Supplementary Methods

Amos Apreku

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
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 becor
```

```
library(dplyr)
library(ggrepel)
```

```
dat <- readRDS("../data/dat.rds")
```

```
names(dat)
```

```
 [1] "date"            "mmwr_year"       "mmwr_week"       "state"
 [5] "state_name"      "population"      "region"          "region_name"
 [9] "cases"           "hosp"            "deaths"          "series_complete"
[13] "booster"
```

```
head(dat)
```

```
        date mmwr_year mmwr_week state state_name population region
1 2020-01-25      2020         4    AK     Alaska     732441     10
```

```
2 2020-02-01      2020         5    AK     Alaska     732441     10
3 2020-02-08      2020         6    AK     Alaska     732441     10
4 2020-02-15      2020         7    AK     Alaska     732441     10
5 2020-02-22      2020         8    AK     Alaska     732441     10
6 2020-02-29      2020         9    AK     Alaska     732441     10
        region_name cases hosp deaths series_complete booster
1 Pacific Northwest     0   NA      0              NA      NA
2 Pacific Northwest     0   NA      0              NA      NA
3 Pacific Northwest     0   NA      0              NA      NA
4 Pacific Northwest     0   NA      0              NA      NA
5 Pacific Northwest     0   NA      0              NA      NA
6 Pacific Northwest     0   NA      0              NA      NA
```

```r
# Load required libraries
library(dplyr)
library(knitr)
library(kableExtra)
```

```
Attaching package: 'kableExtra'
```

```
The following object is masked from 'package:dplyr':

    group_rows
```

```r
# Assuming your data is loaded into a dataframe called 'dat'
# Summarize metrics by region and state
summary_table <- dat %>%
  group_by(region_name, state_name) %>%
  summarize(
    Total_Cases = sum(cases, na.rm = TRUE),
    Total_Hospitalizations = sum(hosp, na.rm = TRUE),
    Total_Deaths = sum(deaths, na.rm = TRUE),
    Avg_Vaccination_Complete = mean(series_complete, na.rm = TRUE),
    .groups = 'drop'
  )

# Create a formatted table
summary_table %>%
  kbl(caption = "Descriptive Summary of COVID-19 Metrics by State and Region") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE)
```

Table 1: Descriptive Summary of COVID-19 Metrics by State and Region

region_name	state_name	Total_Cases	Total_Hospitalizations	Total_Deaths	Avg_Vac
Central Plains	Iowa	908936	59833	11516	
Central Plains	Kansas	945923	62167	10418	
Central Plains	Missouri	1790763	144374	24614	
Central Plains	Nebraska	575134	35319	5809	
Mid-Atlantic	Delaware	333772	22056	3049	
Mid-Atlantic	District of Columbia	178904	18098	1803	
Mid-Atlantic	Maryland	1379385	105562	19494	
Mid-Atlantic	Pennsylvania	3559331	270480	56413	
Mid-Atlantic	Virginia	2310846	138398	24954	
Mid-Atlantic	West Virginia	650556	47718	8621	
Midwest	Illinois	4136659	254531	40821	
Midwest	Indiana	2076326	151659	28472	
Midwest	Michigan	3106362	211790	39289	
Midwest	Minnesota	1795771	87985	16543	
Midwest	Ohio	3441458	284963	52592	
Midwest	Wisconsin	2030717	137940	18018	
Mountain States	Colorado	1783663	106829	16228	
Mountain States	Montana	333959	32324	3563	
Mountain States	North Dakota	291093	16657	2605	
Mountain States	South Dakota	282895	21397	3031	
Mountain States	Utah	1097475	51431	5769	
Mountain States	Wyoming	187034	12198	1454	
NY/NJ/PR/VI	New Jersey	3075271	192382	37375	
NY/NJ/PR/VI	New York	3572043	422486	45110	
NY/NJ/PR/VI	Puerto Rico	1122076	32013	6885	
New England	Connecticut	982335	72554	13040	
New England	Maine	322982	17137	3079	
New England	Massachusetts	2242176	119380	22699	
New England	New Hampshire	382013	20544	2791	
New England	Rhode Island	441466	14945	3641	
New England	Vermont	154243	7354	447	
Pacific	Arizona	2451004	165278	31618	
Pacific	California	12251820	674498	115326	
Pacific	Hawaii	374264	22461	1474	
Pacific	Nevada	898164	68699	12504	
Pacific Northwest	Alaska	297588	10553	1066	
Pacific Northwest	Idaho	526118	30883	5464	
Pacific Northwest	Oregon	975856	52436	9945	
Pacific Northwest	Washington	1957759	85807	16432	
South Central	Arkansas	995043	73224	13083	
South Central	Louisiana	1566171	103105	18346	
South Central	New Mexico	681242	41362	9103	
South Central	Oklahoma	1306350	117246	19548	
South Central	Texas	8508204	642929	108446	
Southeast	Alabama	1659936	128837	22441	
Southeast	Florida	7572282	610281	86809	
Southeast	Georgia	3080929	256882	38284	
Southeast	Kentucky	1743117	165575	21936	
Southeast	Mississippi	1000415	59403	15791	

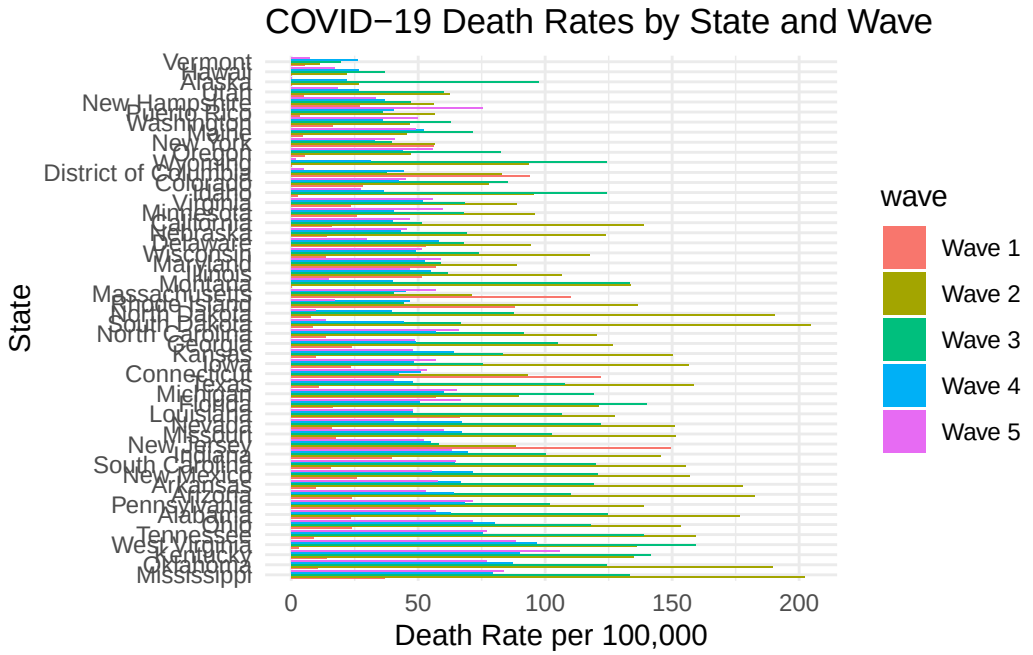
```r
dat <- dat %>%
  mutate(wave = case_when(
    date >= as.Date("2020-01-01") & date <= as.Date("2020-06-30") ~ "Wave 1",
    date >= as.Date("2020-07-01") & date <= as.Date("2021-02-28") ~ "Wave 2",
    date >= as.Date("2021-03-01") & date <= as.Date("2021-12-31") ~ "Wave 3",
    date >= as.Date("2022-01-01") & date <= as.Date("2022-06-30") ~ "Wave 4",
    date >= as.Date("2022-07-01") & date <= as.Date("2024-12-31") ~ "Wave 5",
    TRUE ~ NA_character_
  ))

# Calculate death rates by state and wave
death_rates <- dat %>%
  group_by(state_name, wave) %>%
  summarize(
    total_deaths = sum(deaths, na.rm = TRUE),
    population = max(population, na.rm = TRUE)
  ) %>%
  mutate(death_rate_per_100k = (total_deaths / population) * 100000)
```

```
`summarise()` has grouped output by 'state_name'. You can override using the
`.groups` argument.
```

```r
# View the death rates
print(death_rates)
```

```
# A tibble: 260 x 5
# Groups:   state_name [52]
   state_name wave   total_deaths population death_rate_per_100k
   <chr>      <chr>         <dbl>      <dbl>               <dbl>
 1 Alabama    Wave 1         1168    5024803                23.2
 2 Alabama    Wave 2         8905    5039877               177.
 3 Alabama    Wave 3         6273    5039877               124.
 4 Alabama    Wave 4         3184    5073903                62.8
 5 Alabama    Wave 5         2911    5108468                57.0
 6 Alaska     Wave 1            0     732441                 0
 7 Alaska     Wave 2          194     732673                26.5
 8 Alaska     Wave 3          713     732673                97.3
 9 Alaska     Wave 4          159     733276                21.7
10 Alaska     Wave 5            0     733406                 0
# i 250 more rows
```

```
# Plot death rates by state and wave
ggplot(death_rates, aes(x = reorder(state_name, -death_rate_per_100k), y = death_rate_per_10
  geom_bar(stat = "identity", position = "dodge") +
  coord_flip() +
  labs(title = "COVID-19 Death Rates by State and Wave",
       x = "State",
       y = "Death Rate per 100,000") +
  theme_minimal() +
  theme(axis.text.y = element_text(size = ))  # Set x-axis text size to 5
```




```
library(ggplot2)
library(dplyr)
library(maps)
```

```
Attaching package: 'maps'

The following object is masked from 'package:purrr':

    map
```

```r
library(viridis)
```

```
Loading required package: viridisLite
```

```
Attaching package: 'viridis'
```

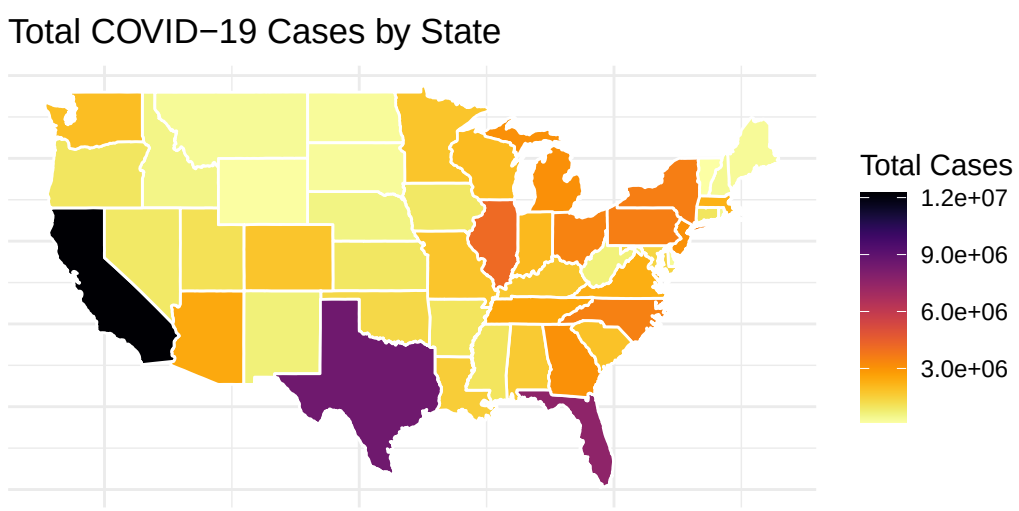
```
The following object is masked from 'package:maps':

    unemp
```

```r
state_summary <- dat %>%
  group_by(state_name) %>%
  summarize(
    case_rate = sum(cases, na.rm = TRUE) / sum(population, na.rm = TRUE) * 100000,
    death_rate = sum(deaths, na.rm = TRUE) / sum(population, na.rm = TRUE) * 100000,
    hosp_rate = sum(hosp, na.rm = TRUE) / sum(population, na.rm = TRUE) * 100000
  ) %>%
  arrange(desc(case_rate))
```

```r
library(dplyr)
library(DT)

# Calculate total counts and rates per 100,000 for each state
state_summary <- dat %>%
  group_by(state_name) %>%
  summarize(
    total_cases = sum(cases, na.rm = TRUE),
    total_deaths = sum(deaths, na.rm = TRUE),
    total_hosp = sum(hosp, na.rm = TRUE),
    case_rate = round(sum(cases, na.rm = TRUE) / sum(population, na.rm = TRUE) * 100000),
    death_rate = round(sum(deaths, na.rm = TRUE) / sum(population, na.rm = TRUE) * 100000),
    hosp_rate = round(sum(hosp, na.rm = TRUE) / sum(population, na.rm = TRUE) * 100000)
  ) %>%
  arrange(desc(case_rate))
```

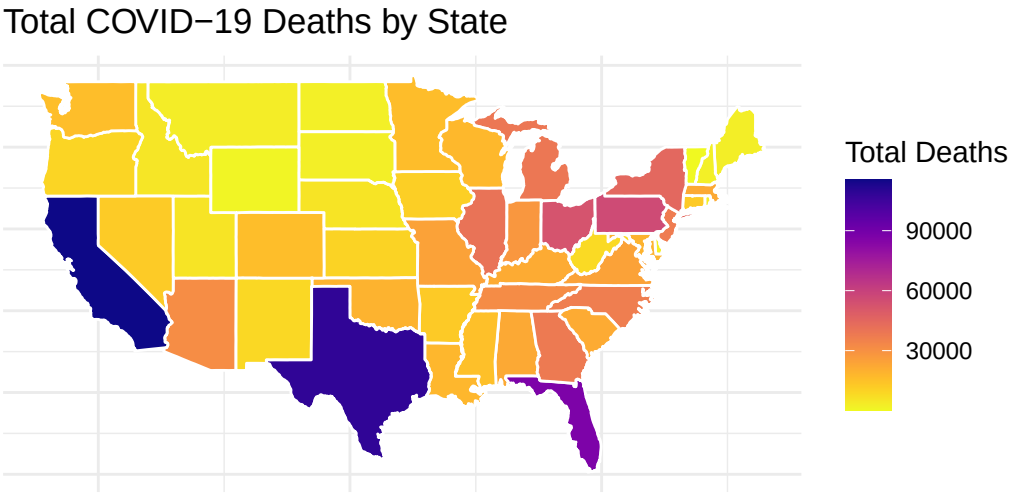
```r
# Load US map data
us_states <- map_data("state")

# Prepare the data by converting state names to lowercase for matching
```

```r
cases_data <- state_summary %>%
  mutate(state_name = tolower(state_name))

# Join the map data with the cases data
map_cases <- us_states %>%
  left_join(cases_data, by = c("region" = "state_name"))

# Plot for total cases
ggplot(map_cases, aes(x = long, y = lat, group = group, fill = total_cases)) +
  geom_polygon(color = "white") +
  coord_fixed(1.3) +
  scale_fill_viridis(option = "inferno", direction = -1, na.value = "grey80",
                     name = "Total Cases") +
  labs(title = "Total COVID-19 Cases by State") +
  theme_minimal() +
  theme(axis.text = element_blank(), axis.ticks = element_blank(),
        axis.title = element_blank())
```




```r
# Plot for total deaths
ggplot(map_cases, aes(x = long, y = lat, group = group, fill = total_deaths)) +
  geom_polygon(color = "white") +
  coord_fixed(1.3) +
  scale_fill_viridis(option = "plasma", direction = -1, na.value = "grey80",
                     name = "Total Deaths") +
  labs(title = "Total COVID-19 Deaths by State") +
  theme_minimal() +
  theme(axis.text = element_blank(), axis.ticks = element_blank(),
```

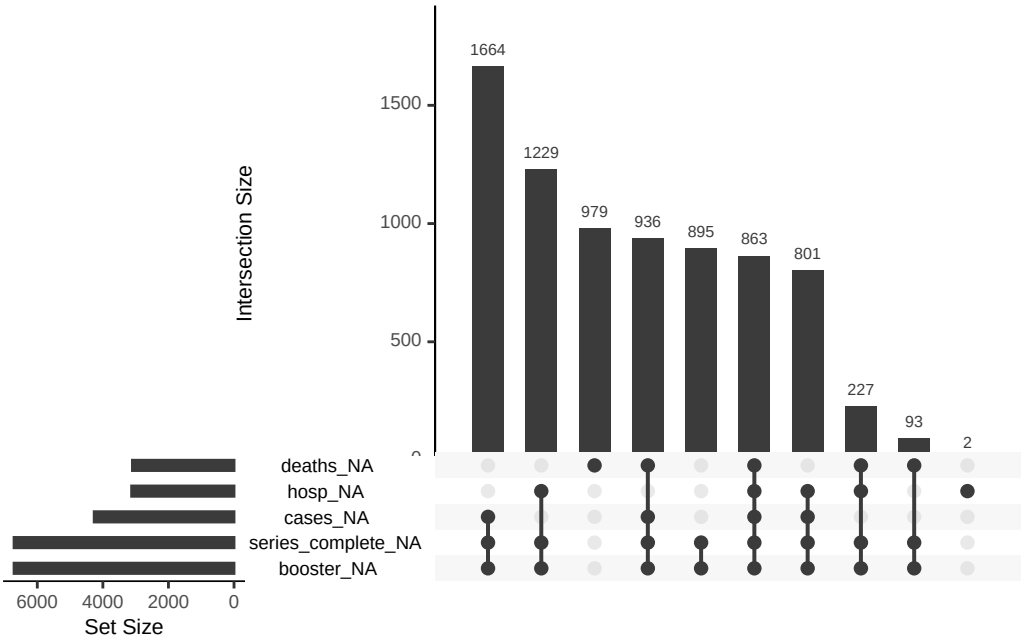
```
        axis.title = element_blank())
```




```
library(dplyr)
library(tidyr)
library(ggplot2)
library(naniar)

# Check the overall missingness in the dataset
miss_summary <- dat %>%
  summarise(
    missing_cases = sum(is.na(cases)),
    missing_hosp = sum(is.na(hosp)),
    missing_deaths = sum(is.na(deaths)),
    missing_series_complete = sum(is.na(series_complete)),
    missing_booster = sum(is.na(booster))
  )

print(miss_summary)
```

```
  missing_cases missing_hosp missing_deaths missing_series_complete
1          4264         3122           3098                    6708
  missing_booster
1            6708
```

```
# Visualize missing data patterns
gg_miss_upset(dat)
```

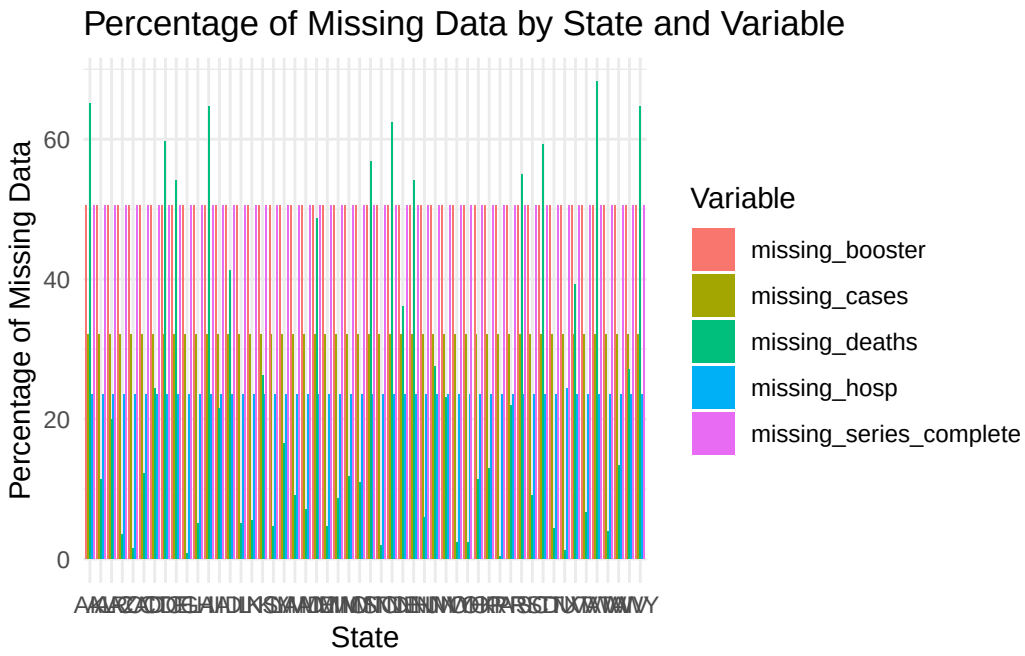



```r
# Calculate the percentage of missing data by state and variable
missing_by_state <- dat %>%
  group_by(state) %>%
  summarise(
    missing_cases = mean(is.na(cases)) * 100,
    missing_hosp = mean(is.na(hosp)) * 100,
    missing_deaths = mean(is.na(deaths)) * 100,
    missing_series_complete = mean(is.na(series_complete)) * 100,
    missing_booster = mean(is.na(booster)) * 100
  )

print(missing_by_state)
```

```
# A tibble: 52 x 6
   state missing_cases missing_hosp missing_deaths missing_series_complete
   <chr>         <dbl>        <dbl>          <dbl>                   <dbl>
 1 AK             32.2         23.5         65.1                      50.6
 2 AL             32.2         23.5         11.4                      50.6
 3 AR             32.2         23.5         20                        50.6
 4 AZ             32.2         23.5          3.53                     50.6
 5 CA             32.2         23.5          1.57                     50.6
 6 CO             32.2         23.5         12.2                      50.6
 7 CT             32.2         23.5         24.3                      50.6
 8 DC             32.2         23.5         59.6                      50.6
```

```
 9 DE              32.2         23.5         54.1                      50.6
10 FL              32.2         23.5          0.784                    50.6
# i 42 more rows
# i 1 more variable: missing_booster <dbl>
```

```r
# Plot missing data by state for each variable
missing_by_state_long <- missing_by_state %>%
  pivot_longer(
    cols = c(missing_cases, missing_hosp, missing_deaths, missing_series_complete, missing_bo
    names_to = "variable",
    values_to = "percent_missing"
  )

ggplot(missing_by_state_long, aes(x = state, y = percent_missing, fill = variable)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(
    title = "Percentage of Missing Data by State and Variable",
    x = "State",
    y = "Percentage of Missing Data",
    fill = "Variable"
  ) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  theme_minimal()
```



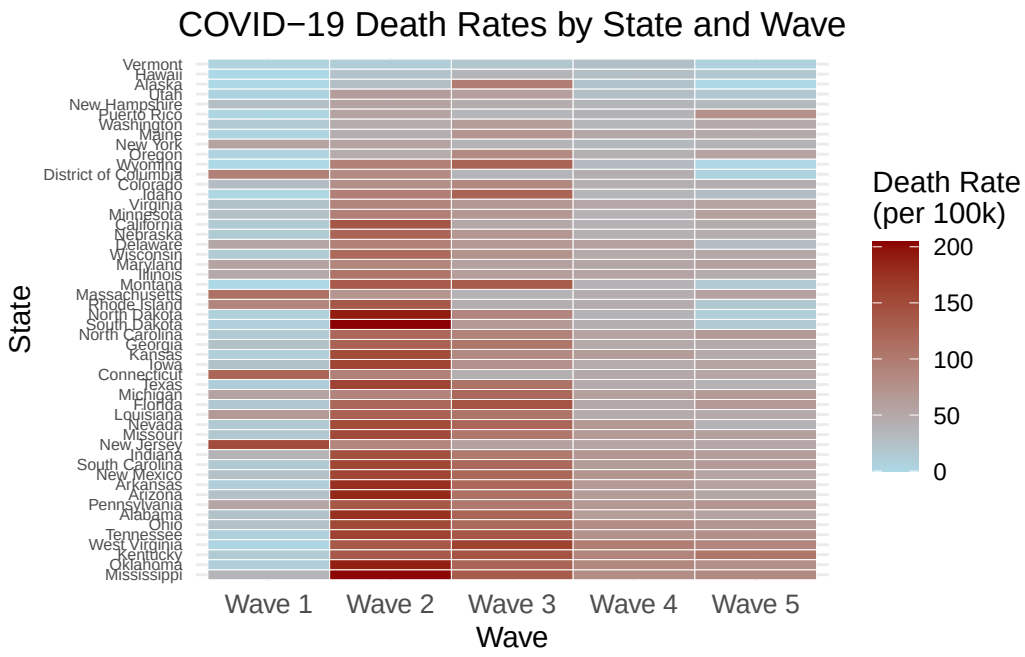
```r
dat <- dat %>%
  mutate(wave = case_when(
    date >= as.Date("2020-01-01") & date <= as.Date("2020-06-30") ~ "Wave 1",
    date >= as.Date("2020-07-01") & date <= as.Date("2021-02-28") ~ "Wave 2",
    date >= as.Date("2021-03-01") & date <= as.Date("2021-12-31") ~ "Wave 3",
    date >= as.Date("2022-01-01") & date <= as.Date("2022-06-30") ~ "Wave 4",
    date >= as.Date("2022-07-01") & date <= as.Date("2024-12-31") ~ "Wave 5",
    TRUE ~ NA_character_
  ))

# Calculate death rates by state and wave
death_rates <- dat %>%
  group_by(state_name, wave) %>%
  summarize(
    total_deaths = sum(deaths, na.rm = TRUE),
    population = max(population, na.rm = TRUE)
  ) %>%
  mutate(death_rate_per_100k = (total_deaths / population) * 100000)
```

```
`summarise()` has grouped output by 'state_name'. You can override using the
`.groups` argument.
```

```r
# View the death rates
print(death_rates)
```

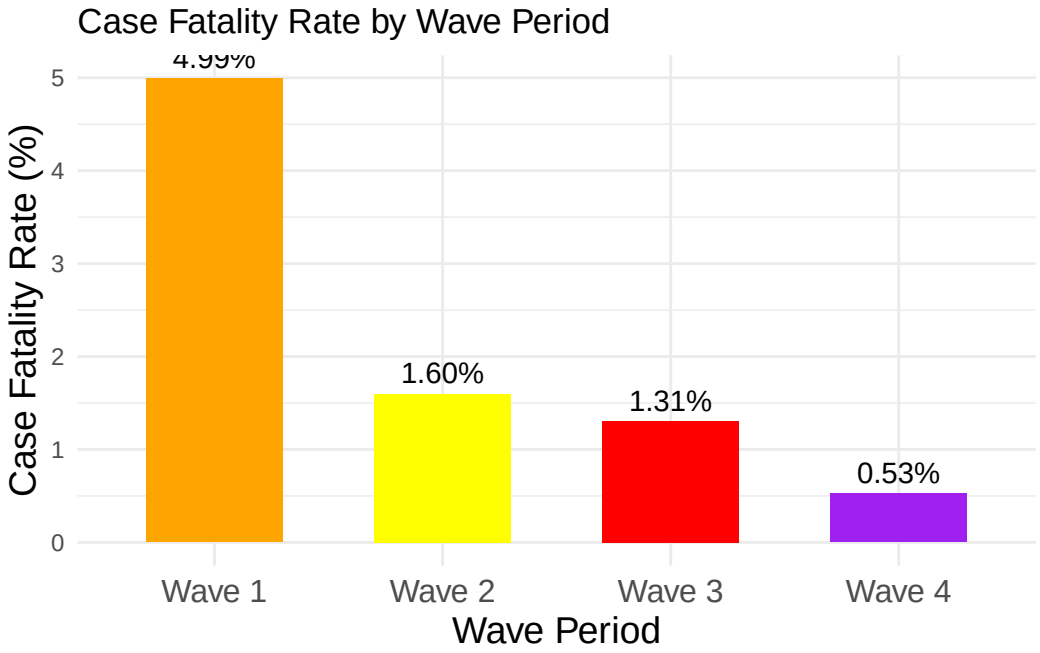
```
# A tibble: 260 x 5
# Groups:   state_name [52]
   state_name wave   total_deaths population death_rate_per_100k
   <chr>      <chr>         <dbl>      <dbl>               <dbl>
 1 Alabama    Wave 1         1168    5024803                23.2
 2 Alabama    Wave 2         8905    5039877               177.
 3 Alabama    Wave 3         6273    5039877               124.
 4 Alabama    Wave 4         3184    5073903                62.8
 5 Alabama    Wave 5         2911    5108468                57.0
 6 Alaska     Wave 1            0     732441                 0
 7 Alaska     Wave 2          194     732673                26.5
 8 Alaska     Wave 3          713     732673                97.3
 9 Alaska     Wave 4          159     733276                21.7
10 Alaska     Wave 5            0     733406                 0
# i 250 more rows
```

```r
library(ggplot2)

# Create a heatmap of death rates by state and wave
ggplot(death_rates, aes(x = wave, y = reorder(state_name, -death_rate_per_100k), fill = deatl
  geom_tile(color = "white") +
  scale_fill_gradient(low = "lightblue", high = "darkred") +
  labs(
    title = "COVID-19 Death Rates by State and Wave",
    x = "Wave",
    y = "State",
    fill = "Death Rate\n(per 100k)"
  ) +
  theme_minimal() +
  theme(
    axis.text.y = element_text(size = 6),
    axis.text.x = element_text(size = 10),
    plot.title = element_text(hjust = 0.5)
  )
```



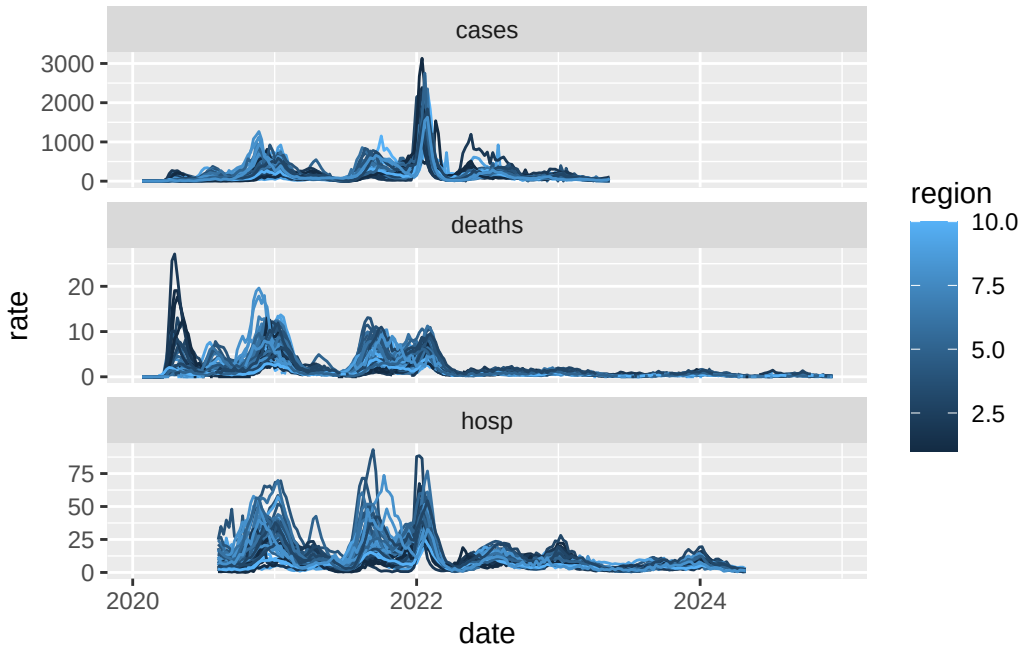

```r
# Use the existing wave definitions in your dataset
dat <- dat %>%
  mutate(wave = case_when(
    date >= as.Date("2020-01-01") & date <= as.Date("2020-06-30") ~ "Wave 1",
```

```
    date >= as.Date("2020-07-01") & date <= as.Date("2021-02-28") ~ "Wave 2",
    date >= as.Date("2021-03-01") & date <= as.Date("2021-12-31") ~ "Wave 3",
    date >= as.Date("2022-01-01") & date <= as.Date("2022-06-30") ~ "Wave 4",
    date >= as.Date("2022-07-01") & date <= as.Date("2024-12-31") ~ "Wave 5",
    TRUE ~ NA_character_
  ))

# Compute case fatality rates for Waves 1 to 4 only
cfr_data <- dat %>%
  filter(wave %in% c("Wave 1", "Wave 2", "Wave 3", "Wave 4")) %>%
  group_by(wave) %>%
  summarize(
    total_cases = sum(cases, na.rm = TRUE),
    total_deaths = sum(deaths, na.rm = TRUE),
    case_fatality_rate = ifelse(total_cases > 0, total_deaths / total_cases * 100, NA), # Av
    .groups = "drop"
  )

# Visualize CFR across Waves 1 to 4
library(ggplot2)

ggplot(cfr_data, aes(x = wave, y = case_fatality_rate, fill = wave)) +
  geom_bar(stat = "identity", width = 0.6) +  # Set bar width for better spacing
  geom_text(aes(label = sprintf("%.2f%%", case_fatality_rate)), vjust = -0.5, size = 4) +  #
  labs(
    title = "Case Fatality Rate by Wave Period",
    x = "Wave Period",
    y = "Case Fatality Rate (%)"
  ) +
  scale_fill_manual(values = c("Wave 1" = "orange", "Wave 2" = "yellow", "Wave 3" = "red", "
  theme_minimal() +
  theme(
    legend.position = "none",  # Remove legend since fill is self-explanatory
    axis.text.x = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```




```
p <- dat |> mutate(cases =  cases/population*100000,
                   hosp = hosp/population*100000,
                  deaths = deaths/population*100000) |>
  select(date, cases, hosp, deaths, state, region) |>
  pivot_longer(c(cases, deaths, hosp), values_to = "rate", names_to = "outcome") |>
  ggplot(aes(date, rate, color = region, group = state)) +
  geom_line() +
  facet_wrap(~outcome, nrow = 3, scales = "free_y")
print(p)
```

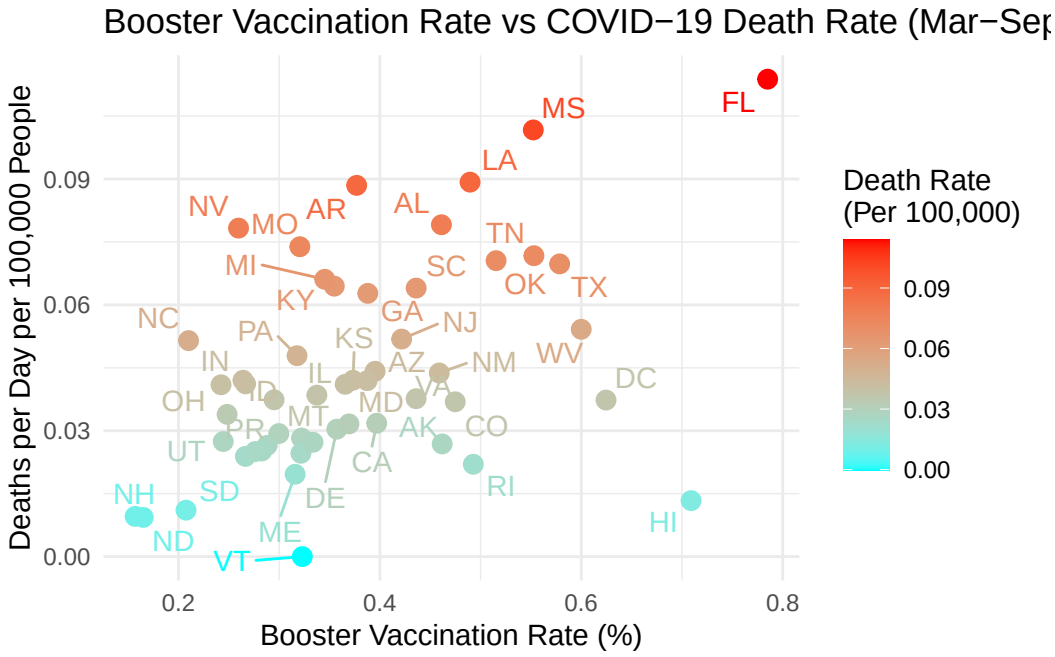
```
Warning: Removed 1664 rows containing missing values or values outside the scale range
(`geom_line()`).
```



i

```
#Figure 6
# Calculate deaths per day per 100,000 people and booster rate
time_period_booster <- dat |>
  filter(date >= "2021-03-01" & date <= "2021-09-01") |>
  group_by(state) |>
  summarize(
    deaths_per_100k = sum(as.numeric(deaths), na.rm = TRUE) /
                      sum(as.numeric(population), na.rm = TRUE) * 100000 / n_distinct(date),
    booster_rate = max(as.numeric(booster) / as.numeric(population) * 100, na.rm = TRUE)
  )

# Scatter plot with color indicating death rate and ggrepel for state labels
ggplot(time_period_booster, aes(x = booster_rate, y = deaths_per_100k, label = state, color =
  geom_point(size = 3) +
  geom_text_repel() +
  scale_color_gradient(low = "cyan", high = "red") + # Gradient from low to high
  labs(
    title = "Booster Vaccination Rate vs COVID-19 Death Rate (Mar-Sep 2021)",
    x = "Booster Vaccination Rate (%)",
    y = "Deaths per Day per 100,000 People",
    color = "Death Rate\n(Per 100,000)"
  ) +
```

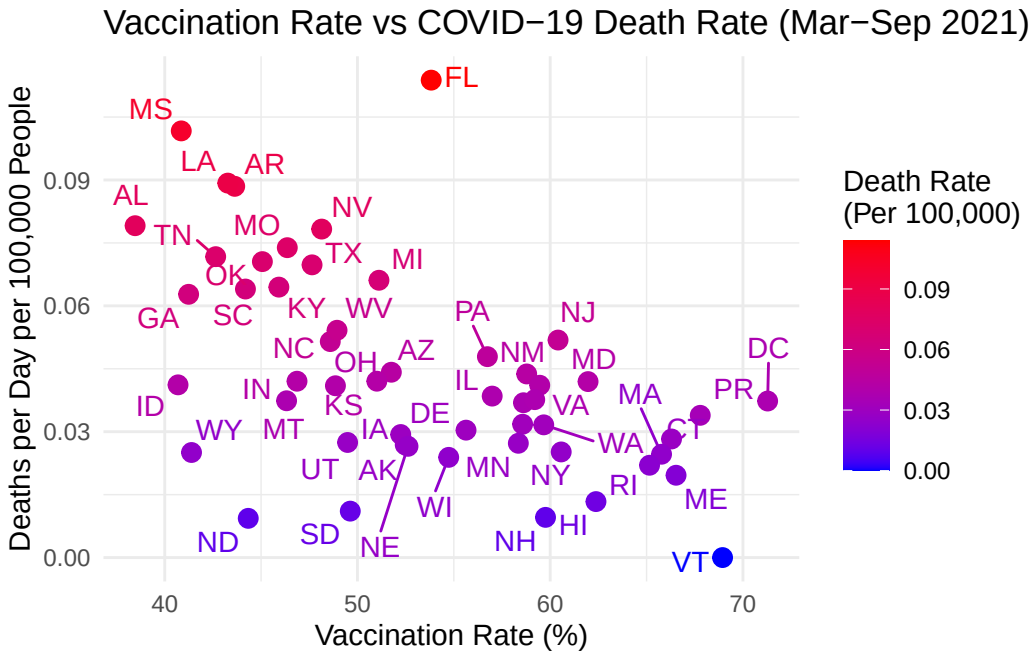
```
  theme_minimal()
```




```
#Figure 5
# Calculate deaths per day per 100,000 people and vaccination rate
time_period_1 <- dat |>
  filter(date >= "2021-03-01" & date <= "2021-09-01") |>
  group_by(state) |>
  summarize(
    deaths_per_100k = sum(as.numeric(deaths), na.rm = TRUE) /
                      sum(as.numeric(population), na.rm = TRUE) * 100000 / n_distinct(date),
    vax_rate = max(as.numeric(series_complete) / as.numeric(population) * 100, na.rm = TRUE)
  )

# Scatter plot with color indicating death rate and ggrepel for state labels
ggplot(time_period_1, aes(x = vax_rate, y = deaths_per_100k, label = state, color = deaths_p
  geom_point(size = 3) +
  geom_text_repel() +
  scale_color_gradient(low = "blue", high = "red") + # Gradient from low to high
  labs(
    title = "Vaccination Rate vs COVID-19 Death Rate (Mar-Sep 2021)",
    x = "Vaccination Rate (%)",
    y = "Deaths per Day per 100,000 People",
    color = "Death Rate\n(Per 100,000)"
```

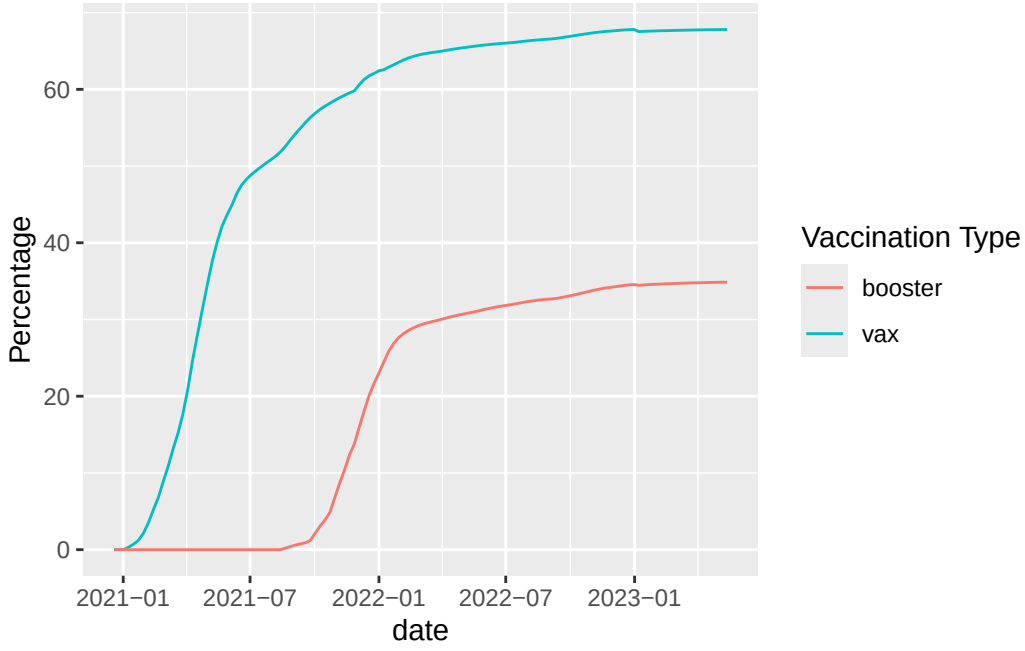
```
  ) +
  theme_minimal()
```

```
Warning: ggrepel: 3 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```




```
#Figure 2
# Vaccination Progress Over Time in US
dat |>
  filter(!is.na(series_complete),
         !is.na(booster),
         !is.na(population)) |>
   select(date,state, population, series_complete, booster) |>
  mutate(booster = as.numeric(booster))|>
  mutate(series_complete= as.numeric(series_complete)) |>

  group_by(date) |>
  mutate(
    vax_perc = (sum(series_complete) / sum(population)) * 100,
    booster_perc = (sum(booster) / sum(population)) * 100
  )|>
```

```
  ggplot(aes(x=date)) +
  geom_line(aes(y=vax_perc, colour = "vax")) +
  geom_line(aes(y = booster_perc, colour = "booster")) +
  labs( y = "Percentage", color = "Vaccination Type")
```




```
theme_minimal()
```

```
List of 136
 $ line                            :List of 6
  ..$ colour       : chr "black"
  ..$ linewidth    : num 0.5
  ..$ linetype     : num 1
  ..$ lineend      : chr "butt"
  ..$ arrow        : logi FALSE
  ..$ inherit.blank: logi TRUE
  ..- attr(*, "class")= chr [1:2] "element_line" "element"
 $ rect                            :List of 5
  ..$ fill         : chr "white"
  ..$ colour       : chr "black"
  ..$ linewidth    : num 0.5
  ..$ linetype     : num 1
  ..$ inherit.blank: logi TRUE
```

```
 ..- attr(*, "class")= chr [1:2] "element_rect" "element"
$ text                            :List of 11
 ..$ family       : chr ""
 ..$ face         : chr "plain"
 ..$ colour       : chr "black"
 ..$ size         : num 11
 ..$ hjust        : num 0.5
 ..$ vjust        : num 0.5
 ..$ angle        : num 0
 ..$ lineheight   : num 0.9
 ..$ margin       : 'margin' num [1:4] 0points 0points 0points 0points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : logi FALSE
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ title                           : NULL
$ aspect.ratio                    : NULL
$ axis.title                      : NULL
$ axis.title.x                    :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : NULL
 ..$ vjust        : num 1
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 2.75points 0points 0points 0points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.title.x.top                :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : NULL
 ..$ vjust        : num 0
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 0points 0points 2.75points 0points
 .. ..- attr(*, "unit")= int 8
```

```
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.title.x.bottom             : NULL
$ axis.title.y                    :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : NULL
 ..$ vjust        : num 1
 ..$ angle        : num 90
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 0points 2.75points 0points 0points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.title.y.left               : NULL
$ axis.title.y.right              :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : NULL
 ..$ vjust        : num 1
 ..$ angle        : num -90
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.75points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.text                       :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : chr "grey30"
 ..$ size         : 'rel' num 0.8
 ..$ hjust        : NULL
 ..$ vjust        : NULL
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : NULL
```

```
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.text.x                    :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : NULL
 ..$ vjust        : num 1
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 2.2points 0points 0points 0points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.text.x.top                :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : NULL
 ..$ vjust        : num 0
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 0points 0points 2.2points 0points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.text.x.bottom             : NULL
$ axis.text.y                    :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : num 1
 ..$ vjust        : NULL
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 0points
 .. ..- attr(*, "unit")= int 8
```

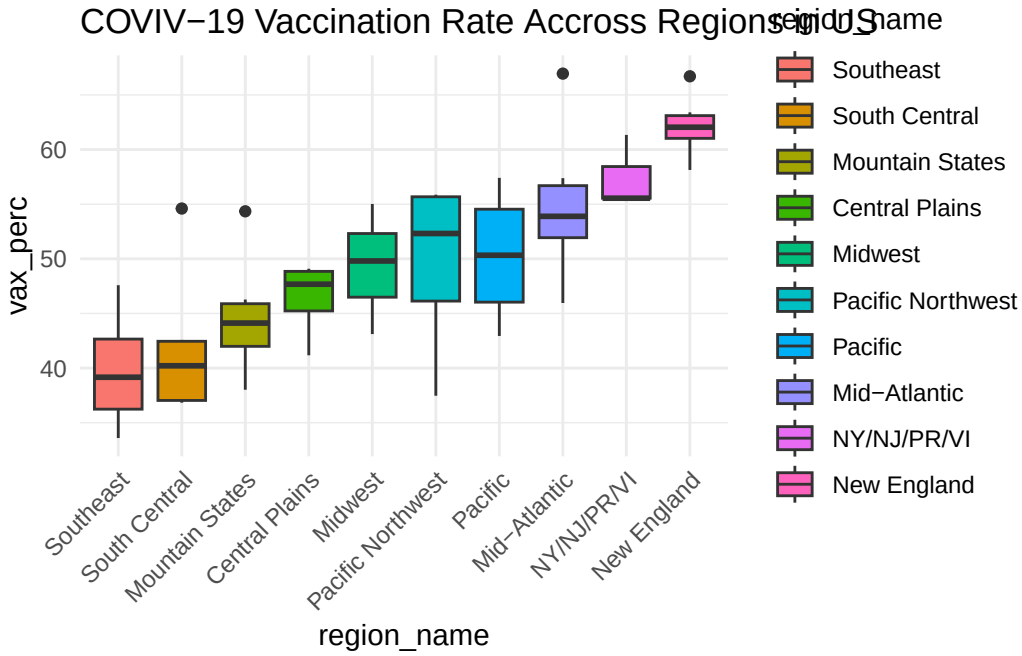
```
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.text.y.left                : NULL
$ axis.text.y.right               :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : num 0
 ..$ vjust        : NULL
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.2points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.text.theta                 : NULL
$ axis.text.r                     :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : num 0.5
 ..$ vjust        : NULL
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 2.2points
 .. ..- attr(*, "unit")= int 8
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ axis.ticks                      : list()
 ..- attr(*, "class")= chr [1:2] "element_blank" "element"
$ axis.ticks.x                    : NULL
$ axis.ticks.x.top                : NULL
$ axis.ticks.x.bottom             : NULL
$ axis.ticks.y                    : NULL
$ axis.ticks.y.left               : NULL
$ axis.ticks.y.right              : NULL
$ axis.ticks.theta                : NULL
$ axis.ticks.r                    : NULL
```

```
$ axis.minor.ticks.x.top          : NULL
$ axis.minor.ticks.x.bottom       : NULL
$ axis.minor.ticks.y.left         : NULL
$ axis.minor.ticks.y.right        : NULL
$ axis.minor.ticks.theta          : NULL
$ axis.minor.ticks.r              : NULL
$ axis.ticks.length               : 'simpleUnit' num 2.75points
 ..- attr(*, "unit")= int 8
$ axis.ticks.length.x             : NULL
$ axis.ticks.length.x.top         : NULL
$ axis.ticks.length.x.bottom      : NULL
$ axis.ticks.length.y             : NULL
$ axis.ticks.length.y.left        : NULL
$ axis.ticks.length.y.right       : NULL
$ axis.ticks.length.theta         : NULL
$ axis.ticks.length.r             : NULL
$ axis.minor.ticks.length         : 'rel' num 0.75
$ axis.minor.ticks.length.x       : NULL
$ axis.minor.ticks.length.x.top   : NULL
$ axis.minor.ticks.length.x.bottom: NULL
$ axis.minor.ticks.length.y       : NULL
$ axis.minor.ticks.length.y.left  : NULL
$ axis.minor.ticks.length.y.right : NULL
$ axis.minor.ticks.length.theta   : NULL
$ axis.minor.ticks.length.r       : NULL
$ axis.line                       : list()
 ..- attr(*, "class")= chr [1:2] "element_blank" "element"
$ axis.line.x                     : NULL
$ axis.line.x.top                 : NULL
$ axis.line.x.bottom              : NULL
$ axis.line.y                     : NULL
$ axis.line.y.left                : NULL
$ axis.line.y.right               : NULL
$ axis.line.theta                 : NULL
$ axis.line.r                     : NULL
$ legend.background               : list()
 ..- attr(*, "class")= chr [1:2] "element_blank" "element"
$ legend.margin                   : 'margin' num [1:4] 5.5points 5.5points 5.5points 5.5poi
 ..- attr(*, "unit")= int 8
$ legend.spacing                  : 'simpleUnit' num 11points
 ..- attr(*, "unit")= int 8
$ legend.spacing.x                : NULL
$ legend.spacing.y                : NULL
```

```
$ legend.key                      : list()
 ..- attr(*, "class")= chr [1:2] "element_blank" "element"
$ legend.key.size                 : 'simpleUnit' num 1.2lines
 ..- attr(*, "unit")= int 3
$ legend.key.height               : NULL
$ legend.key.width                : NULL
$ legend.key.spacing              : 'simpleUnit' num 5.5points
 ..- attr(*, "unit")= int 8
$ legend.key.spacing.x            : NULL
$ legend.key.spacing.y            : NULL
$ legend.frame                    : NULL
$ legend.ticks                    : NULL
$ legend.ticks.length             : 'rel' num 0.2
$ legend.axis.line                : NULL
$ legend.text                     :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : 'rel' num 0.8
 ..$ hjust        : NULL
 ..$ vjust        : NULL
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : NULL
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ legend.text.position            : NULL
$ legend.title                    :List of 11
 ..$ family       : NULL
 ..$ face         : NULL
 ..$ colour       : NULL
 ..$ size         : NULL
 ..$ hjust        : num 0
 ..$ vjust        : NULL
 ..$ angle        : NULL
 ..$ lineheight   : NULL
 ..$ margin       : NULL
 ..$ debug        : NULL
 ..$ inherit.blank: logi TRUE
 ..- attr(*, "class")= chr [1:2] "element_text" "element"
$ legend.title.position           : NULL
$ legend.position                 : chr "right"
```

```
 $ legend.position.inside          : NULL
 $ legend.direction                : NULL
 $ legend.byrow                    : NULL
 $ legend.justification            : chr "center"
 $ legend.justification.top        : NULL
 $ legend.justification.bottom     : NULL
 $ legend.justification.left       : NULL
 $ legend.justification.right      : NULL
 $ legend.justification.inside     : NULL
 $ legend.location                 : NULL
 $ legend.box                      : NULL
 $ legend.box.just                 : NULL
 $ legend.box.margin               : 'margin' num [1:4] 0cm 0cm 0cm 0cm
  ..- attr(*, "unit")= int 1
 $ legend.box.background           : list()
  ..- attr(*, "class")= chr [1:2] "element_blank" "element"
 $ legend.box.spacing              : 'simpleUnit' num 11points
  ..- attr(*, "unit")= int 8
  [list output truncated]
 - attr(*, "class")= chr [1:2] "theme" "gg"
 - attr(*, "complete")= logi TRUE
 - attr(*, "validate")= logi TRUE
```

```r
# Figure 4
# Exploring Regional Variations: Insights on Vaccination Differences Across Regions

dat |>
  filter(mmwr_week == epiweek(as_date("2021-07-01")),
         mmwr_year == epiyear(as_date("2021-07-01"))) |>
  mutate(booster = as.numeric(booster)) |>
  mutate(series_complete = as.numeric(series_complete)) |>
  mutate(vax_perc = (series_complete / population) * 100,
         region_name = reorder(as.factor(region_name), vax_perc)) |>
  ggplot(aes(x = region_name, y = vax_perc, fill = region_name)) +
  geom_boxplot() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1)) +
  ggtitle("COVIV-19 Vaccination Rate Accross Regions in US")
```


COVIV−19 Vaccination Rate Accross Regions in U.S
vax_perc
60
50
40
region_name
Southeast
South Central
Mountain States
Central Plains
Midwest
Pacific Northwest
Pacific
Mid−Atlantic
NY/NJ/PR/VI
New England
region_name
Southeast
South Central
Mountain States
Central Plains
Midwest
Pacific Northwest
Pacific
Mid−Atlantic
NY/NJ/PR/VI
New England