed data.frame) into a csv format data file, you can use

write.table() and write.csv()

```
au_subset <- gapminder[gapminder$country == “Australia”, ]

write.table(au_subset,
            file = “folder/australia.csv”,
            sep = “,”)
# or
write.csv(au_subset,
            file = “folder/australia.csv”,
            sep = “,”)
```

Summary

`ggplot` takes some time to learn, and we all learn by making mistakes!

There are many blogs and tutorials on how to make specific types of plots.

Read this book: https://ggplot2-book.org

Website and user guide by tidyverse team https://ggplot2.tidyverse.org

https://swcarpentry.github.io/r-novice-gapminder/08-plot-ggplot2/index.html

https://swcarpentry.github.io/r-novice-gapminder/11-writing-data/index.html