Alabama Covid-19 Cases

Tom Plunkett

2025-03-03

Introduction:

I live in Huntsville, Alabama so I used R to analyze Covid-19 cases for the State of Alabama. I compared COVID cases per thousand with respect to deaths per thousand. I used COVID-19 data sourced from the Johns Hopkins GitHub repository. These datasets provide daily updates on the total number of confirmed COVID-19 cases and deaths for each country worldwide, as well as for each state in the United States. The data sets can be found at the following URL: https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series.

Libraries

```
library(tidyverse)
library(lubridate)
library(dplyr)
library(ggplot2)
library(readr)
```

Importing Dataset and reading data from csv file

First, import the required dataset from the public repository from the following source: https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/

```
url_in <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covi

file_names <- c("time_series_covid19_confirmed_global.csv", "time_series_covid19_deaths_global.csv","ti
                "time_series_covid19_deaths_US.csv")

urls <- str_c(url_in, file_names)
global_cases <- read_csv(urls[1])
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
global_deaths <- read_csv(urls[2])
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
US_cases <- read.csv(urls[3])
US_deaths <- read.csv(urls[4])
```

TIDY AND TRANSFORM

Now tidy the dataset and transform it into useful form for further data analysis.

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

global_deaths <- global_deaths %>%
  pivot_longer(cols = -c('Province/State', 'Country/Region', Lat, Long), names_to = "date", values_to =
  select(-c(Lat,Long))

global <- global_cases %>% full_join(global_deaths) %>%
  rename(Country_Region = 'Country/Region',
         Province_State = 'Province/State') %>%
  mutate(date = mdy(date))
```

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

```
global <- global %>% filter(cases > 0)

global <- global %>%
  unite("Combined_Key", c(Province_State, Country_Region),
        sep = ", ",
        na.rm = TRUE,
        remove = FALSE)

uid_lookup_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/U

uid <- read_csv(uid_lookup_url) %>% select(-c(Lat, Long_, Combined_Key, code3, iso2, iso3, Admin2))
```

```
## Rows: 4321 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): iso2, iso3, FIPS, Admin2, Province_State, Country_Region, Combined_Key
## dbl (5): UID, code3, Lat, Long_, Population
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
global <- global %>%
  left_join(uid, by = c("Province_State", "Country_Region")) %>% select(-c(UID, FIPS)) %>% select(Provi

global
```

```
## # A tibble: 306,827 x 7
##    Province_State Country_Region date       cases deaths Population Combined_Key
##    <chr>          <chr>          <date>     <dbl>  <dbl>      <dbl> <chr>
##  1 <NA>           Afghanistan    2020-02-24     5      0   38928341 Afghanistan
##  2 <NA>           Afghanistan    2020-02-25     5      0   38928341 Afghanistan
##  3 <NA>           Afghanistan    2020-02-26     5      0   38928341 Afghanistan
##  4 <NA>           Afghanistan    2020-02-27     5      0   38928341 Afghanistan
##  5 <NA>           Afghanistan    2020-02-28     5      0   38928341 Afghanistan
##  6 <NA>           Afghanistan    2020-02-29     5      0   38928341 Afghanistan
##  7 <NA>           Afghanistan    2020-03-01     5      0   38928341 Afghanistan
##  8 <NA>           Afghanistan    2020-03-02     5      0   38928341 Afghanistan
##  9 <NA>           Afghanistan    2020-03-03     5      0   38928341 Afghanistan
## 10 <NA>           Afghanistan    2020-03-04     5      0   38928341 Afghanistan
## # i 306,817 more rows
```

```
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:306827      Length:306827      Min.   :2020-01-22   Min.   :        1
##  Class :character   Class :character   1st Qu.:2020-12-12   1st Qu.:     1316
##  Mode  :character   Mode  :character   Median :2021-09-16   Median :    20365
##                                        Mean   :2021-09-11   Mean   :  1032863
##                                        3rd Qu.:2022-06-15   3rd Qu.:   271281
##                                        Max.   :2023-03-09   Max.   :103802702
##
##      deaths          Population        Combined_Key
##  Min.   :      0   Min.   :6.700e+01   Length:306827
##  1st Qu.:      7   1st Qu.:7.866e+05   Class :character
##  Median :    214   Median :6.948e+06   Mode  :character
##  Mean   :  14405   Mean   :2.890e+07
##  3rd Qu.:   3665   3rd Qu.:2.914e+07
##  Max.   :1123836   Max.   :1.380e+09
##                    NA's   :6729
```

Removing character "X" and converting date to date data type

Now convert the date to a date data type in the US data set, removing the "X" character from the date. Also, we can rename the column.

```
US_cases <- US_cases %>% pivot_longer(cols = -(UID:Combined_Key),
                                      names_to = "date",
                                      values_to = "cases") %>%
  select(Admin2:cases) %>%
```

```
  select(-c(Lat, Long_))

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

US_cases$date <- gsub("X", "", as.character(US_cases$date))
US_deaths$date <- gsub("X", "", as.character(US_deaths$date))

US <- US_cases %>% full_join(US_deaths) %>%
  mutate(date = mdy(date))
```

Organizing the data set

Now organize our dataset as per our analysis by grouping all the countries in a new dataframe.

```
global_country <- global %>% group_by(Province_State, Country_Region, date) %>% summarize(cases = sum(ca
```

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

```
global_country
```

```
## # A tibble: 306,827 x 7
##    Province_State Country_Region date       cases deaths deaths_per_mill
##    <chr>          <chr>          <date>     <dbl>  <dbl>           <dbl>
##  1 Alberta        Canada         2020-03-06     1      0               0
##  2 Alberta        Canada         2020-03-07     2      0               0
##  3 Alberta        Canada         2020-03-08     4      0               0
##  4 Alberta        Canada         2020-03-09     7      0               0
##  5 Alberta        Canada         2020-03-10     7      0               0
##  6 Alberta        Canada         2020-03-11    19      0               0
##  7 Alberta        Canada         2020-03-12    19      0               0
##  8 Alberta        Canada         2020-03-13    29      0               0
##  9 Alberta        Canada         2020-03-14    29      0               0
## 10 Alberta        Canada         2020-03-15    39      0               0
## # i 306,817 more rows
## # i 1 more variable: Population <dbl>
```

```
total_all <- global_country %>%
  group_by(Country_Region) %>%
  summarize(deaths = max(deaths), cases = max(cases), population = max(Population), cases_per_thou = 100

total_all %>% slice_min(deaths_per_thou, n = 10)
```

```
## # A tibble: 10 x 6
##    Country_Region deaths cases population cases_per_thou deaths_per_thou
##    <chr>           <dbl> <dbl>      <dbl>          <dbl>           <dbl>
```

```
##  1 Holy See            0    29        809     35.8             0
##  2 Tuvalu              0  2828      11792    240.              0
##  3 Korea, North        6     1   25778815      0.0000388       0.000233
##  4 Burundi            38 53631   11890781      4.51            0.00320
##  5 Chad              194  7679   16425859      0.467           0.0118
##  6 South Sudan       138 18368   11193729      1.64            0.0123
##  7 Niger             315  9508   24206636      0.393           0.0130
##  8 Tajikistan        125 17786    9537642      1.86            0.0131
##  9 Benin             163 27999   12123198      2.31            0.0134
## 10 Tanzania          846 42906   59734213      0.718           0.0142
```

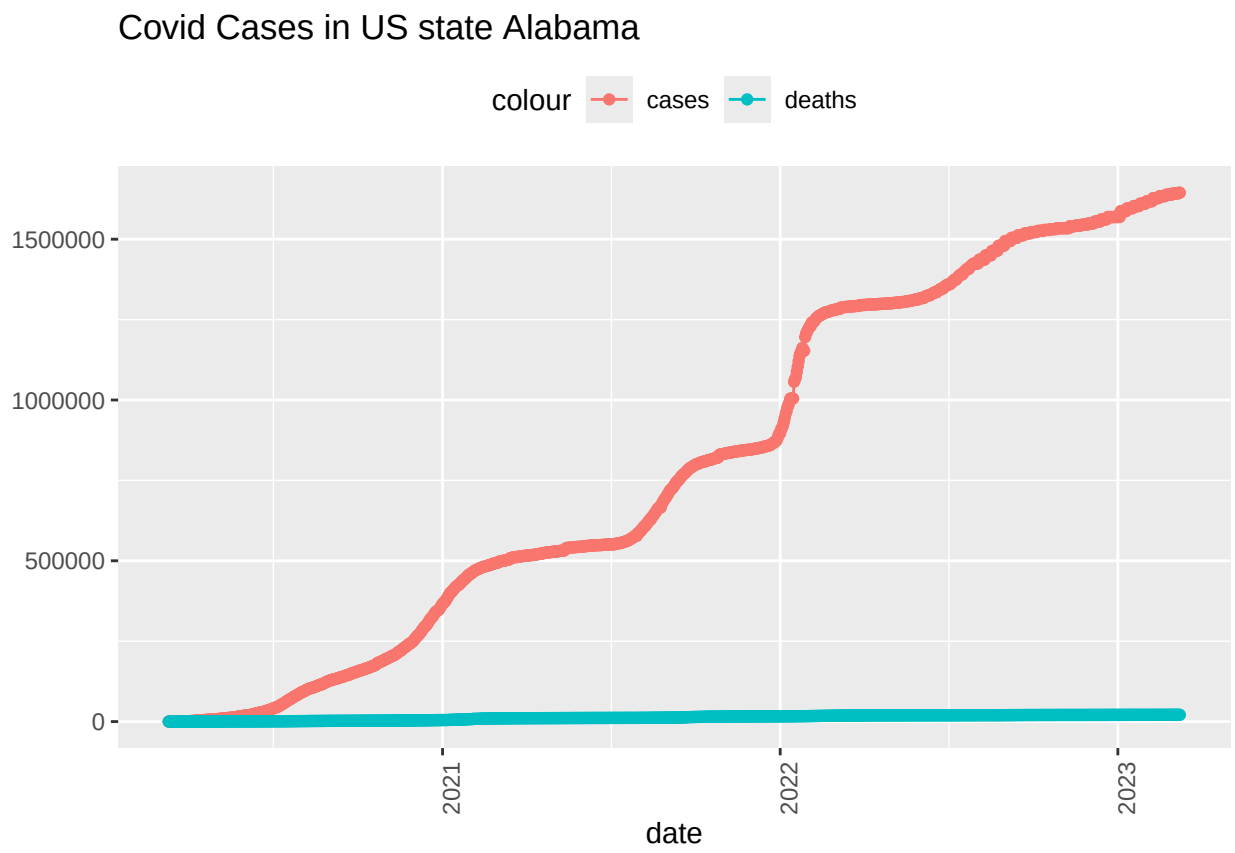
Visualization of COVID cases in Alabama

Data visualization is the graphical representation of information and data. First, visualize the COVID cases in Alabama using the absolute numbers of cases.

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

Let us visualize the COVID cases that occurred in Alabama in the US from the US dataset. The below graphs show the number of cases and deaths in Alabama. First let's visualize the absolute numbers.

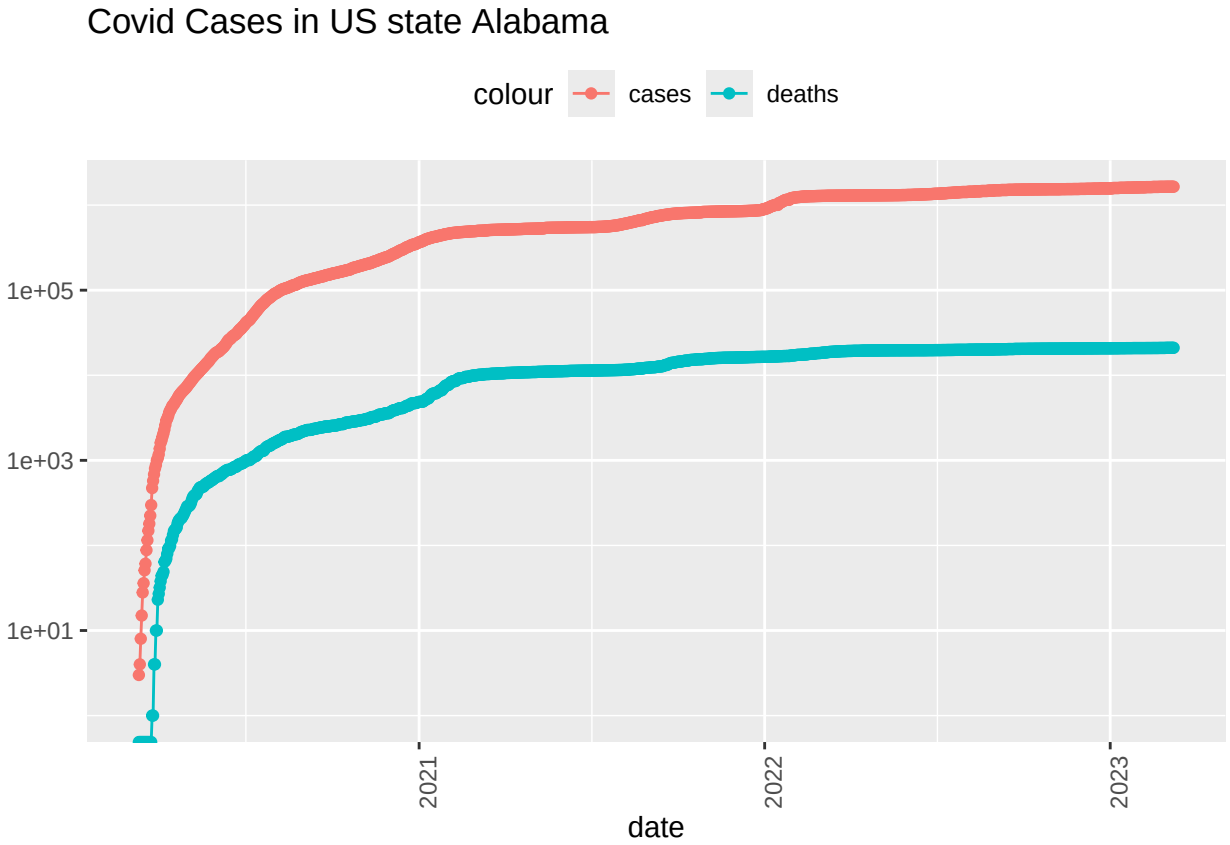
```
state <- "Alabama"
US_by_state %>% filter(Province_State == state) %>% filter(cases > 0) %>%
  ggplot(aes(x = date, y = cases)) +
  geom_line(aes(color = "cases")) +
  geom_point(aes(color = "cases")) +
  geom_line(aes(y = deaths, color = "deaths")) +
  geom_point(aes(y = deaths, color = "deaths")) +
  theme(legend.position = "top",
        axis.text.x = element_text(angle = 90)) +
  labs(title = str_c("Covid Cases in US state ", state), y = NULL)
```



Now lets visualize using log on the y axis.

```
state <- "Alabama"
US_by_state %>% filter(Province_State == state) %>% filter(cases > 0) %>%
  ggplot(aes(x = date, y = cases)) +
  geom_line(aes(color = "cases")) +
  geom_point(aes(color = "cases")) +
  geom_line(aes(y = deaths, color = "deaths")) +
  geom_point(aes(y = deaths, color = "deaths")) +
  scale_y_log10() +
  theme(legend.position = "top",
        axis.text.x = element_text(angle = 90)) +
  labs(title = str_c("Covid Cases in US state ", state), y = NULL)
```

```
## Warning in scale_y_log10(): log-10 transformation introduced infinite values.
## log-10 transformation introduced infinite values.
```



ANALYSING AND MODELING

Let us now delve deeper into the data set by comparing COVID cases per thousand with respect to deaths per thousand. Here is a linear model to show the correlation between them. The below model roughly follows the pandemic trend in the region. The yellow curve shows the observed deaths per thousand cases. The black one represents the linear model's estimate of cases per thousand.

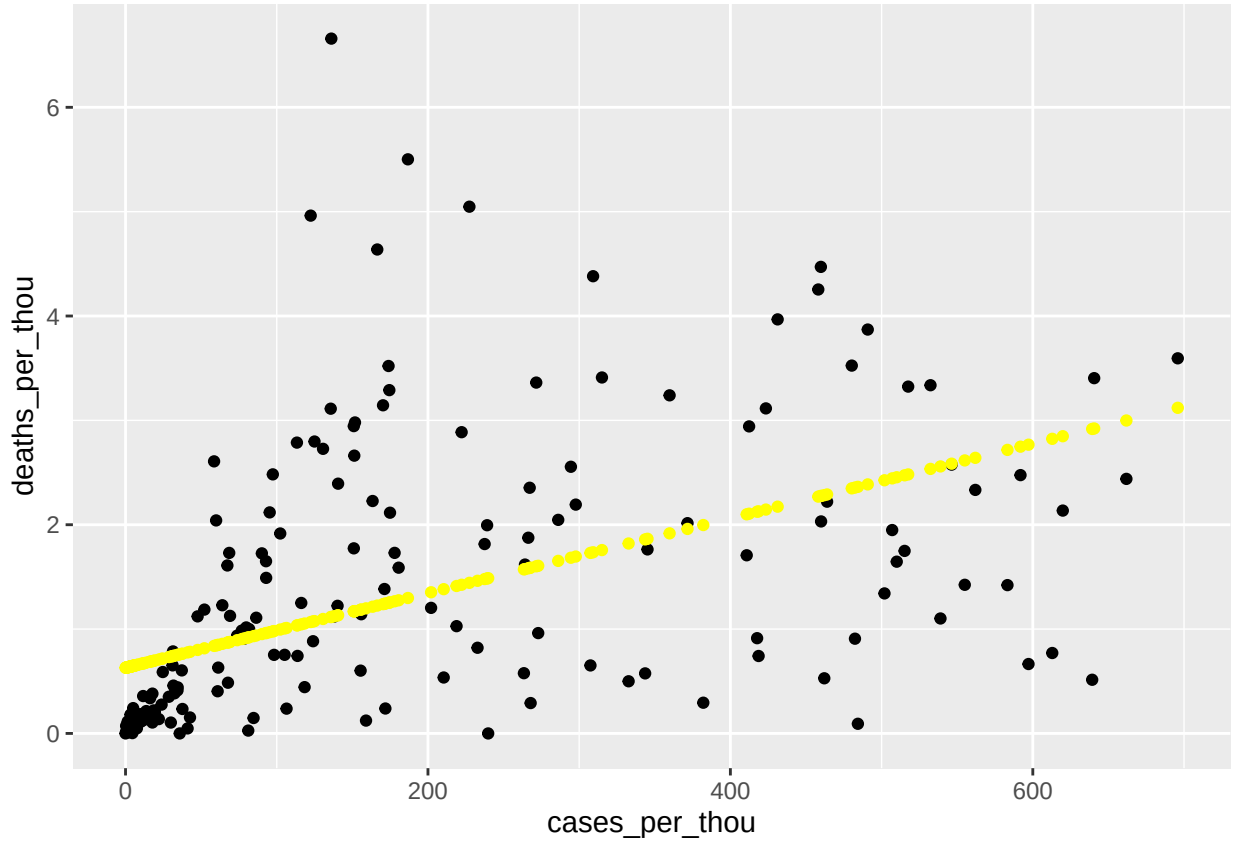
```
mod <- lm(deaths_per_thou ~ cases_per_thou, data = total_all)
summary(mod)
```

```
##
## Call:
## lm(formula = deaths_per_thou ~ cases_per_thou, data = total_all)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4034 -0.6103 -0.3892  0.4718  5.5419
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.6292479  0.1116264   5.637 6.10e-08 ***
## cases_per_thou 0.0035799  0.0004383   8.167 4.13e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.151 on 192 degrees of freedom
## Multiple R-squared:  0.2578, Adjusted R-squared:  0.254
```

```
## F-statistic:  66.7 on 1 and 192 DF,  p-value: 4.132e-14
```

```
x_grid <- seq(1, 151)
new_df <- tibble(cases_per_thou = x_grid)
total_pred <- total_all %>% mutate(pred = predict(mod))
```

```
total_pred %>% ggplot() +
  geom_point(aes(x = cases_per_thou, y = deaths_per_thou), color = "black") +
  geom_point(aes(x = cases_per_thou, y = pred), color = "yellow")
```



BIAS IDENTIFCATION:

Aggregating data from various sources introduces variability in quality, accuracy, and timeliness. The reliability of each source impacts the overall dataset. Inconsistent reporting of COVID cases and deaths across regions can skew the data. Varying definitions of COVID-related deaths adds complexity. Confirmed cases may not capture all infections due to testing limitations.

Researchers and data collectors may unintentionally introduce bias based on their perspectives. Undiagnosed or asymptomatic cases may be unreported. This introduces uncertainty about the actual prevalence of COVID-19.

CONCLUSION

The data visualizions showed how cases have increased over time. Yet the success of the vacinnes have signficantly reduced the deaths that were present during the early phase of the pandemic.

This John Hopkings Covid-19 data set helps researchers look for trends in the data on cases and death. More insights can be found by digging deeper into this data set.

```
sessionInfo()
```

```
## R version 4.4.3 (2025-02-28 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 11 x64 (build 26100)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Chicago
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.9.4 forcats_1.0.0   stringr_1.5.1   dplyr_1.1.4
##  [5] purrr_1.0.4     readr_2.1.5     tidyr_1.3.1     tibble_3.2.1
##  [9] ggplot2_3.5.1   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] bit_4.5.0.1       gtable_0.3.6      crayon_1.5.3      compiler_4.4.3
##  [5] tidyselect_1.2.1  parallel_4.4.3    scales_1.3.0      yaml_2.3.10
##  [9] fastmap_1.2.0     R6_2.6.1          labeling_0.4.3    generics_0.1.3
## [13] curl_6.2.1        knitr_1.49        munsell_0.5.1     pillar_1.10.1
## [17] tzdb_0.4.0        rlang_1.1.5       utf8_1.2.4        stringi_1.8.4
## [21] xfun_0.51         bit64_4.6.0-1     timechange_0.3.0  cli_3.6.4
## [25] withr_3.0.2       magrittr_2.0.3    digest_0.6.37     grid_4.4.3
## [29] vroom_1.6.5       rstudioapi_0.17.1 hms_1.1.3         lifecycle_1.0.4
## [33] vctrs_0.6.5       evaluate_1.0.3    glue_1.8.0        farver_2.1.2
## [37] colorspace_2.1-1  rmarkdown_2.29    tools_4.4.3       pkgconfig_2.0.3
## [41] htmltools_0.5.8.1
```