

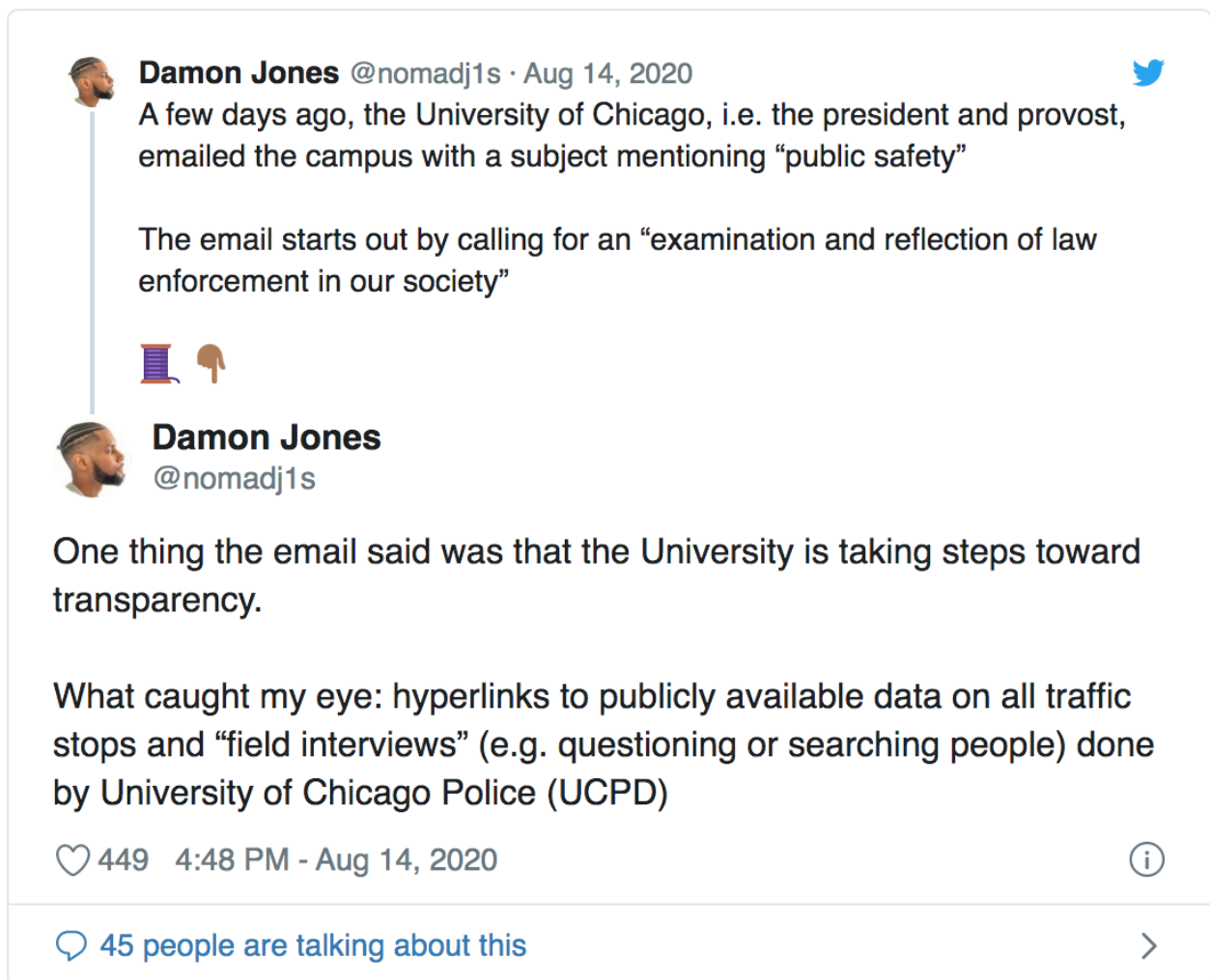
Lab Session 4: Read and manipulate data

Ari Anisfeld

8/29/2021

We expect you to **watch the class 4 material**, [here](#) prior to lab. In addition, **read the background and data section** before lab.

Background and data



Damon Jones @nomadj1s · Aug 14, 2020

A few days ago, the University of Chicago, i.e. the president and provost, emailed the campus with a subject mentioning “public safety”

The email starts out by calling for an “examination and reflection of law enforcement in our society”

📖 👉

Damon Jones
@nomadj1s

One thing the email said was that the University is taking steps toward transparency.

What caught my eye: hyperlinks to publicly available data on all traffic stops and “field interviews” (e.g. questioning or searching people) done by University of Chicago Police (UCPD)

📄 449 4:48 PM - Aug 14, 2020 ⓘ

💬 45 people are talking about this >

Follow the [tweet thread](#) and you’ll see that Prof. Damon Jones, of Harris, gets that data and does some analysis. In this lab, you’re going to follow his lead and dig into traffic stop data from the University of Chicago Police Department, one of the largest private police forces in the world.

Download the data [here](https://github.com/harris-coding-lab/harris-coding-lab.github.io/raw/master/data/data_traffic.csv). You can save the file directly from your browser using ctrl + s or cmd + s. Alternatively, you can read the csv directly from the internet like we saw in lab 0 using the link https://github.com/harris-coding-lab/harris-coding-lab.github.io/raw/master/data/data_traffic.csv

Warm-up

1. Open a new Rmd and save it in your coding lab folder; if you downloaded the data, move your data file to your preferred data location.
2. In your Rmd, write code to load your packages. If you load packages in the console, you will get an error when you knit because knitting starts a fresh R session.
3. Load `data_traffic.csv` and assign it to the name `traffic_data`. This data was scrapped from the UCPD website and partially cleaned by Prof. Jones.
4. Recall that `group_by()` operates silently. Below I create a new data frame called `grouped_data`.

```
grouped_data <-  
  traffic_data %>%  
    group_by(Race, Gender)
```

- a. How can you tell `grouped_data` is different from `traffic_data`?
 - b. How many groups (Race-Gender pairs) are in the data? (This information should be available without writing additional code!)
 - c. Without running the code, predict the dimensions (number of rows by number of columns) of the tibbles created by `traffic_data %>% summarize(n = n())` and `grouped_data %>% summarize(n = n())`.
 - d. Now check you intuition by running the code.
5. Use `group_by()` and `summarize()` to recreate the following table.

```
## # A tibble: 6 x 2  
##   Race                                n  
##   <chr>                               <int>  
## 1 African American                   3278  
## 2 American Indian/Alaskan Native     12  
## 3 Asian                               226  
## 4 Caucasian                           741  
## 5 Hispanic                             217  
## 6 Native Hawaiian/Other Pacific Islander 4
```

6. Use `count()` to produce the same table.

Moving beyond counts

1. Raw counts are okay, but frequencies (or proportions) are easier to compare across data sets. Add a column with frequencies and assign the new tibble to the name `traffic_stop_freq`. The result should be identical to Prof. Jones's analysis on twitter.

Try on your own first. If you're not sure how to add a frequency though, you could google "add a proportion to count with tidyverse" and find this [stackoverflow post](#). Follow the advice of the number one answer. The green checkmark and large number of upvotes indicate the answer is likely reliable.

2. The frequencies out of context are not super insightful. What additional information do we need to argue the police are disproportionately stopping members of a certain group? (Hint: Prof. Jones shares

the information in his tweets.)¹

3. For the problem above, your groupmate tried the following code. Explain why the frequencies are all 1.²

```
traffic_stop_freq_bad <-  
traffic_data %>%  
  group_by(Race) %>%  
  summarize(n = n(),  
            freq = n / sum(n))  
  
traffic_stop_freq_bad
```

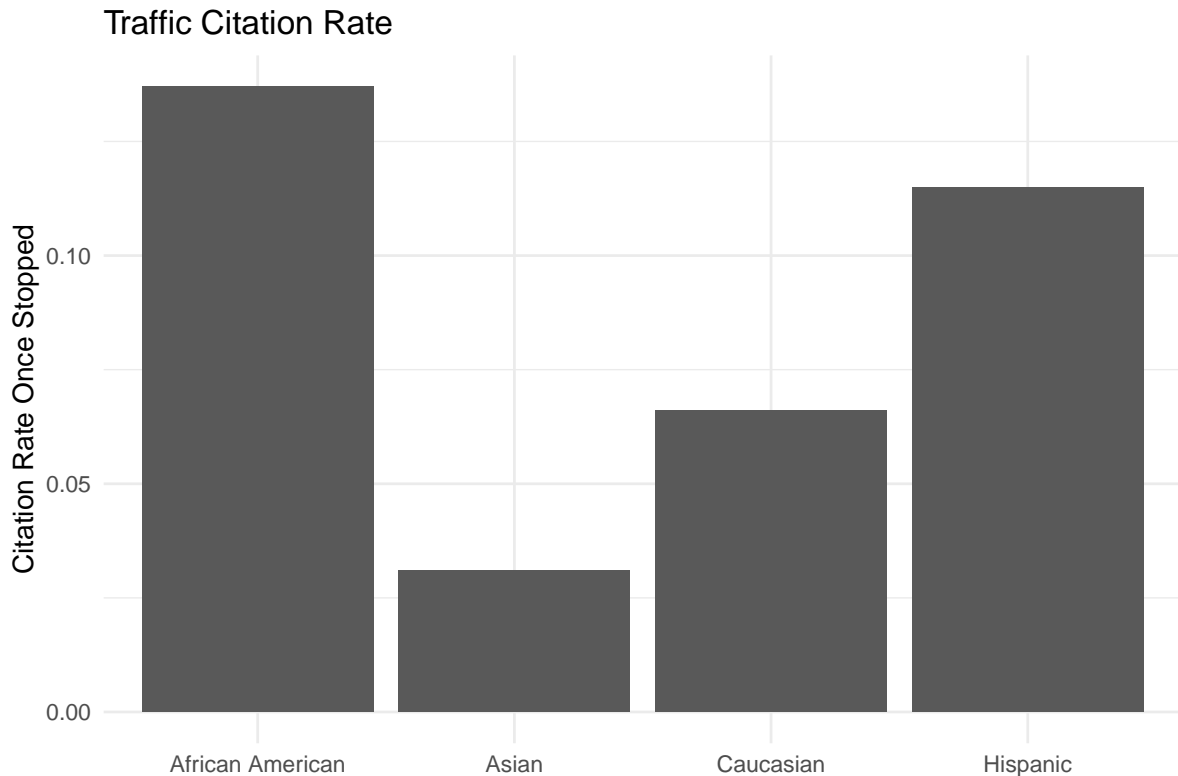
4. Now we want to go a step further.³ Do outcomes differ by race? In the first code block below, I provide code so you can visualize disposition by race. “Disposition” is police jargon that means the current status or final outcome of a police interaction.

```
citation_strings <- c("citation issued", "citations issued", "citation issued" )  
  
arrest_strings <- c("citation issued, arrested on active warrant",  
                  "citation issued; arrested on warrant",  
                  "arrested by cpd",  
                  "arrested on warrant",  
                  "arrested",  
                  "arrest")  
  
disposition_by_race <-  
  traffic_data %>%  
    mutate(Disposition = str_to_lower(Disposition),  
           Disposition = case_when(Disposition %in% citation_strings ~ "citation",  
                                   Disposition %in% arrest_strings ~ "arrest",  
                                   TRUE ~ Disposition)) %>%  
    count(Race, Disposition) %>%  
    group_by(Race) %>%  
    mutate(freq = round(n / sum(n), 3))  
  
disposition_by_race %>%  
  filter(n > 5, Disposition == "citation") %>%  
  ggplot(aes(y = freq, x = Race)) +  
  geom_col() +  
  labs(y = "Citation Rate Once Stopped", x = "", title = "Traffic Citation Rate") +  
  theme_minimal()
```

¹To be fair, even with this information, this is crude evidence that can be explained away in any number of ways. One job of a policy analyst is to bring together evidence from a variety of sources to better understand the issue.

²Hint: This is a lesson about `group_by()`!

³The analysis that follows is partially inspired by Eric Langowski, a Harris alum, who was also inspired to investigate by the existence of this data (You may have seen Prof. Jones retweet him at the end of the thread.)



Let's break down how we got to this code. First, I ran `traffic_data %>% count(Race, Disposition)` and noticed that we have a lot of variety in how officers enter information into the system.⁴ I knew I could deal with some of the issue by standardizing capitalization.

- a. In the console, try out `str_to_lower(...)` by replacing the `...` with different strings. The name may be clear enough, but what does `str_to_lower()` do?⁵

After using `mutate` with `str_to_lower()`, I piped into `count()` again and looked for strings that represent the same `Disposition`. I stored terms in character vectors (e.g. `citation_strings`). The purpose is to make the `case_when()` easier to code and read. Once I got that right, I added frequencies to finalize `disposition_by_race`.

5. To make the graph, I first tried to get all the disposition data on the same plot.

```
disposition_by_race %>%
  ggplot(aes(y = freq, x = Race, fill = Disposition)) +
  geom_col()
```

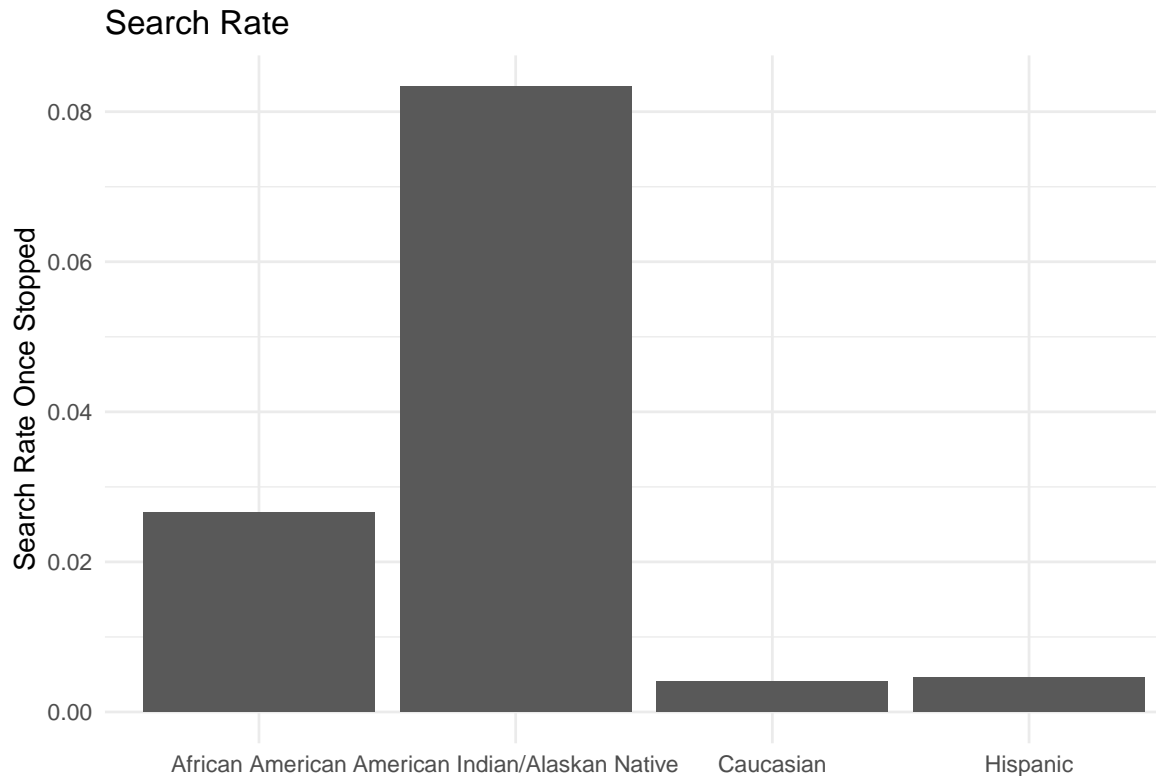
By default, the bar graph is stacked. Look at the resulting graph and discuss the pros and cons of this plot with your group.

6. I decided I would focus on citations only and added the `filter(n > 5, Disposition == "citation")` to the code.⁶ What is the impact of filtering based on `n > 5`? Would you make the same choice? This question doesn't have a "right" answer. You should try different options and reflect.
7. Now, you can create a similar plot based called "Search Rate" using the `Search` variable. Write code to re-

⁴Try it yourself!

⁵This code comes from the `stringr` package. Checkout `?str_to_lower` to learn about some related functions.

⁶Notice that I get the data exactly how I want it using `dplyr` verbs and then try to make the graph.



produce this plot.

Extension: Revisiting world inequality data

When we explored the World Inequality Database data in lab 1, we mimicked grouped analysis by filtering the data to only show data for France and then repeated the analysis for Russia. Using `group_by()`, we can complete the analysis for each country simultaneously.

1. Read in the `wid_data`.⁷

```
wid_data_raw <-
  # You will like have to adjust the file path
  readxl::read_xlsx("../data/world_wealth_inequality.xlsx",
    col_names = c("country", "indicator", "percentile", "year", "value")) %>%
  separate(indicator, sep = "[\\r]?\\n", into = c("row_tag", "type", "notes"))

wid_data <- wid_data_raw %>%
  select(-row_tag) %>%
  select(-notes, everything()) %>%
  # some students had trouble because excel added "\\r" to the end
  # of each string. mutate standardizes the string across platforms.
  mutate(type = ifelse(str_detect(type, "Net personal wealth"),
    "Net personal wealth", type)) %>%
  filter(type == "Net personal wealth")
```

2. Create a table that tells us the number of years observed per country and first and last year we have data⁸ for each country.⁹ For example, India has 6 years of observations and has data from 1961 to 2012.

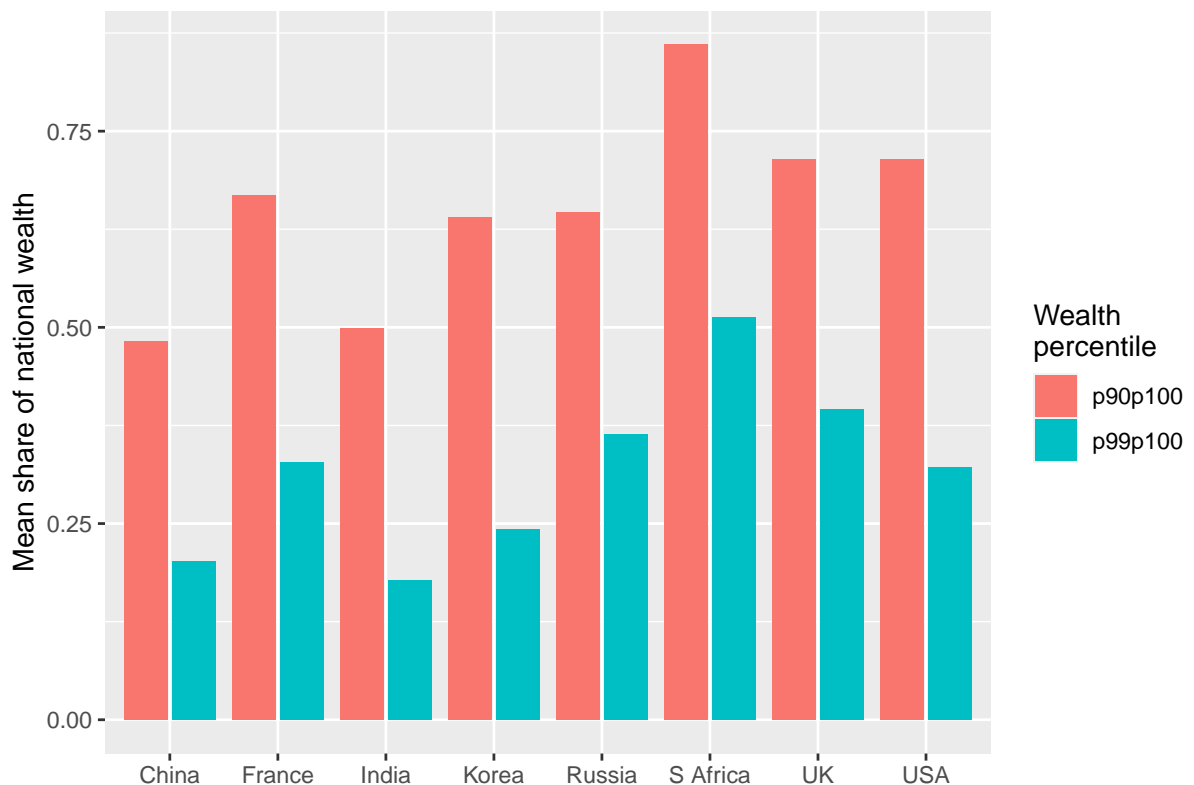
⁷If you are still having trouble, you may want to re-download the file [here](#) and do not open with Excel!

⁸i.e. not NAs.

⁹Hint: `?summarize` lists “Useful functions” for summarizing data. Look at the “Range” or “Position” functions. If you are going to use the “Position” functions, make sure the data is sorted properly.

3. Create a table that provides the mean and standard deviation of the share of wealth owned by the top 10 percent and top 1 percent for each country. Call the resulting tibble `mean_share_per_country`.
 - a. Which country has the smallest standard deviation in share of wealth owned by the top 10 percent? Use `arrange()` to order the countries by standard deviation. Compare the order to your results above about the number of observation and time horizon.¹⁰
 - b. If your code worked, you should be able to make this bar chart.

```
mean_share_per_country %>%
  mutate(country = case_when(country == "Russian Federation" ~ "Russia",
                             country == "United Kingdom" ~ "UK",
                             country == "South Africa" ~ "S Africa",
                             TRUE ~ country)) %>%
  ggplot(aes(x = country, y = mean_share, fill = percentile)) +
  geom_col(position = "dodge2") +
  labs(y = "Mean share of national wealth", x = "", fill = "Wealth\npercentile")
```



4. **Challenge** Write code to create `mean_share_per_country_with_time` a tibble that produces the following graph which lets us see how the share of national wealth held by the top 10 and 1 percent change over time.¹¹

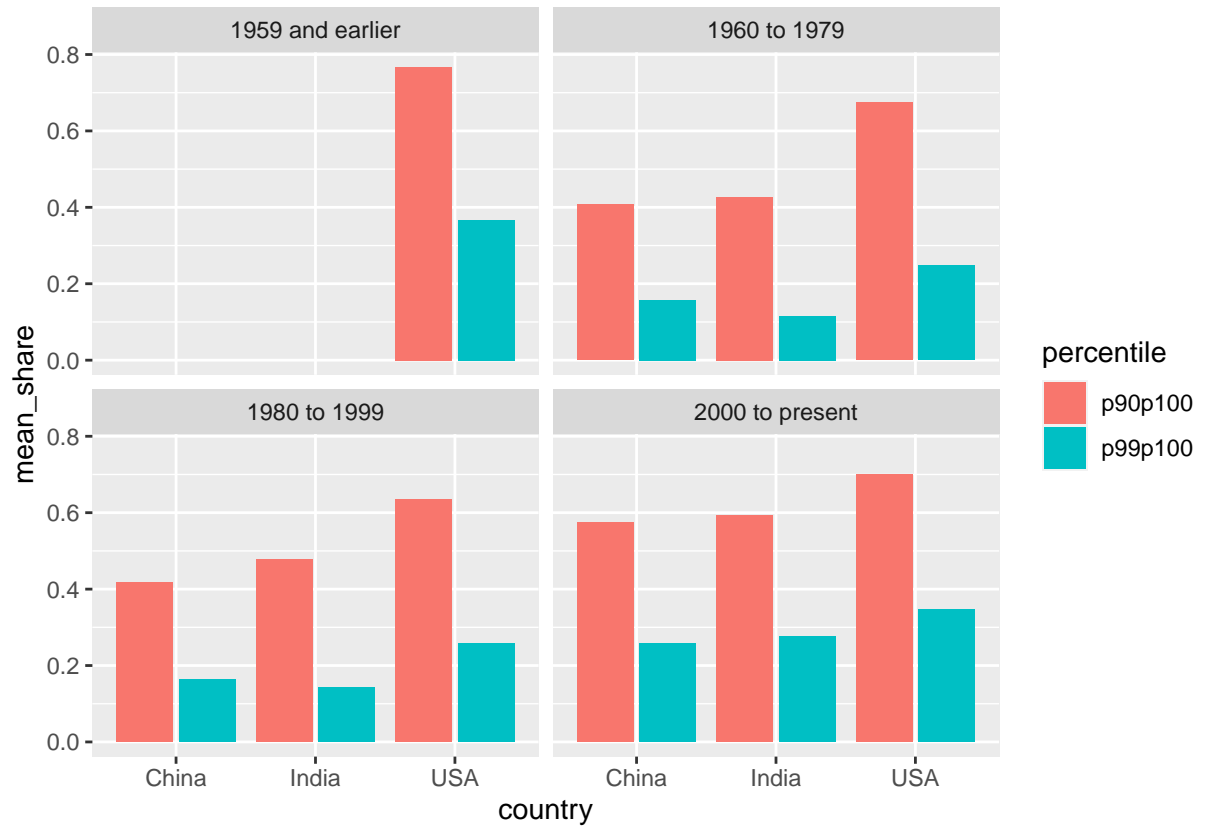
```
mean_share_per_country_with_time %>%
  ggplot(aes(x = country, y = mean_share, fill = percentile)) +
```

¹⁰Usually when we get more data we expect variances to decrease, but that reasoning assumes independence between observations. In this case, there is high temporal correlation, which means if the top 10 percent own 50 percent of wealth this year, they'll own some proportion near 50 percent in the next year. South Korea only has observations during a short and highly stable historical period, so that explains the low variance.

¹¹Hint: use `case_when` or several `ifelse` to create a new column called `time_period` that labels data as "1959 and earlier", "1960 to 1979", "1980 to 1999", or "2000 to present". Then, add `time_period` to your `group_by()` along with other relevant grouping variables.

```
geom_col(position = "dodge2") +  
facet_wrap(~time_period)
```

```
## Warning: Removed 4 rows containing missing values (geom_col).
```



Want to improve this tutorial? Report any suggestions/bugs/improvements on [here!](#) We're interested in learning from you how we can make this tutorial better.