Time Series and Geographic Maps

INFO 526 Data Analysis and Visualization

Adriana Picoral

Covid in the US data

For this part of the tutorial we will be using COVID-19 Estimated Inpatient Beds Occupied by COVID-19 Patients by State. I downloaded the file from the data.gov website and added to my data folder.

```
library(tidyverse)
covid_19_inpatient_beds <- read_csv("data/COVID-19_Estimated_Inpatient_Beds_Occupied_by_COVID-19_Patien
```

First we need to ensure the date variable is in actual date data format.

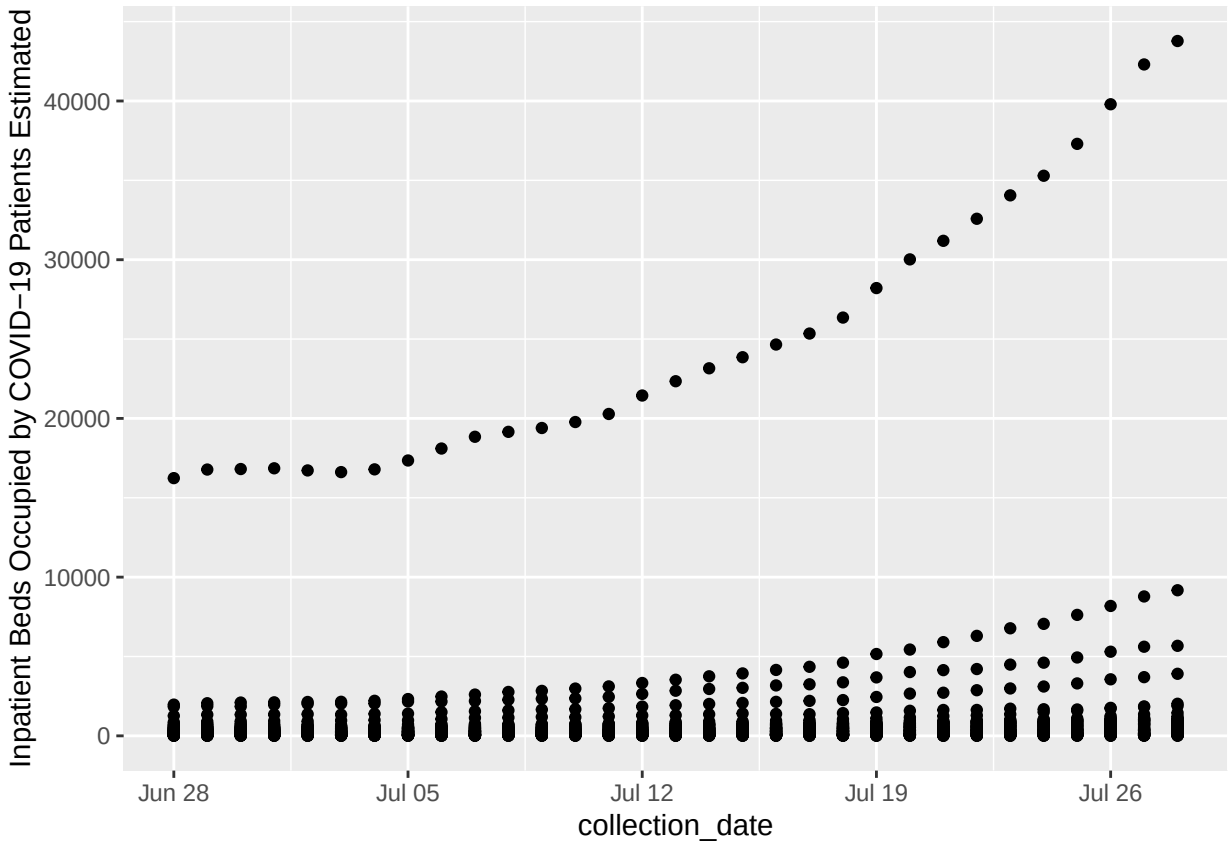
```
covid_19_inpatient_beds <- covid_19_inpatient_beds %>%
  mutate(collection_date = as.Date(collection_date))

glimpse(covid_19_inpatient_beds)
```

```
## Rows: 1,612
## Columns: 12
## $ state                                                                <chr> ~
## $ collection_date                                                      <date> ~
## $ `Inpatient Beds Occupied by COVID-19 Patients Estimated`             <dbl> ~
## $ `Count LL`                                                           <dbl> ~
## $ `Count UL`                                                           <dbl> ~
## $ `Percentage of Inpatient Beds Occupied by COVID-19 Patients Estimated` <dbl> ~
## $ `Percentage LL`                                                      <dbl> ~
## $ `Percentage UL`                                                      <dbl> ~
## $ `Total Inpatient Beds`                                               <dbl> ~
## $ `Total LL`                                                           <dbl> ~
## $ `Total UL`                                                           <dbl> ~
## $ geocoded_state                                                       <chr> ~
```

The second thing we need in addition to a date variable is an aggregated value. Our data already provides that for us, with the variable `Inpatient Beds Occupied by COVID-19 Patients Estimated`.

The simplest way to plot a time series is by using `geom_point()`. The x axis is usually mapped to the date variable, and the y variable to the numeric variable.

```
covid_19_inpatient_beds %>%
  ggplot(aes(x = collection_date,
             y = `Inpatient Beds Occupied by COVID-19 Patients Estimated`)) +
  geom_point()
```




```
covid_19_inpatient_beds %>%
  distinct(state) %>%
  nrow()
```

```
## [1] 52
```

DC is one the extra categories under state. The other one must the a total for the country, which shows are the much higher line in the plot above.

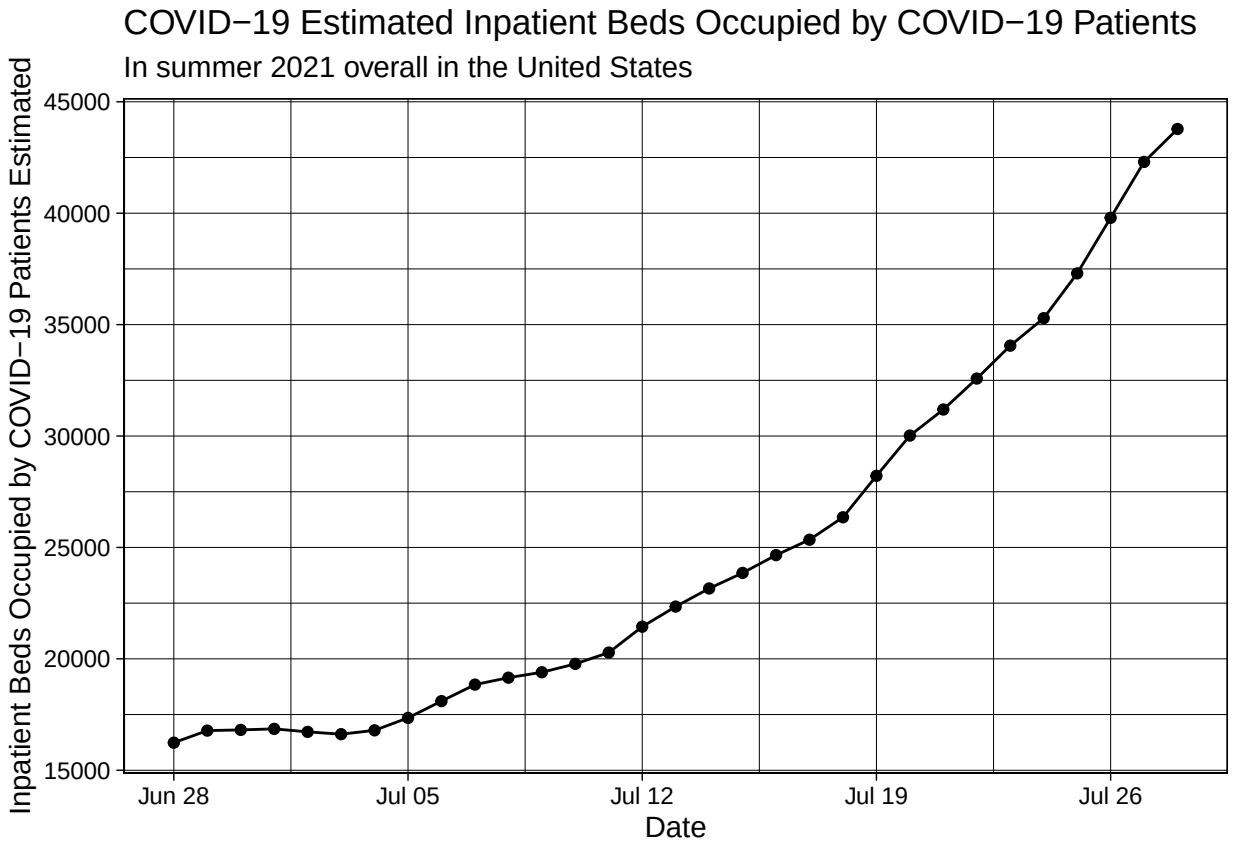
```
covid_19_inpatient_beds %>%
  group_by(state) %>%
  summarize(max_beds = max(`Inpatient Beds Occupied by COVID-19 Patients Estimated`)) %>%
  slice_max(n = 1, order_by = max_beds)
```

```
## # A tibble: 1 x 2
##   state max_beds
##   <chr>    <dbl>
## 1 CW       43779
```

The code "CW" must mean country wide. let's plot only that.

```
covid_19_inpatient_beds %>%
  filter(state == "CW") %>%
  ggplot(aes(x = collection_date,
             y = `Inpatient Beds Occupied by COVID-19 Patients Estimated`)) +
```

```
  geom_point() +
  geom_line() +
  labs(title = "COVID-19 Estimated Inpatient Beds Occupied by COVID-19 Patients",
       subtitle = "In summer 2021 overall in the United States",
       x = "Date") +
  theme_linedraw()
```



Plotting a map fo the US

In this part of the tutorial, we will be plotting US maps, so we need to call (and install if you don't have it yet) the `usmap` library.

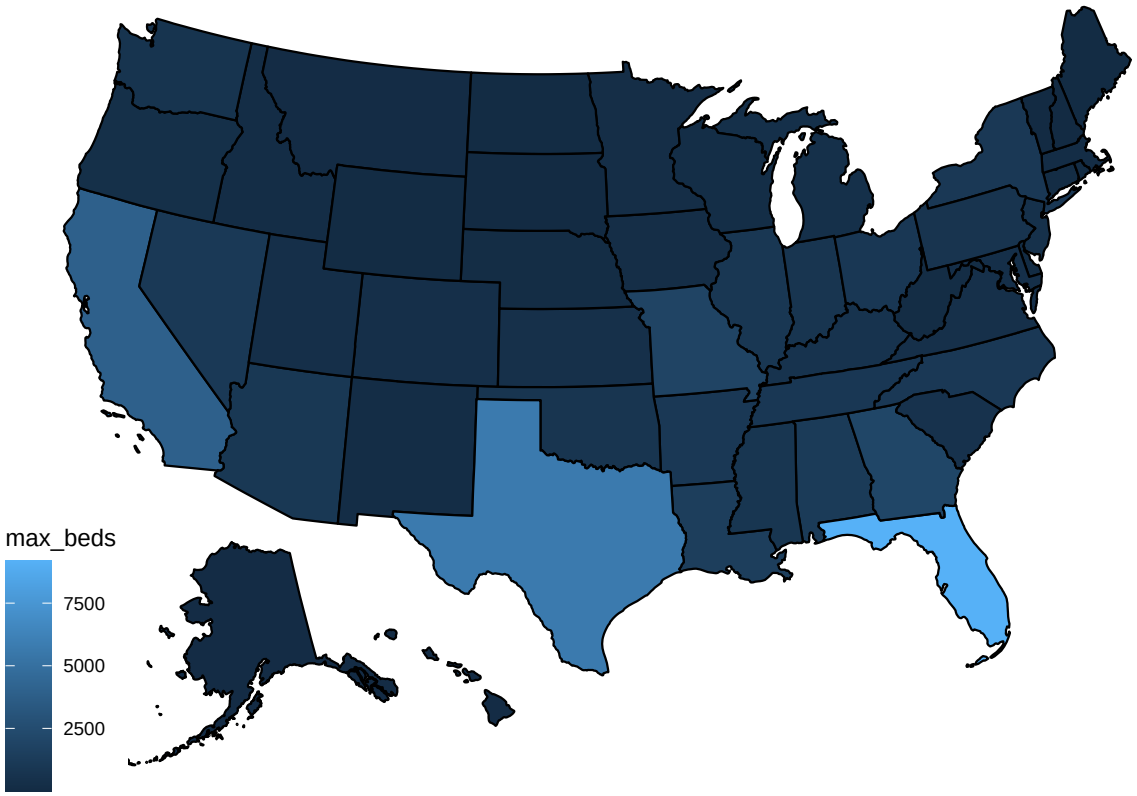
```
library(usmap)
```

Since for a static map we will not be plotting time, we will get the maximun number of inpatient beds per state to plot.

```
max_beds_per_state <- covid_19_inpatient_beds %>%
  filter(state != "CW") %>%
  group_by(state) %>%
  summarize(max_beds = max(`Inpatient Beds Occupied by COVID-19 Patients Estimated`))
```

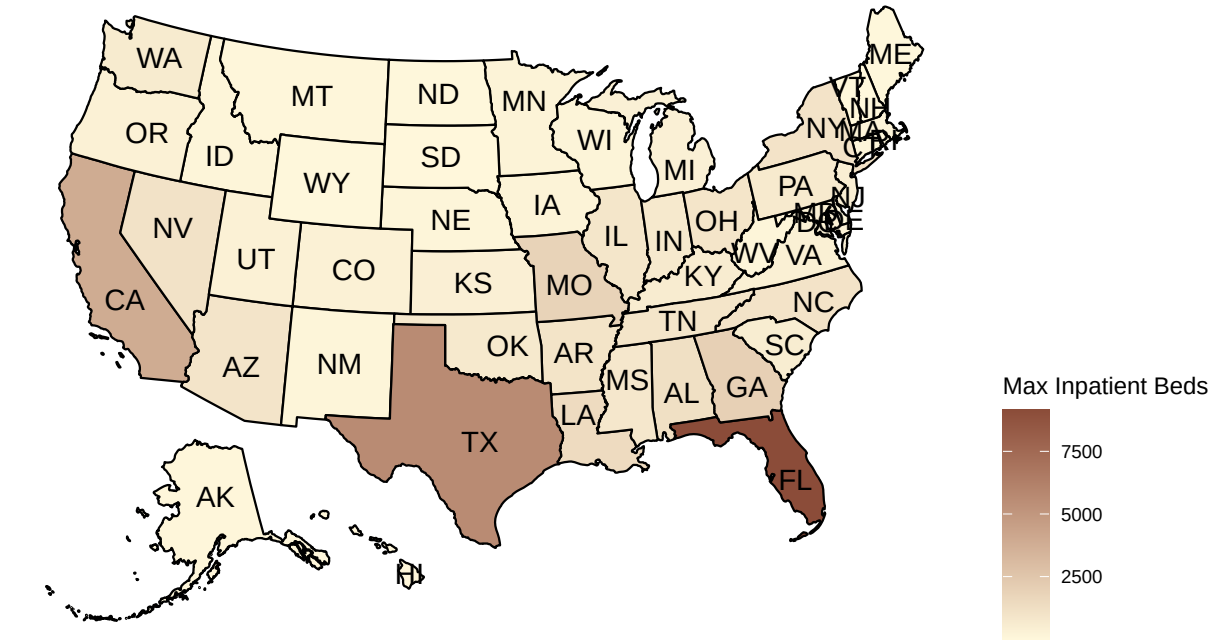
To plot a US map is simple with `plot_usmap` from the `usmap` library. Unfortunately, the name of the column (i.e, "max_beds") needs to go between quotes with this function (which is not the case with ggplot).

```
max_beds_per_state %>%
  plot_usmap(data = .,
             values = "max_beds")
```



We can improve on this a lot. Let's move the legend to the right, and change the low and high colors in the color scheme (with dark meaning higher numbers and light color meaning lower numbers). You can check the list of R color names for a reference on what names we can use.

```
# move legend to the right
# change scale_fill name and colors for low and high colors
# add a title
max_beds_per_state %>%
  plot_usmap(data = .,
             values = "max_beds",
             labels = TRUE) +
  theme(legend.position = "right") +
  scale_fill_continuous(name = "Max Inpatient Beds",
                        low = "cornsilk",
                        high = "salmon4") +
  labs(title = "Max Number of Inpatient Beds Occupied by COVID-19 Patients Estimated",
       subtitle = "In Summer 2021")
```



To add the actual label for each state, we need to get coordinates first (i.e, our centroids for each state) and then combine this information with our data. I added the centroid data found at the usmapdata GitHub repo to the data folder.

```
# get centroids
centroid_labels <- read_csv("data/us_states_centroids.csv") %>%
  rename(state = abbr) %>%
  mutate(fips = stringr::str_pad(fips, 2, pad = "0"))
```

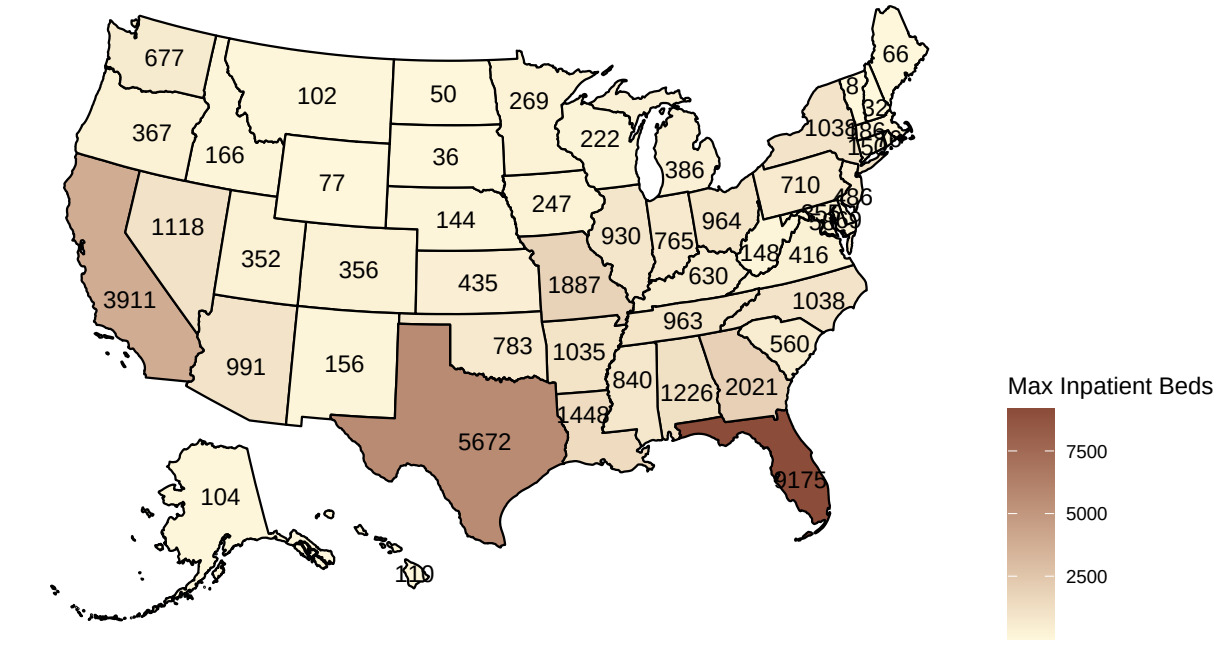
We can now join centroids with our `total_per_state` dataframe.

```
# join max_beds_per_state data to centroids
max_beds_per_state_complete <- max_beds_per_state %>%
  left_join(centroid_labels)
```

We need to start our plot with the original data that contains only two columns (a usmap restriction) and then change the data for the `geom_text()` function.

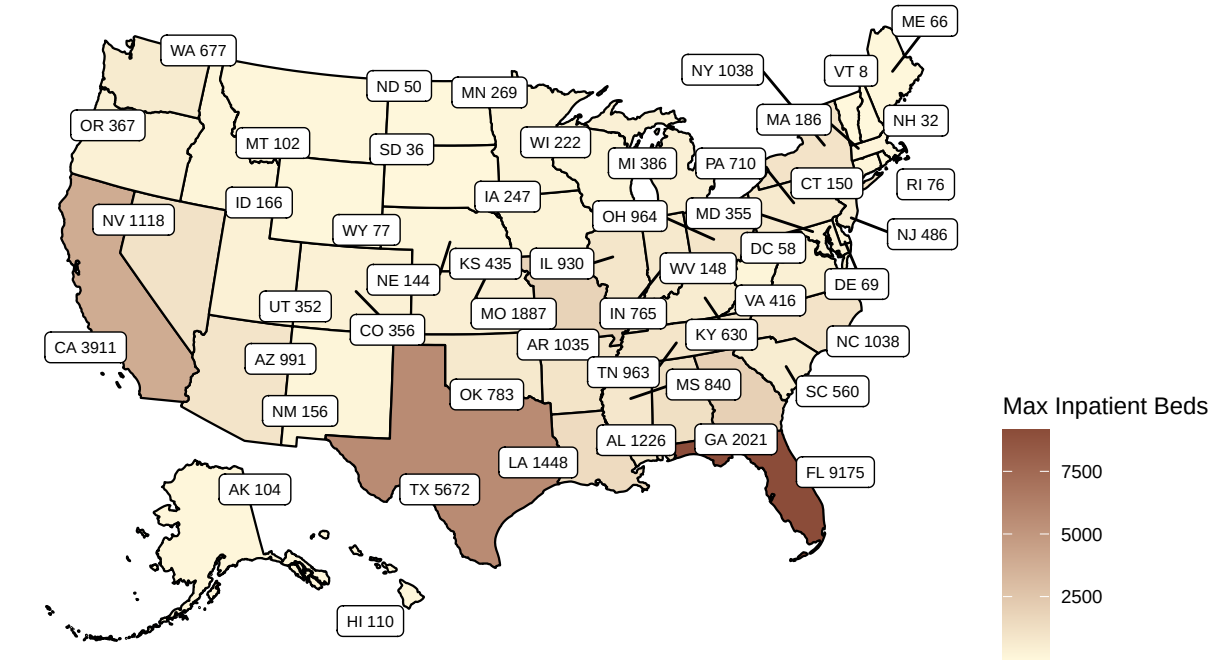
```
# add geom_text with rounded values for doses per person
max_beds_per_state %>%
  plot_usmap(data = .,
             values = "max_beds") +
  theme(legend.position = "right") +
  scale_fill_continuous(name = "Max Inpatient Beds",
                        low = "cornsilk",
                        high = "salmon4") +
  ggtitle("COVID-19 Vaccine Allocations per Person") +
  labs(title = "Max Number of Inpatient Beds Occupied by COVID-19 Patients Estimated",
       subtitle = "In Summer 2021") +
```

```
  geom_text(data = max_beds_per_state_complete,
            aes(x = x,
                y = y,
                label = max_beds),
            size = 3)
```



There's a lot of overlap. We can use another library, `ggrepel` that contains the function `geom_label_repel` that repels labels from each other, so there's no label overlapping.

```
# load ggrepel library
library(ggrepel)

# use geom_label_repel
# paste state abbreviation to label
max_beds_per_state %>%
  plot_usmap(data = .,
             values = "max_beds") +
  theme(legend.position = "right") +
  scale_fill_continuous(name = "Max Inpatient Beds",
                        low = "cornsilk",
                        high = "salmon4") +
  ggtitle("COVID-19 Vaccine Allocations per Person") +
  labs(title = "Max Number of Inpatient Beds Occupied by COVID-19 Patients Estimated",
       subtitle = "In Summer 2021") +
  geom_label_repel(data = max_beds_per_state_complete,
             aes(x = x,
                 y = y,
                 label = paste(state, max_beds)),
             size = 2)
```



Plotting World Maps

For this tutorial, we are visualizing how many people have been vaccinated against COVID-19 in the world, using a dataset from Kaggle.

```
covid_vaccines_world <- read_csv("data/country_vaccinations.csv")
```

We will start with getting the maximum of `people_vaccinated_per_hundred`, since there are multiple weeks in the data and we are interested in the most current value.

```
people_vaccinated <- covid_vaccines_world %>%
  filter(!is.na(people_vaccinated_per_hundred)) %>%
  group_by(country) %>%
  summarize(people_vaccinated_per_hundred = max(people_vaccinated_per_hundred))
```

Let's inspect our results.

```
people_vaccinated %>%
  head()
```

```
## # A tibble: 6 x 2
##   country     people_vaccinated_per_hundred
##   <chr>                               <dbl>
## 1 Afghanistan                          12.4
## 2 Albania                              44.2
## 3 Algeria                              16.7
## 4 Andorra                              74.7
```

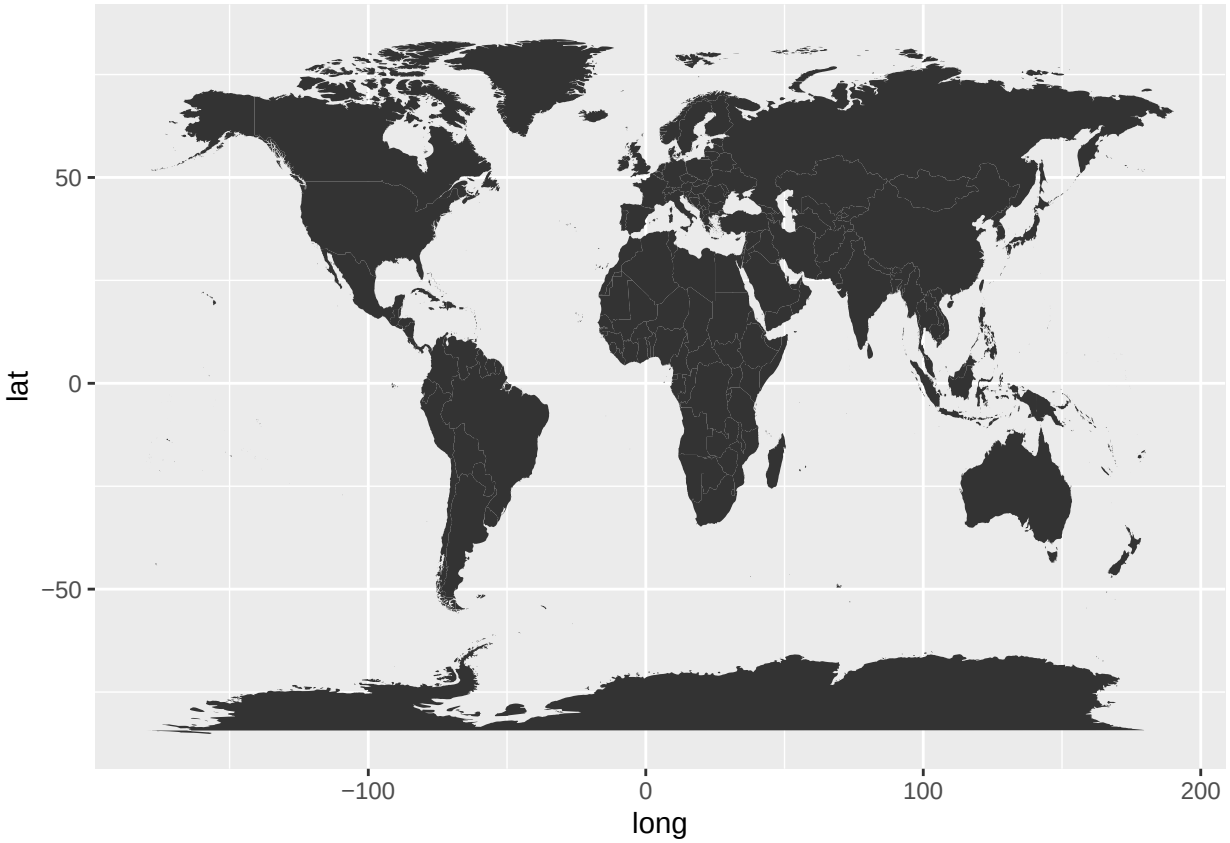
```
## 5 Angola                                31.8
## 6 Anguilla                              69.0
```

To draw a world map, we will first get the data we need (with latitude and longitude values) using the `map_data` function.

```
world <- map_data("world")
```

We can now plot the data using `geom_map`.

```
world %>%
  ggplot(aes(x = long,
             y = lat,
             map_id = region)) +
  geom_map(map = world)
```



We want the percentage vaccinated from our `people_vaccinated` dataset. However, the country names in `people_vaccinated` might not match what we have for region in `world`. Let's check what country names are not in region in the world dataset.

```
people_vaccinated %>%
  filter(!(country %in% unique(world$region))) %>%
  pull(country)
```

```
##  [1] "Antigua and Barbuda"              "Bonaire Sint Eustatius and Saba"
```

```
##  [3] "British Virgin Islands"          "Congo"
##  [5] "Cote d'Ivoire"                   "Czechia"
##  [7] "Democratic Republic of Congo"    "England"
##  [9] "Eswatini"                        "Faeroe Islands"
## [11] "Gibraltar"                       "Hong Kong"
## [13] "Macao"                           "Northern Cyprus"
## [15] "Northern Ireland"                "Pitcairn"
## [17] "Saint Kitts and Nevis"           "Saint Vincent and the Grenadines"
## [19] "Scotland"                        "Sint Maarten (Dutch part)"
## [21] "Timor"                           "Tokelau"
## [23] "Trinidad and Tobago"             "Tuvalu"
## [25] "United Kingdom"                  "United States"
## [27] "Wales"
```

That's quite a few countries. We can change the country name in `people_vaccinated` to match what we have in `world` using `case_when`.

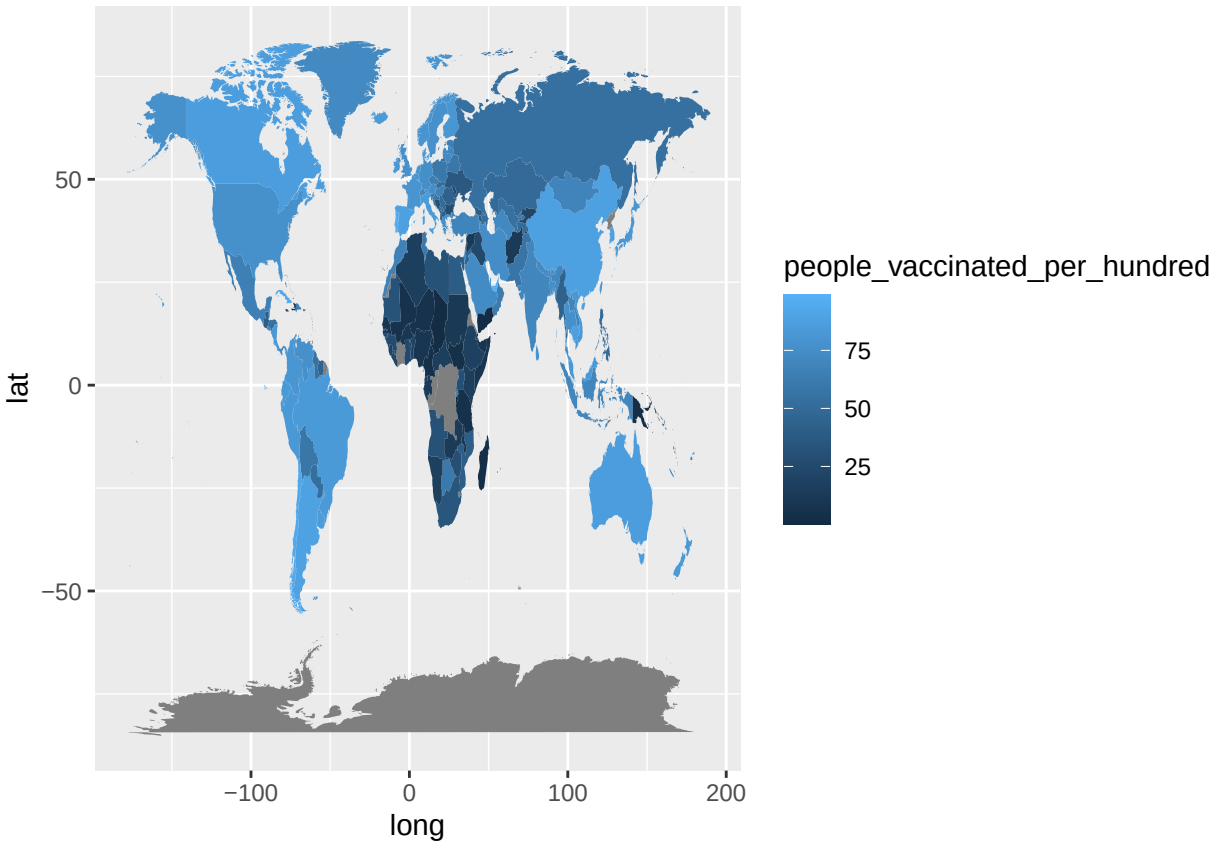
```
people_vaccinated <- people_vaccinated %>%
  mutate(country = case_when(country == "United States" ~ "USA",
                             country == "United Kingdom" ~ "UK",
                             country == "Czechia" ~ "Czech Republic",
                             TRUE ~ country))
```

After fixing country so the names match, we can join both datasets.

```
people_vaccinated_world <- world %>%
 left_join(people_vaccinated,
           by = c("region" = "country"))
```

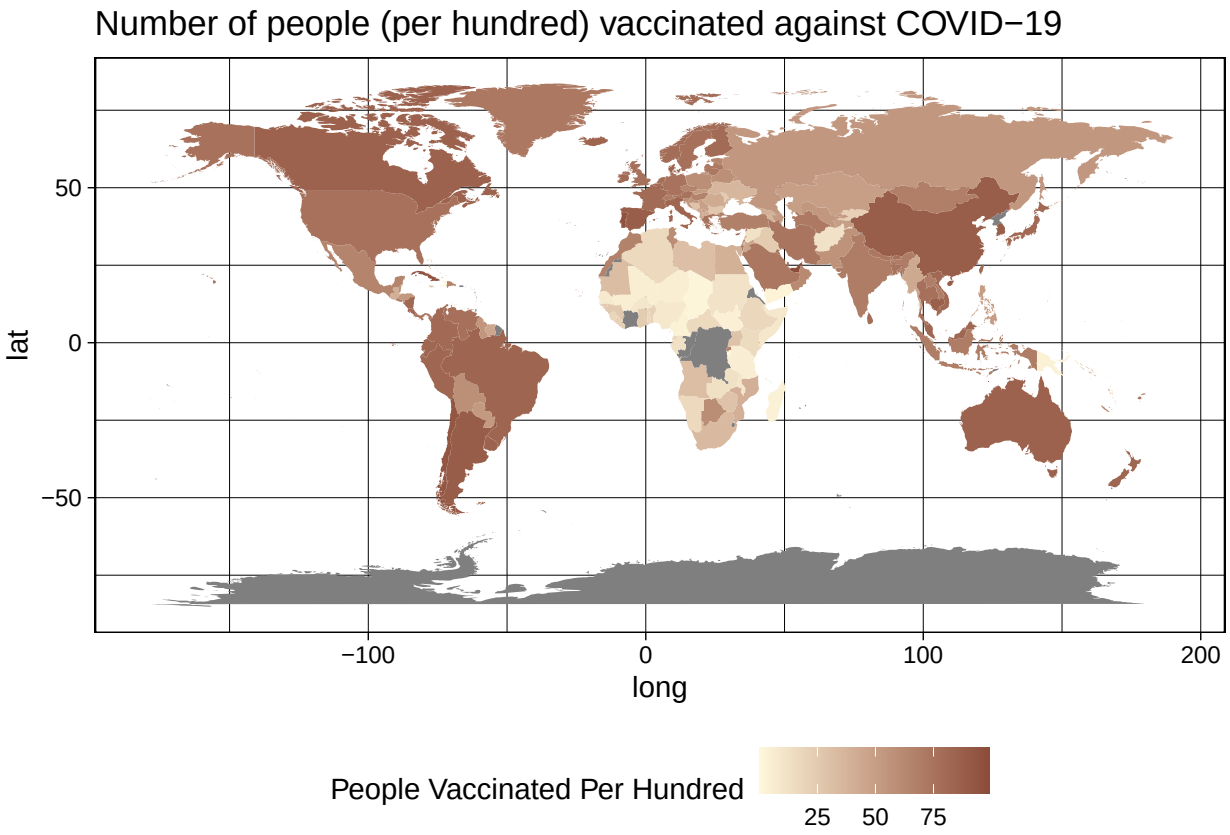
We can now plot `people_vaccinated_world` using `geom_map` to fill each country with `people_vaccinated_per_hundred`.

```
people_vaccinated_world %>%
  ggplot(aes(x = long,
             y = lat,
             fill = people_vaccinated_per_hundred,
             map_id = region)) +
  geom_map(map = world)
```



Let's make some adjustments to the plot (e.g., move legend to the bottom, add a title, and improve on the colors).

```
people_vaccinated_world %>%
  ggplot(aes(x = long,
             y = lat,
             fill = people_vaccinated_per_hundred,
             map_id = region)) +
  geom_map(map = world) +
  theme_linedraw() +
  theme(legend.position = "bottom") +
  ggtitle("Number of people (per hundred) vaccinated against COVID-19") +
  scale_fill_continuous(name = "People Vaccinated Per Hundred",
                        low = "cornsilk",
                        high = "salmon4")
```


Number of people (per hundred) vaccinated against COVID−19
lat
50
0
−50
−100
0
100
200
long
People Vaccinated Per Hundred
25
50
75