Creating US Maps3

Tawny Spinelli

2024-12-19

Visualizing Data with plot_usmap()

Below, I've created example code to help visualize national data. You can use this code to set a part states or create heat maps. As there are several ways to create these types of maps in R, my aim is to add helpful examples to the overall pool and to share some of the code I've created.

Helpful Resources

To create the maps below, I relied heavily on information I found on the web and other users' code. I tried to notate those resources throughout this document.

The basics with examples:

https://jtr13.github.io/cc19/different-ways-of-plotting-u-s-map-in-r.html

https://www.rdocumentation.org/packages/usmap/versions/0.7.1/topics/plot_usmap

There are several ways to plot maps. I used plot_usmap() because it included Alaska and Hawaii already in the map and because I could use my ggplot2 knowledge to add layers like text below the map. You can also use ggplot by itself but extra steps are necessary to display Alaska and Hawaii in the plot.

Setting Up

First, load necessary packages. You may need to install if these are not already in your library.

```r
library(ggplot2)
library(usmap)
library(ggthemes)
```

Since this is an ongoing project, I created a dummy database called, "Duplicate Data for Sharing." It's uploaded here. Feel free to download it, set your working directory, and read in the file, if you want to follow along. It should read in as 51 observations of 8 variables.

Base Code for US Maps

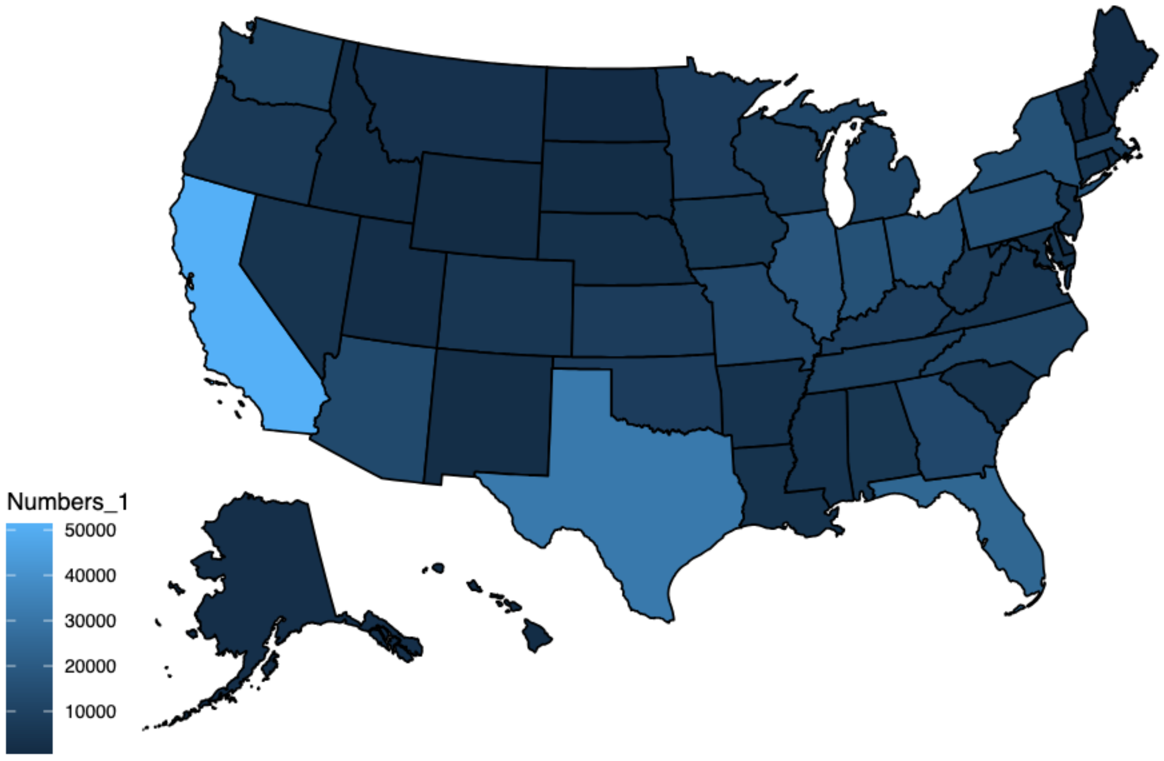
[**Important**]: To use plot_usmap, your database / dataframe *MUST* have a column named “fips” that corresponds to the Federal Information Processing Standards (FIPS) codes that are used by the postal service and for the census. States have 2-digit codes and counties has 5-digit codes. You can find an easily downloadable version of the fips codes here: https://gist.github.com/dantonnoriega/bf1acd2290e15b91e6710b6fd3be0a53.

Be sure that the codes match up to how your states / counties are organized. For instance, D.C. is included in the list. Also, while the list appears to go sequentially, it skips numbers (e.g., no 3). So, as with anything, double check.

The code below is the most basic to generate a map highlighting the values you want to use.

```r
setwd("/Users/tawny/Desktop")
Duplicate_Map_Data <- read.csv("Duplicate Data for Sharing.csv")
```

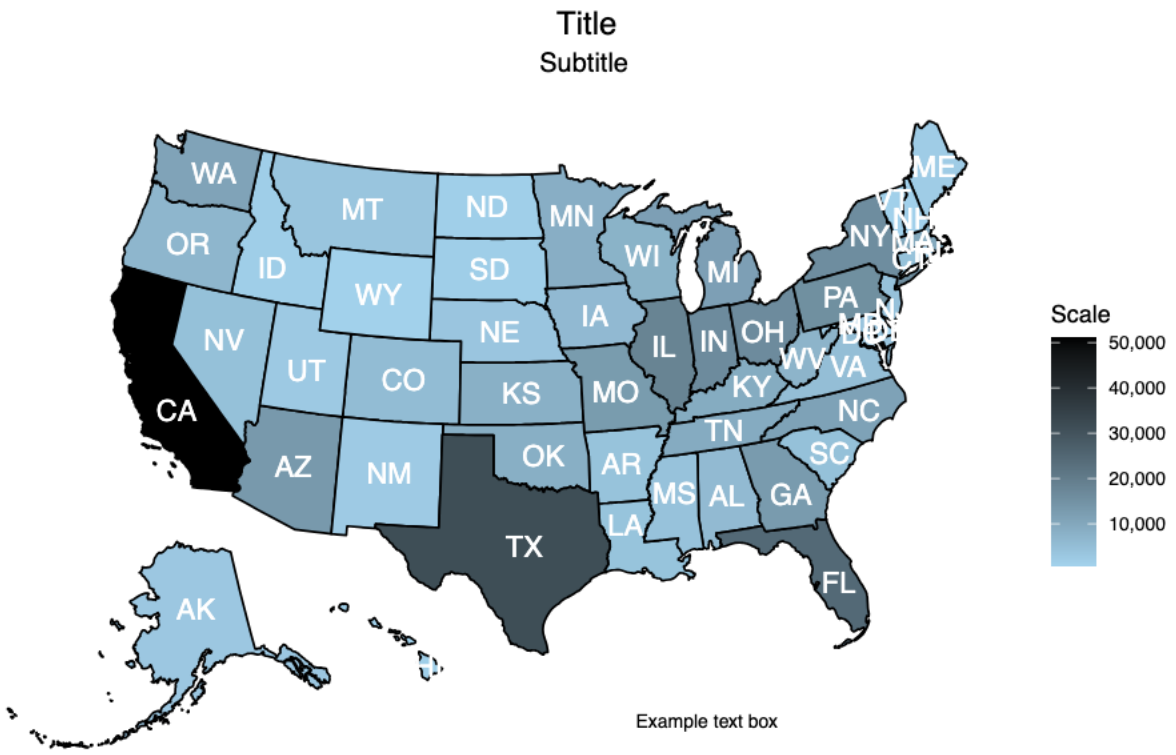
```r
plot_usmap(data = Duplicate_Map_Data, values = "Numbers_1")
```



Map Elements

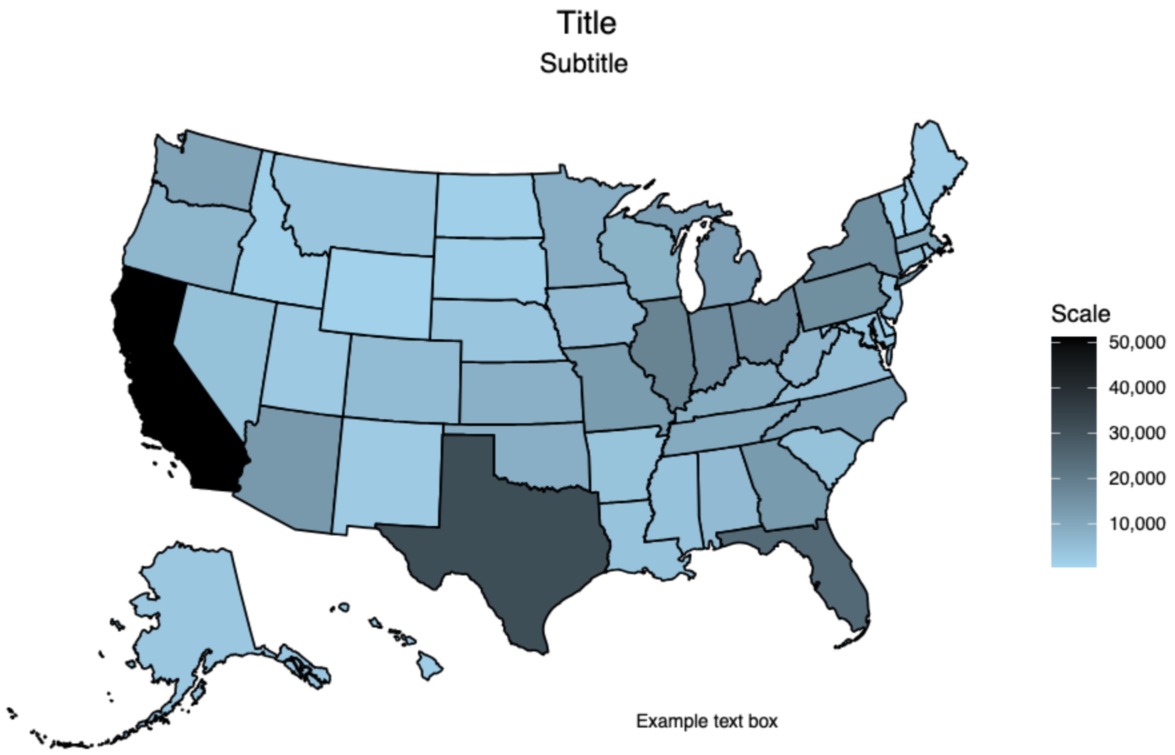
I wanted to add in more elements and make it more aesthetically pleasing. The code below contains several options. You can add additional options or take out options depending on your needs.

```r
frequencies_map1 <-
  plot_usmap(data = Duplicate_Map_Data, values = "Numbers_1", color = "black",
             regions = "states", include = c(), labels = TRUE, label_color = "white") +
  labs(title = "Title") +
  labs(subtitle = "Subtitle") +
  theme(plot.title = element_text(hjust = 0.6)) +
  theme(plot.subtitle = element_text(hjust = 0.6)) +
  theme(plot.title = element_text(size = 12)) +
  theme(plot.subtitle = element_text(size = 10)) +
  scale_fill_continuous(low = "lightskyblue2", high = "black",
                        name = "Scale", label = scales::comma) +
  theme(legend.position = "right") +
  annotate(geom="text", x = 1, y= 1, label= "Example text box",
           color="black", hjust = -1, vjust = 35, size = 2.6)
frequencies_map1
```



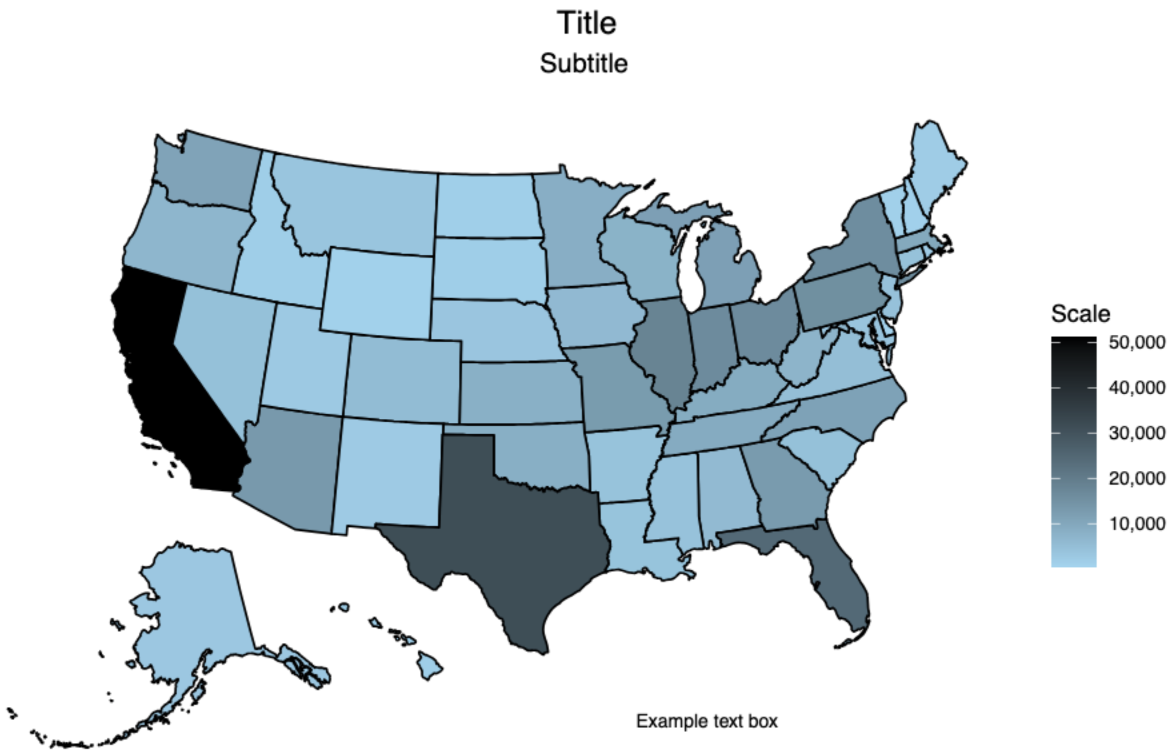
For instance, you can turn off the labels to get rid of the state abbreviations:

```r
frequencies_map2 <-
  plot_usmap(data = Duplicate_Map_Data, values = "Numbers_1", color = "black",
             regions = "states", include = c(), labels = FALSE, label_color = "white") +
  labs(title = "Title") +
  labs(subtitle = "Subtitle") +
  theme(plot.title = element_text(hjust = 0.6)) +
  theme(plot.subtitle = element_text(hjust = 0.6)) +
  theme(plot.title = element_text(size = 12)) +
  theme(plot.subtitle = element_text(size = 10)) +
  scale_fill_continuous(low = "lightskyblue2", high = "black",
                        name = "Scale", label = scales::comma) +
  theme(legend.position = "right") +
  annotate(geom="text", x = 1, y= 1, label= "Example text box",
           color="black", hjust = -1, vjust = 35, size = 2.6)
frequencies_map2
```



You can also just delete that portion of the code as the default is FALSE:

```r
frequencies_map3 <-
  plot_usmap(data = Duplicate_Map_Data, values = "Numbers_1", color = "black",
             regions = "states", include = c()) +
  labs(title = "Title") +
  labs(subtitle = "Subtitle") +
  theme(plot.title = element_text(hjust = 0.6)) +
  theme(plot.subtitle = element_text(hjust = 0.6)) +
  theme(plot.title = element_text(size = 12)) +
  theme(plot.subtitle = element_text(size = 10)) +
  scale_fill_continuous(low = "lightskyblue2", high = "black",
                        name = "Scale", label = scales::comma) +
  theme(legend.position = "right") +
  annotate(geom="text", x = 1, y= 1, label= "Example text box",
           color="black", hjust = -1, vjust = 35, size = 2.6)
frequencies_map3
```



Map Colors

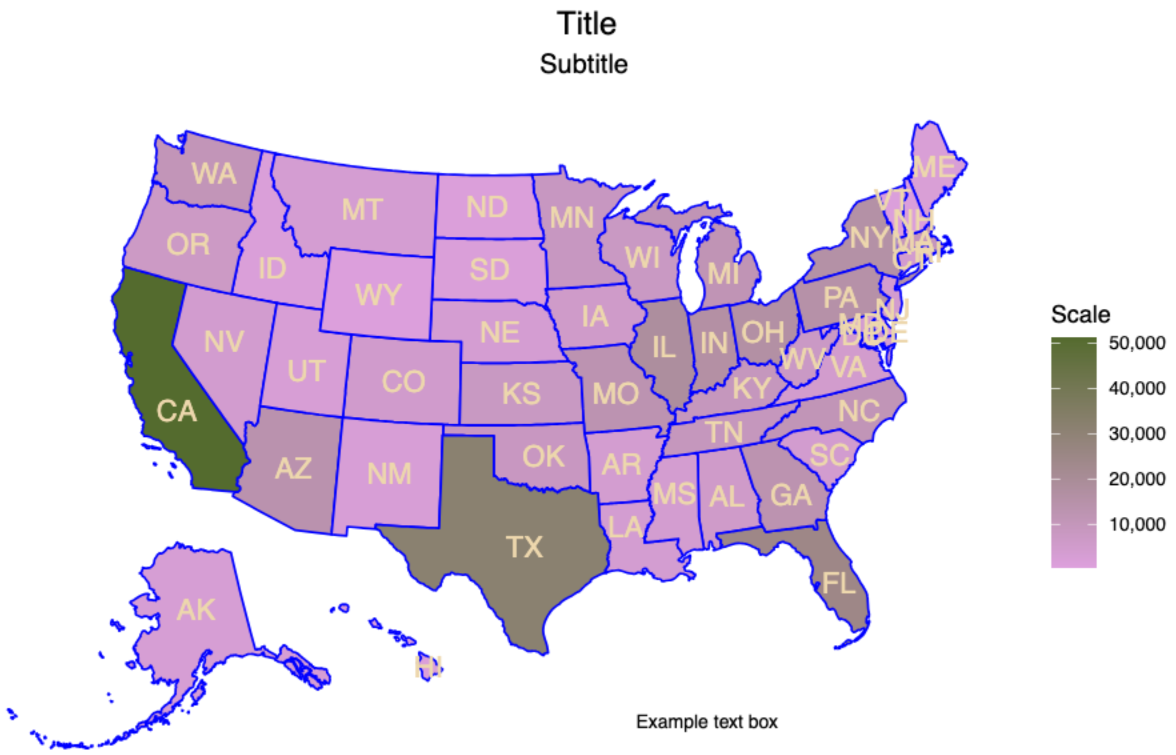
You can change colors to nearly whatever you want but consider how it may look if printed in black and white and for persons who are colorblind. In addition, you'll want enough color distinction to tell the various categories a part. Here are a few resources related to color:

Color Palette Cheatsheet: https://www.nceas.ucsb.edu/sites/default/files/2020-04/colorPaletteCheatsheet.pdf

Page 3 has a list of the predefined colors in R, which is also available in other places. I did not use any of the other color packages for any of the maps shown here.

R Cookbook Colors (ggplot2): http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/

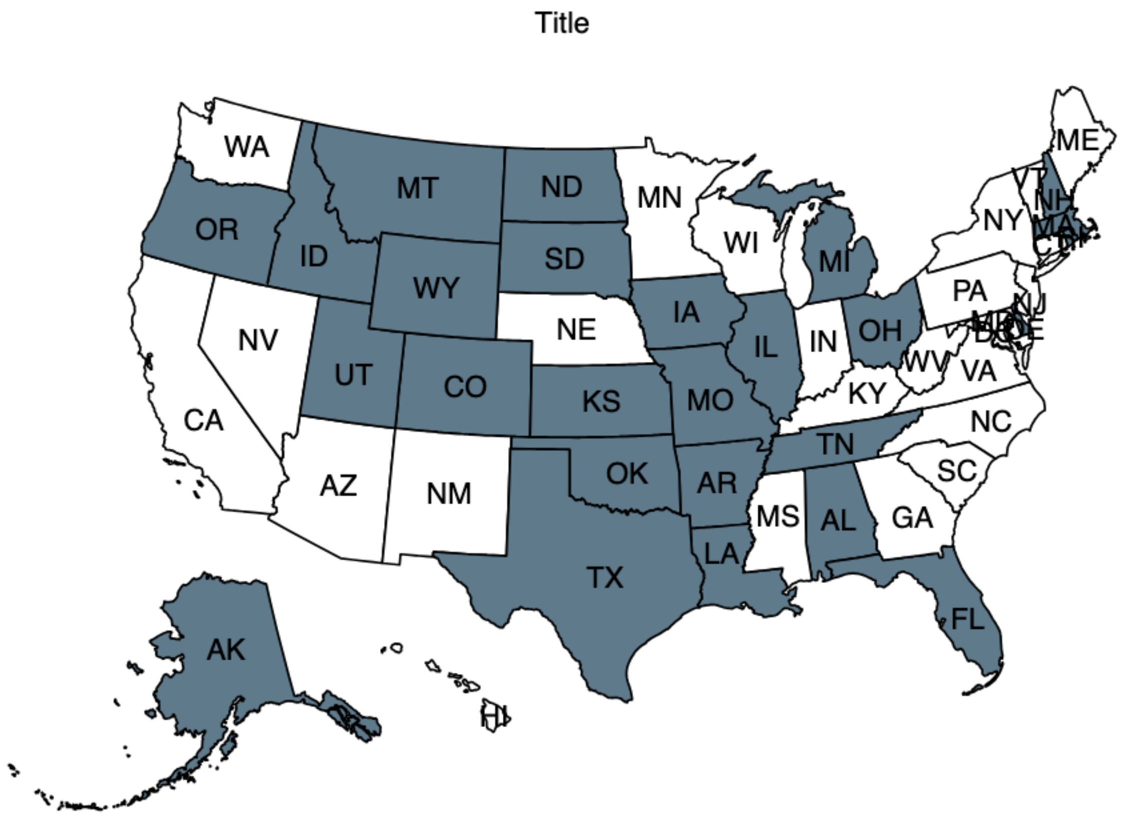
```
frequencies_map4 <-
  plot_usmap(data = Duplicate_Map_Data, values = "Numbers_1", color = "blue",
             regions = "states", include = c(), labels = TRUE, label_color = "wheat2") +
  labs(title = "Title") +
  labs(subtitle = "Subtitle") +
  theme(plot.title = element_text(hjust = 0.6)) +
  theme(plot.subtitle = element_text(hjust = 0.6)) +
  theme(plot.title = element_text(size = 12)) +
  theme(plot.subtitle = element_text(size = 10)) +
  scale_fill_continuous(low = "plum", high = "darkolivegreen",
                        name = "Scale", label = scales::comma) +
  theme(legend.position = "right") +
  annotate(geom="text", x = 1, y= 1, label= "Example text box",
           color="black", hjust = -1, vjust = 35, size = 2.6)
frequencies_map4
```



Binary Maps

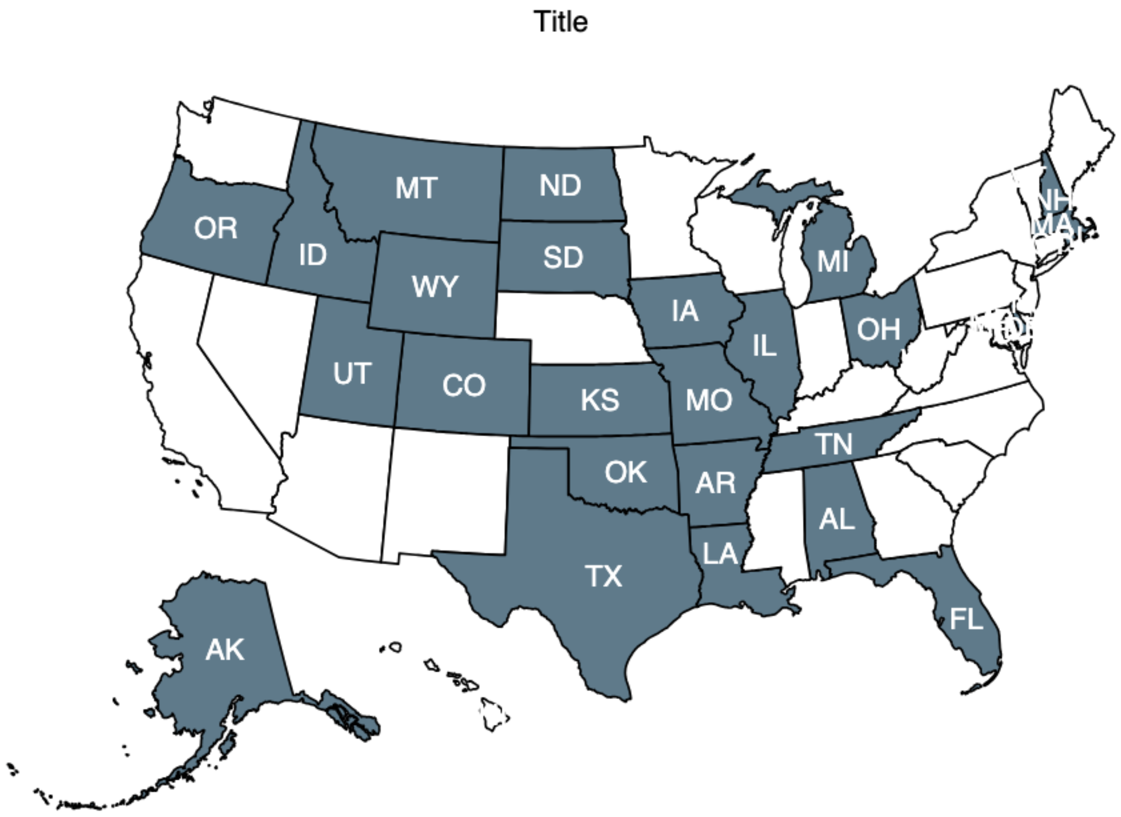
The above maps were great when I wanted to display distribution of some continuous variable across the US, but when I wanted to demonstrate which states had something other states did not, I found this code to visualize better:

```
binary_map1 <-
  plot_usmap(data = Duplicate_Map_Data, values = "Var_1", color = "black",
             regions = "states", include = c(), labels = TRUE, label_color = "black") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(plot.subtitle = element_text(hjust = 0.5)) +
  scale_fill_continuous(low = "white", high = "lightskyblue4",
                        name = "Has Variation", label = scales::comma) +
  labs(title = "Title") +
  theme(legend.position = "none")
binary_map1
```



If you really want states to pop, you can camouflage the state abbreviations by changing the color:

```r
binary_map2 <-
  plot_usmap(data = Duplicate_Map_Data, values = "Var_1", color = "black",
             regions = "states", include = c(), labels = TRUE, label_color = "white") +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(plot.subtitle = element_text(hjust = 0.5)) +
  scale_fill_continuous(low = "white", high = "lightskyblue4",
                        name = "Has Variation", label = scales::comma) +
  labs(title = "Title") +
  theme(legend.position = "none")
binary_map2
```



Saving Maps

Make sure to save your maps! There are many options for saving. More info below.

https://ggplot2.tidyverse.org/reference/ggsave.html

```r
ggsave("binarymap.jpeg", plot = binary_map1)
```

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