Report on COVID19 Data

2023-11-27

R code chunk "load_library" to load libraries

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
```

1. Introduction

This report is to analyze COVID19 Data provided by Johns Hopkins University through its GitHub public repo. R Markdown is used for this analysis.

About dataset

This is the dataset for the 2019 Novel Coronavirus Visual Dashboard operated by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE). Also, Supported by ESRI Living Atlas Team and the Johns Hopkins University Applied Physics Lab (JHU APL).

Questions which will be addressed through this analysis

1. Was trend of Covid19 Cases and deaths for US similar as of China where outbreak started?
2. Was trend of Covid19 Cases and deaths similar in US East and West Coast's States ?
3. Whether there relation between number of deaths and number of cases ?

2. Report - Analysis and Visualizations

We will be analysing and visualizing these trends:

- Trend of Covid19 Cases and deaths for US and China per year
- Trend of trend of Covid19 Cases and deaths US East and West Coast State.

Step 1 - Identify and Import the Data

Create URLs for Importing Data from Johns Hopkins University's Github Repo

R code chunk "get_jhu_data" to create URL for importing datasets

```
##Get current data in four files
# They all begin in same way
url_in <- "https://raw.github.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_
file_names <- c("time_series_covid19_confirmed_global.csv", "time_series_covid19_deaths_global.csv", "ti
urls <- str_c(url_in,file_names)
uid_lookup_url <- "https://raw.github.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/UID_ISO_FIPS
```

Let's now read/import the main CSVs.

R code chunk "import_data" import datasets

```
global_cases_raw  <- read_csv(urls[1])
global_deaths_raw<- read_csv(urls[2])
US_cases_raw <- read_csv(urls[3])
US_deaths_raw <- read_csv(urls[4])
uid_raw <- read_csv(uid_lookup_url)
```

Step 2 - Tidy up datasets

After looking at global cases and global deaths, let tidy up the datasets

R code chunk "tidy_global_data" to tidy up global raw dataset

```
# create global_cases dataframe
global_cases <- global_cases_raw %>%
  pivot_longer( cols = -c(`Province/State`,
                          `Country/Region`, Lat, Long ),
                names_to = "date",
                values_to = "cases") %>%
  select(-c(Lat,Long))

# create global_deaths dataframe
global_deaths <- global_deaths_raw %>%
  pivot_longer( cols = -c(`Province/State`,
                          `Country/Region`, Lat, Long ),
                names_to = "date",
                values_to = "deaths") %>%
  select(-c(Lat,Long))

# create uid dataframe which will be used later to add population field
uid <- uid_raw %>%
  select (-c(Lat, Long_, Combined_Key, iso2:code3))

# create global dataframe with population field
global <- global_cases %>%
    full_join(global_deaths) %>%
  rename( Country_Region = 'Country/Region',
          Province_State = 'Province/State') %>%
```

```
  mutate( date = mdy(date)) %>%
  filter(cases>0) %>%
  unite("Combined_key",
        c(Province_State,Country_Region),
        sep = (", "),
        na.rm = TRUE,
        remove = FALSE) %>%
  left_join(uid, by = c("Province_State", "Country_Region" )) %>%
  select(-c(UID,FIPS)) %>%
  select(Province_State, Country_Region, date,
         cases, deaths, Population,
         Combined_key)
```

```
## Joining with `by = join_by(`Province/State`, `Country/Region`, date)`
```

```
# create global_US_China dataframe with only US and China data from global dataframe
global_US_China <- global %>%
  filter(Country_Region %in% c("US","China")) %>%
  mutate(month = month(date,label = TRUE), year=year(date) )
```

R code chunk "tidy_US_data" to tidy up US raw dataset

```
# create US_cases dataframe
US_cases <- US_cases_raw %>%
  pivot_longer( cols = -c(UID:Combined_Key),
                names_to = "date",
                values_to = "cases") %>%
  select(Admin2:cases) %>%
  mutate(date = mdy(date)) %>%
  select(-c(Lat,Long_))

# create US_deaths dataframe
US_deaths <- US_deaths_raw %>%
  pivot_longer( cols = -c(UID:Population),
                names_to = "date",
                values_to = "deaths") %>%
  select(Admin2:deaths) %>%
  mutate(date = mdy(date)) %>%
  select(-c(Lat,Long_))

# create US dataframe by joining US_cases and US_deaths
US <- US_cases %>%
      full_join(US_deaths)
```

```
## Joining with `by = join_by(Admin2, Province_State, Country_Region,
## Combined_Key, date)`
```

```
# create US_NY_CA dataframe  by only including "New York" and California" for east and west coast data
US_NY_CA <- US %>%
  filter(Province_State %in% c("New York","California")) %>%
  mutate(month = month(date,label = TRUE), year=year(date) )
```

Step 3 - Analyze Data

Let's now analyze data of US and China

R code chunk "analyse_global_US_China" to create summerized dataframe global_US_China_sum

```
global_US_China_sum <- global_US_China %>%
  group_by(Country_Region, year, month) %>%
  summarize(cases = sum(cases), deaths = sum(deaths),
            Population = sum(Population)) %>%
  mutate(cases_per_mill = deaths *1000000 / Population) %>%
  mutate(deaths_per_mill = deaths *1000000 / Population) %>%
  select(Country_Region, year, month,
         cases, deaths,cases_per_mill, deaths_per_mill, Population) %>%
    mutate(new_cases = cases - lag(cases),
         new_deaths = deaths - lag(deaths)) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'Country_Region', 'year'. You can override
## using the `.groups` argument.
```

R code chunk "analyse_global_US_China" to create summerized dataframe US_NY_CA_sum

```
US_NY_CA_sum <- US_NY_CA %>%
  group_by(Province_State, year, month) %>%
  summarize(cases = sum(cases), deaths = sum(deaths),
            Population = sum(Population)) %>%
  mutate(cases_per_mill = deaths *1000000 / Population) %>%
  mutate(deaths_per_mill = deaths *1000000 / Population) %>%
  select(Province_State, year, month,
         cases, deaths,cases_per_mill, deaths_per_mill, Population) %>%
    mutate(new_cases = cases - lag(cases),
         new_deaths = deaths - lag(deaths)) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'Province_State', 'year'. You can override
## using the `.groups` argument.
```

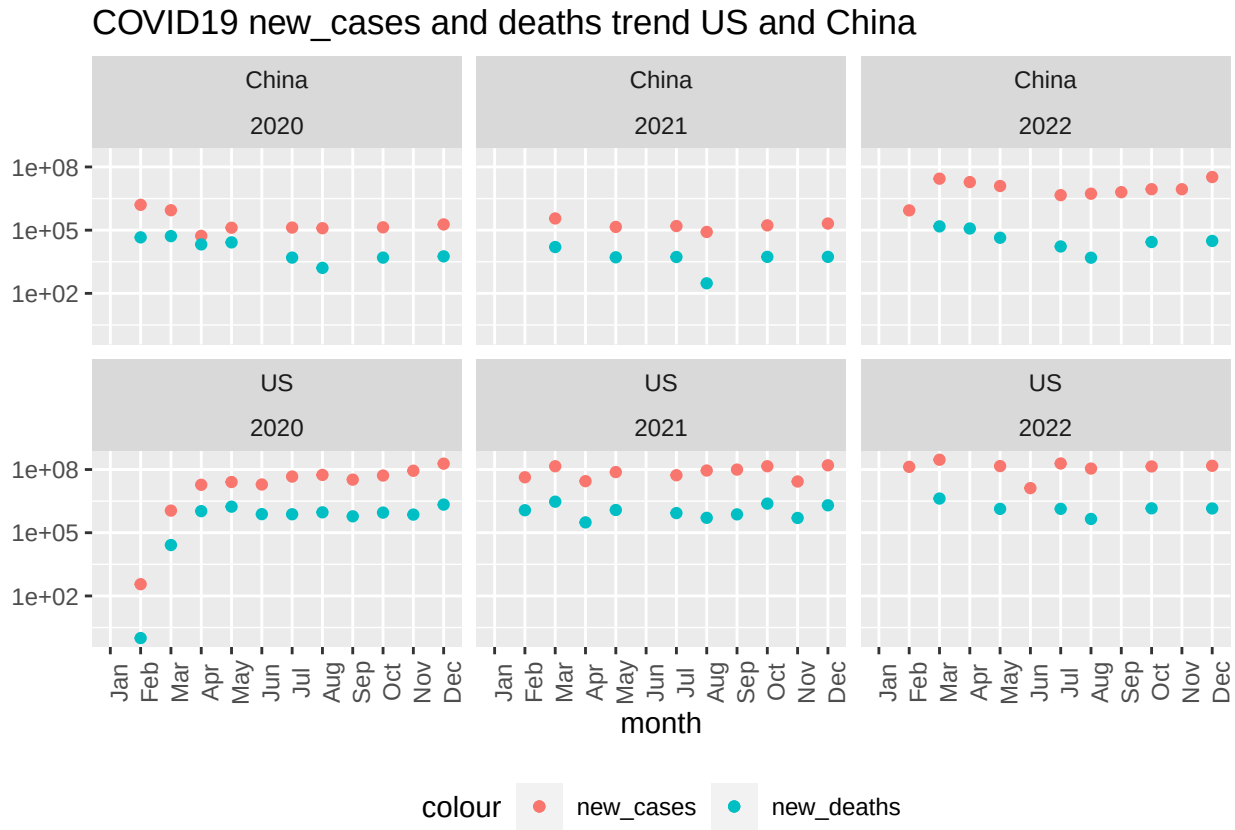
Step 4 - Visualizing Data

1. Let's visualize US and China new cases and new deaths per year.

R code chunk "visualize_global_US_China" to visualize US and China new cases and new deaths per year

```
global_US_China_sum %>%
  filter(!year == "2023") %>%
  ggplot(aes(x = month , y = new_cases)) +
  geom_point(aes(y = new_cases, color = "new_cases")) +
  geom_point(aes(y = new_deaths, color = "new_deaths")) +
  facet_wrap(~Country_Region + year) +
  scale_y_log10() +
```

```
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 90)) +
    labs(title = "COVID19 new_cases and deaths trend US and China" , y = NULL)
```



Based on above graph, below can be concluded:

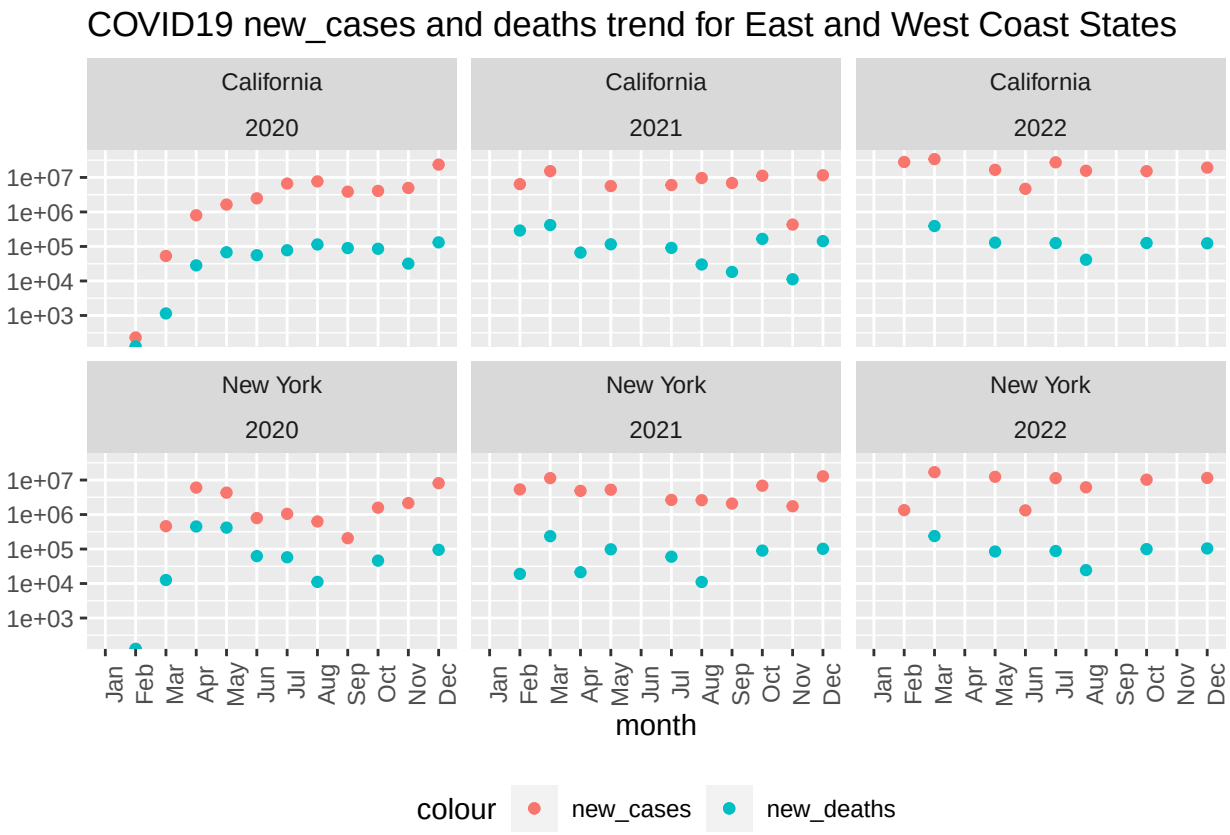
- In 2020, China was recovering from Covid cases and death. However, there was rise of cases and death in US.
- In 2022, trend for US cases are less than China.
- In 2023, China and US cases trend are similar.

2. Let's visualize New York and California new cases and new deaths per year. This is to compare east coast and west coast states.

R code chunk "visualize_US_NY_CA_sum" to visualize US and China new cases and new deaths per year

```
US_NY_CA_sum %>%
  filter(!year == "2023") %>%
  ggplot(aes(x = month , y = new_cases)) +
  geom_point(aes(y = new_cases, color = "new_cases")) +
  geom_point(aes(y = new_deaths, color = "new_deaths")) +
  facet_wrap(~Province_State + year) +
  scale_y_log10() +
  theme(legend.position = "bottom",
```

```
        axis.text.x = element_text(angle = 90)) +
    labs(title = "COVID19 new_cases and deaths trend for East and West Coast States" , y = NULL)
```



Based on above graph, below can be concluded:

- In 2020, there was constant surge of cases in West Coast (California). For East Coast, there was reduction of cases between June to September.
- Apart from 2020, trend for east cost (New York) and West Coast (California) is almost same.

Step 5 - Modeling

Based on the trend in US, lets check dependecy between cases per thousand and death per thousand. This is to check if mortality was more when number of cases increased.

Let's first create dataframe having year, summary of total shooting incident and shooting incident involving the victim of age group 18-44.

R code chunk "analyse_US_state_data_for_model" to get US_state_totals dataframe for linear model

```
US_by_state <- US %>%
  group_by(Province_State, Country_Region, date) %>%
  summarize(cases = sum(cases), deaths = sum(deaths),
            Population = sum(Population)) %>%
  mutate(deaths_per_mill = deaths *1000000 / Population) %>%
```

```
  select(Province_State, Country_Region, date,
         cases, deaths, deaths_per_mill, Population) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'Province_State', 'Country_Region'. You can
## override using the `.groups` argument.
```

```
US_state_totals <- US_by_state %>%
  group_by(Province_State) %>%
  summarize(deaths = max(deaths), cases = max(cases),
            population = max(Population),
            cases_per_thou = 1000* cases / population ,
            death_per_thou = 1000* deaths  / population ) %>%
  filter(cases > 0 , population > 0 )
```

Linear model and summary of the linear model.

R code chunk "model_linear_death_per_thousand" to do liner modelling based on case per thousand and death_per_thousand and get perdication value

```
mod <- lm(death_per_thou ~ cases_per_thou, data= US_state_totals)
summary(mod)
```

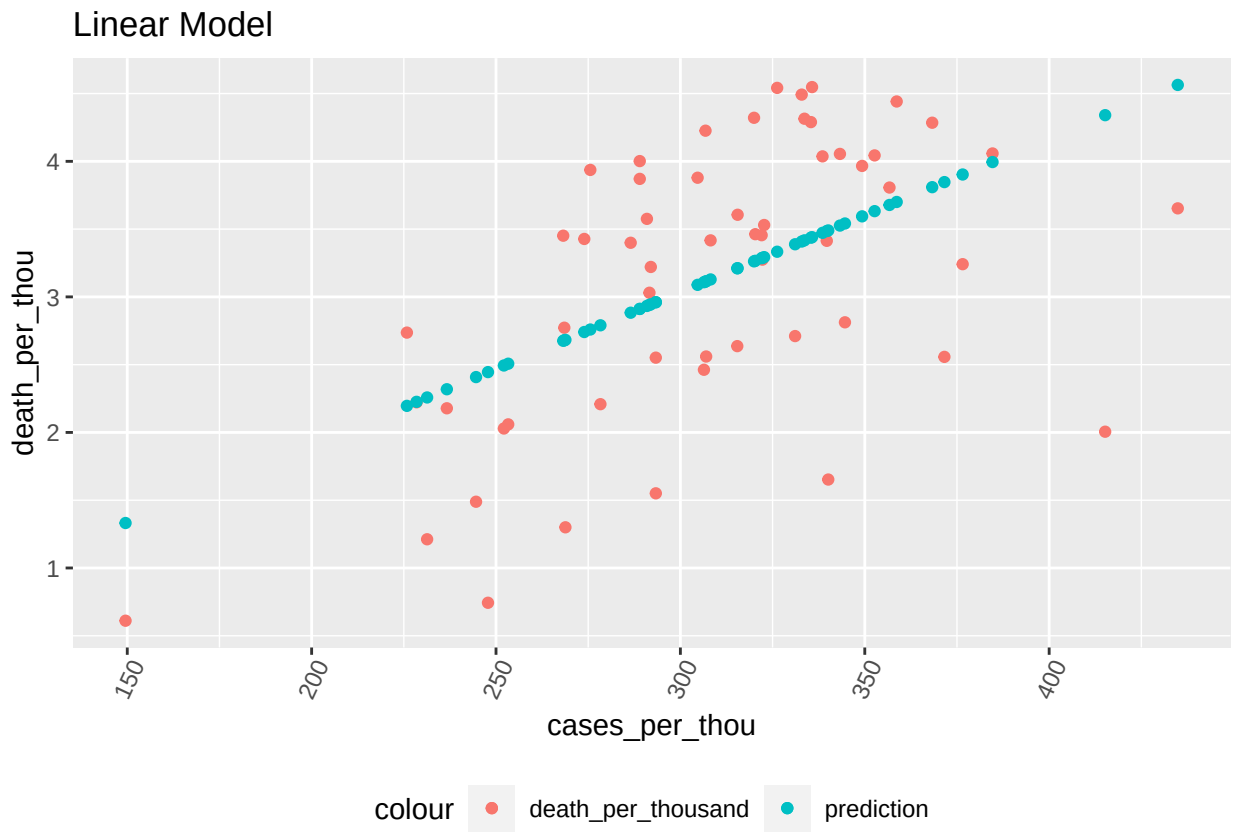
```
##
## Call:
## lm(formula = death_per_thou ~ cases_per_thou, data = US_state_totals)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3352 -0.5978  0.1491  0.6535  1.2086
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -0.36167    0.72480  -0.499     0.62
## cases_per_thou  0.01133    0.00232   4.881 9.76e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8615 on 54 degrees of freedom
## Multiple R-squared:  0.3061, Adjusted R-squared:  0.2933
## F-statistic: 23.82 on 1 and 54 DF,  p-value: 9.763e-06
```

```
US_tot_w_pred <- US_state_totals %>%
  mutate(pred = predict(mod))
```

R code chunk "visualize_model" visualize death per thousand and predicted value

```
US_tot_w_pred  %>%
  ggplot() +
  geom_point(aes(x = cases_per_thou, y = death_per_thou, color = "death_per_thousand")) +
  geom_point(aes(x = cases_per_thou, y = pred, color = "prediction")) +
```

```
theme(legend.position = "bottom",
      axis.text.x = element_text(angle = 90)) +
labs(title = "Linear Model" ) +
theme(axis.text.x = element_text(angle=65, vjust=0.6))
```



Based on above graph, below can be concluded:

- Prediction is linear.
- Model is reasonably correct in predicting in more on lower and somewhat in higher end.
- This also proves with current trend, mortality rate was dependant on no of cases. This might because of limited medical facity to handle surge in cases.

3. Final Summary

After analyzing COVID19 dataset, we can conclude below results of the questions stated at the start of the analysis :-

1. Was trend of Covid19 Cases and deaths for US similar as of China where outbreak started? As per Analysis, we can say in 2020, China was already recovering when in US COVID was speading. However, by 2022, there was similar situation in both US and China.

2. Was trend of Covid19 Cases and deaths similar in US East and West Coast's States? As per Analysis, in 2020, we can see constant rise of number of cases in West Coast. This might be due to more number of travellers. After 2020, we can see similar trend for East and West coast.

3. Whether there relation between number of deaths and number of cases ? As per modelling, there is probability of deaths when number of cases increases. This might be beause of capacity of medical faclity to support sudden surge of cases.

Bias Indentification

Personal Bias These are the two personal bias for this analysis:

- China was more impacted than US.
- US East coast was more impacted than West Coast due to cold weather.

Bias in Data There can be chances that cases of other countries are not accurate than of the US.

To overcome bias, I trusted dataset and started analysis without any pre-judgment.

R code chunk to display session information

```
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-apple-darwin20 (64-bit)
## Running under: macOS Ventura 13.5.2
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRlapack.dylib;  LAPACK v
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: Asia/Kolkata
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.9.3 forcats_1.0.0   stringr_1.5.0   dplyr_1.1.3
##  [5] purrr_1.0.2     readr_2.1.4     tidyr_1.3.0     tibble_3.2.1
##  [9] ggplot2_3.4.4   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] bit_4.0.5         gtable_0.3.4      crayon_1.5.2      compiler_4.3.1
##  [5] tidyselect_1.2.0  parallel_4.3.1    scales_1.2.1      yaml_2.3.7
##  [9] fastmap_1.1.1     R6_2.5.1          labeling_0.4.3    generics_0.1.3
## [13] curl_5.1.0        knitr_1.44        munsell_0.5.0     pillar_1.9.0
## [17] tzdb_0.4.0        rlang_1.1.1       utf8_1.2.4        stringi_1.7.12
## [21] xfun_0.40         bit64_4.0.5       timechange_0.2.0  cli_3.6.1
## [25] withr_2.5.1       magrittr_2.0.3    digest_0.6.33     grid_4.3.1
## [29] vroom_1.6.4       rstudioapi_0.15.0 hms_1.1.3         lifecycle_1.0.3
## [33] vctrs_0.6.4       evaluate_0.22     glue_1.6.2        farver_2.1.1
## [37] fansi_1.0.5       colorspace_2.1-0  rmarkdown_2.25    tools_4.3.1
## [41] pkgconfig_2.0.3   htmltools_0.5.6.1
```