# Data Exploration with New York Times COVID-19 Case Data

CHRP202 2020.04.21

Questions: Margaret Antonio antmarge@stanford.edu

## To do before class on Wednesday, April 21

1. Watch the week 4 lecture videos on Canvas
2. Work through this R notebook (read the text and run the code blocks) up to Step #4. Continuing through Step #4 is optional.
3. (Optional) Read the articles in the appendix at the end of this notebook

## Goal for this workshop

The goal of this workshop is to build a framework of 1) questions we should ask when approaching a new dataset & visualizing it and 2) tools we can use to answer those questions. Don't get bogged down about the syntax and code - just know that these are accessible tools that are available to you.

#### By the end of the workshop, you will know:

1. How to load a dataset into R and "clean it"
2. How to critique ("proof read") a plot
3. How to effectively visualize data in R
4. About packages in R that are useful for manipulating data frames and creating effective graphics

### Housekeeping notes:

#### This is an R Notebook. What's an R notebook?

R Markdown Notebooks are a neat way to keep track of code and free text. It's similar to an R script. There are two file types of the same file:

1. `.Rmd` - "R markdown" - load this version in Rstudio and run the code blocks to see the results. Try executing a code chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*. When you execute code within the notebook, the results appear beneath the code.
2. `.pdf` - this is just a PDF format of the file

#### Data: COVID-19 Data from New York Times

The New York Times has been collected case counts for COVID-19 since the beginning of the pandemic. All of their data can be found on Github, an open-source repository for code and data. https://github.com/nytimes/covid-19-data

## Step 1: Know your data before downloading

Before even downloading data, it's always important to read about the dataset. Downloading data can be difficult and we don't want to dive into that if the data is not what we want. So always read about it first! Read the introduction on Github. Based on the description of the dataset, try to answer these questions: (no need to look at the actual data, the description should be sufficient)

1. Why was this data collected?
2. When was the data last updated?
3. What kind of data is it? Is it genetic data, imaging data, text data? Is it descriptive, numeric, categorical?
4. How was the data collected? One or many sources?
5. Are there imperfections in the data? Limitations? E.g. missing data, different data meanings?

## Step #2: Download the data and load necessary packages

Functions: `install.packages()`, `library()`, `dir.create`

Packages are like toolkits of functions that developers make to work with R. Some packages can be focused on making statistical functions, such as calculating mean or variance, easier for R users. Today, we'll use two packages which belong to the R tidyverse, a collection of extremely useful packages

1. `ggplot` (read about ggplot here): a package that allows us to create more detailed graphics (i.e.plots)
2. `dplyr` (read about dplyr here): a package that makes manipulating data frames easier, including handy functions to do common tasks such as create new columns, filter based on criteria, etc.
3. `ggrepel`

Some of these packages were introduced in the Week 4 lecture videos. Be sure to check those out!

```
# Load the ggplot and dplyr libraries
# If they are not already installed,
# then install by uncommenting the following lines

# install.packages("ggplot2")
# install.packages("dplyr")
# install.packages("ggrepel")
# install.packages("scales")


# Now load the libraries.
# Packages only need to be installed once,
# but they need to be loaded everytime R is started
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)
library(ggrepel)
```

```r
library(scales)

# Set your working directory -
# this is where we'll keep code, data,
# and any files or plots generated by this script
# Make sure to change this to a directory on YOUR computer
project_dir = "~/Documents/projects/covid19/2021"

# Create a new directory inside the project directory, called "data"
data_dir = paste0(project_dir, "/data")
dir.create(data_dir)
setwd(data_dir)

getwd()
```

```
## [1] "/Users/margaretantonio/Documents/projects/covid19/2021/data"
```

```r
# There are several files in the Github repo,
# but we only need one csv file: us-states.csv
# A csv file is a file which is comma-separated (as opposed to tabs or spaces)

data_url = "https://github.com/nytimes/covid-19-data/raw/master/us-states.csv"
```

**Comprehension check**

What are "packages" in R? How do we install them and how do we load them in R?

## Step #3: Read the data and make sure it's clean

Functions: `read.csv`, `head()`, `summary()`, `as.Date()`, `ggplot()`,`filter()`

Now that we have the data downloaded, let's read it using R and check that everything makes sense Sometimes data needs to be "cleaned". This can include addressing "NAs", missing values, incorrectly formatted columns, and more

```r
# Read the file into a data frame - "df" for short here
# Remember it's a csv file, so we can let R know that there
# will be commas by reading the file with the read.csv function
# Note that you can download the data directly from the web using the URL

df <- read.csv(data_url,
           header = T)

# Peak at the data by just showing the first few lines
# Use the head() function

# In base R:

#head(df)
# In dplyr, we can "pipe" the data frame into the function

df %>%
  head()
```

```
##         date      state fips cases deaths
## 1 2020-01-21 Washington   53     1      0
## 2 2020-01-22 Washington   53     1      0
## 3 2020-01-23 Washington   53     1      0
## 4 2020-01-24   Illinois   17     1      0
## 5 2020-01-24 Washington   53     1      0
## 6 2020-01-25 California    6     1      0
```

```
# Does the data look ok? Is the correct number of columns there?
# Do you know what all the columns mean?
# If not go back to step 1 and read the data description!

# We can quickly summarize the columns with the summary() function

# In base R: summary(df)
# In dplyr, we can "pipe" the data frame into the function
df %>%
  summary()
```

```
##      date               state                fips           cases
##  Length:23554       Length:23554       Min.   : 1.00   Min.   :      1
##  Class :character   Class :character   1st Qu.:17.00   1st Qu.:   8743
##  Mode  :character   Mode  :character   Median :31.00   Median :  66377
##                                        Mean   :31.94   Mean   : 221769
##                                        3rd Qu.:46.00   3rd Qu.: 259313
##                                        Max.   :78.00   Max.   :3748832
##      deaths
##  Min.   :    0
##  1st Qu.:  192
##  Median : 1402
##  Mean   : 4585
##  3rd Qu.: 5350
##  Max.   :61996
```

**Data check!**

1. Do the summaries make sense?
2. Does it make sense that some of them are character while others are numeric?
3. Is this all we need to know about the data or are there still questions?
4. Look at the date column. Are they numbers or characters/strings? Will R know that these are dates?

**Comprehension check**

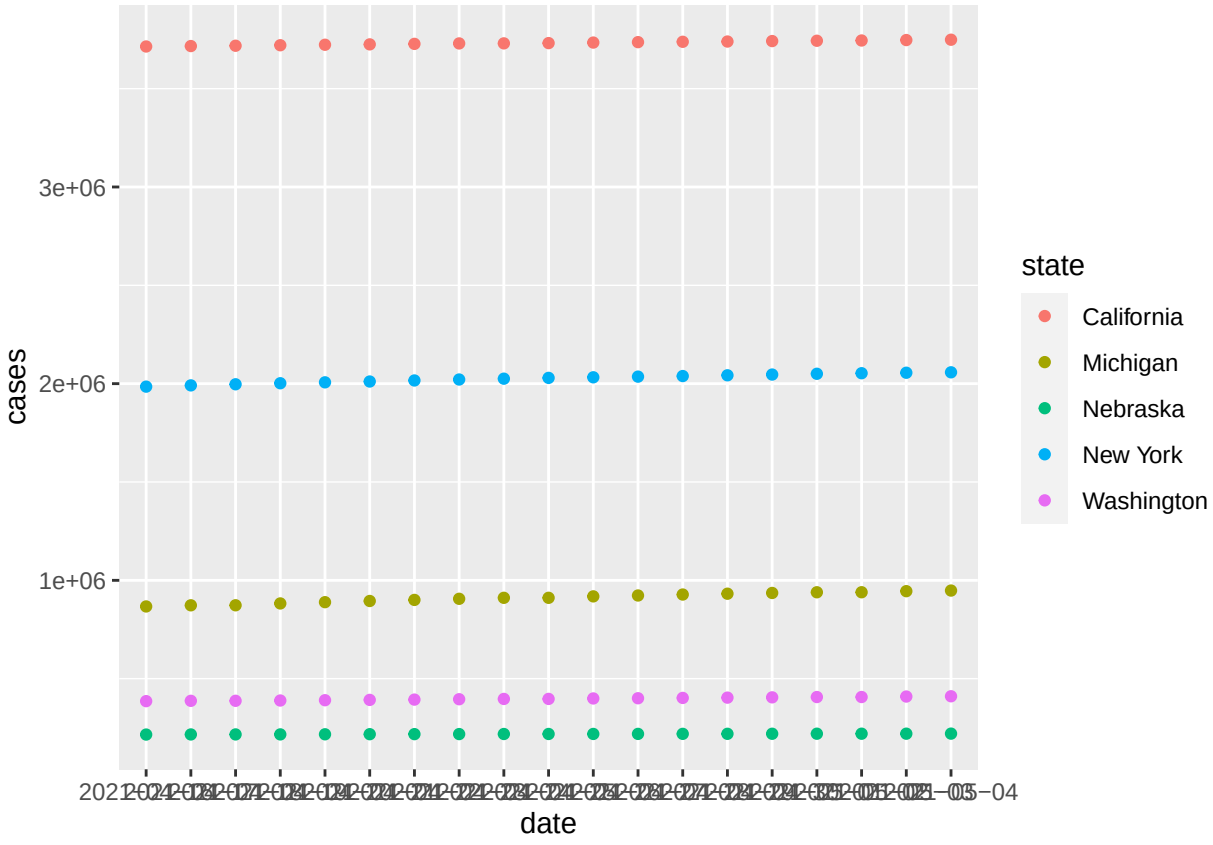
What is `%>%`? What is the difference between putting a variable directly into the function vs piping it into the function using `%>%`?

## Step #4: Let's explore the data visually

Sometimes pictures say more than words. Let's make a plot to answer the question: How have case counts changed for each state over time?

```
# We know there are a lot of states,
# so let's add a filter to choose only a few states
```

```r
fav_states = c("California", "Masachusetts",
               "Michigan", "Nebraska",
               "Washington", "New York")

# Plot using ggplot
df %>%
  filter(state %in% fav_states) %>%
  filter(date > "2021-04-15") %>%
  # ggplot function begins here:
  ggplot() +
  # Add points
  # The aes() aesthetic mapping can be in the ggplot() function or in the layer
  geom_point(aes(x = date, y = cases,
                 color = state))
```




### Comprehension check

To make a plot with `ggplot`, what are the essential components? Is this the right kind of plot for our data?

### Reformat the date so R knows it's a date

There is a base R function called `as.Date()` that will convert a character to a date. This is similar to `as.numeric()` and `as.character()` functions which change values to numbers or characters, respectively.

```r
# What type of variable is date currently?
class(df$date)
```

```
## [1] "character"
```

```r
# Reformat it to be a date
df$new_date <- as.Date(df$date,format="%Y-%m-%d")

# What type of variable is date after converting it?
class(df$new_date)
```

```
## [1] "Date"
```

## Break out room! Proofread this plot

Proof reading a plot is similar to proof reading an essay. The plot should communicate a clear message, supported by the evidence. Everything included (e.g. colors, shapes) should have a purpose. Here are some questions that should be clearly answered using the plot. Try to answer these questions for your own plots and for plots you see in a research paper or even on the news.

1. What's on the x and the y axis?
2. What does a single point on the plot represent?
3. What do the colors and/or shapes mean?
4. What is the main message of this plot?
5. Is it easy to read the text and labels on the axis?

If it was difficult to answer any of those questions, then that means we need to make some changes to the plot.

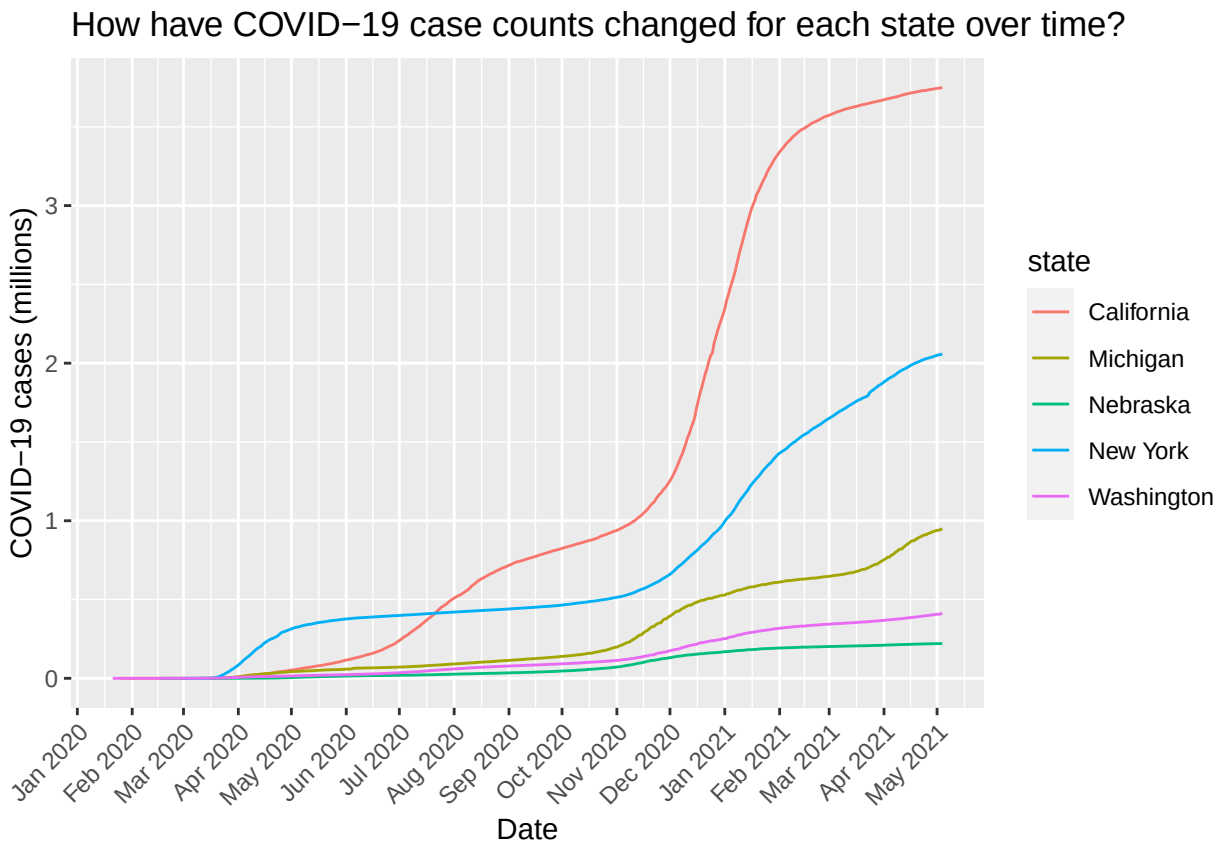
### Remake the plot

For example, let's make these obvious changes

1. Every plot should have an informative title
2. On the axes: text should be readable, with clear units, and labels

```r
plot_title = "How have COVID-19 case counts changed for each state over time?"

df_states <- df %>%
  filter(state %in% fav_states)

c  = 1000000

g <- ggplot(data = df_states) +
  # Lines instead of points?
  # Change the y-axis to more readable numbers
  geom_line(aes(x = new_date, y = cases/c,
                color = state)) +

  # Add a plot title
  ggtitle(plot_title) +
  # Make y and x axis labels more informative
  ylab("COVID-19 cases (millions)") +
  xlab("Date") +
  scale_x_date(labels = date_format("%b %Y"),
```

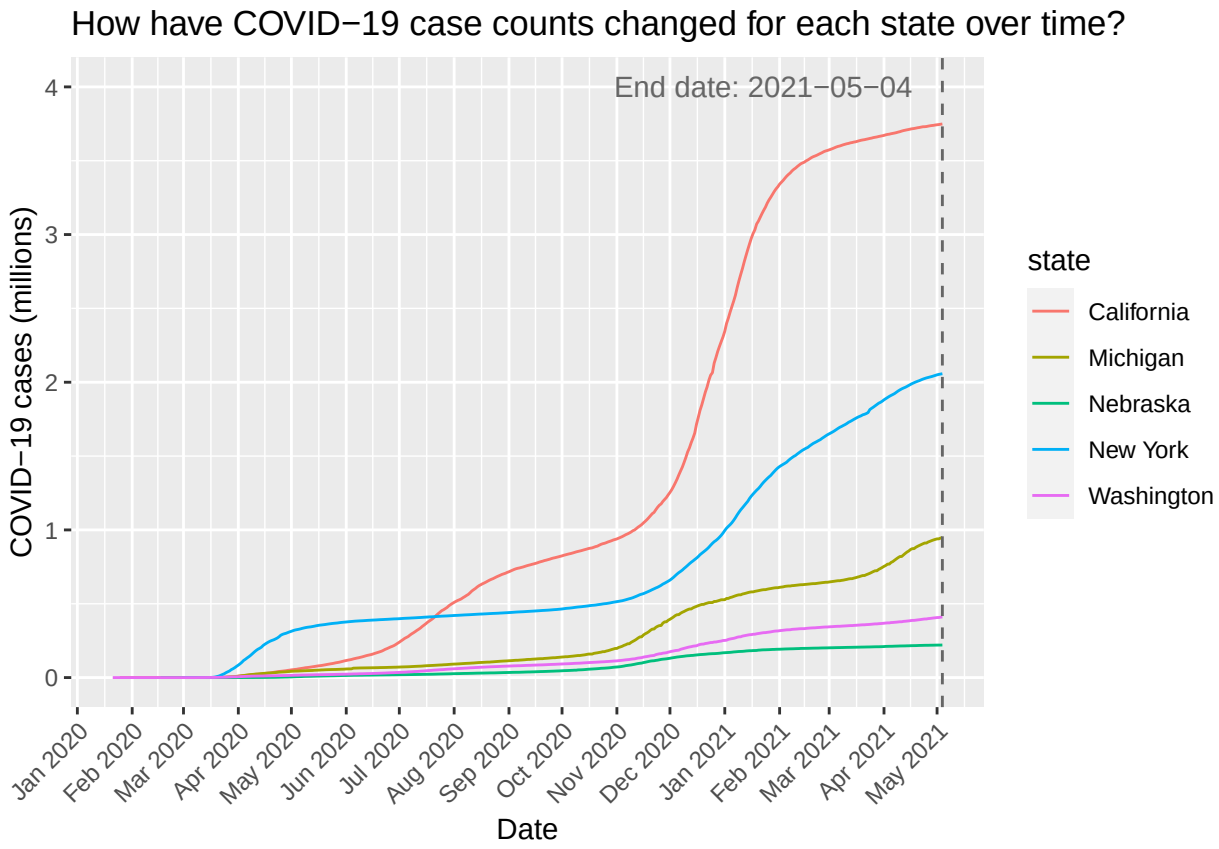
```
               date_breaks = "months") +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1)
  )


# show plot
g
```




**When possible, add annotations directly on the plot**

```
# Let's add a line that specifies the end date

g2 <- g +
  # Add annotations
  geom_vline(xintercept = max(df_states$new_date),
             linetype = "dashed",
             color = "gray40") +
  annotate(x = max(df_states$new_date),
           y = 4,
           hjust = 1.1,
           color = "gray40",
           label = paste0("End date: ",max(df_states$new_date)),
           "text")
```

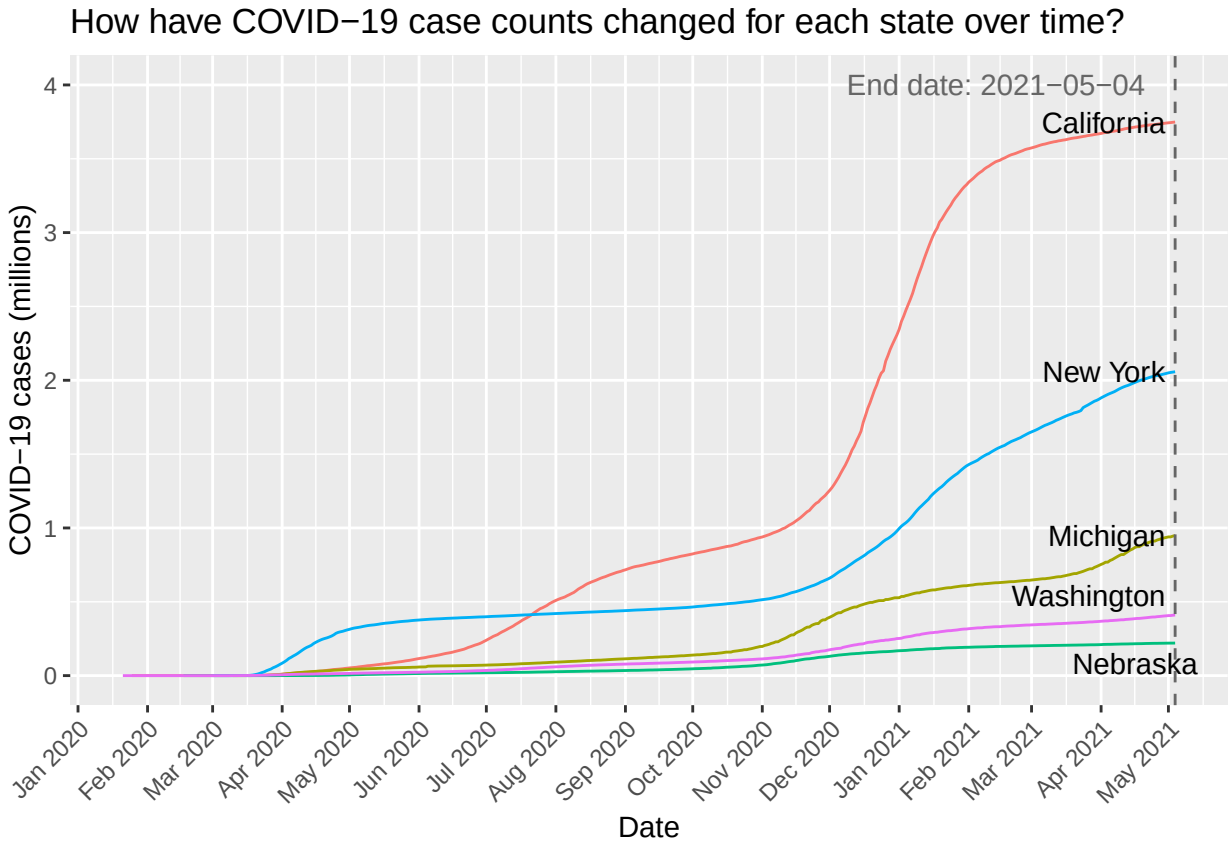
```
g2
```




```
# Let's get rid of the legend
g3 <- g2 +
  geom_text_repel(data = df_states %>%
                group_by(state) %>%
                summarize(max_date = max(new_date),
                          max_cases = max(cases)) %>%
                ungroup(),
              aes(x = max_date, y = max_cases/c,
                  label = state)) +
  guides(color = F)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
  g3
```




**Proof read the plot again**

1. What's on the x and the y axis?
2. What does a single point on the plot represent?
3. What do the colors and/or shapes mean?
4. What is the main message of this plot?
5. Is it easy to read the text and labels on the axis?

Are these questions easier to answer than with the original plot?

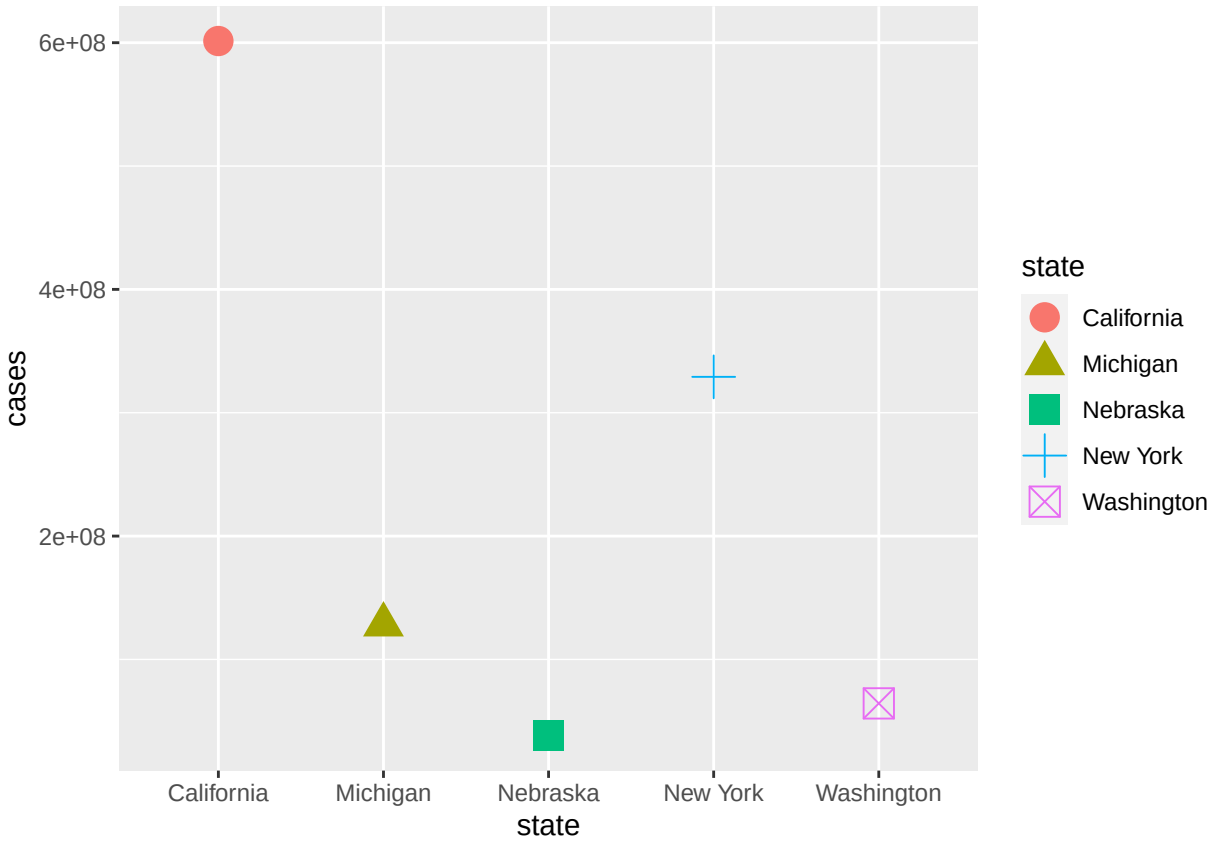
**How to choose the right kind of plot**

Question: What are the total numbers of cases in each state?

```
# Summarize the data to get totals for each state
df_states_sum <- df_states %>%
         group_by(state) %>%
         summarize(deaths = sum(deaths),
                   cases = sum(cases))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
# Plot total deaths for each state
ggplot(data = df_states_sum,
       aes(x = state, y = cases, shape = state)) +
  geom_point(aes(color = state), size = 5)
```




## Break out room! Proofread this plot

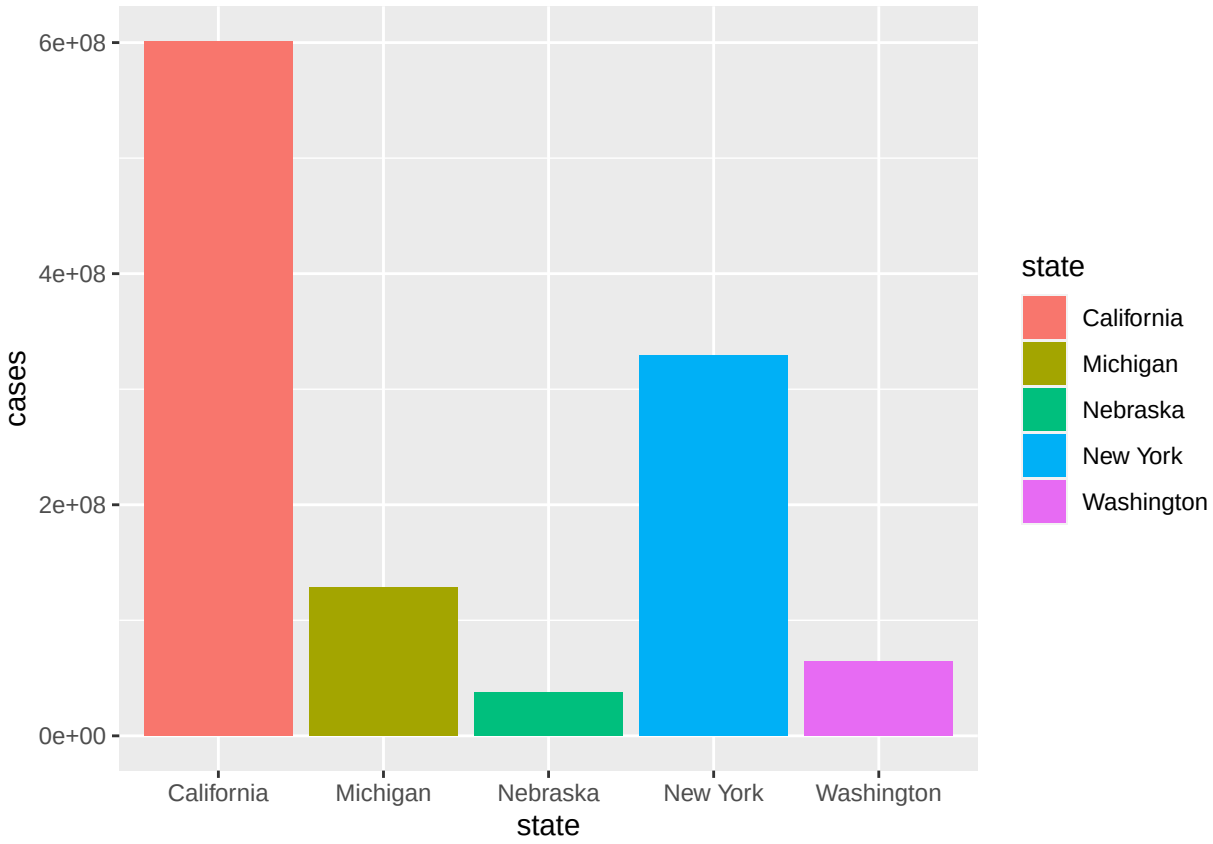
1. What's on the x and the y axis?
2. What does a single point on the plot represent?
3. What do the colors and/or shapes mean?
4. What is the main message of this plot?
5. Is it easy to read the text and labels on the axis?

### Is this the right type of plot?

1. What's wrong with using points in this plot?
2. Is time still important for answering this question?
3. How about using lines? What do lines communicate about the data? In this plot, how would the lines be drawn?

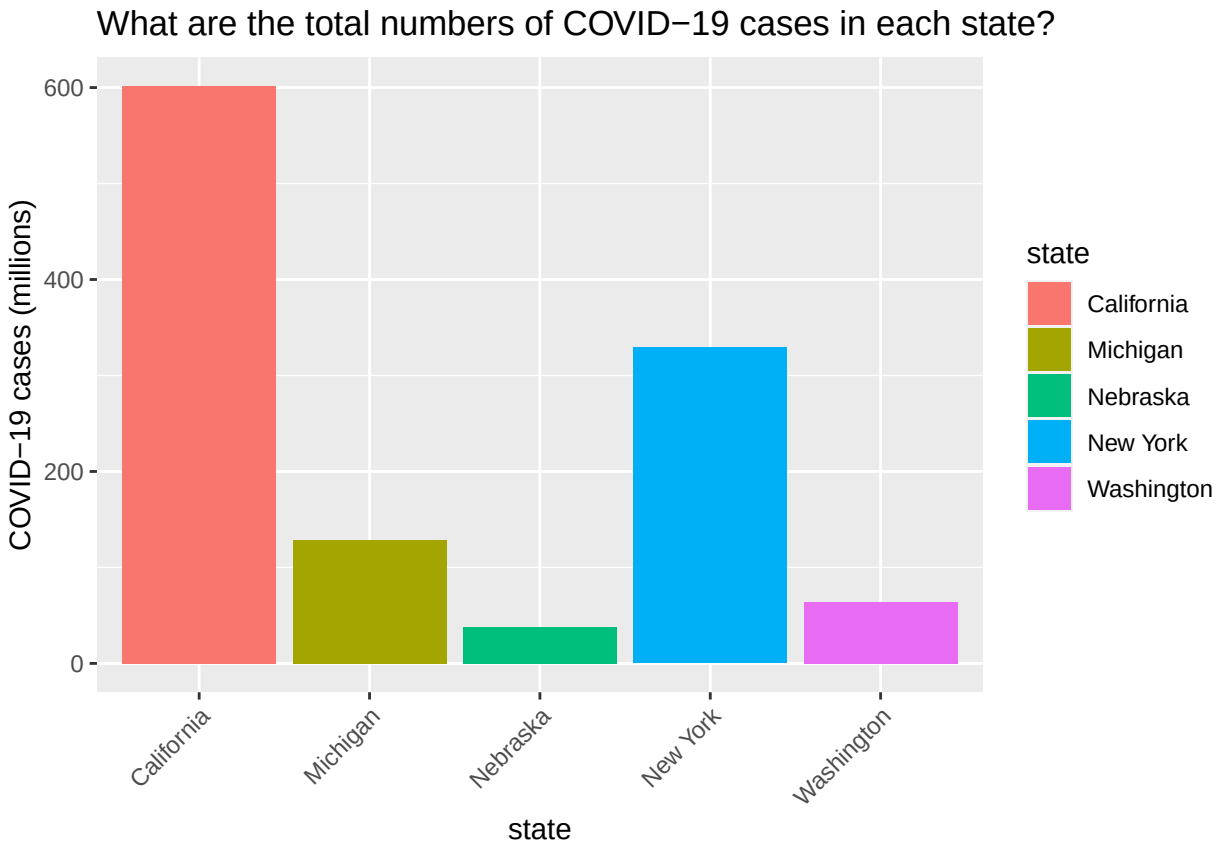
Remember, we're interested in totals and a given point in time.

```
ggplot(data = df_states_sum,
       aes(x = state, y = cases)) +
  geom_bar(stat = "identity",
                 aes(fill = state))
```



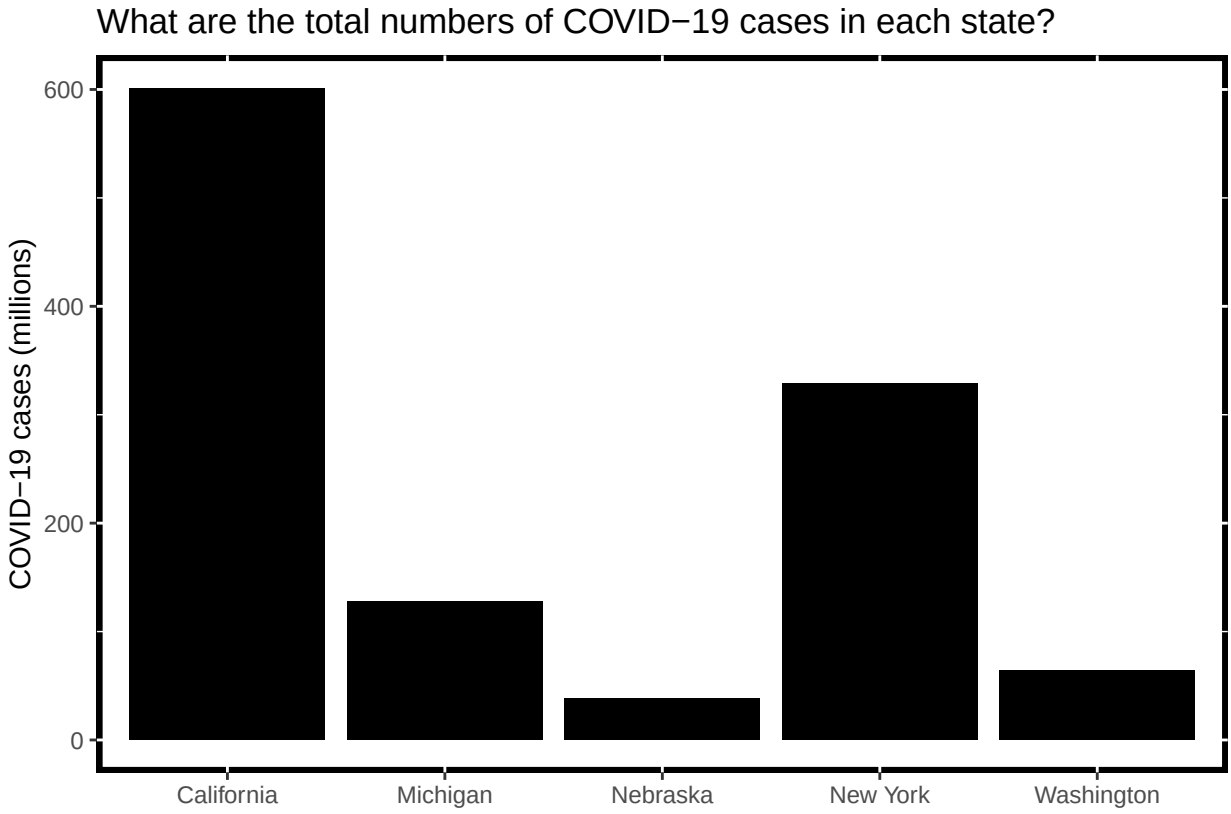

Proof read the plot again. What would you change about it to better communicate the message or answer the question? What are some other ways to visualize this data?

```
plot_title2 = "What are the total numbers of COVID-19 cases in each state?"

# Add some features from before
ggplot(data = df_states_sum,
       aes(x = state, y = cases/c)) +
  geom_bar(stat = "identity",
           aes(fill = state)) +
  # Add plot and axis labels
  ggtitle(plot_title2) +
  ylab("COVID-19 cases (millions)") +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```




**Extra information is just distracting from the main point**

```
# Add some features from before
ggplot(data = df_states_sum,
       aes(x = state, y = cases/c)) +
  geom_bar(stat = "identity",
           fill = "black") +
  # Add plot and axis labels
  ggtitle(plot_title2) +
  ylab("COVID-19 cases (millions)") +
  xlab("") +
  theme(
      panel.background = element_rect(fill = "white", colour = "black",
                                      size = 2, linetype = "solid"),
      panel.grid.major = element_line(size = 0.5, linetype = 'solid',
                                      colour = "white")
  )
```




## Conclusion

We used the NY Times COVID-19 data to put some data exploration and visualization principles into practice. We used the R packages ggplot and dplyr as tools along the way.

**Main points we went over as an introduction to data exploration:**

1. Learn about the data before downloading it
2. Check the data and do any necessary cleaning or reformatting
3. Initial data exploration begins with simple questions answered with tables and plots

**When making plots, there are a few key data visualization principles to keep in mind**

1. Proof read plots. Always proof read plots throughout the process
2. Stick to the main point. Remove anything (e.g. annotations, colors) that does not support the main point (even if it is interesting!)
3. Aesthetics do matter! If the font is too small or the colors are not distinguishable that's just as important as the actual data.

## Appendix

**More reading about data visualization**

1. Bad Data Visualization in the Time of COVID-19. This is a great article about plots that were

popular on Twitter or in the News and why they can be misleading. For each example, think about the implications of a poor visualization of data.
2. Visualizations that Really Work. This article provides a more detailed perspective surrounding many of the concepts we discussed, such as effective communication with graphics and simplicity over clutter.
3. Data Visualization 101: How to Choose the Right Chart or Graph for Your Data. A fun article about different types of plots. Today, we only went over scatter plots, line graphs, and bar plots. There are so many more though! If you decide to try one of the visualization challenges below, check out this article for other visualization ideas

**If you're interested in more data visualization challenges with this dataset:**

1. Download the `us-counties.csv` file in the NY Times Github Repo. Are there certain counties that have higher case counts in California?
2. Add more states to the plots we made above. What aesthetics (e.g. labels, colors, font size) do you need to adapt now that there is more data?
3. If you like math and statistics - Plot the distribution of cases. What kind of distribution is this (e.g. normal, exponential, logistic)?
4. What's the relationship between deaths and cases? Come up with a hypothesis. Plot deaths vs cases. Calculate the intercept and slope. What does the intercept mean? What does the slope mean? Try adding this line along with annoations for slope and intercept to the plot. Are deaths and cases correlated?
5. What kinds of data or datasets would be interesting to co-analyze with these case and death counts?