COMP2501 Assignment 1

Sibo Ding

Spring 2023

Requirements

Submission deadline: Feb 28th, 2023 at 23:59.

Mark of assignment 1: 33/33.

For the following questions, please:

1. Replace all [Input here] places with your information or your answer (for multiple choice).
2. Complete the code block by adding your own code to fulfill the requirements in each question. Please use the existing code block and do not add your own code block.

Please make sure your Rmd file is a valid Markdown document and can be successfully knitted.

For assignment submission, please knit your final Rmd file into a Word document, and submit both your **Rmd** file and the knitted **Microsoft Word** document file to Moodle. You get 0 score if 1) the Rmd file you submitted cannot be knitted, and 2) you have not submitted a Word document. For each visualization question, please make sure that the generated plot is shown in-place with the question and after the code block.

Name and UID

Name: Sibo Ding

UID: 3035637204

Environmental setup

You need to have the `dslabs`, `dplyr`, and `ggplot2` packages installed. If not yet, please run `install.packages(c("dslabs", "dplyr", "ggplot2"))` in your R environment. If you have installed the `tidyverse` package, `dplyr` is installed by default.

```r
# Load the packages and dataset.
library(dslabs)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)
data("murders")
```

Example question. Print the first 2 records of the murders dataset.

```r
head(murders, 2)

##     state abb region population total
## 1 Alabama  AL  South    4779736   135
## 2  Alaska  AK   West     710231    19
```

1. (1 points) Define a matrix `mat` of your own choice, print the entries of both row 2 and columns 2 to 4 simultaneously.

```r
mat <- matrix(1:16, 4, 4)
mat[2, 2:4]

## [1]  6 10 14
```

2. (2 points) Write a function `compute_s_n` that for any given n, computes the $S_n = n * \sqrt{(n+9)} * log_{10}(n)$. Print the S_n with $n = 500$.

```r
compute_s_n <- function(n){
  n * sqrt(n+9) * log10(n)
}

n <- 500
s_n <- compute_s_n(n)
s_n

## [1] 30445.77
```

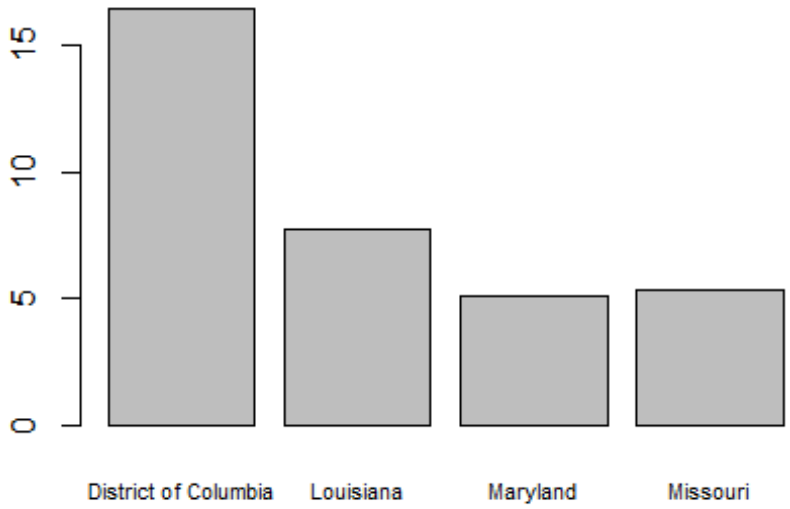
3. (2 points) Compute the murder rate per 100,000 people for each state and store it in an object called murder_rate. Then use logical operators to find which state has a murder rate per 100,000 people higher than 5. Find these states, print their names and murder rate per 100,000 people.

```r
murder_rate <- murders$total / murders$population * 100000
```

```
murder_state_5 <- murders$state[which(murder_rate > 5)]
murder_state_5

## [1] "District of Columbia" "Louisiana"            "Maryland"
## [4] "Missouri"

murder_rate_5 <- murder_rate[which(murder_rate > 5)]
murder_rate_5

## [1] 16.452753  7.742581  5.074866  5.359892
```

4. (2 points) For all states having a murder rate per 100,000 people higher than 5, use the barplot function to create a barplot with the x-axis being the state name, and the y-axis being the murder rate per 100,000 people of each state. (Hint: check some barplot examples at https://r-graph-gallery.com/210-custom-barplot-layout.html)

```
barplot(murder_rate_5, names.arg = murder_state_5, cex.names = 0.7)
```



5. (1 points) Examine the built-in dataset `Orange`. Which of the following is true?

- a. Orange is tidy data: it has one observation for each row.
- b. Orange is not tidy: we need at least one column with a character vector.
- c. Orange is not tidy: it is a matrix instead of a data frame.

- d. Orange is tidy data: all small datasets are tidy by definition.

Your answer is: [a]

6. (3 points) Base on the `murders` dataset, create a table called `my_states` that contains rows for states satisfying two conditions: 1) it is in either West or South, and 2) the murder rate per 100,000 people is less than 2.0. Use `select` to show only the state name, the region and the murder rate, and use top_n function to find the 3 safest states among them.

```
murders <- murders |> mutate(murder_rate) # Add murder_rate to
dataframe
# murders$murder_rate <- murder_rate

my_states <- murders |>
  filter((region == "West" | region == "South") & murder_rate < 2)
my_states |> select(state, region, murder_rate)
```

```
##           state region murder_rate
## 1      Colorado   West   1.2924531
## 2        Hawaii   West   0.5145920
## 3         Idaho   West   0.7655102
## 4       Montana   West   1.2128379
## 5        Oregon   West   0.9396843
## 6          Utah   West   0.7959810
## 7    Washington   West   1.3829942
## 8 West Virginia  South   1.4571013
## 9       Wyoming   West   0.8871131
```

```
my_states |> top_n(-3) # 3 safest states among them
```

```
## Selecting by murder_rate
```

```
##    state abb region population total murder_rate
## 1 Hawaii  HI   West    1360301     7   0.5145920
## 2  Idaho  ID   West    1567582    12   0.7655102
## 3   Utah  UT   West    2763885    22   0.7959810
```

7. (2 points) By using the murders dataset, compute the average murder rate per 100,000 people in the four regions respectively of the U.S., and sort the results by murder rate in ascending order.

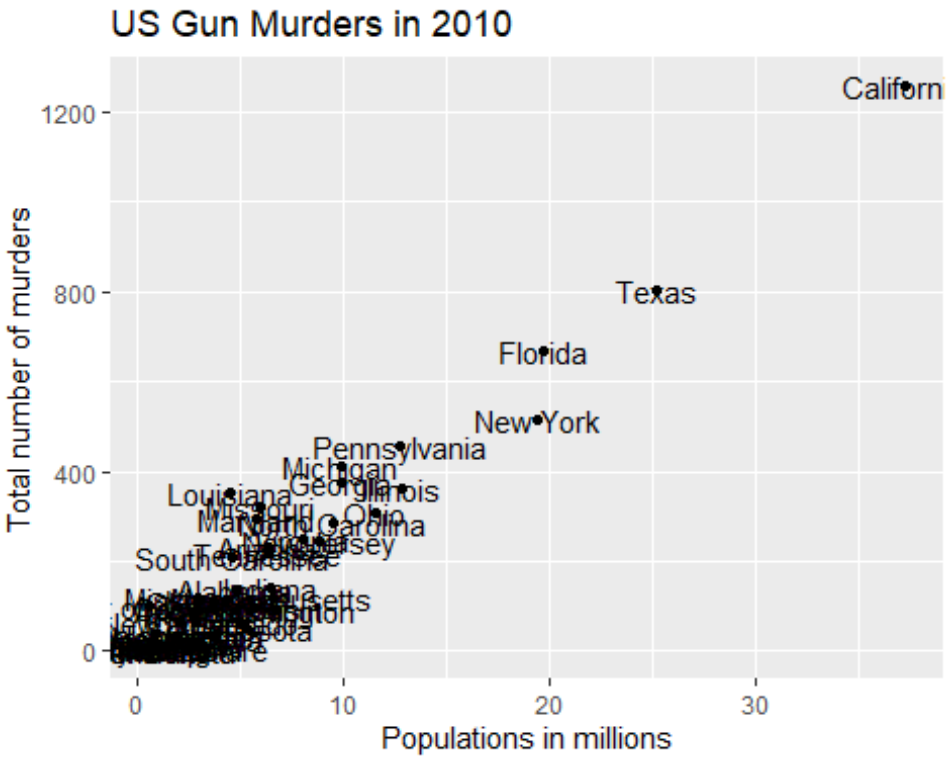
```
# Note: rates cannot be averaged
murders |> group_by(region) |>
  summarize(average_murder_rate = sum(total) / sum(population) *
100000) |>
  arrange(average_murder_rate)
```

```
## # A tibble: 4 × 2
##   region        average_murder_rate
##   <fct>                       <dbl>
```

```
## 1 Northeast                    2.66
## 2 West                         2.66
## 3 North Central                2.73
## 4 South                        3.63
```

8. (3 points) Use the ggplot2 package to create a scatterplot from the murders dataset, where the x-axis is the number of population, the y-axis is the total number of murders, and each point in the scatterplot is labeled with the state name. Please add an appropriate title, and axis labels to the plot.

```
ggplot(data = murders, aes(x = population/10^6, y = total, label =
state)) +
  geom_point() + # points layer
  geom_text() + # labels layer
  xlab("Populations in millions") + # x-axis label layer
  ylab("Total number of murders") + # y-axis label layer
  ggtitle("US Gun Murders in 2010") # title layer
```



9. (17 points) Explore the tidyverse with the COVID-19 dataset (http://www.bio8.cs.hku.hk/comp2501/covid.csv), and answer the following questions.

a. (2 points) Read the CSV formatted dataset. Find out how many observations (rows) and variables (columns) are in the dataset. Print the names of all variables.

```
covid <- read.csv("covid.csv")

nrow(covid) # number of observations
```

```
## [1] 47480

ncol(covid) # number of variables

## [1] 12

names(covid)

##  [1] "dateRep"
##  [2] "day"
##  [3] "month"
##  [4] "year"
##  [5] "cases"
##  [6] "deaths"
##  [7] "countriesAndTerritories"
##  [8] "geoId"
##  [9] "countryterritoryCode"
## [10] "popData2019"
## [11] "continentExp"
## [12] "Cumulative_number_for_14_days_of_COVID.19_cases_per_100000"
```

b. (1 points) List the observation with the largest Cumulative_number_for_14_days_of_COVID.19_cases_per_100000.

```
covid |> top_n(1,
Cumulative_number_for_14_days_of_COVID.19_cases_per_100000)

##      dateRep day month year cases deaths countriesAndTerritories
geoId
## 1 20/08/2020  20     8 2020   175      1                   Aruba
AW
##   countryterritoryCode popData2019 continentExp
## 1                  ABW      106310      America
##   Cumulative_number_for_14_days_of_COVID.19_cases_per_100000
## 1                                                   1058.226
```

c. (2 points) How many unique countriesAndTerritories are in the dataset? How many unique continentExp are in the dataset?

```
covid |> distinct(countriesAndTerritories) |> count()

##     n
## 1 210

covid |> distinct(continentExp) |> count()

##   n
## 1 6
```

d. (3 points) For 1) the whole dataset, 2) different `countriesAndTerritories`, and 3) different `continentExp`, compute both i) the sum of cases, and ii) the sum of deaths. Sort the results by the sum of cases descendingly. Use head() if there are too many rows in the results.

```
# The whole dataset
covid |> summarize(total_cases = sum(cases), total_deaths =
sum(deaths))
```

```
##   total_cases total_deaths
## 1    35848254      1048181
```

```
# countriesAndTerritories
covid |> group_by(countriesAndTerritories) |>
  summarize(total_cases = sum(cases), total_deaths = sum(deaths)) |>
  arrange(desc(total_cases)) |> head()
```

```
## # A tibble: 6 × 3
##   countriesAndTerritories  total_cases total_deaths
##   <chr>                         <int>        <int>
## 1 United_States_of_America     7501612       210909
## 2 India                        6757131       104555
## 3 Brazil                       4969141       147494
## 4 Russia                       1237504        21663
## 5 Colombia                      869808        27017
## 6 Peru                          832929        32914
```

```
# continentExp
covid |> group_by(continentExp) |>
  summarize(total_cases = sum(cases), total_deaths = sum(deaths)) |>
  arrange(desc(total_cases))
```

```
## # A tibble: 6 × 3
##   continentExp total_cases total_deaths
##   <chr>              <int>        <int>
## 1 America         17445678       578079
## 2 Asia            11233759       203583
## 3 Europe           5605508       228689
## 4 Africa           1528213        36828
## 5 Oceania            34400          995
## 6 Other                696            7
```

e. (2 points) Add a new column `date` with the standard date format "YYYY-MM-DD" to the data table according to the `dateRep` column. Be reminded the format of `dateRep` is "DD/MM/YYYY". Please use head() to show the result.

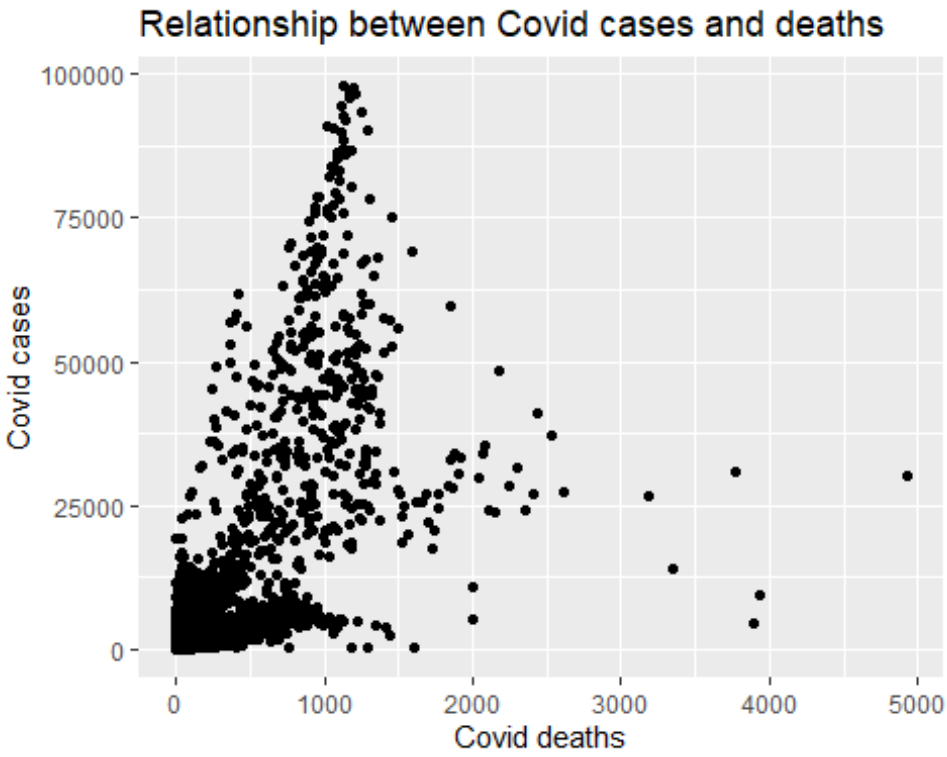
```
covid <- covid |> mutate(date = as.Date(dateRep, "%d/%m/%Y"))
# covid$date <- as.Date(covid$dateRep, "%d/%m/%Y")
head(covid)
```

```
##      dateRep day month year cases deaths countriesAndTerritories
geoId
```

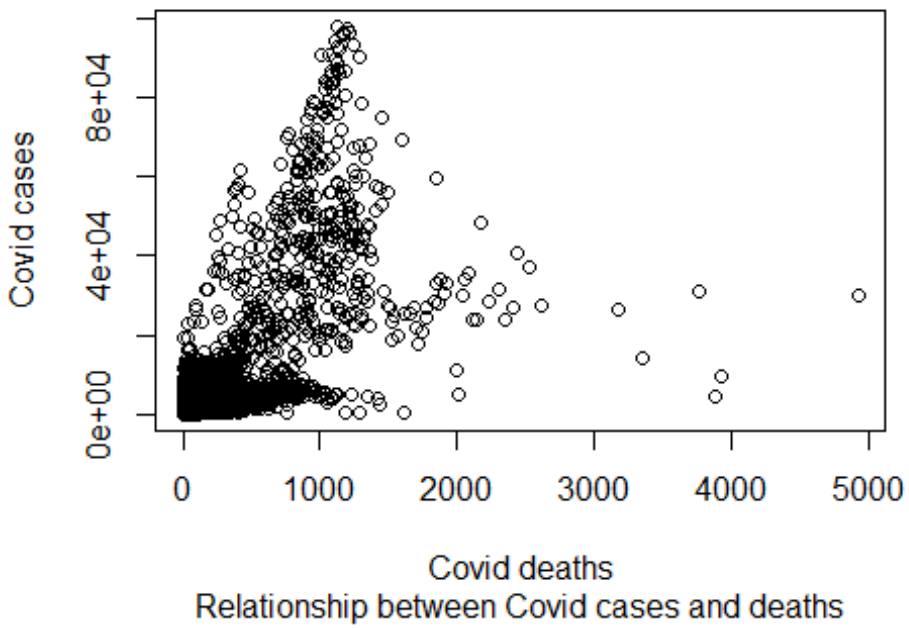
```
## 1 07/10/2020   7    10 2020    62      2             Afghanistan
AF
## 2 06/10/2020   6    10 2020   145      5             Afghanistan
AF
## 3 05/10/2020   5    10 2020    44      0             Afghanistan
AF
## 4 04/10/2020   4    10 2020     7      4             Afghanistan
AF
## 5 03/10/2020   3    10 2020     5      0             Afghanistan
AF
## 6 02/10/2020   2    10 2020    17      0             Afghanistan
AF
##   countryterritoryCode popData2019 continentExp
## 1                  AFG    38041757         Asia
## 2                  AFG    38041757         Asia
## 3                  AFG    38041757         Asia
## 4                  AFG    38041757         Asia
## 5                  AFG    38041757         Asia
## 6                  AFG    38041757         Asia
##   Cumulative_number_for_14_days_of_COVID.19_cases_per_100000
date
## 1                                                  1.0593622 2020-
10-07
## 2                                                  1.0830204 2020-
10-06
## 3                                                  0.7807210 2020-
10-05
## 4                                                  0.6650587 2020-
10-04
## 5                                                  0.9752441 2020-
10-03
## 6                                                  1.0856491 2020-
10-02
```

f. (1 points) Create a scatterplot showing cases vs. deaths. Set an appropriate plot title and axis titles.

```
# Note: 25 observations with negative cases or deaths are excluded
covid_filter <- covid |> filter(cases >= 0 & deaths >= 0)

ggplot(data = covid_filter, aes(x = deaths, y = cases)) +
  geom_point() +
  xlab("Covid deaths") +
  ylab("Covid cases") +
  ggtitle("Relationship between Covid cases and deaths")
```

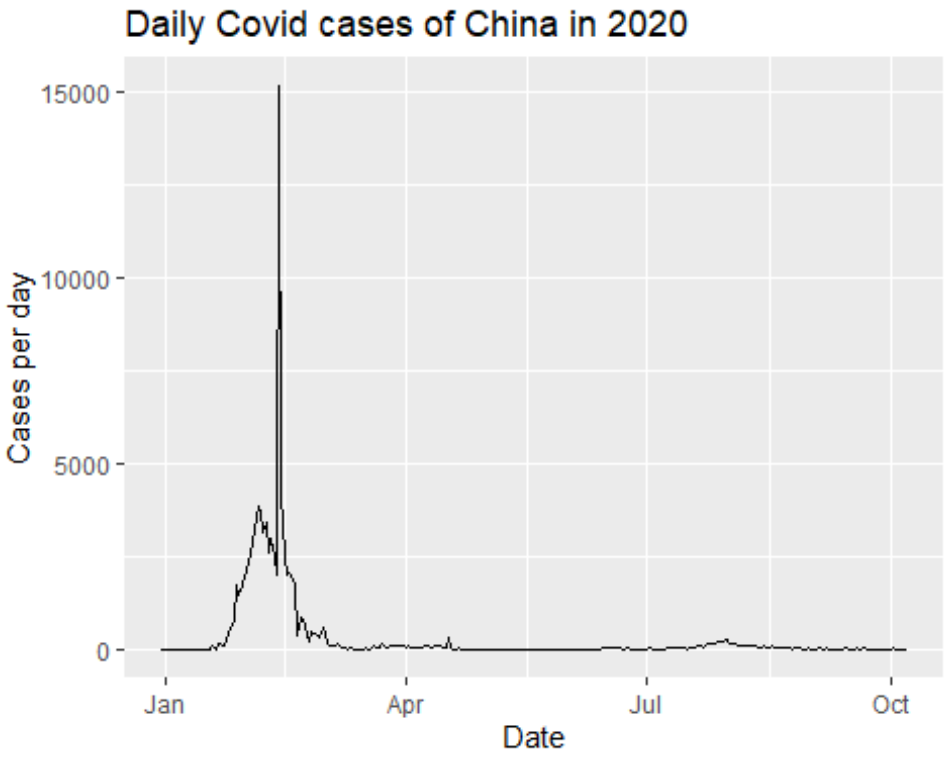



```
plot(covid_filter$deaths, covid_filter$cases,
     sub = "Relationship between Covid cases and deaths",
     xlab = "Covid deaths",
     ylab = "Covid cases")
```

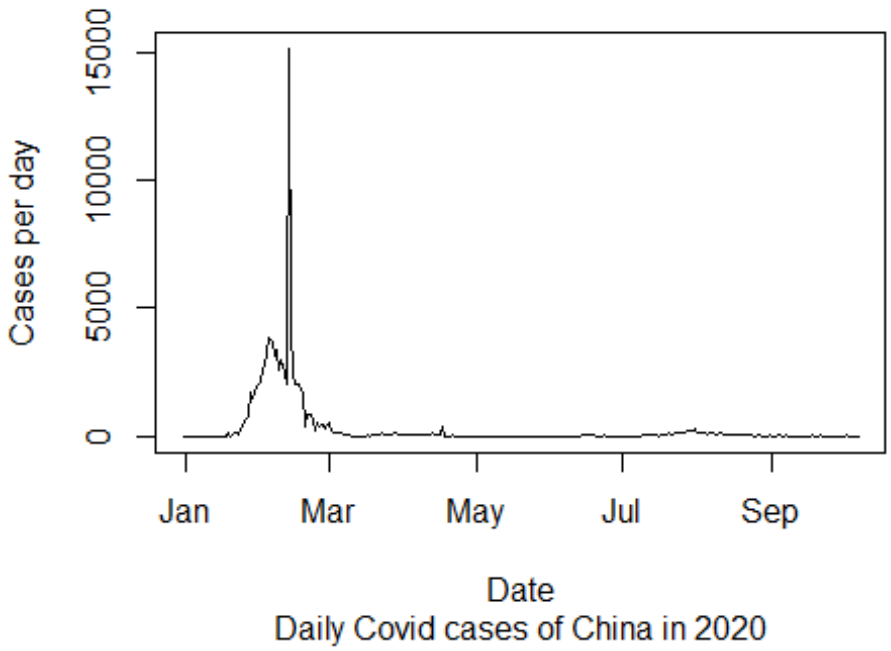


g. (2 points) Create a line plot using data with `countriesAndTerritories=="China"`, showing date on the x-axis and cases per day on the y-axis. Set an appropriate plot title and axis titles.

```
china <- covid_filter |> filter(countriesAndTerritories == "China")
ggplot(data = china, aes(x = date, y = cases)) +
  geom_line() +
  xlab("Date") +
  ylab("Cases per day") +
  ggtitle("Daily Covid cases of China in 2020")
```

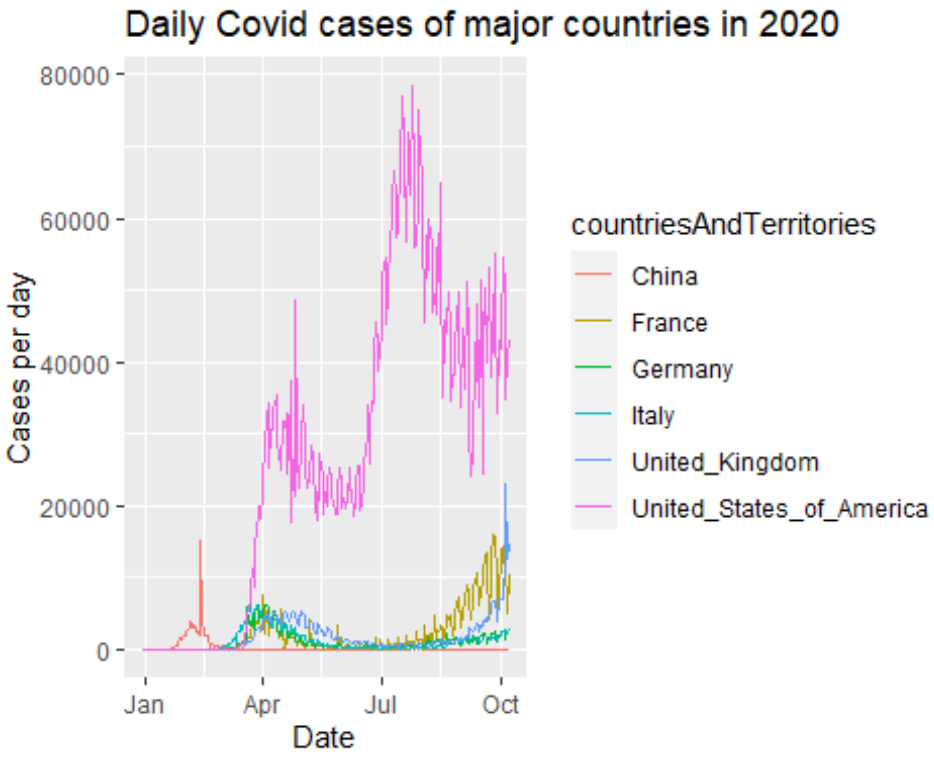



```
plot(china$date, china$cases,
     type = "l",
     sub = "Daily Covid cases of China in 2020",
     xlab = "Date",
     ylab = "Cases per day")
```



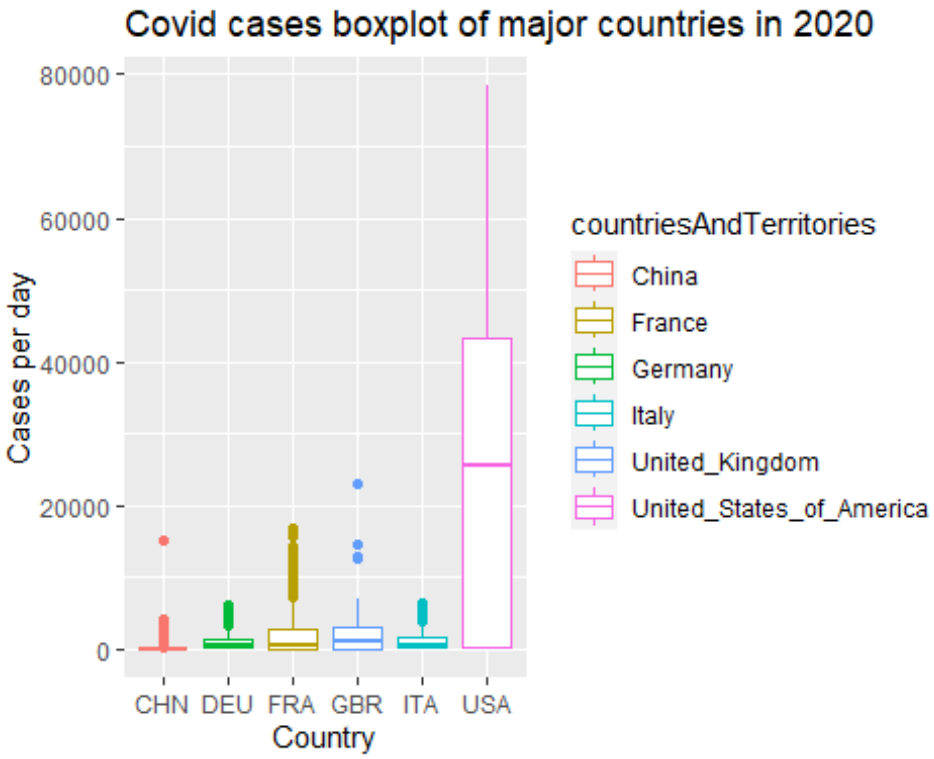
h. (2 points) Similar to above, create a line plot using the data of six countries including "China", "United_States_of_America", "United_Kingdom", "France", "Germany", and "Italy". Use different line colors for each country. Set an appropriate plot title and axis titles.

```
six_countries <- covid_filter |> filter(
  countriesAndTerritories %in%
    c("China", "United_States_of_America", "United_Kingdom",
      "France", "Germany", "Italy"))

ggplot(data = six_countries, aes(x = date, y = cases, col =
countriesAndTerritories)) +
  geom_line() +
  xlab("Date") +
  ylab("Cases per day") +
  ggtitle("Daily Covid cases of major countries in 2020")
```

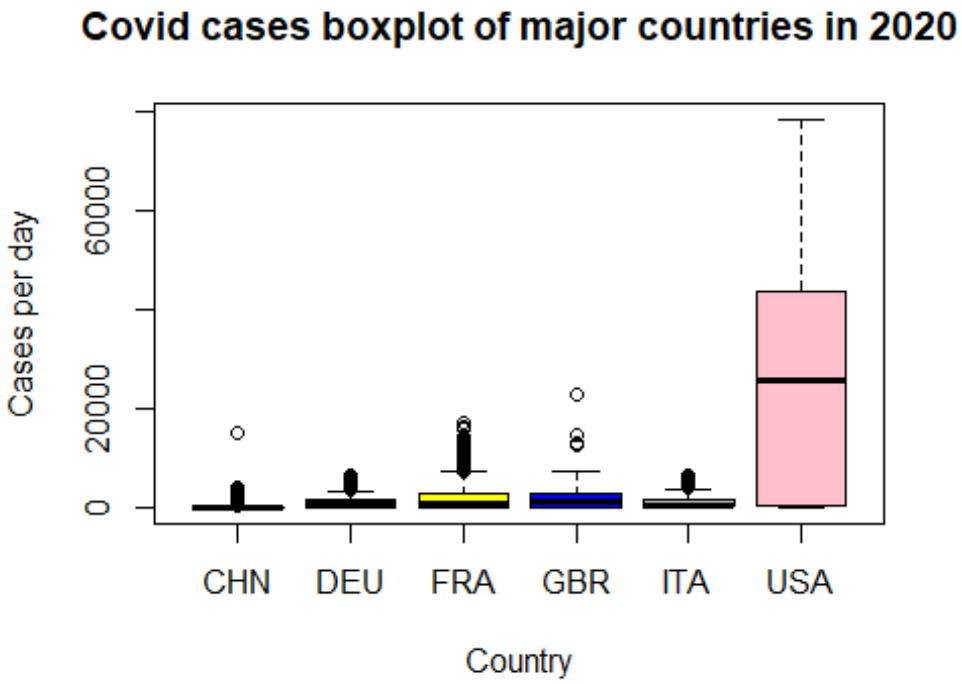


i. (2 points) Similar to question h, create a boxplot instead.

```
ggplot(data = six_countries, aes(x = countryterritoryCode, y = cases,
col = countriesAndTerritories)) +
  geom_boxplot() +
  xlab("Country") +
  ylab("Cases per day") +
  ggtitle("Covid cases boxplot of major countries in 2020")
```




```
boxplot(six_countries$cases ~ six_countries$countryterritoryCode,
        col = c("red","green", "yellow", "blue", "gray", "pink"),
        main = "Covid cases boxplot of major countries in 2020",
        xlab = "Country",
        ylab = "Cases per day",
        )
```


Covid cases boxplot of major countries in 2020
Cases per day
0
20000
60000
CHN
DEU
FRA
GBR
ITA
USA
Country