Pub460 Lab2

Joseph Yang

2/24/2021

Introduction: This section should summarize and provide a link to the original post. The link of original post: https://www.r-bloggers.com/2021/01/covidcast-package-for-covid-19-related-data/ The original article is about an analysis of what you can do with the covidcast package with multiple data visualizations.

Analysis: The package is not available on CRAN yet but can be downloaded from Github using the devtool: devtools::install_github("cmu-delphi/covidcast", ref = "main", subdir = "R-packages/covidcast")

We first set up all the libraries that we need. The code below pulls data on cumulative COVID cases per 100k people on 2020-12-31 at the county level. The function covidcast_signal is use for pulling data, and it returns an object of class c("covidcast_signal", "data.frame").

```
library(covidcast)
```

```
## We encourage COVIDcast API users to register on our mailing list:
## https://lists.andrew.cmu.edu/mailman/listinfo/delphi-covidcast-api
## We'll send announcements about new data sources, package updates,
## server maintenance, and new features.
```

```
# Cumulative COVID cases per 100k people on 2020-12-31
df <- covidcast_signal(data_source = "usa-facts",
                       signal = "confirmed_cumulative_prop",
                       start_day = "2020-12-31", end_day = "2020-12-31")
```

```
## Fetched day 2020-12-31 to 2020-12-31: num_entries = 3142
```

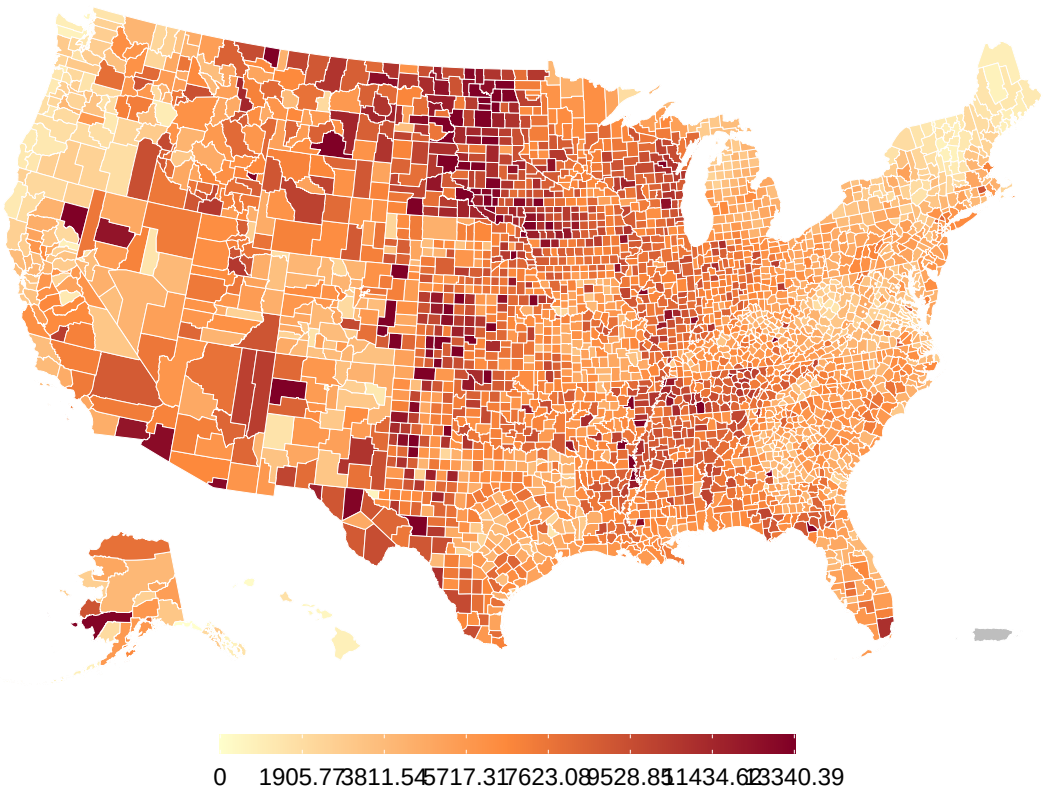
Let's take a look of what we actually have right now and have a brief understanding of what the data contains.

```
summary(df)
```

```
## A `covidcast_signal` data frame with 3142 rows and 9 columns.
##
## data_source : usa-facts
## signal      : confirmed_cumulative_prop
## geo_type    : county
##
## first date                          : 2020-12-31
## last date                           : 2020-12-31
## median number of geo_values per day : 3142
```

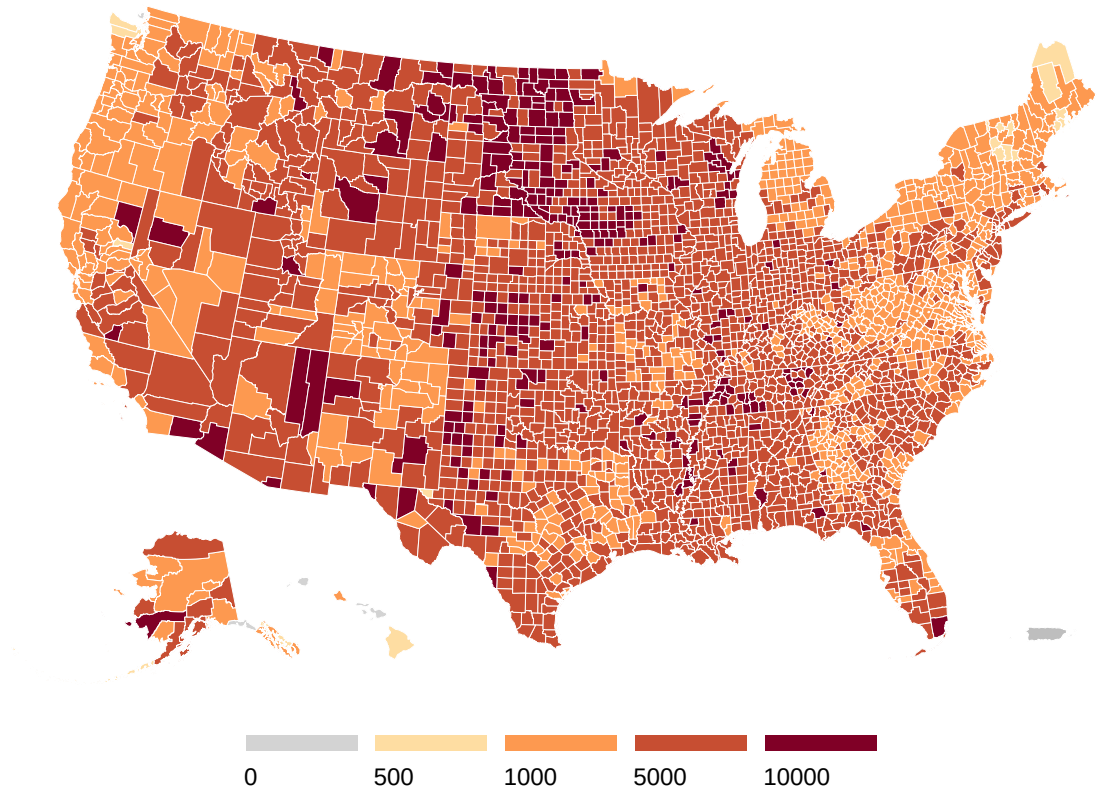
Next we check out what we get using the plot function.

```
plot(df)
```



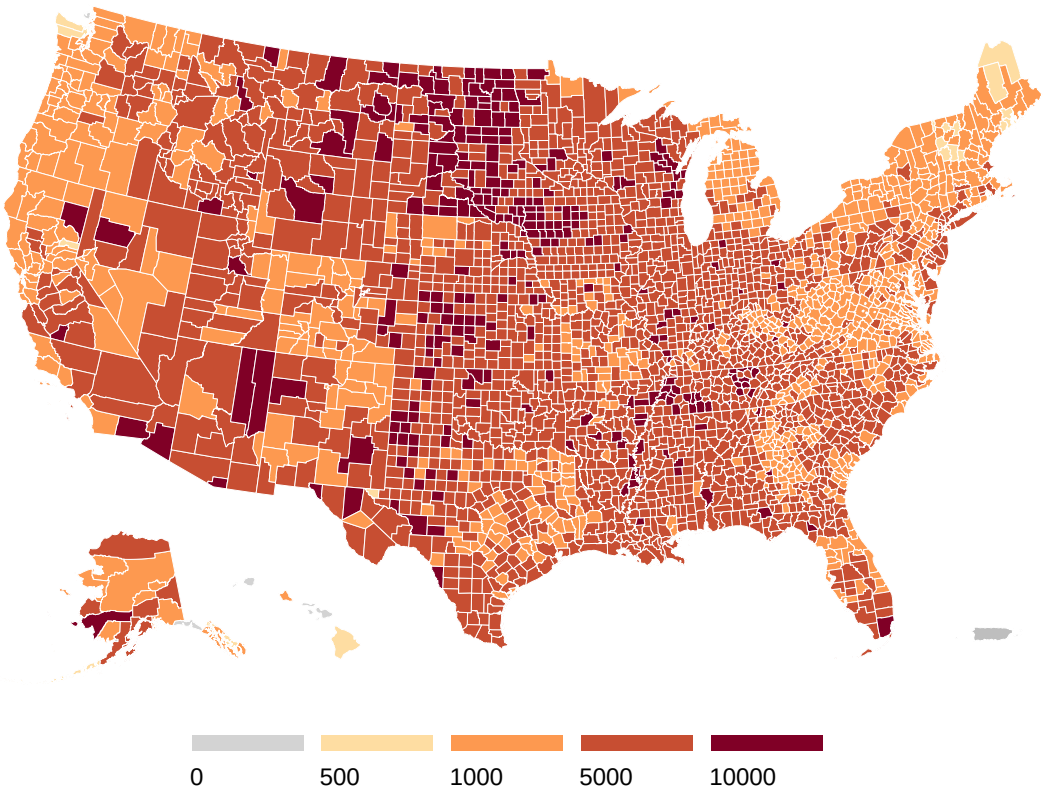
With more specification we can do things differently maybe by breaking it up by 0, 500, 1000, 5000 and 10000.

```
breaks <- c(0, 500, 1000, 5000, 10000)
colors <- c("#D3D3D3", "#FEDDA2",  "#FD9950", "#C74E32", "#800026")
plot(df, choro_col = colors, choro_params = list(breaks = breaks),
     title = "Cumulative COVID cases per 100k people on 2020-12-31")
```



The plot returned is actually created using the ggplot2 package, so we can regenerate the same thing out of ggplot:

```
library(ggplot2)
plot(df, choro_col = colors, choro_params = list(breaks = breaks),
     title = "Cumulative COVID cases per 100k people on 2020-12-31") +
  theme(title = element_text(face = "bold"))
```



Follow-up: Instead of visualizing all the COVID cases I think visualizing the number of confirmed deaths would also be helpful for people to see how this tragic pandemic has impacted us so far. We start by extracting the data from the API and this time we are extracting from the JUH data source which is made available by the Center for Systems Science and Engineering at Johns Hopkins University. We name it "df_deaths" and we'll take a look at the data on 2020-11-22.

```
df_deaths <- covidcast_signal(data_source = "jhu-csse",
                  signal = "deaths_cumulative_num",
                  start_day = "2020-11-22", end_day = "2020-11-22")
```

```
## Fetched day 2020-11-22 to 2020-11-22: num_entries = 3276
```

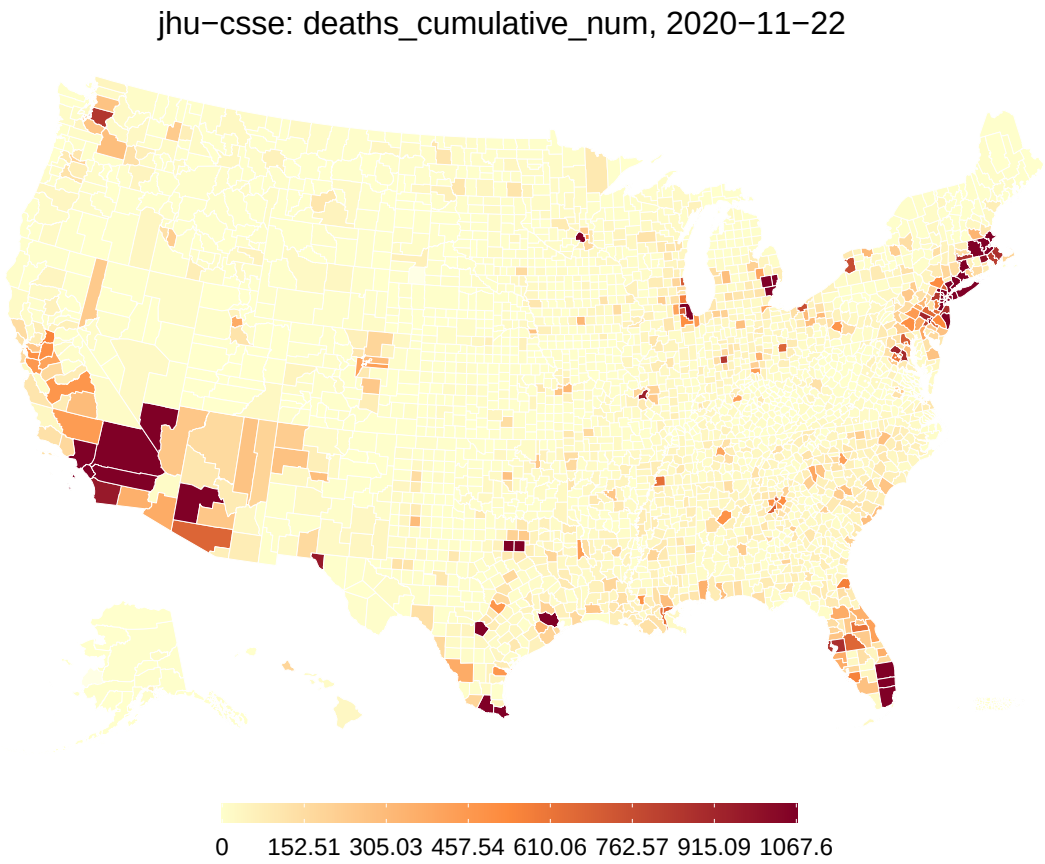
Always good to keep track of what we have currently so we'll pull up the summary function to check out the data we got.

```
summary(df_deaths)
```

```
## A `covidcast_signal` data frame with 3276 rows and 9 columns.
##
## data_source : jhu-csse
## signal      : deaths_cumulative_num
## geo_type    : county
##
## first date                          : 2020-11-22
## last date                           : 2020-11-22
## median number of geo_values per day : 3276
```

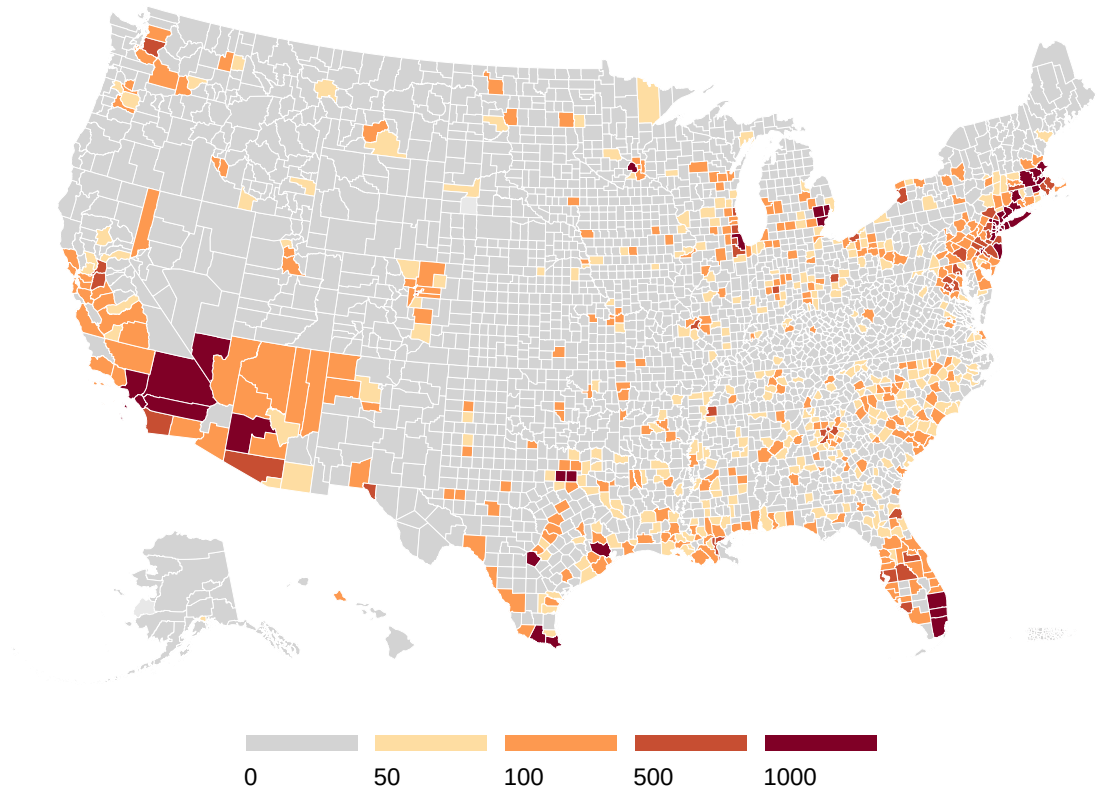
We use the plot function to get some idea of the default setting.

```
plot(df_deaths)
```



Next, we want to customize the mapping a little bit so it's more balanced and straightforward by breaking it up by 0, 50, 100, 500 and 1000 combined with the color of our choice.

```
breaks <- c(0, 50, 100, 500, 1000)
colors <- c("#D3D3D3", "#FEDDA2",  "#FD9950", "#C74E32", "#800026")
plot(df_deaths, choro_col = colors, choro_params = list(breaks = breaks),
     title = "Cumulative number of confirmed deaths due to COVID-19 on 2020-11-22")
```



We could definitly use ggplot to do the same thing and it'll look like this:

```
#library(ggplot2)
#plot(df_deaths, choro_col = colors, choro_params = list(breaks = breaks),
#     title = "Cumulative COVID cases per 100k people on 2020-12-31") +
#  theme(title = element_text(face = "bold"))
```