# Lab Session 0: Intro to R, RStudio and Tidyverse

Solutions

9/3/2020

We expect you to watch the `class 0` material, here prior to lab and have:

1. Installed R from https://www.r-project.org
2. Installed Rstudio from http://www.rstudio.com

If you find yourself in lab without R installed, you can use google colab colab.fan/r ) to run R in a notebook environment. You'll lose some of the RStudio functionality, but it'll work.

## Installation completion

1. If you have not yet, run the following code to install the `tidyverse`.

```
install.packages("tidyverse")
```

2. This code returns an error. Why? Fix it and install the packages. We'll see what they're for in the next lecture

```
install.packages(haven)
install.packages(readxl)
```

*Solution:*

The code returns an error because the package names need to be in quotes:

```
install.packages("haven")
install.packages("readxl")
```

3. In order to have access to `tidyverse` functions, you need to load the library.

```
library("tidyverse")
```

4. Run the following code to make sure you have installed successfully. Then, assign the resulting dataframe to the name `big_storms`.

```
# storms is a dataset that comes with the tidyverse
# use View(storms) to see the full dataset
storms %>%
  group_by(name, year) %>%
  filter(max(category) == 5)
```

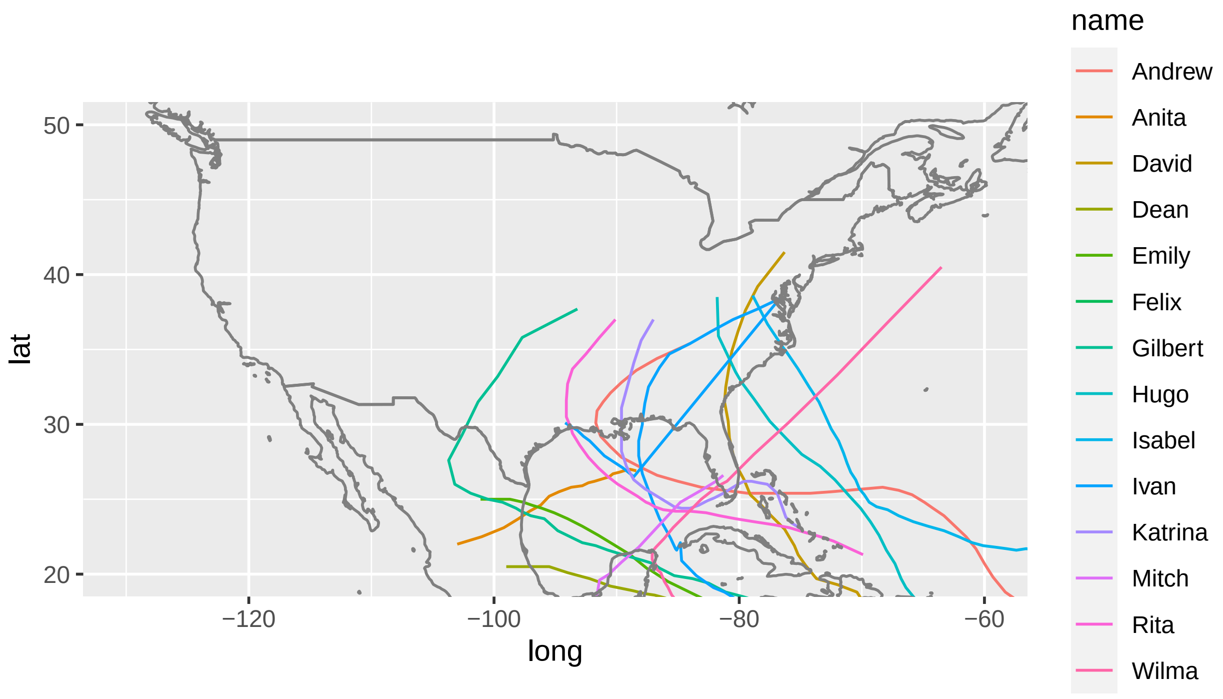
*Solution:*

We use `<-` for assigning variables in R. We can assign all sorts of objects to names, including data sets. All we have to do is add `big_storms <-` at the beginning of the code; that will tell R to take the output of the code and assign it to the name `big_storms`:

```
big_storms <- storms %>%
  group_by(name, year) %>%
  filter(max(category) == 5)
```

5. This code makes the map seen below. If you run it you will get an error that says your code is missing a required package. Use the error to figure out what package is missing, install it and then reproduce the map on your computer. [This package is not essential, so if you're using a lot of time (> 3 min) trying to install it. It's okay to move on or try on google colab.]

```
  ggplot(aes(x = long, y = lat, color = name), data = big_storms) +
        geom_path() +
        borders("world") +
        coord_quickmap(xlim = c(-130, -60), ylim = c(20, 50))
```



*Solution:*

Most of you probably saw the following error when you ran the code:

```
Error:   Package ‘maps‘ required for ‘map_data‘.
  Please install and try again.
Run ‘rlang::last_error()‘ to see where the error occurred.
```

The solution is to install and load the `maps` package:

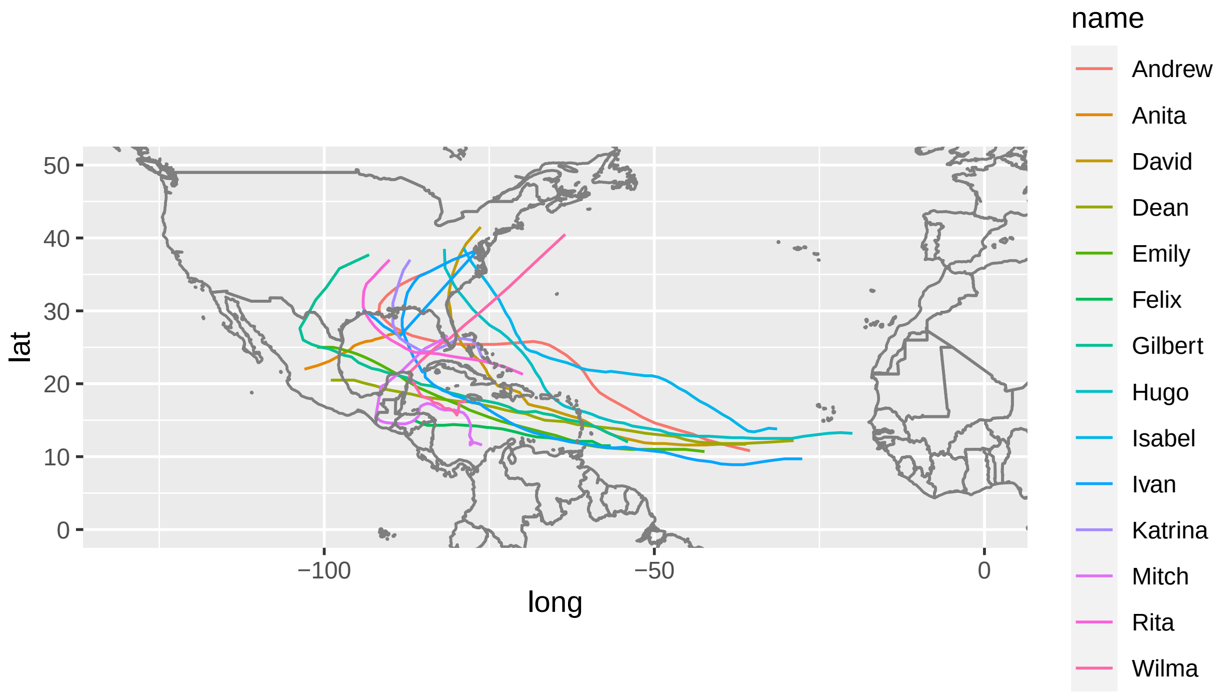
```
install.packages("maps")
library("maps")
```

6. The map is nice, but we cannot see where the storm baths begin. Play with the code to expand our view to capture the full path.

*Solution:*

Looking at the code, you'll notice that the `xlim` and `ylim` parameters look like the longitude and latitude positions where the graph cuts off. Changing these limits will also change the part of the world the graph shows. You can play around with these numbers until you have a graph you like, such as:

```
ggplot(aes(x = long, y = lat, color = name), data = big_storms) +
  geom_path() +
  borders("world") +
  coord_quickmap(xlim = c(-130, 0), ylim = c(0, 50))
```



## Processing and analyzing data, an example:

Now you'll see an abridged version of the data exploration I undertook to make the racial disparities of covid-19 plot discussed in lecture.

The idea is to preview code we'll see in coding lab and get you tinkering.

Copy and paste the following code to download and load data from CDC. It's a large enough file that it might take a minute to run. Note the original names of the variables `diff` and `perc_diff`. This code mainly does some pre-processing of the data for you.

```
covid_data <-
  read_csv(paste0("https://data.cdc.gov/api/views/qfhf-uhaa/rows.csv?",
                  "accessType=DOWNLOAD&bom=true&format=true%20target="),
           col_types = cols(Suppress = col_character())) %>%
    mutate(week = ‘Week Ending Date‘,
           race_ethnicity = ‘Race/Ethnicity‘,
           n_deaths = ‘Number of Deaths‘,
           diff = ‘Difference from 2015-2019 to 2020‘,
           expected_deaths = n_deaths - diff,
           perc_diff = ‘Percent Difference from 2015-2019 to 2020‘,
           year = MMWRYear,
           week_no = MMWRWeek,
           jurisdiction = Jurisdiction,
           state = ‘State Abbreviation‘
           )  %>%
    filter(‘Time Period‘ == "2020", Outcome == "All Cause", Type != "Unweighted") %>%
    select(jurisdiction, state, week, year, week_no,
           race_ethnicity, n_deaths, expected_deaths, diff, perc_diff)
```

1. Now that you have `covid_data` loaded. Use `View()` to examine the data.

   Analysis is easier with data that is "tidy", by which we mean

   - each row is an observation
   - each column is a variable.

   `storms` is tidy and the unit of observation is storm-time which is captured by the columns `name`, `year`, `month`, `day` and `hour`.

   Is `covid_data` tidy? What is the unit of observation for each row?

*Solution:*

`covid_data` appears to be tidy. Each row represents one week of deaths for one race/ethnicity within one state.

2. R has a built in function to summarize each column called `summary()`. Try calling the function. Your data goes in the first position in the parenthesis. It's not the prettiest, but it gives you some information with little code. Look at `n_deaths`. What information are you getting from `summary()`. Does anything surprise you?

*Solution:*

`summary(covid_data)` shows us a summary of each variable within the `covid_data` data frame. Most of the variables are fairly self-explanatory, but for some reason `n_deaths`, `expected_deaths`, `diff`, and `perc_diff` have some very big outliers within their data.

3. In `summary()`, you saw that the Max values are quite high compared to the third quartile. That's worthy of investigation. Run the following code and explain why we have some super large values in our data set.

```
covid_data %>%
  filter(n_deaths > 10000)
```

*Solution:*

The reason we have outliers in the data is that, while most of the data is at the state level, the CDC decided to also include aggregated data for the entire country as if it were also a state.

4. Ah, that's actually quite useful. We have US data already aggregated for us. Let's start by aggregating the data so we have a single number for each `race_ethnicity`. Use the `<-` to assign the results to the name `us_deaths_by_race`.

```
covid_data %>%
  filter(state == "US") %>%
  group_by(race_ethnicity) %>%
 summarize(expected_deaths = sum(expected_deaths, na.rm = TRUE),
           total_additional_deaths = sum(diff, na.rm = TRUE),
            percent_diff = 100 * total_additional_deaths / expected_deaths
            )
```

*Solution:*

```
us_deaths_by_race <- covid_data %>%
  filter(state == "US") %>%
  group_by(race_ethnicity) %>%
  summarize(
    expected_deaths = sum(expected_deaths, na.rm = TRUE),
    total_additional_deaths = sum(diff, na.rm = TRUE),
    percent_diff = 100 * total_additional_deaths / expected_deaths
  )
```

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

5. It's often nice to see the data. Run the code to make a bar graph.

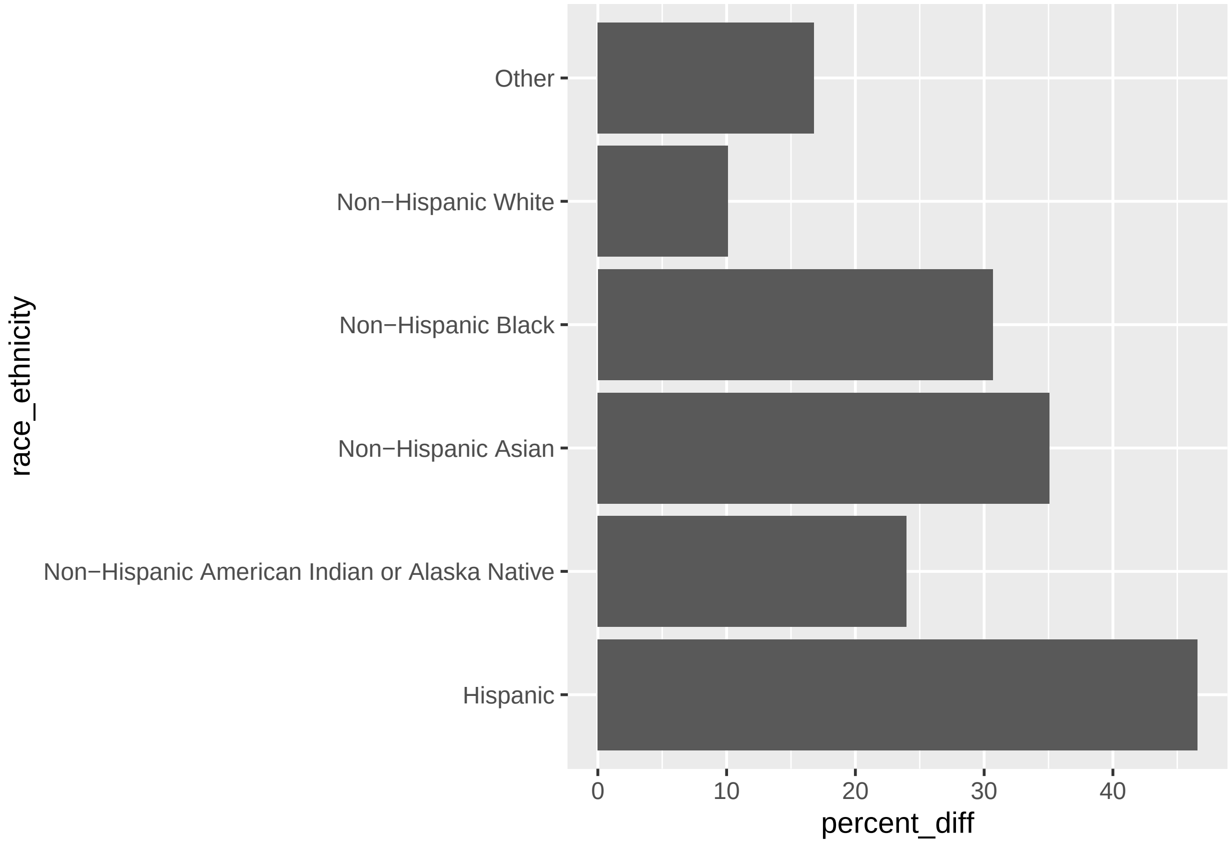
```
us_deaths_by_race %>%
  ggplot(aes(y = race_ethnicity, x = percent_diff)) +
  geom_col() +
labs(x = "Percent above expected death count",
     y = "",
     title = "Racial disparities of Covid-19 (USA, 2020)" )
```

6. Remove the `+ labs()` function from the above code. Describe what `labs()` does. What is the default plotting behavior?

*Solution:*

We run the following code and get a new graph:

```r
us_deaths_by_race %>%
  ggplot(aes(y = race_ethnicity, x = percent_diff)) +
  geom_col()
```



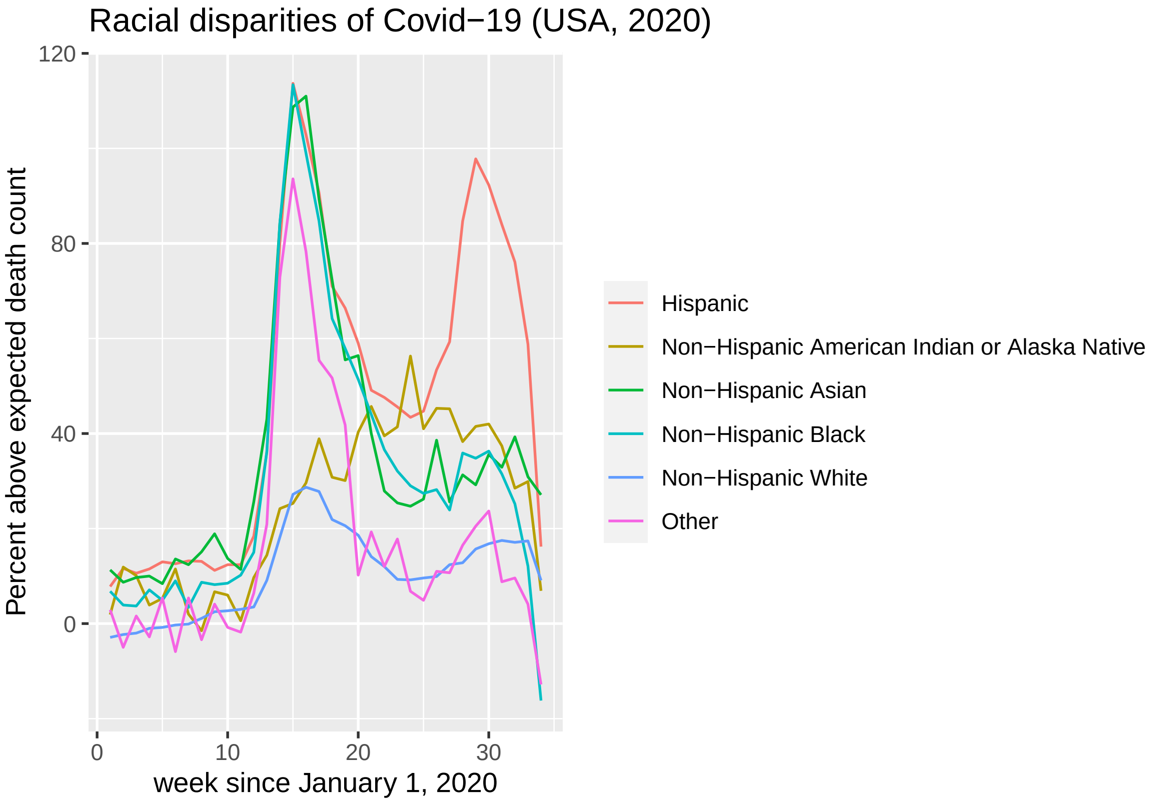
Notice that the new graph uses the variable names as the axis labels, whereas the original graph had nicer descriptive labels. This implies that `labs()` sets labels for the graph that override the default behavior, which is to use the names of the variables being graphed.

7. Remake the graph but for the state of IL. You should be able to do this by changing the above code in two places.

*Solution:*

8. Seeing this graph, I thought to myself. I can tell a better story and understand the data better if I included the time series element. This would allow us to see how the disparities change over time.

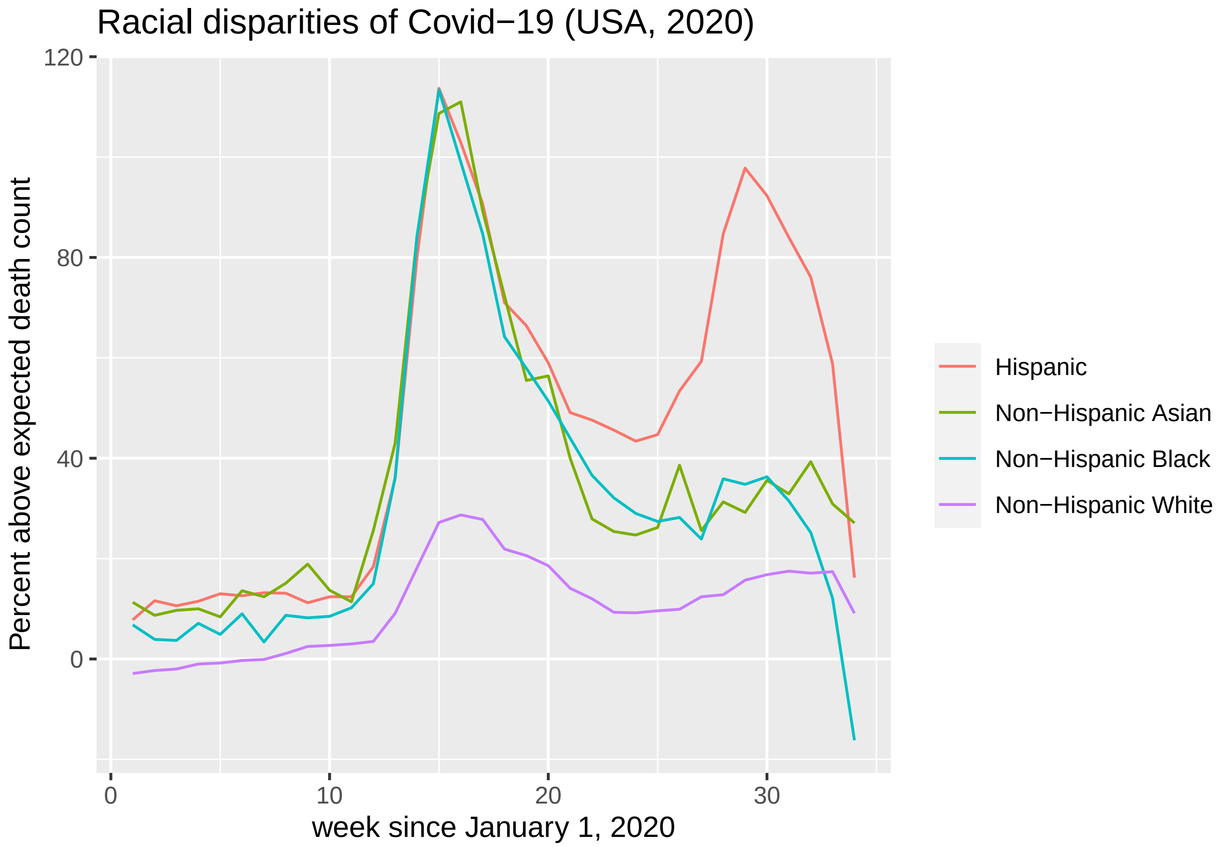
```r
covid_data %>%
  filter(state == "US") %>%
  # filter(race_ethnicity %in%
  # c("Hispanic", "Non-Hispanic White", "Non-Hispanic Black", "Non-Hispanic Asian")) %>%
  ggplot(aes(x = week_no, y = perc_diff, color = race_ethnicity)) +
  geom_line()  +
labs(y = "Percent above expected death count",
     x = "week since January 1, 2020",
     title = "Racial disparities of Covid-19 (USA, 2020)",
    color = "" )
```



9. Notice that groups with smaller populations have noisier time-series. The unfortunately-named "Other" time series is saw toothed, while the "Non-Hispanic White" line is much smoother. Remove the `#` from the code above and run it again. `#` is used for "comments" in R. R ignores code that comes after `#` until a new line is started.

*Solution:*

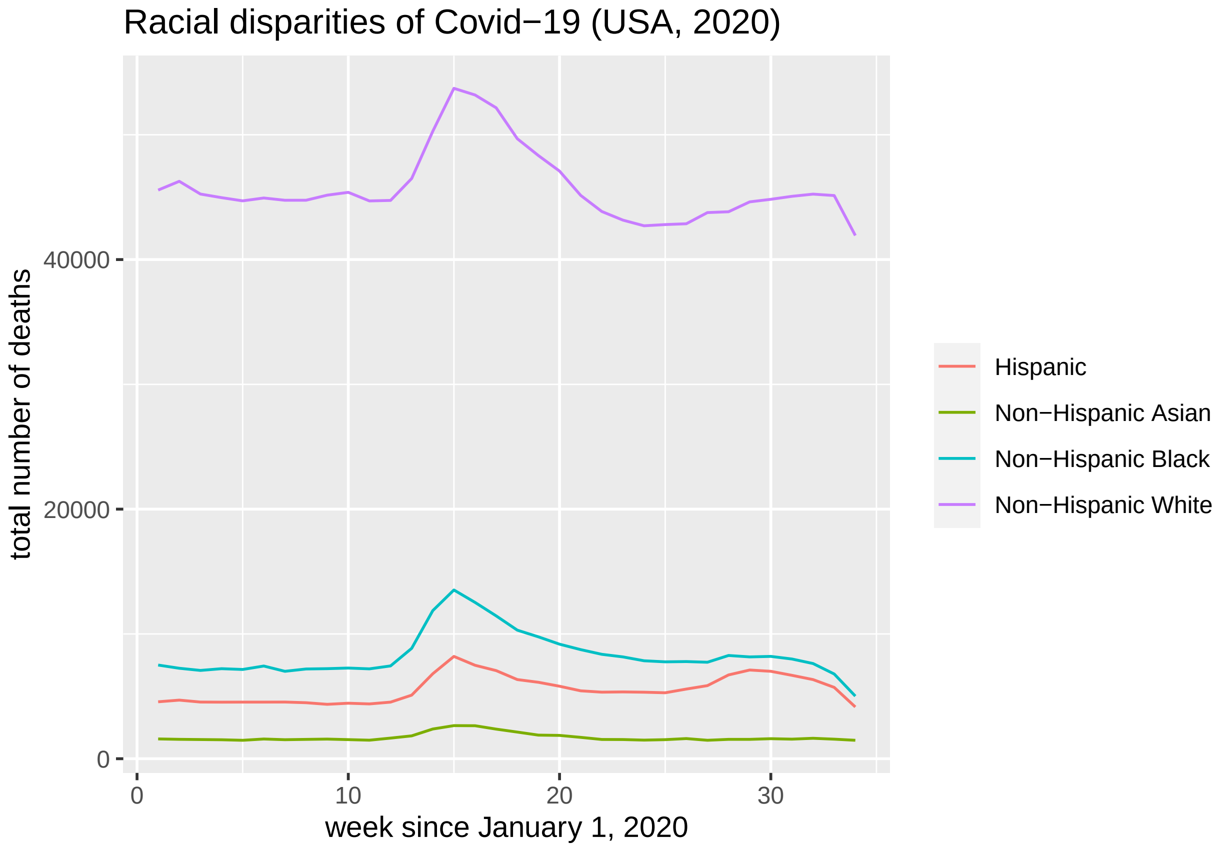
```
covid_data %>%
  filter(state == "US") %>%
  filter(race_ethnicity %in%
    c("Hispanic", "Non-Hispanic White", "Non-Hispanic Black", "Non-Hispanic Asian")) %>%
  ggplot(aes(x = week_no, y = perc_diff, color = race_ethnicity)) +
  geom_line()  +
labs(y = "Percent above expected death count",
     x = "week since January 1, 2020",
     title = "Racial disparities of Covid-19 (USA, 2020)",
     color = "" )
```



10. Finally, you might notice that it looks like there's a suspicious drop off in our "Percent above expected death count" after week 30. Let's investigate. Is this because there actually fewer deaths? Change the `y` value from `perc_diff` to a different column in your data set to graph death counts.

*Solution:*

```
covid_data %>%
  filter(state == "US") %>%
  filter(race_ethnicity %in%
    c("Hispanic", "Non-Hispanic White", "Non-Hispanic Black", "Non-Hispanic Asian")) %>%
  ggplot(aes(x = week_no, y = n_deaths, color = race_ethnicity)) +
  geom_line()  +
labs(y = "total number of deaths",
     x = "week since January 1, 2020",
     title = "Racial disparities of Covid-19 (USA, 2020)",
     color = "" )
```

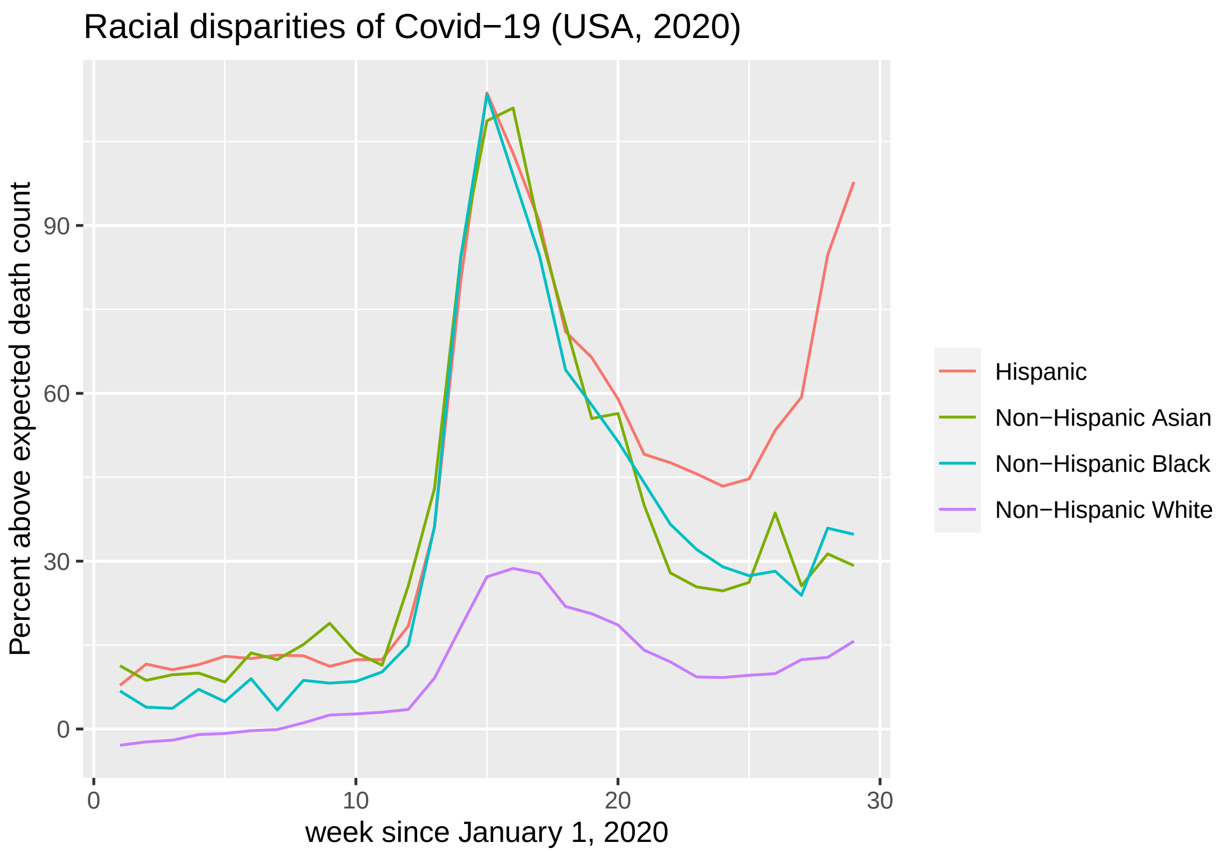


11. With a little investigation you discover that the CDC updates these counts over time and the last 6 weeks are likely to increase substantially. Add this line `filter(week_no > 30) %>%` to your code to cutoff the data sooner.

*Solution:*

We made a typo in the initial assignment; rather than `filter(week_no > 30) %>%`, it should be `filter(week_no < 30) %>%`; that is to say, we want to filter the data set so it only includes data from *before* week 30, not *after* week 30.

```
covid_data %>%
  filter(state == "US") %>%
  filter(race_ethnicity %in%
    c("Hispanic", "Non-Hispanic White", "Non-Hispanic Black", "Non-Hispanic Asian")) %>%
  filter(week_no < 30) %>%
  ggplot(aes(x = week_no, y = perc_diff, color = race_ethnicity)) +
  geom_line()  +
labs(y = "Percent above expected death count",
     x = "week since January 1, 2020",
     title = "Racial disparities of Covid-19 (USA, 2020)",
     color = "" )
```


Racial disparities of Covid−19 (USA, 2020)
Percent above expected death count
90
60
30
0
0
10
20
30
week since January 1, 2020
Hispanic
Non−Hispanic Asian
Non−Hispanic Black
Non−Hispanic White