R Codes - graphical descriptive statistics

Subham Kailthya

2023-12-22

Load libraries

```
library("knitr")          # knitting doc
library("dplyr")          # data wrangling
library("ggplot2")        # plotting data
library("janitor")        # manage tables
library("sf")             # spatial data
library("rnaturalearth")  # spatial database
```

Import Freedom House Data

Here we import FH data from `/path/link/<data-file>`, drop the variable `Region_Code`, convert `Status`, `Region_Name` and `is_ldc` to factor data type, and keep observations corresponding to the year 2020.

`select()`, `mutate()` and `filter()` are functions from the `ddplyr` package. To know more about a function, say `mutate()`, type `help("mutate")` or `?mutate()` in `R` console.

```
path <- "https://raw.githubusercontent.com/subhamk/"
link <- "ec122_share/main/data/"

freedom <- readr::read_csv(paste0(path, link, "freedom.csv")) # import csv

freedom_2020 <- freedom %>%
  dplyr::select(-Region_Code) %>%           # drop Region_Code
  mutate(Status = factor(Status),
         Region_Name = factor(Region_Name),
         is_ldc = factor(is_ldc)) %>%       # convert to factor
  filter(year == 2020)                      # keep data for the year 2020

head(freedom_2020)                          # print first few lines of the data
```

```
## # A tibble: 6 x 8
##    ...1 country              year    CL    PR Status Region_Name is_ldc
##   <dbl> <chr>               <dbl> <dbl> <dbl> <fct>  <fct>       <fct>
## 1    26 Afghanistan          2020     6     5 NF     Asia        1
## 2    52 Albania              2020     3     3 PF     Europe      0
## 3    78 Algeria              2020     5     6 NF     Africa      0
## 4   104 Andorra              2020     1     1 F      Europe      0
## 5   130 Angola               2020     5     6 NF     Africa      1
## 6   156 Antigua and Barbuda  2020     2     2 F      Americas    0
```

The function `glimpse()` provides a quick overview of the dataset.

```
glimpse(freedom_2020)
```

```
## Rows: 193
## Columns: 8
## $ ...1         <dbl> 26, 52, 78, 104, 130, 156, 182, 208, 234, 260, 286, 312, 3~
## $ country      <chr> "Afghanistan", "Albania", "Algeria", "Andorra", "Angola", ~
## $ year         <dbl> 2020, 2020, 2020, 2020, 2020, 2020, 2020, 2020, 2020, 2020~
## $ CL           <dbl> 6, 3, 5, 1, 5, 2, 2, 4, 1, 1, 6, 1, 6, 5, 1, 6, 1, 1, 2, 4~
## $ PR           <dbl> 5, 3, 6, 1, 6, 2, 2, 4, 1, 1, 7, 1, 7, 5, 1, 7, 1, 2, 4, 2~
## $ Status       <fct> NF, PF, NF, F, NF, F, F, PF, F, F, NF, F, NF, PF, F, NF, F~
## $ Region_Name <fct> Asia, Europe, Africa, Europe, Africa, Americas, Americas, ~
## $ is_ldc       <fct> 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1~
```

Cross Tables

This tabulates the number of observations per cell where each cell is a combination of `Stutus` and `is_ldc`.

```
freedom_2020 %>%
  tabyl(is_ldc, Status) %>%              # create table
  adorn_totals(c("row", "col")) %>%      # add row and col totals
  kable()                                # print table
```

is_ldc	F	NF	PF	Total
0	76	35	36	147
1	5	19	22	46
Total	81	54	58	193

Bar Chart

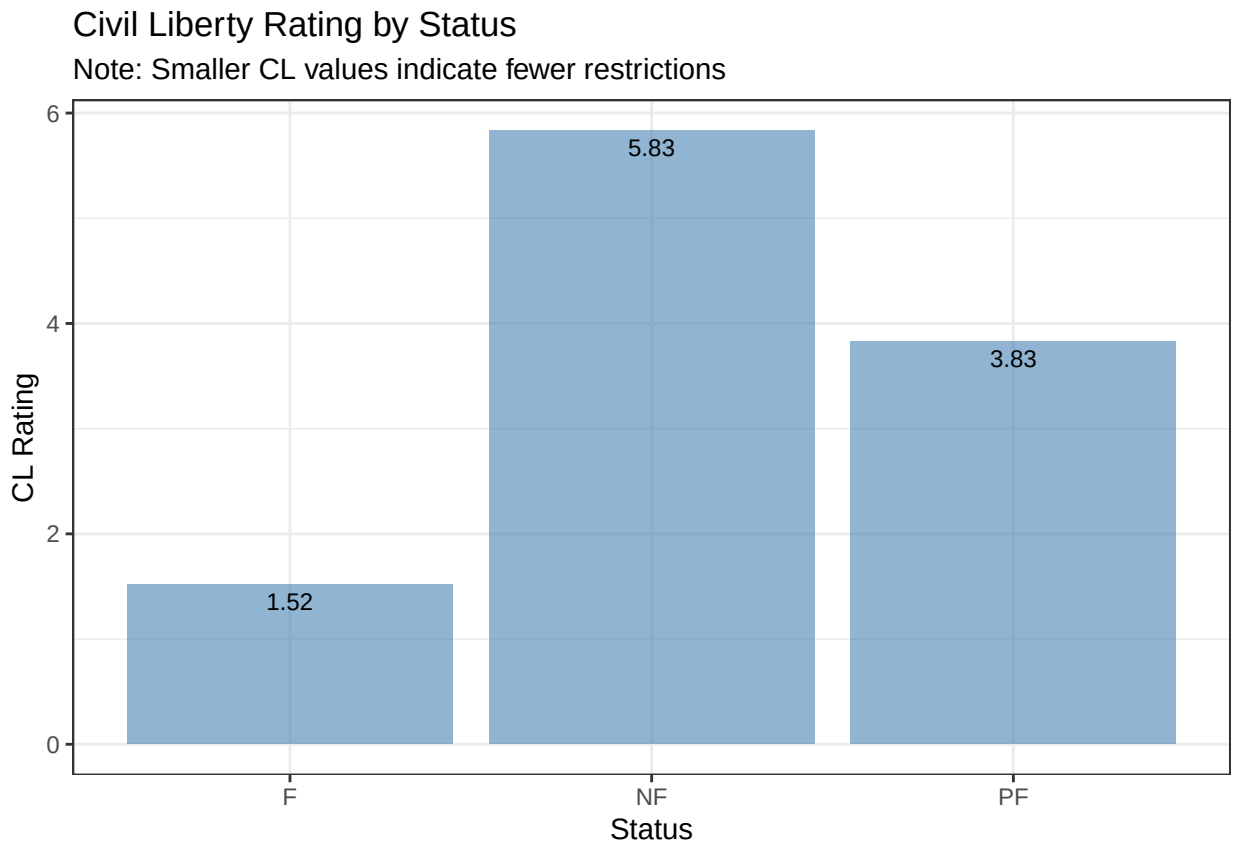
The following code chunk plots a bar chart of average `CL` score by group `Status`. `summarise` creates a new data (`tibble` in tidy language) which contains the mean of `CL` and we name this `fh_ranking`.

```
fh_ranking <- freedom_2020 %>%           # use 2020 FH data
  group_by(Status) %>%                   # group data by Status
  summarise(mean_cl = mean(CL, na.rm = TRUE))   # calculate the mean by group var.
fh_ranking
```

```
## # A tibble: 3 x 2
##   Status mean_cl
##   <fct>    <dbl>
## 1 F         1.52
## 2 NF        5.83
## 3 PF        3.83
```

This reproduces the bar chart in the slides. We use `ggplot2` library to produce charts. Hadley Wickham's ggplot2 book is a detailed resource.

```r
ggplot(data = fh_ranking, aes(x = Status, y = mean_cl)) +
  geom_bar(stat = 'identity',
           fill = 'steelblue',
           alpha = 0.6) +
  geom_text(
    aes(label = round(mean_cl, 2)),
    colour = "black",
    size = 3,
    vjust = 1.5,
    position = position_dodge(.9)
  ) +
  labs(
    title = "Civil Liberty Rating by Status",
    subtitle = "Note: Smaller CL values indicate fewer restrictions",
    x = "Status",
    y = "CL Rating"
  ) +
  theme_bw()
```



Bar Chart with Multiple Groups

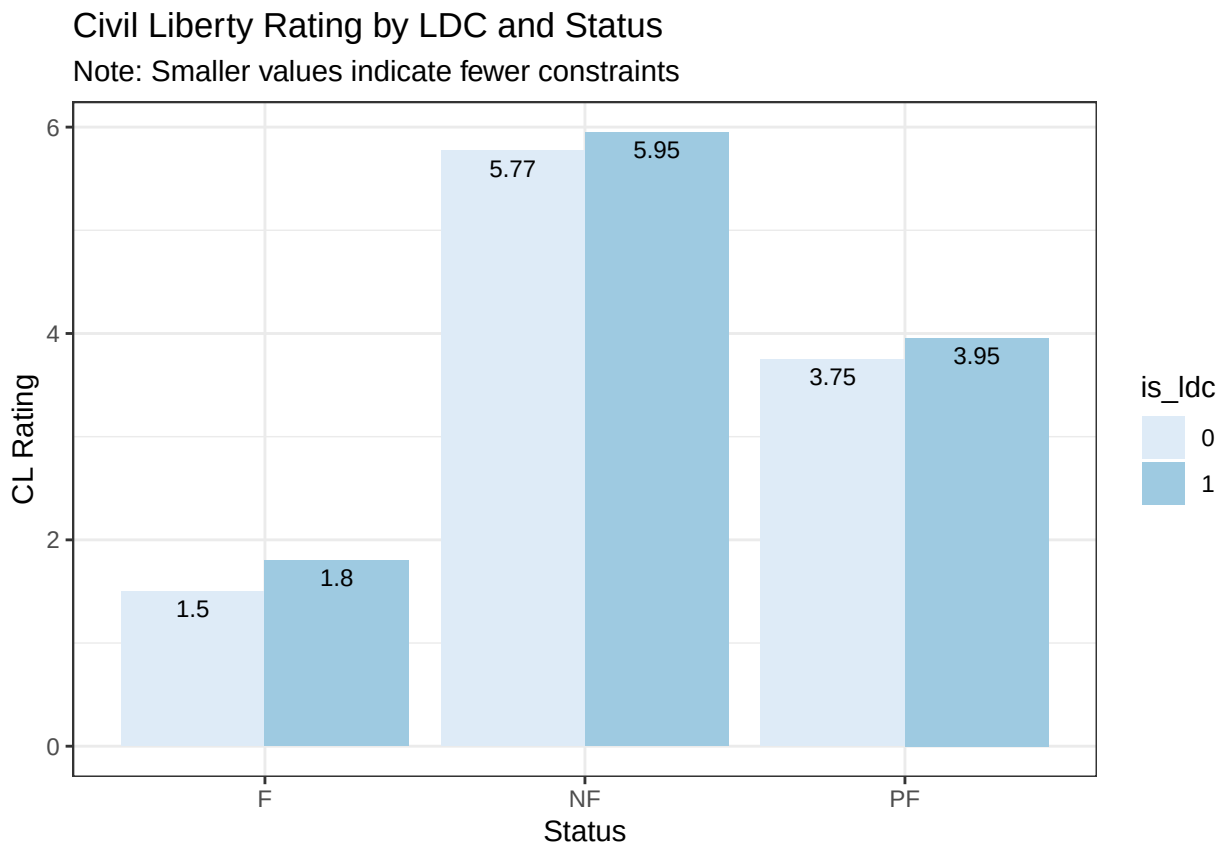
Here, we calculate the mean by both `Status` and `is_ldc` and present data for each of the groups side by side. We construct a new variable `label_y` – the cumulative sum of `mean_cl` – which helps in placing the value labels in the stacked bar plot (next section) in the right place.

```r
fh_ranking_m <- freedom_2020 %>%
  group_by(Status, is_ldc) %>%          # group data
  summarise(mean_cl = mean(CL, na.rm = TRUE)) %>%   # get averages
  mutate(label_y = cumsum(mean_cl))     # cumulative sum
fh_ranking_m
```

```
## # A tibble: 6 x 4
## # Groups:   Status [3]
##   Status is_ldc mean_cl label_y
##   <fct>  <fct>    <dbl>   <dbl>
## 1 F      0         1.5     1.5
## 2 F      1         1.8     3.3
## 3 NF     0         5.77    5.77
## 4 NF     1         5.95   11.7
## 5 PF     0         3.75    3.75
## 6 PF     1         3.95    7.70
```

This reproduces multiple bar plot in the slides.

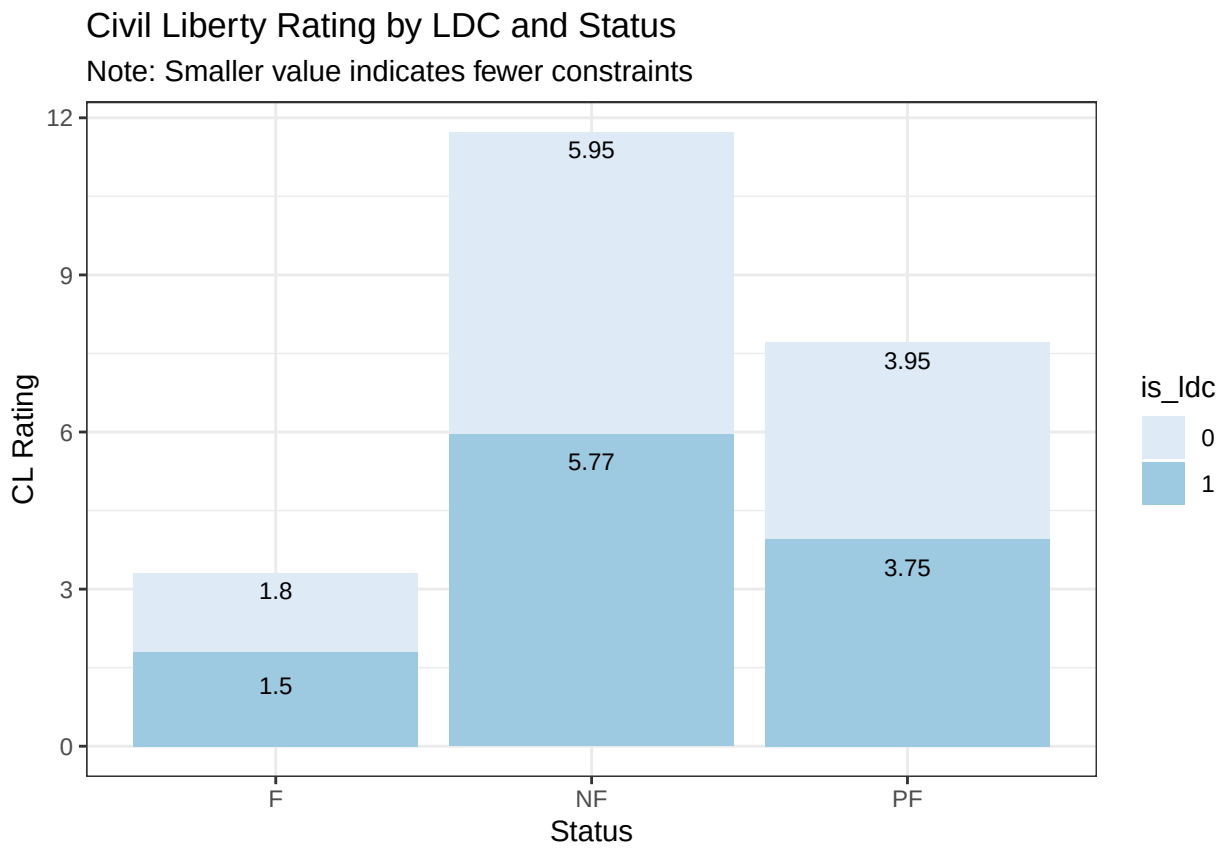
```r
ggplot(data = fh_ranking_m, aes(x = Status, y = mean_cl, fill = is_ldc)) +
  geom_bar(stat = 'identity', position = position_dodge()) +
  scale_fill_brewer(palette = "Blues") +
  geom_text(
    aes(label = round(mean_cl, 2)),
    colour = "black",
    size = 3,
    vjust = 1.5,
    position = position_dodge(.9)
  ) +
  labs(
    title = "Civil Liberty Rating by LDC and Status",
    subtitle = "Note: Smaller values indicate fewer constraints",
    x = "Status",
    y = "CL Rating"
  ) +
  theme_bw()
```



Stacked Bar Chart

We stack the bars instead of presenting them side by side. This is particularly useful when there are several categories for a variable of interest.

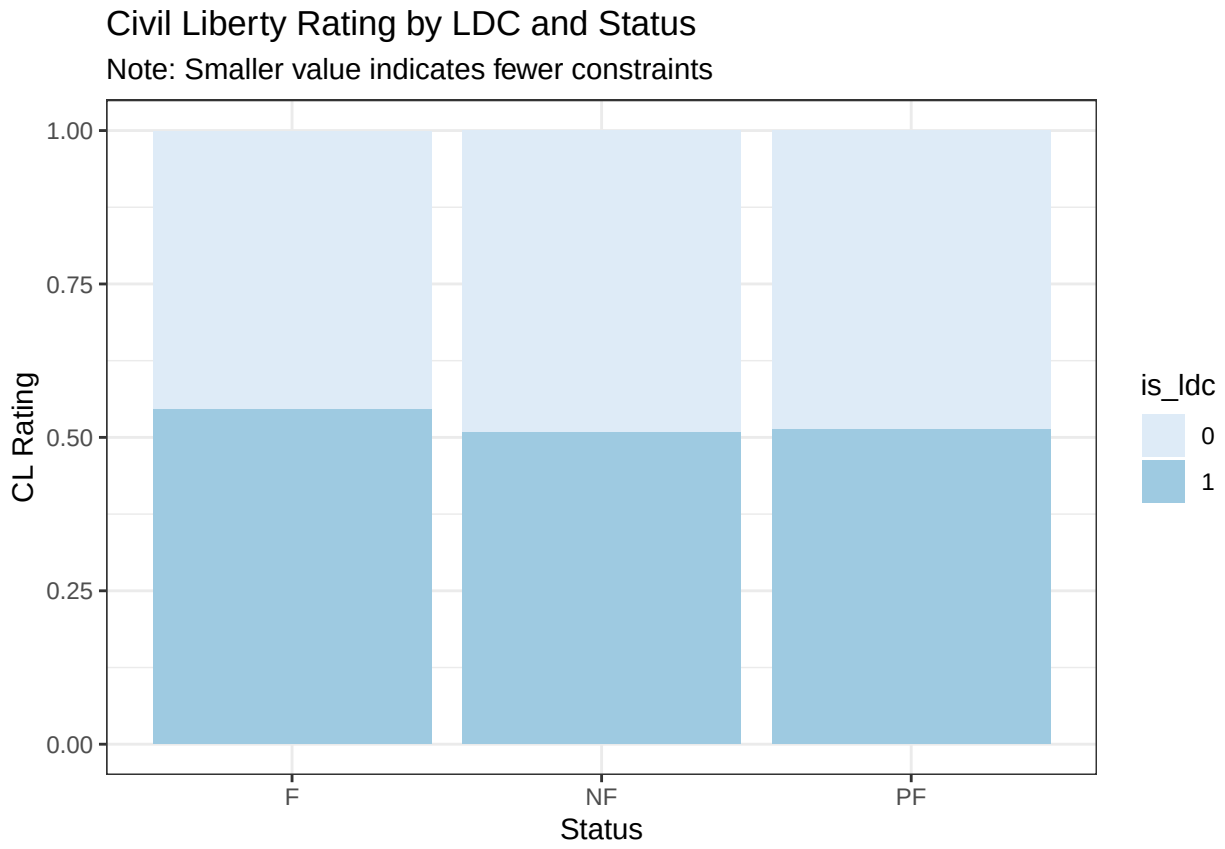
```
ggplot(data = fh_ranking_m, aes(x = Status, y = mean_cl, fill = is_ldc)) +
  geom_bar(stat = 'identity') +
  scale_fill_brewer(palette = "Blues") +
  geom_text(
    aes(y = label_y, label = round(mean_cl, 2)),
    colour = "black",
    size = 3,
    vjust = 1.5
  ) +
  labs(
    title = "Civil Liberty Rating by LDC and Status",
    subtitle = "Note: Smaller value indicates fewer constraints",
    x = "Status",
    y = "CL Rating"
  ) +
  theme_bw()
```



Stacked Bar Chart - Proportional

We could stack proportions instead.

```
ggplot(data = fh_ranking_m, aes(x = Status, y = mean_cl, fill = is_ldc)) +
  geom_col(position = "fill") +
  scale_fill_brewer(palette = "Blues") +
  labs(
    title = "Civil Liberty Rating by LDC and Status",
    subtitle = "Note: Smaller value indicates fewer constraints",
    x = "Status",
    y = "CL Rating"
  ) +
  theme_bw()
```

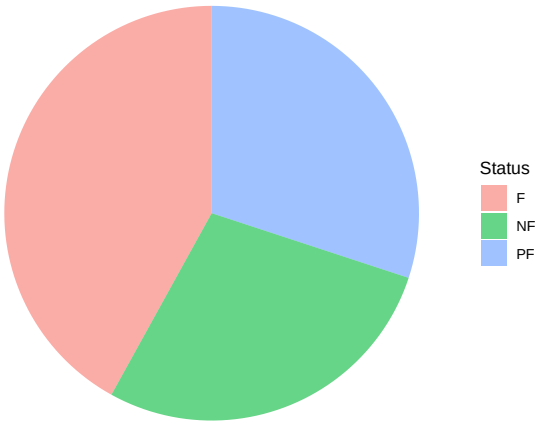


Pie Chart

```
dat <- freedom_2020 %>%
  janitor::tabyl(Status) %>%    # creates a table
  arrange(desc(n))              # descending order
dat
```

```
##  Status  n   percent
##       F 81 0.4196891
##      PF 58 0.3005181
##      NF 54 0.2797927
```

```
ggplot(dat, aes(x = "", y = percent, fill = Status)) +
  geom_bar(stat = "identity",
           width = 1,
           alpha = 0.6) +
  coord_polar("y", start = 0) +
  theme_void()
```



Merging data

First, we load PWT 10.1 and name this data object `pwt`. We then create a new data object `pwt_1` which subsets `pwt` to keep only GDP data for the year 2010.

```
pwt <- readr::read_csv(paste0(path, link, "pwt1001.csv"))
pwt_1 <- pwt %>%
  filter(year == 2010) %>%
  dplyr::select(countrycode, country, rgdpo)

head(pwt_1)
```

```
## # A tibble: 6 x 3
##   countrycode country                rgdpo
##   <chr>       <chr>                  <dbl>
## 1 ABW         Aruba                  3933.
## 2 AGO         Angola               159625.
## 3 AIA         Anguilla               382.
## 4 ALB         Albania              31021.
## 5 ARE         United Arab Emirates 618895.
## 6 ARG         Argentina            784810.
```

We then merge FH data with GDP data based on `country` (country names). We drop observations where `rgdpo` are missing (indicated by `NA` in `R`) and then take the log of GDP (`log_rgdpo`).

```
freedom_gdp <- freedom_2020 %>%
  dplyr::left_join(pwt_1, by = join_by(country)) %>%
  tidyr::drop_na(rgdpo) %>%
  mutate(log_rgdpo = log(rgdpo))
head(freedom_gdp)
```

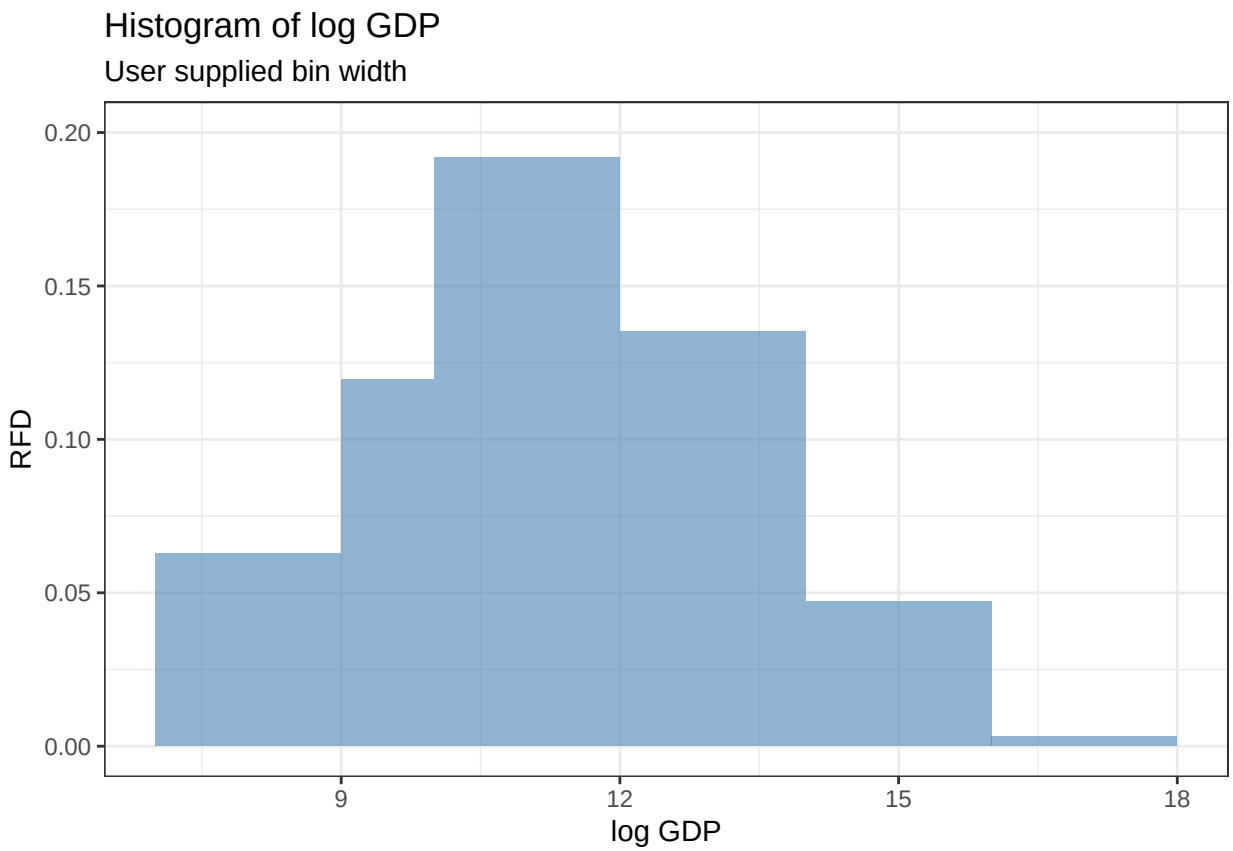
```
## # A tibble: 6 x 11
##    ...1 country    year    CL    PR Status Region_Name is_ldc countrycode  rgdpo
##   <dbl> <chr>     <dbl> <dbl> <dbl> <fct>  <fct>       <fct>  <chr>        <dbl>
```

```
## 1    52 Albania    2020     3     3 PF     Europe      0      ALB         3.10e4
## 2    78 Algeria    2020     5     6 NF     Africa      0      DZA         5.33e5
## 3   130 Angola     2020     5     6 NF     Africa      1      AGO         1.60e5
## 4   156 Antigua ~  2020     2     2 F      Americas    0      ATG         1.85e3
## 5   182 Argentina  2020     2     2 F      Americas    0      ARG         7.85e5
## 6   208 Armenia    2020     4     4 PF     Asia        0      ARM         2.56e4
## # i 1 more variable: log_rgdpo <dbl>
```

Histogram

The code plots the distribution of log GDP for the year 2010 across all the countries in our sample. We specify the breaks in `br`.

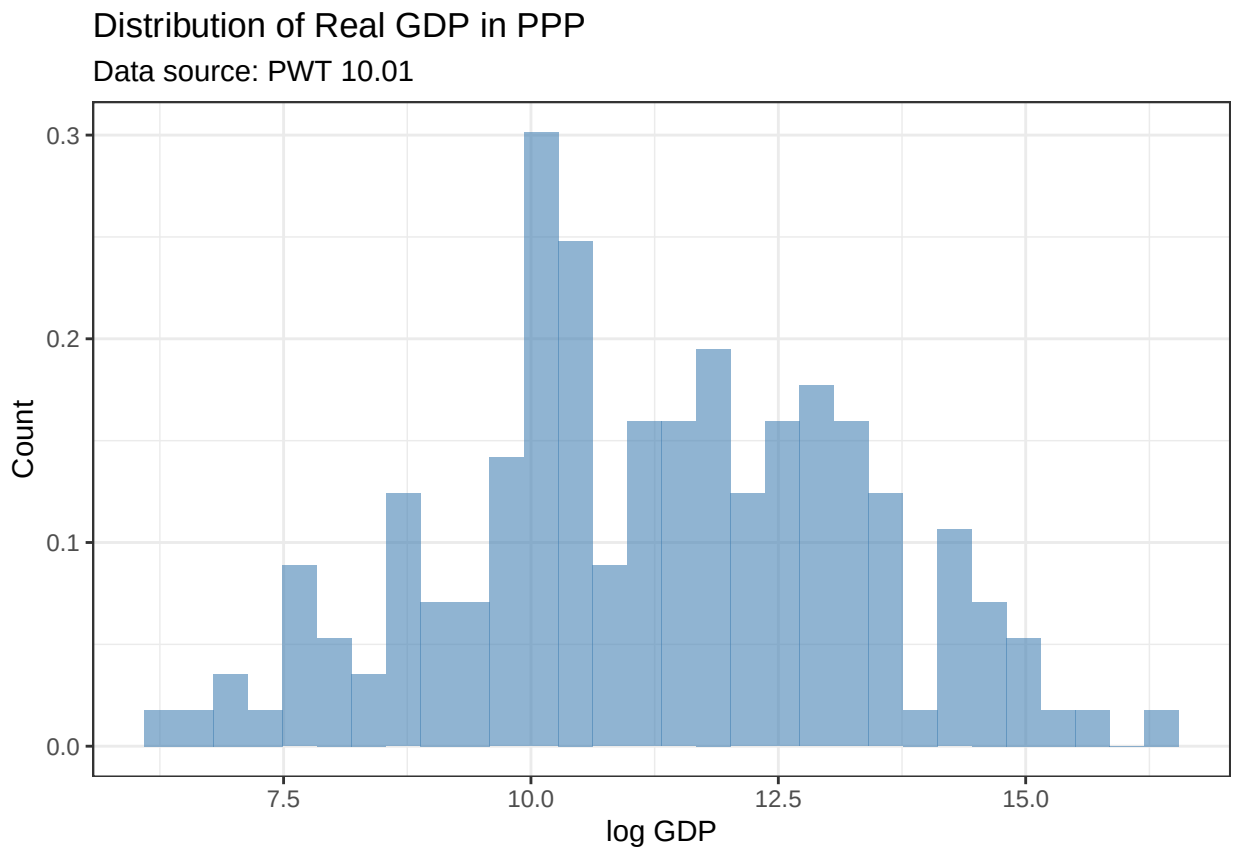
```
br <- c(7, 9, 10, 12, 14, 16, 18)        # specify breaks for histogram

ggplot(data = freedom_gdp, aes(x = log_rgdpo, stat(density))) +
  geom_histogram(breaks = br,
                 fill = 'steelblue',
                 alpha = 0.6) +
  ylim(0, 0.2) +
  labs(
    title = 'Histogram of log GDP',
    subtitle = 'User supplied bin width',
    x = 'log GDP',
    y = 'RFD'
  ) +
  theme_bw()
```



Histogram - Automatic bin selection

Here, we plot the distribution of `log_rgdpo` but allow `R` to automatically select bin size. This results in a more elegant plot.

```
ggplot(data = freedom_gdp, aes(x = log_rgdpo, y = stat(density))) +
  geom_histogram(fill = 'steelblue', alpha = 0.6) +
  labs(
    title = 'Distribution of Real GDP in PPP',
    subtitle = 'Data source: PWT 10.01',
    x = 'log GDP',
    y = 'Count'
  ) +
  theme_bw()
```



Time Series Plot

We might be interested in how a variable evolves over time. Suppose we are interested in how the UK economy has grown over time.

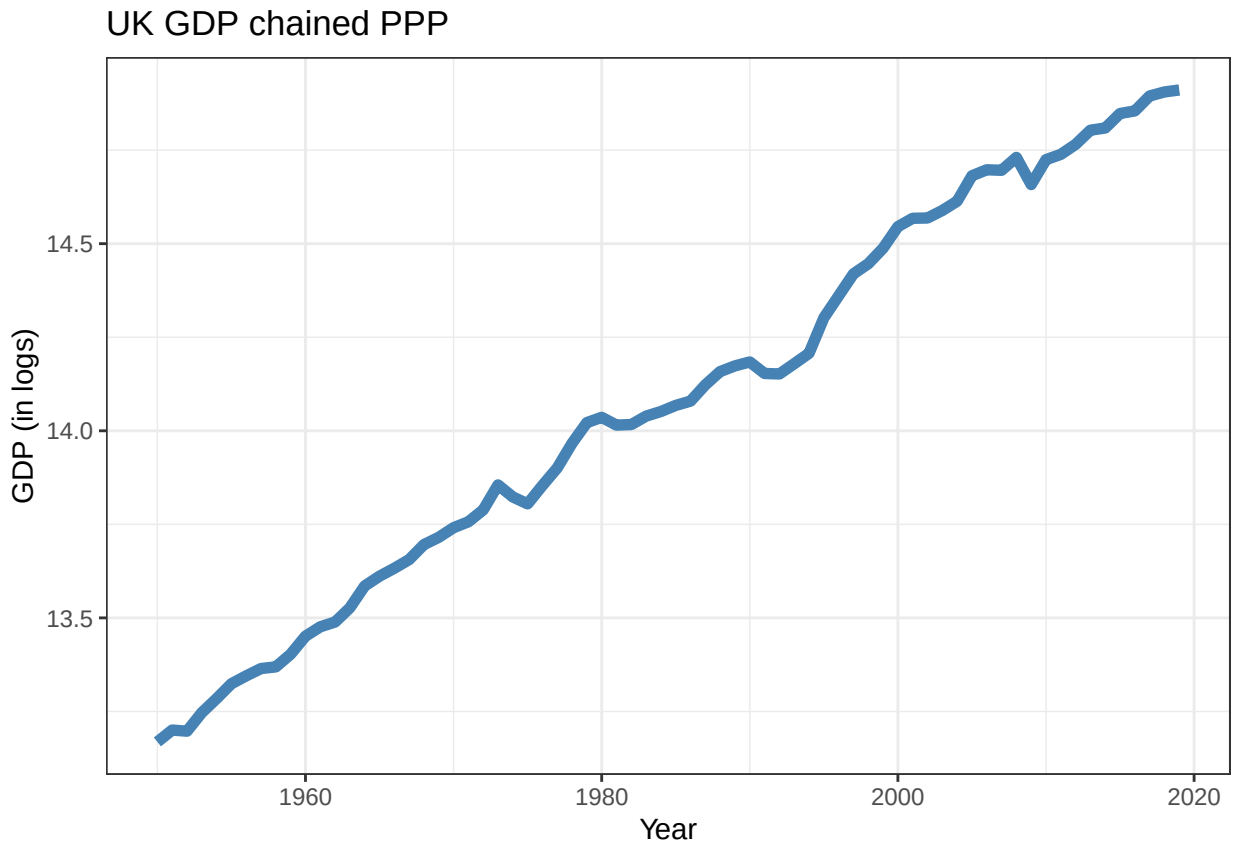
To do this, we start with `pwt` data object, subset observations corresponding to the UK, select the variables `year` and `rgdpo` that are of interest. Note that `rgdpo` is in chained PPP. We then create a new variable `log_gdp` which log transforms real UK GDP.

```
uk_gdp <- pwt %>%
  filter(countrycode == "GBR") %>%
  dplyr::select(year, rgdpo) %>%
  mutate(log_gdp = log(rgdpo))
head(uk_gdp)
```

```
## # A tibble: 6 x 3
##    year   rgdpo log_gdp
##   <dbl>   <dbl>   <dbl>
## 1  1950 523645.    13.2
## 2  1951 540520.    13.2
## 3  1952 538869.    13.2
## 4  1953 566293.    13.2
## 5  1954 587977.    13.3
## 6  1955 611782.    13.3
```

We then plot `log_gdp` against `year` to show how the UK economy has grown over time.

```
ggplot(data = uk_gdp, aes(x = year, y = log_gdp)) +
  geom_line(linewidth = 2, col = 'steelblue') +
  labs(title = 'UK GDP chained PPP',
       x = 'Year',
       y = "GDP (in logs)") +
  theme_bw()
```



Geo-spatial data

R has advanced features to deal with spatial data. We obtain the map of the world from naturalearth, merge `fh_ranking_2020` to get the civil liberty scores by country and plot it on a map.

```
world <- ne_countries(scale = "medium", returnclass = "sf")

fh_ranking_2020 <- freedom_2020 %>%
  mutate(
    country = replace(
      country,
      country == "United Kingdom of Great Britain and Northern Ireland",
      "United Kingdom"
    )
  ) %>%
```

```
  dplyr::select(CL, PR, country)

world <- world %>%
  left_join(fh_ranking_2020, by = join_by(geounit == country))
```

```
ggplot(data = world) +
  geom_sf(aes(fill = factor(CL))) +
  labs(title = 'Civil Liberty Ranking in 2020',
       subtitle = 'Smaller score indicates fewer constraints') +
  theme_bw()
```