BLM Protests Data Project

PLSC31101 Final Project

Neomi Rao

December 11, 2020

Brief Background/Goal

The first wave of Black Lives Matter (BLM) protests in 2014 happened after the police murders of Eric Garner and Michael Brown. The second wave of BLM protests in 2020 began after the police murders of George Floyd and Breonna Taylor. Previous research has linked the geographic locations of the 2014 wave of protests to overall reduction in racial resentment in those areas (Mazumder, 2019). Other research has also suggested that local protest activity may be related to rates of voter registration, support for movement goals (e.g. defunding police), and awareness of movement frames.

I wanted to collect BLM protest data by date and location with 2 primary goals:

1. Create pretty visualizations of BLM protest location data
2. Prepare a dataset for future analysis against the various measures mentioned above

Collecting Data

Data collection was probably the most challenging part of this project. I collected data via 2 main steps:

1. Scraping BLM protest data via Selenium
2. Adding latlong geolocation data via the Google Map API

```r
# Load required libraries
library(RSelenium)
library(wdman)
library(tidyverse)
library(rvest)
library(lubridate)
library(stringr)
library(ggmap)
library(purrr)
library(sp)
library(maptools)
```

Webscraping with Selenium

Selenium is a web browser automation tool that acts as a real user who is surfing the web. Its primary use is front-end website testing, but it can also use for web scraping.

Why use Selenium instead of RVest? Elephrame, the site hosting the BLM protest data, uses dynamic JavaScript, so RVest is not sufficient.

See Scraping.R for code.

```r
# load the raw data from webscraping
blm_df_distinct <- read.csv("data/BLM_data.csv", header = TRUE)
```

Geolocation: Google Map API

I used the ggmap package to get latitude and longitude for locations from the Google Maps API. To use ggmap, I needed to set up a Google Cloud Console account and get my unique Google API key. These instructions describe how to do the set-up for ggmap. Note that Google requires payment information and only gives you $200 worth of free API usage per month, so be careful not to overuse the API.

See Geolocation.R for code.

```
# load the dataframe with longlat geolocation data
blm_df_longlat <- read.csv("data/BLM_data_longlat.csv", header = TRUE)
```

Cleaning/Pre-processing Data

To pre-process the data for visualization and further analysis, I aggregated the dates for each protest to a week/year and year format. I also split the data into datasets for only US and only non-US protests. Finally, I created a dataset with county data of only US protests from 2014 and from after May 2020 for future comparative analyses of the 2 waves of BLM protests.

See Preprocessing.R for code.

```
library(maps)
# load the dataframe with weekly data & adjust date columns
blm_df_ll_wk <- read.csv("data/BLM_data_weeks.csv", header = TRUE)
blm_df_ll_wk <- blm_df_ll_wk %>%
  mutate(date = ymd(date),
         weekof = ymd(weekof))
```

```
# load the dataframe with US-only data & adjust date columns
blm_df_usa <- read.csv("data/BLM_data_USA.csv", header = TRUE)
blm_df_usa <- blm_df_usa %>%
  mutate(date = ymd(date),
         weekof = ymd(weekof))
```

```
# load the dataframe with comparative county data
# adjust date columns, make into spatial sf data (must load library(sf))
library(sf)
blm_df_county <- read.csv("data/BLM_data_county.csv", header = TRUE)
blm_df_county <- blm_df_county %>%
  mutate(date = ymd(date),
         weekof = ymd(weekof))
```

```
# load the dataframe with 2014 protest counts data
blm_2014 <- read.csv("data/BLM_counts_2014.csv", header = TRUE)
# replace leading 0s on fips codes removed by csv format, and make into character format
blm_2014$county_fips <- str_pad(blm_2014$county_fips, width = 5, side = "left", pad = "0")
```

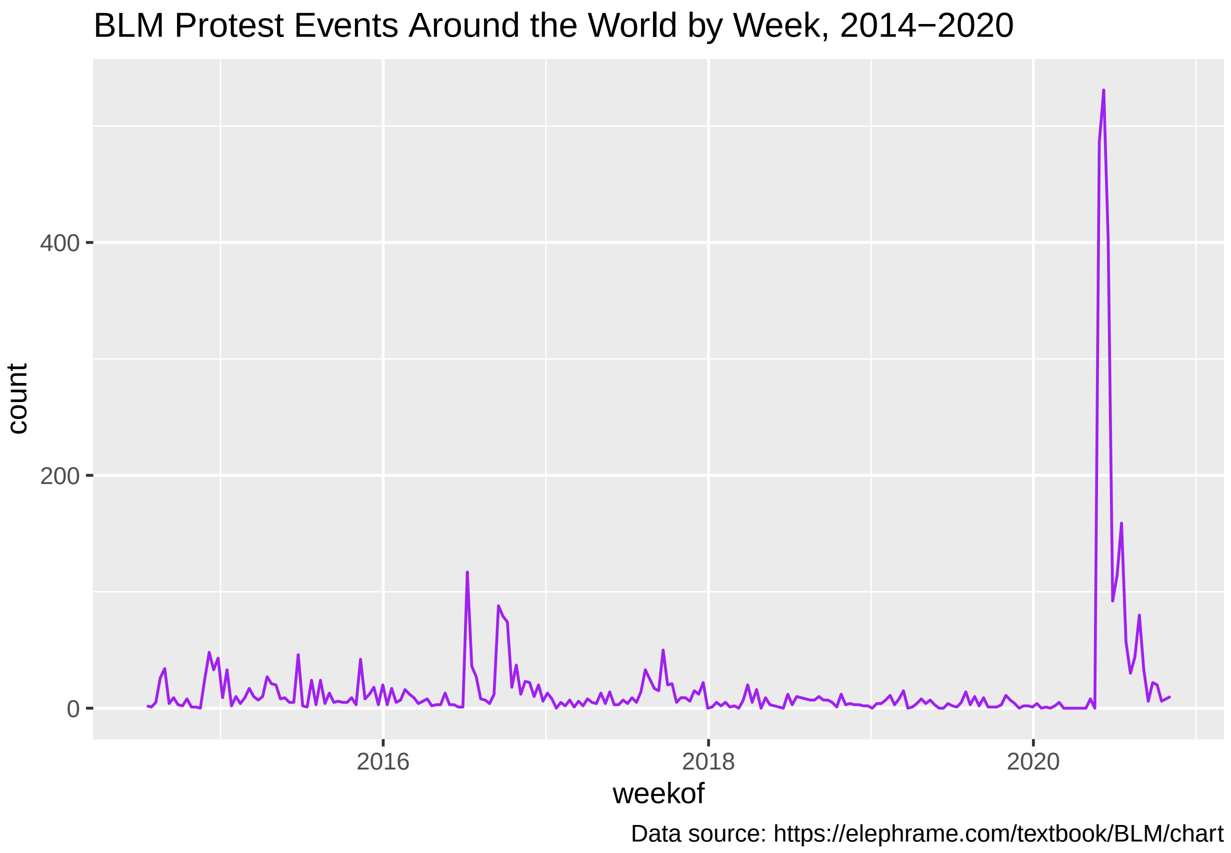
```
# load the dataframe with 2020 protest counts data
blm_2020 <- read.csv("data/BLM_counts_2020.csv", header = TRUE)
# replace leading 0s on fips codes removed by csv format, and make into character format
blm_2020$county_fips <- str_pad(blm_2020$county_fips, width = 5, side = "left", pad = "0")
```

Analysis and Visualization

First, I quickly checked the frequency of all protest events over time, to see if it is as expected.

```
ggplot(data = blm_df_ll_wk, aes(x=weekof)) +
  geom_line(aes(fill=..count..), stat="bin", binwidth=10, color = "purple") +
  labs(title = "BLM Protest Events Around the World by Week, 2014-2020",
       caption = "Data source: https://elephrame.com/textbook/BLM/chart")
```

```
## Warning: Ignoring unknown aesthetics: fill
```



But I wanted to map this data not only by time but also by geospatial location.

Static Map: World

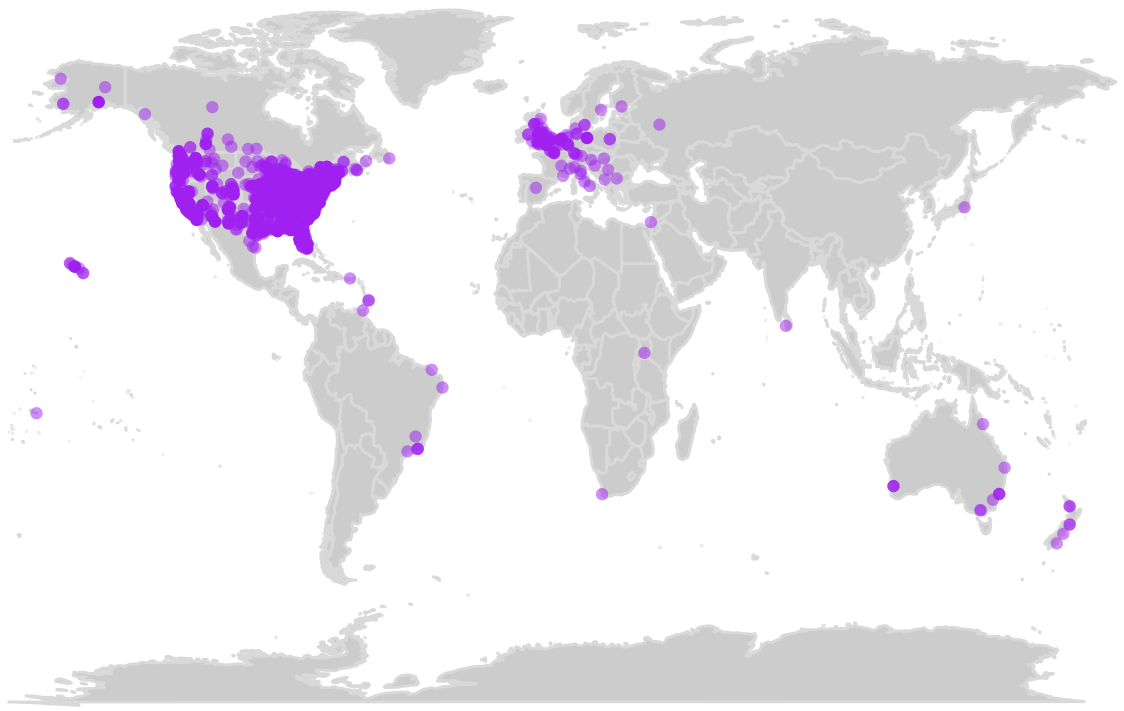
To start with visualization, I made a static map of protest locations around the world, using the ggplot2 and maps packages.

```
library(ggplot2)
library(maps)
library(ggthemes)
```

```
world <- ggplot() +
  borders("world", colour = "gray85", fill = "gray80") +
  borders("state", colour = "gray85", fill = "gray80") +
  theme_map()

# Map worldwide BLM protests by
map <- world +
  geom_point(aes(x = lon, y = lat),
             data = blm_df_ll_wk,
             colour = 'purple', alpha = .5) +
  labs(title = "BLM Protest Events Around the World, 2014-2020",
       caption = "Data source: https://elephrame.com/textbook/BLM/chart")
```

```
map
```

BLM Protest Events Around the World, 2014−2020



Data source: https://elephrame.com/textbook/BLM/chart

Now, I had separate temporal and spatial visualizations of the data, but neither were satisfactory for showing BLM protest locations over time.

Map GIF: World

Then, I used gganimate and gifski packages to make a GIF of an animated visualization to show protest locations around the world over time.

```
library(gganimate)
library(gifski)
library(glue)
library(plotly)
```

Note that the animate() step takes several minutes to run as the frames are rendered, and GIFs are tricky to knit to a PDF, so this code is suppressed when knit to RMarkdown. GIFs can be found in the results folder.

```
# Create the base animation
blm_map <- map +
  # add animation by specifing what is changing (week of the year)
  transition_manual(frames = weekof) +
  labs(title = "BLM Protest Events Around the World, 2014-2020",
       subtitle = 'Week of {current_frame}',
       caption = "Data source: https://elephrame.com/textbook/BLM/chart")
# Make a GIF - Note, this takes a while to run
BLM_map_gif_week <- animate(blm_map, nframes = length(unique(blm_df_ll_wk$weekof)),
                            fps = 5, renderer = gifski_renderer(), end_pause = 20,
                            height = 600, width = 800)
```

```r
# Save the GIF
anim_save("results/BLMmap_world.gif", animation = BLM_map_gif_week)
```

Map GIF: United States

I also created a GIF for US-only BLM protests. (This code is also suppressed for knitting to RMarkdown)

```r
usa <- ggplot() +
  borders("world", c("USA", "hawaii"), colour = "gray85", fill = "gray80") +
  borders("state", colour = "gray85", fill = "gray80") +
  theme_map() +
  coord_cartesian(xlim= c(-170, -65)) #limit to just US + HI + AK
# Create base animation
blm_usa_map <- usa +
  geom_point(aes(x = lon, y = lat),
             data = blm_df_usa, colour = 'purple', alpha = .5) +
  # add animation by specifing what is changing (dates)
  transition_manual(frames = weekof) +
  labs(title = "BLM Protest Events Around in the US, 2014-2020",
       subtitle = 'Week of {current_frame}',
       caption = "Data source: https://elephrame.com/textbook/BLM/chart")
# Make GIF
BLM_usa_gif <- animate(blm_usa_map, nframes = length(unique(blm_df_usa$weekof)),
                       fps = 5, renderer = gifski_renderer(), end_pause = 20)
BLM_usa_gif
# Save GIF
anim_save("results/BLMmap_usa.gif", animation = BLM_usa_gif)
```

Cloropleth Maps: US Counties, 2014 & 2020

First, we have to do a spatial join on the 2014 & 2020 protest couts data to get mappable data.

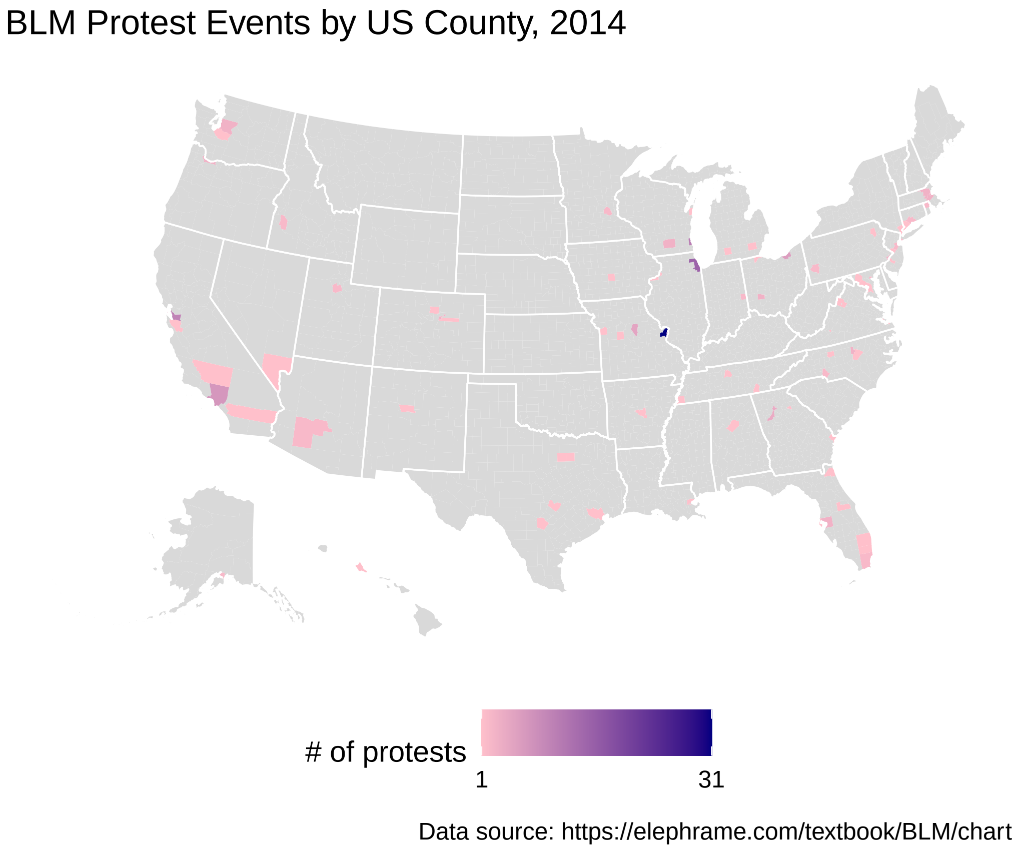
```r
library(urbnmapr)
# Get spatial data for all US states and counties
states_sf <- get_urbn_map(map = "states", sf = TRUE)
counties_sf <- get_urbn_map(map = "counties", sf = TRUE)

# Create a dataframe of all US counties with counts of BLM protests for 2014
blm_count_2014 <-
  left_join(counties_sf, # spatial data for all US counties
            blm_2014, # counts of BLM protests by county
            by = "county_fips") # join by the unique county FIPS code

# Create a dataframe of all US counties with counts of BLM protests for 2014
blm_count_2020 <-
  left_join(counties_sf, # spatial data for all US counties
            blm_2020, # counts of BLM protests by county
            by = "county_fips") # join by the unique county FIPS code
```

Using the spatial US county map data created during pre-processing, I used the urbnmapr package to create a cloropleth map to show BLM protest density in 2014.

```r
# Plot the map with sf spatial layers
blm_2014_map <- ggplot() +
  # Plot county cloropleth data
  geom_sf(data = blm_count_2014, mapping = aes(fill = protests), color = NA) +
  # Overlay state outlines
  geom_sf(data = states_sf, fill = NA, color = "#ffffff", size = 0.3) +
  # Add scale, themes, labels
  coord_sf(datum = NA) +
  scale_fill_gradient(name = "# of protests",
                      low='pink', high='navyblue', na.value="gray85",
                      breaks=c(1, max(blm_count_2014$protests, na.rm=T))) +
  theme_bw() +
  theme(legend.position="bottom", panel.border = element_blank()) +
  labs(title = "BLM Protest Events by US County, 2014",
       caption = "Data source: https://elephrame.com/textbook/BLM/chart")
# Save map as vector image
ggsave(filename = "results/BLM_Map_County_2014.pdf", plot = blm_2014_map)
```

```
## Saving 6.5 x 4.5 in image
```

```r
blm_2014_map
```

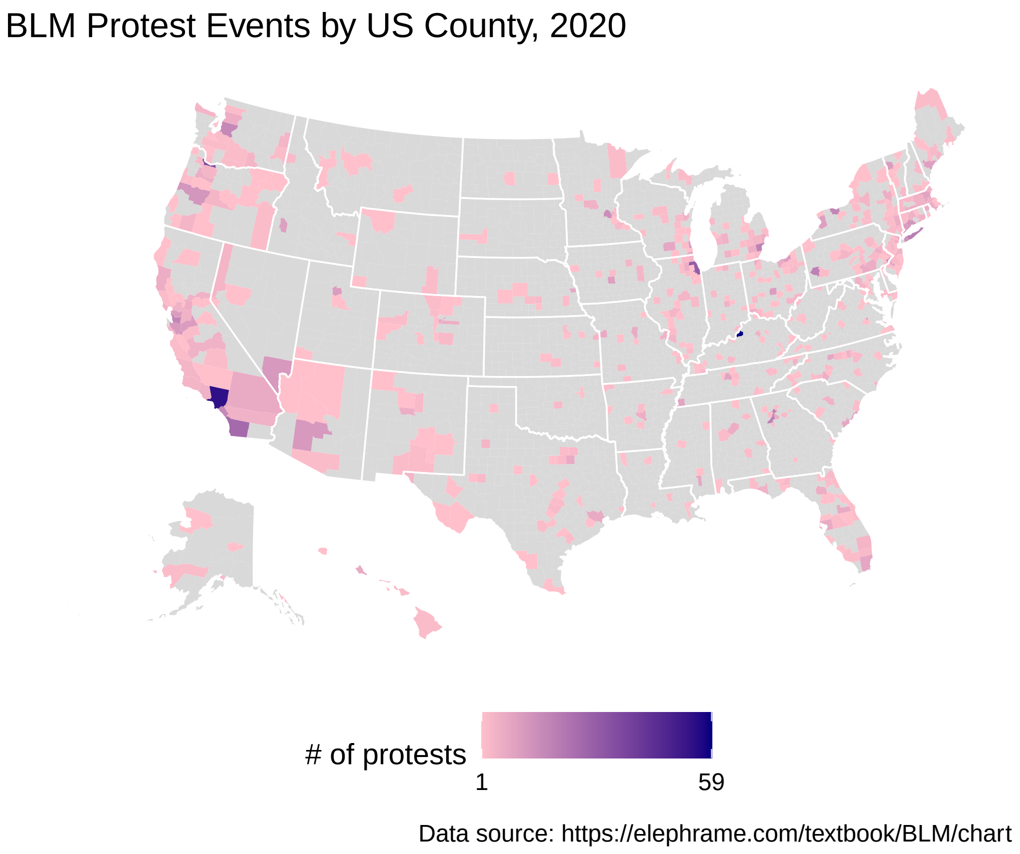

BLM Protest Events by US County, 2014
# of protests
1
31
Data source: https://elephrame.com/textbook/BLM/chart

I did the same thing for 2020 data.

```r
# Plot the map with sf spatial layers
blm_2020_map <- ggplot() +
  # Plot county cloropleth data
  geom_sf(data = blm_count_2020, mapping = aes(fill = protests), color = NA) +
  # Overlay state outlines
  geom_sf(data = states_sf, fill = NA, color = "#ffffff", size = 0.3) +
  # Add scale, themes, labels
  coord_sf(datum = NA) +
  scale_fill_gradient(name = "# of protests",
                      low='pink', high='navyblue', na.value="gray85",
                      breaks=c(1, max(blm_count_2020$protests, na.rm=T))) +
  theme_bw() +
  theme(legend.position="bottom", panel.border = element_blank()) +
  labs(title = "BLM Protest Events by US County, 2020",
       caption = "Data source: https://elephrame.com/textbook/BLM/chart")
# Save map as vector image
ggsave(filename = "results/BLM_Map_County_2020.pdf", plot = blm_2020_map)
```

```
## Saving 6.5 x 4.5 in image
```

```r
blm_2020_map
```


BLM Protest Events by US County, 2020
# of protests
1
59
Data source: https://elephrame.com/textbook/BLM/chart

Future work

There are many future directions that this data project could take. Just a few include:

1. Creating an R Shiny app to allow easy, interactive visualization of the spatiotemporal data
    - Including leaflet functionality for interactive maps
    - Automatically updating data from the source website as new events are added
2. Extending Mazumder, 2019 to see whether the link between BLM protests and reduced racial resentment continues to be shown in 2020.
3. Compare protest location data with:
    - Support for BLM and movement demands by location, before and after the 2020 wave.
    - Voter registration numbers by location.
    - Awareness about BLM (via geo-located Google Trends)