Exploratory Data Analysis with R

Visualizing Geospatial Data

Xuemao Zhang
East Stroudsburg University

November 7, 2022

Outline

- Introduction
- Displaying maps with R packages
- Displaying maps using shapefiles
- Plotting locations
- Plotting symbols
- Plotting choropleth maps
- `You can disable/enable mouse click advance by pressing key 'k', when viewing the presentation.`

Introduction

- For analysis of spatial/geographical data, R can provide maps of data in the same computing environment that the data analysis is being performed in.
- The main place to go to get an overview of the kinds and capabilities of the **spatial packages** in R is the Spatial Task Views on CRAN http://cran.us.r-project.org/web/views/Spatial.html.
  - *R has been acquiring much of the functionality of traditional GIS packages such as* `sp`, `rgeos`, `spData`, `shapefiles`, `maptools` *and* `raster`.
- The maps can be *static* or *interactive*.
- Another classification of maps:
  - *Locations only*
  - *Choropleth map: coloring/sizing regions/locations by data values.*

Introduction

- Packages for US maps:
  - *`usmap`, `fiftystater`, ...*
- Packages for static visualization:
  - *`ggmap`; `maps`, ...*
  - *`mapdata` provides additional map databases for the `maps` package*
- Packages for interactive visualization:
  - *`leaflet`, `mapview`, `tmap`, ...*
  - *`tmap` is an actively maintained open-source R-library for drawing thematic maps. The API is based on A Layered Grammar of Graphics and resembles the syntax of `ggplot2`.*
- Packages for reading and writing spatial data
  - *`shapefiles`, `maptools`, `sf`, ...*
- We use several R pacakages to illustrate how to draw static and interactive choropleth maps.

Introduction - points

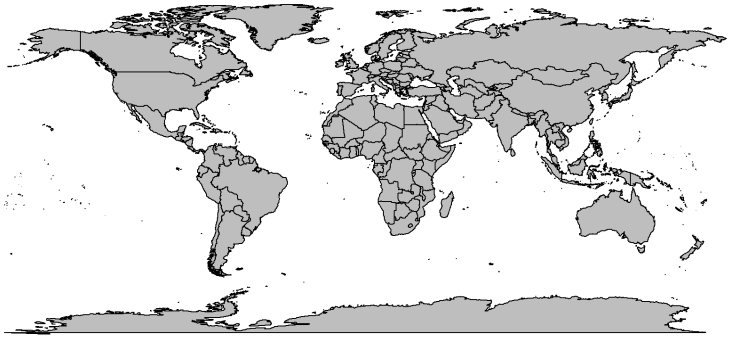
- Geographical points are on a sphere while maps are ploted on a flat surface.
- There are two different types of Coordinate Reference Systems in GIS (Geographic Information System).
  - *Geographic coordinate systems: coordinate systems that span the entire globe (e.g. latitude / longitude).*
  - *Projected coordinate systems: coordinate systems that are localized to minimize visual distortion in a particular region (e.g. Robinson, **UTM**, State Plane)*
- For example,
  - *The package `maps` and `fiftystater` use geographic coordinate systems.*
  - *The package `usmap` uses UTM Coordinates.*
- For more information, see Penn State E-education UTM Coordinate System.
- The function `convUL()` in the package `PBSmapping` can be used to convert coordinates between UTM and Lon/Lat.

Introduction - choropleth map

- The **choropleth map** colors/sizes regions/points by data values.
  - *It's a widely used mapping method that a good proportion of people seem to understand, so it's often a good choice if you want to show or look at regional patterns.*
- The package usmap plot US map including Alaska and Hawaii. The function `usmap::plot_usmap` returns a ggplot object, which means we can add ggplot layers to the plot right out of the box.
- The package tmap offers a flexible, layer-based, and easy to use approach to create thematic maps. It resembles the syntax of `ggplot2`.
- The package leaflet https://rstudio.github.io/leaflet/ is one of the most popular open-source JavaScript libraries for mobile-friendly interactive maps.

Displaying maps: World map

```
library(maps)
maps::map('world',fill=TRUE,border='black',col='gray', bg='white')
```



Displaying maps: World map

- To map individual countries, we use the regions parameter.

```
par(mfrow=c(1,3));
map("world", "italy");
map("world", "france");
map("world", "spain");
```

Displaying maps: World map

- `map_data()` in `ggplot2` can easily turn data from the `maps` package in to a data frame suitable for plotting with ggplot2.
- See ggplot2 themes for more information about themes.

```
library(ggplot2)
worldmap = ggplot2::map_data('world')
str(worldmap)
```

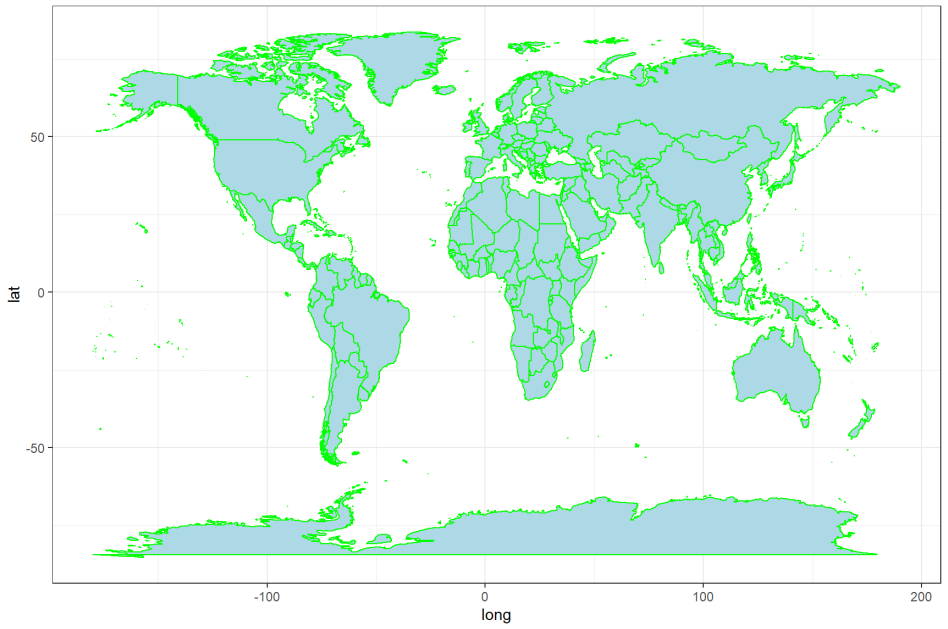
```
## 'data.frame':    99338 obs. of  6 variables:
##  $ long     : num  -69.9 -69.9 -69.9 -70 -70.1 ...
##  $ lat      : num  12.5 12.4 12.4 12.5 12.5 ...
##  $ group    : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ order    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ region   : chr  "Aruba" "Aruba" "Aruba" "Aruba" ...
##  $ subregion: chr  NA NA NA NA ...
```

```
head(worldmap)
```

```
##        long      lat group order region subregion
## 1 -69.89912 12.45200     1     1  Aruba      <NA>
## 2 -69.89571 12.42300     1     2  Aruba      <NA>
## 3 -69.94219 12.43853     1     3  Aruba      <NA>
## 4 -70.00415 12.50049     1     4  Aruba      <NA>
## 5 -70.06612 12.54697     1     5  Aruba      <NA>
## 6 -70.05088 12.59707     1     6  Aruba      <NA>
```

Displaying maps: World map

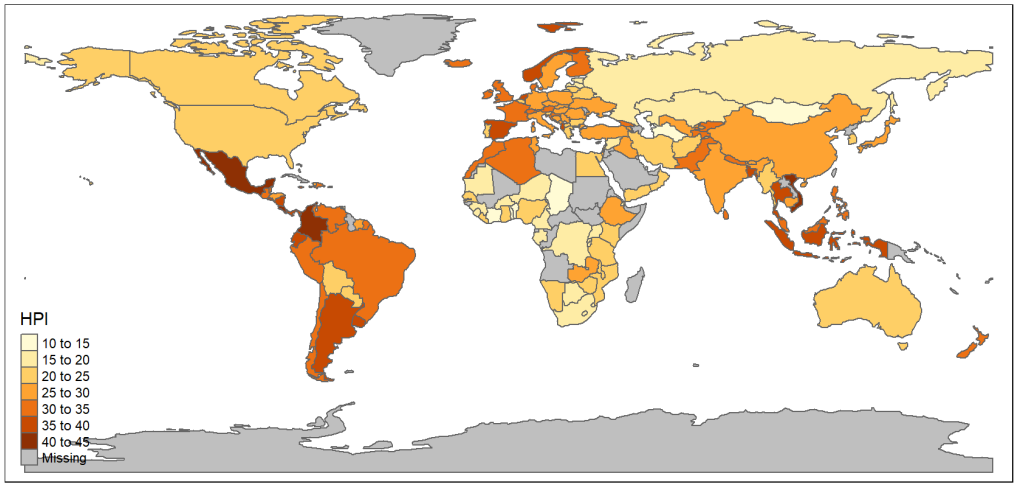
```
ggplot(worldmap, aes(long, lat, group=group)) +
  geom_polygon(fill='lightblue', colour="green") + theme_bw()
```



Displaying maps: World map

- Let's use the package tmap https://cran.r-project.org/web/packages/tmap/vignettes/tmap-getstarted.html

```
library(tmap)
data("World")
#str(World)
#str(World$HPI)
tm_shape(World) +tm_polygons("HPI")
```




```
#It is a choropleth map
```

Displaying maps: World map

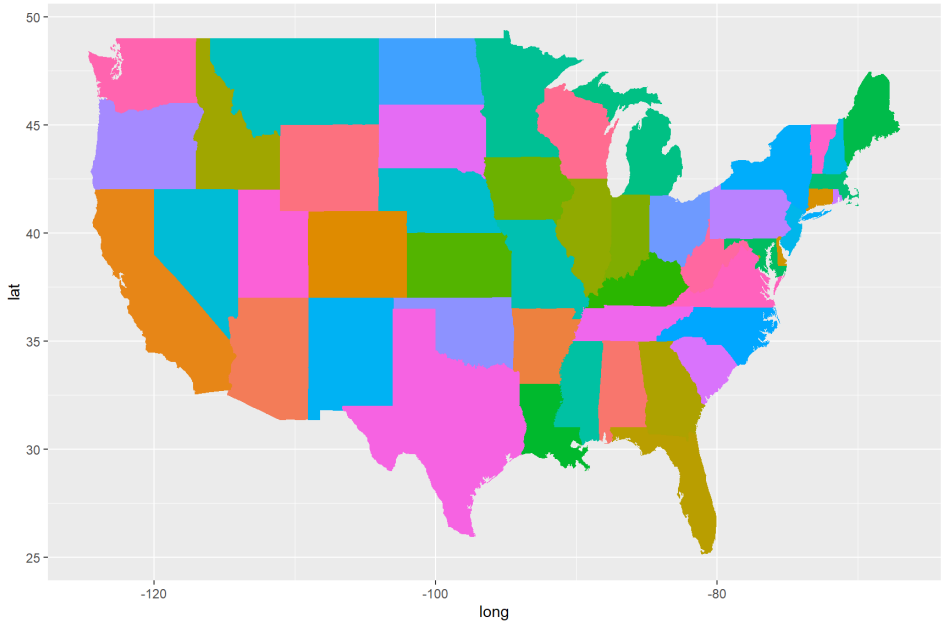
- Each map can be plotted as a static image or viewed interactively using “plot” and “view” modes, respectively.
- Toggling between the modes can be done with the ‘switch’ `ttm()`(which stands for toggle thematic map).

```
tmap_mode("view")
tm_shape(World) + tm_polygons("HPI")
```

Displaying maps: US map

- `maps` package can be used to plot US map.

```
usmap = ggplot2::map_data("state") # US map
ggplot(usmap, aes(long, lat, group=group, fill=region)) +
  geom_polygon(show.legend = F);
```



Displaying maps: US map

- You have to know the names of the regions in the database though. To find that out, enter the following in the console.

- "fips" is for each state's FIPS code. FIPS stands for "Federal Information Processing Standard", and pretty much any geography-based data from the government uses it.

```
maps::state.fips  # See state region names
```

```
##    fips ssa region division abb                          polyname
## 1     1   1      3        6  AL                           alabama
## 2     4   3      4        8  AZ                           arizona
## 3     5   4      3        7  AR                          arkansas
## 4     6   5      4        9  CA                        california
## 5     8   6      4        8  CO                          colorado
## 6     9   7      1        1  CT                       connecticut
## 7    10   8      3        5  DE                          delaware
## 8    11   9      3        5  DC              district of columbia
## 9    12  10      3        5  FL                           florida
## 10   13  11      3        5  GA                           georgia
## 11   16  13      4        8  ID                             idaho
## 12   17  14      2        3  IL                          illinois
## 13   18  15      2        3  IN                           indiana
## 14   19  16      2        4  IA                              iowa
## 15   20  17      2        4  KS                            kansas
## 16   21  18      3        6  KY                          kentucky
## 17   22  19      3        7  LA                         louisiana
## 18   23  20      1        1  ME                             maine
## 19   24  21      3        5  MD                          maryland
## 20   25  22      1        1  MA massachusetts:martha's vineyard
## 21   25  22      1        1  MA               massachusetts:main
## 22   25  22      1        1  MA         massachusetts:nantucket
## 23   26  23      2        3  MI                   michigan:north
## 24   26  23      2        3  MI                   michigan:south
## 25   27  24      2        4  MN                       minnesota
## 26   28  25      3        6  MS                     mississippi
## 27   29  26      2        4  MO                        missouri
## 28   30  27      4        8  MT                         montana
## 29   31  28      2        4  NE                        nebraska
## 30   32  29      4        8  NV                          nevada
## 31   33  30      1        1  NH                   new hampshire
## 32   34  31      1        2  NJ                      new jersey
## 33   35  32      4        8  NM                      new mexico
## 34   36  33      1        2  NY               new york:manhattan
## 35   36  33      1        2  NY                     new york:main
## 36   36  33      1        2  NY          new york:staten island
## 37   36  33      1        2  NY            new york:long island
## 38   37  34      3        5  NC           north carolina:knotts
## 39   37  34      3        5  NC             north carolina:main
## 40   37  34      3        5  NC             north carolina:spit
## 41   38  35      2        4  ND                    north dakota
## 42   39  36      2        3  OH                            ohio
## 43   40  37      3        7  OK                        oklahoma
## 44   41  38      4        9  OR                          oregon
## 45   42  39      1        2  PA                    pennsylvania
## 46   44  41      1        1  RI                    rhode island
## 47   45  42      3        5  SC                  south carolina
## 48   46  43      2        4  SD                    south dakota
```

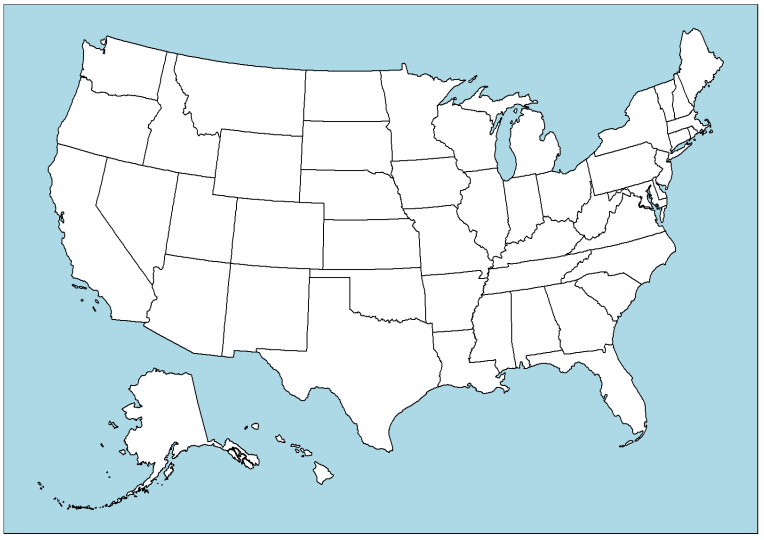
```
## 49   47  44      3       6  TN                    tennessee
## 50   48  45      3       7  TX                        texas
## 51   49  46      4       8  UT                         utah
## 52   50  47      1       1  VT                      vermont
## 53   51  49      3       5  VA           virginia:chesapeake
## 54   51  49      3       5  VA         virginia:chincoteague
## 55   51  49      3       5  VA                virginia:main
## 56   53  50      4       9  WA     washington:san juan island
## 57   53  50      4       9  WA        washington:lopez island
## 58   53  50      4       9  WA        washington:orcas island
## 59   53  50      4       9  WA      washington:whidbey island
## 60   53  50      4       9  WA              washington:main
## 61   54  51      3       5  WV                west virginia
## 62   55  52      2       3  WI                    wisconsin
## 63   56  53      4       8  WY                      wyoming
```

Displaying maps: US map

- Package `usmap` includes Alaska and Hawaii

```
library(usmap)
usmap::plot_usmap(regions = "states", lines = "brown")+
  theme(panel.background = element_rect(colour = "black",
                                        fill = "lightblue"))
```

```
## Warning: Ignoring unknown parameters: lines
```



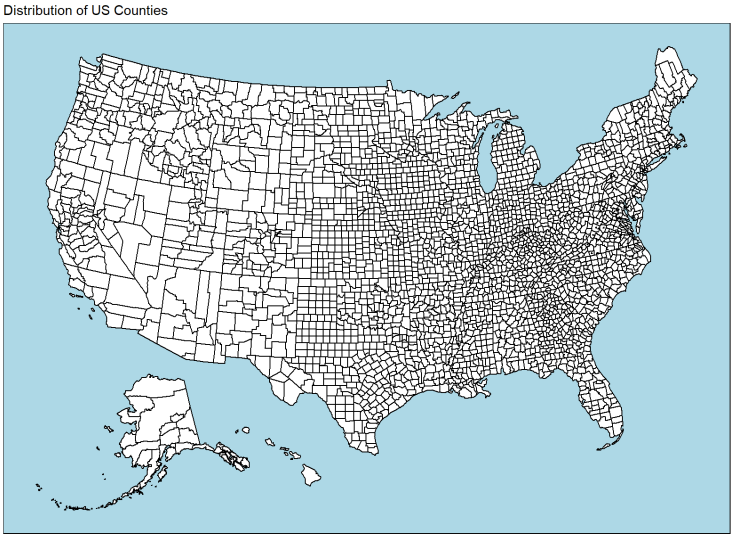
- The function `theme()` in ggplot2 is used to modify a plot's theme.

Displaying maps: US map

- Package `usmap` includes Alaska and Hawaii

```
usmap::plot_usmap(regions = "counties",lines = "brown") +
  labs(title = "Distribution of US Counties") +
  theme(panel.background = element_rect(colour = "black",
                            fill = "lightblue"));
```

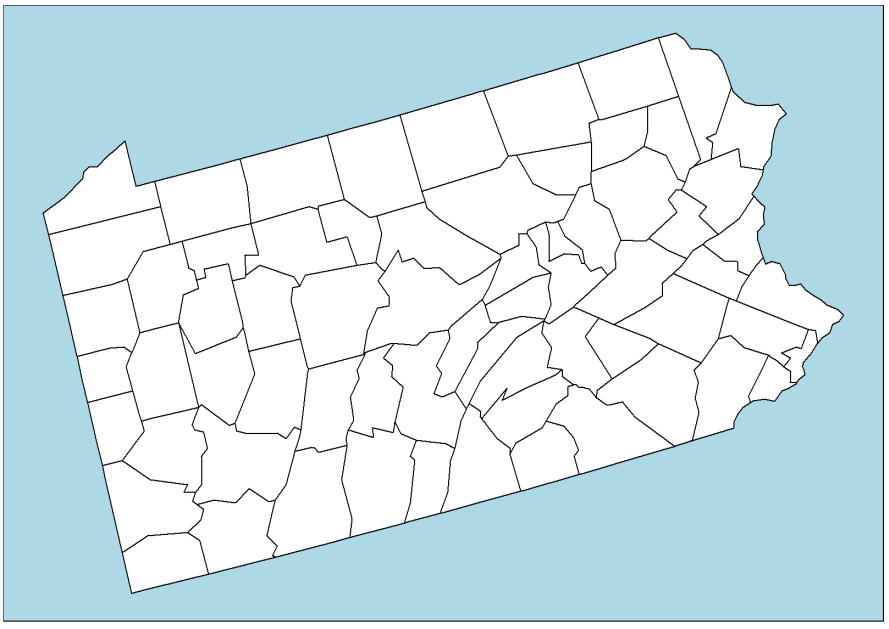
```
## Warning: Ignoring unknown parameters: lines
```



Displaying maps: US map

- We can plot US map with specified states

```
usmap::plot_usmap(regions = "counties", include = c("PA")) +
theme(panel.background = element_rect(colour = "black",
             fill = "lightblue"))
```



Displaying maps: US map

- We can also specify multiple regions to map them all at once.

```
usmap::plot_usmap(include = c("CA", "ID", "NV", "OR", "WA")) +
labs(title = "Western US States",
subtitle = "These are the states in the Pacific Timezone.")
```

Displaying maps using shapefiles

- The disadvantage of using R packages is that you're limited to the regions that the package offers.
- Using shapefiles provides more flexibility.
- Shapefiles are specifcally for geographic data. The file format encodes points, lines, and polygons in geographic space and is a common way to distribute spatial data. The file extension is .shp.
- The `rgdal` package was very easy to use, but not recommenced any more because it will be retired by the end of 2023.

Displaying maps using shapefiles

Using the tigris package, get Census Tiger shapefiles for US census geographies. Tigris will return the shapefile in the sf, or simple features, format. (https://map-rfun.library.duke.edu/031_thematic_mapping.html)

```
library(tigris)
us_geo <- tigris::states(class = "sf", year=2020)
```

```
##
  |
  |                                                                      |   0%
  |
  |                                                                      |   1%
  |
  |=                                                                     |   1%
  |
  |=                                                                     |   2%
  |
  |==                                                                    |   2%
  |
  |==                                                                    |   3%
  |
  |==                                                                    |   4%
  |
  |===                                                                   |   4%
  |
  |===                                                                   |   5%
  |
  |====                                                                  |   5%
  |
  |====                                                                  |   6%
  |
  |=====                                                                 |   6%
  |
  |=====                                                                 |   7%
  |
  |=====                                                                 |   8%
  |
  |======                                                                |   8%
  |
  |======                                                                |   9%
  |
  |=======                                                               |   9%
  |
  |=======                                                               |  10%
  |
  |=======                                                               |  11%
  |
  |========                                                              |  11%
  |
  |========                                                              |  12%
  |
  |=========                                                             |  12%
  |
  |=========                                                             |  13%
  |
  |=========                                                             |  14%
  |
  |==========                                                            |  14%
  |
  |==========                                                            |  15%
```

```
|
|===========                                                    | 15%
|
|===========                                                    | 16%
|
|============                                                   | 17%
|
|============                                                   | 18%
|
|==============                                                 | 18%
|
|==============                                                 | 19%
|
|===============                                                | 19%
|
|===============                                                | 20%
|
|===============                                                | 21%
|
|================                                               | 21%
|
|================                                               | 22%
|
|=================                                              | 23%
|
|==================                                             | 24%
|
|==================                                             | 25%
|
|===================                                            | 25%
|
|===================                                            | 26%
|
|====================                                           | 27%
|
|=====================                                          | 28%
|
|=====================                                          | 29%
|
|======================                                         | 30%
|
|======================                                         | 31%
|
|=======================                                        | 31%
|
|=======================                                        | 32%
|
|========================                                       | 32%
|
|========================                                       | 33%
|
|=========================                                      | 34%
|
|=========================                                      | 34%
|
|==========================                                     | 35%
|
|==========================                                     | 35%
|
|===========================                                    | 36%
|
|============================                                   | 37%
|
|============================                                   | 38%
|
|=============================                                  | 38%
|
|=============================                                  | 39%
|
|==============================                                 | 39%
```

```
|
|============================                                    |  40%
|
|============================                                    |  41%
|
|=============================                                   |  41%
|
|=============================                                   |  42%
|
|==============================                                  |  42%
|
|==============================                                  |  43%
|
|===============================                                 |  44%
|
|===============================                                 |  45%
|
|================================                                |  45%
|
|================================                                |  46%
|
|=================================                               |  47%
|
|=================================                               |  48%
|
|==================================                              |  48%
|
|==================================                              |  49%
|
|===================================                             |  49%
|
|===================================                             |  50%
|
|====================================                            |  51%
|
|====================================                            |  52%
|
|=====================================                           |  52%
|
|=====================================                           |  53%
|
|======================================                          |  54%
|
|======================================                          |  55%
|
|=======================================                         |  55%
|
|=======================================                         |  56%
|
|========================================                        |  57%
|
|========================================                        |  58%
|
|=========================================                       |  58%
|
|=========================================                       |  59%
|
|==========================================                      |  59%
|
|==========================================                      |  60%
|
|===========================================                     |  61%
|
|===========================================                     |  62%
|
|============================================                    |  62%
|
|============================================                    |  63%
|
|=============================================                   |  64%
```

```
|
|=============================================                         | 65%
|
|=============================================                         | 65%
|
|==============================================                        | 66%
|
|==============================================                        | 67%
|
|==============================================                        | 68%
|
|===============================================                       | 68%
|
|===============================================                       | 69%
|
|================================================                      | 70%
|
|================================================                      | 71%
|
|================================================                      | 72%
|
|=================================================                     | 72%
|
|=================================================                     | 73%
|
|=================================================                     | 74%
|
|==================================================                    | 74%
|
|==================================================                    | 75%
|
|==================================================                    | 75%
|
|==================================================                    | 76%
|
|===================================================                   | 76%
|
|===================================================                   | 77%
|
|===================================================                   | 78%
|
|====================================================                  | 78%
|
|====================================================                  | 79%
|
|====================================================                  | 80%
|
|====================================================                  | 81%
|
|=====================================================                 | 81%
|
|=====================================================                 | 82%
|
|=====================================================                 | 82%
|
|======================================================                | 83%
|
|======================================================                | 84%
|
|======================================================                | 84%
|
|=======================================================               | 85%
|
|=======================================================               | 85%
|
|=======================================================               | 86%
|
|========================================================              | 87%
|
|========================================================              | 88%
```

```
|
|============================================================      |  88%
|
|============================================================      |  89%
|
|============================================================      |  89%
|
|============================================================      |  90%
|
|============================================================      |  91%
|
|============================================================      |  91%
|
|============================================================      |  92%
|
|============================================================      |  92%
|
|============================================================      |  93%
|
|============================================================      |  94%
|
|============================================================      |  95%
|
|============================================================      |  95%
|
|============================================================      |  96%
|
|============================================================      |  97%
|
|============================================================      |  98%
|
|============================================================      |  98%
|
|============================================================      |  99%
|
|============================================================      |  99%
|
|============================================================      | 100%
```

Displaying maps using shapefiles

```
library(dplyr)
class(us_geo)
```

```
## [1] "sf"         "data.frame"
```

```
dplyr::glimpse(us_geo)
```

```
## Rows: 56
## Columns: 15
## $ REGION   <chr> "3", "3", "2", "2", "3", "1", "4", "1", "3", "1", "1", "3", "…
## $ DIVISION <chr> "5", "5", "3", "4", "5", "1", "8", "1", "5", "1", "1", "5", "…
## $ STATEFP  <chr> "54", "12", "17", "27", "24", "44", "16", "33", "37", "50", "…
## $ STATENS  <chr> "01779805", "00294478", "01779784", "00662849", "01714934", "…
## $ GEOID    <chr> "54", "12", "17", "27", "24", "44", "16", "33", "37", "50", "…
## $ STUSPS   <chr> "WV", "FL", "IL", "MN", "MD", "RI", "ID", "NH", "NC", "VT", "…
## $ NAME     <chr> "West Virginia", "Florida", "Illinois", "Minnesota", "Marylan…
## $ LSAD     <chr> "00", "00", "00", "00", "00", "00", "00", "00", "00", "00", "…
## $ MTFCC    <chr> "G4000", "G4000", "G4000", "G4000", "G4000", "G4000", "G4000"…
## $ FUNCSTAT <chr> "A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "…
## $ ALAND    <dbl> 62266296765, 138958484319, 143778461053, 206232157570, 251518…
## $ AWATER   <dbl> 489206049, 45975808217, 6216594318, 18949864226, 6979171386, …
## $ INTPTLAT <chr> "+38.6472854", "+28.3989775", "+40.1028754", "+46.3159573", "…
## $ INTPTLON <chr> "-080.6183274", "-082.5143005", "-089.1526108", "-094.1996043…
## $ geometry <MULTIPOLYGON [°]> MULTIPOLYGON (((-81.74725 3..., MULTIPOLYGON (((…
```

Displaying maps using shapefiles

- Data frame view:

```
as_tibble(us_geo)
```

```
## # A tibble: 56 × 15
##    REGION DIVISION STATEFP STATENS  GEOID STUSPS NAME       LSAD  MTFCC FUNCSTAT
##    <chr>  <chr>    <chr>   <chr>    <chr> <chr>  <chr>      <chr> <chr> <chr>
##  1 3      5        54      01779805 54    WV     West Virg… 00    G4000 A
##  2 3      5        12      00294478 12    FL     Florida    00    G4000 A
##  3 2      3        17      01779784 17    IL     Illinois   00    G4000 A
##  4 2      4        27      00662849 27    MN     Minnesota  00    G4000 A
##  5 3      5        24      01714934 24    MD     Maryland   00    G4000 A
##  6 1      1        44      01219835 44    RI     Rhode Isl… 00    G4000 A
##  7 4      8        16      01779783 16    ID     Idaho      00    G4000 A
##  8 1      1        33      01779794 33    NH     New Hamps… 00    G4000 A
##  9 3      5        37      01027616 37    NC     North Car… 00    G4000 A
## 10 1      1        50      01779802 50    VT     Vermont    00    G4000 A
## # … with 46 more rows, and 5 more variables: ALAND <dbl>, AWATER <dbl>,
## #   INTPTLAT <chr>, INTPTLON <chr>, geometry <MULTIPOLYGON [°]>
```

Displaying maps using shapefiles

- Quick Plot using the `geometry` data

```
plot(sf::st_geometry(us_geo))
```

Displaying maps using shapefiles

- `tigris::shift_geometry()` will shift and resize Alaska and Hawaii.

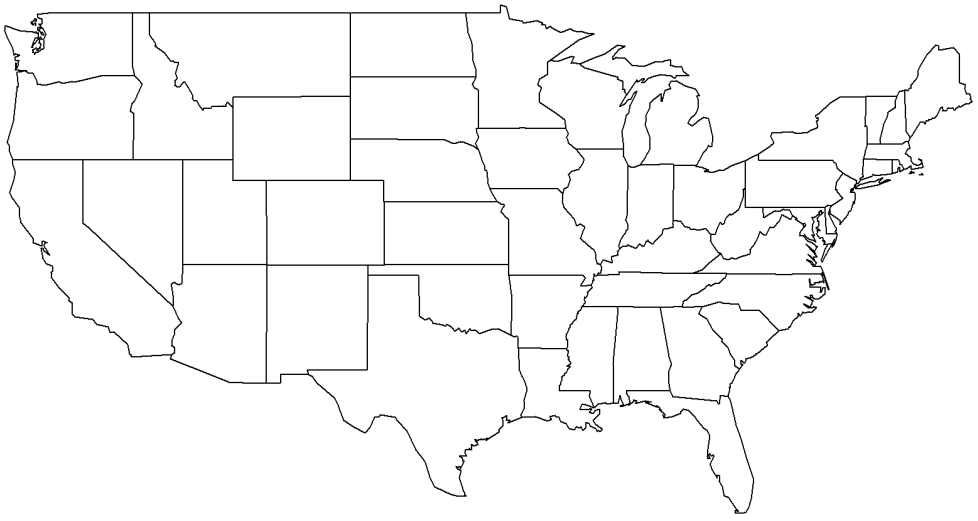
```
contiguous_states <- us_geo%>%
  filter(REGION != 9)%>%
  shift_geometry()
```

```
tm_shape(contiguous_states) + tm_borders()+
  tm_fill("darkolivegreen3")
```

Projections

- Longitude/latitude position points on a sphere; maps are drawn on a flat surface.
- Consider the `maps` package.
- The default uses a **rectangular projection** with the aspect ratio chosen so that longitude and latitude scales are equivalent at the center of the map.

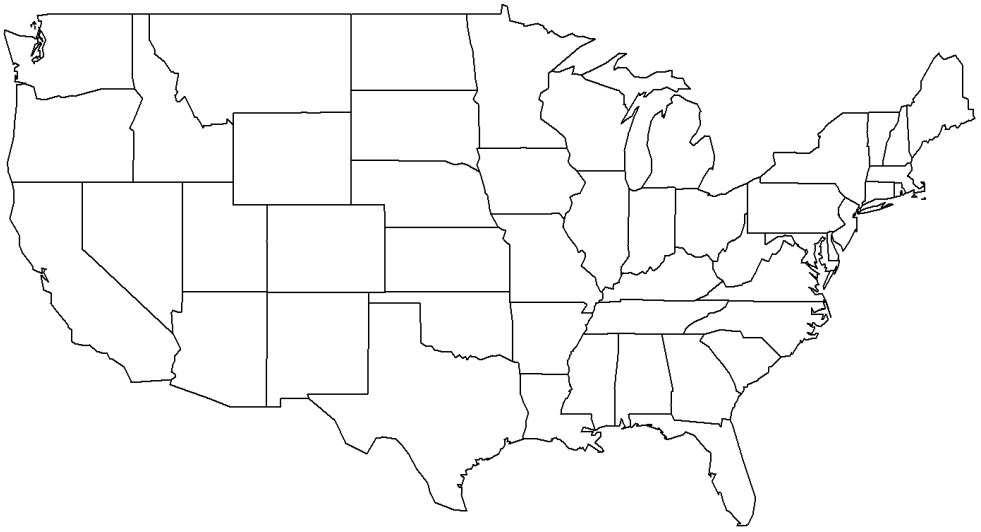
```
maps::map("state");
```



Projections

- Other projections try to preserve angles or areas.

```
library(mapproj)
maps::map("state", project = "mercator");
```



Projections

```
maps::map("state", project = "bonne", param = 45);
```

Plotting locations

- Region boundaries essentially serve as your backdrop.
- For **point-based data**, you draw the map, and then you add things (locations and weighted locations) on top.
- We will use an **airports** data frame with eight variables from the package `nycflights13`.

```
library(nycflights13)
dim(airports)
```

```
## [1] 1458    8
```

```
head(airports)
```

```
## # A tibble: 6 × 8
##   faa   name                            lat   lon   alt    tz dst   tzone
##   <chr> <chr>                         <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 04G   Lansdowne Airport              41.1 -80.6  1044    -5 A     America/Ne…
## 2 06A   Moton Field Municipal Airport  32.5 -85.7   264    -6 A     America/Ch…
## 3 06C   Schaumburg Regional            42.0 -88.1   801    -6 A     America/Ch…
## 4 06N   Randall Airport                41.4 -74.4   523    -5 A     America/Ne…
## 5 09J   Jekyll Island Airport          31.1 -81.4    11    -5 A     America/Ne…
## 6 0A9   Elizabethton Municipal Airport 36.4 -82.2  1593    -5 A     America/Ne…
```

Plotting locations with `maps`

Approximate Centroids

- In mathematics, the centroid of a plane figure is the arithmetic **mean position** of all the points in the figure.

```
usmap = ggplot2::map_data("state") # US map
str(usmap)
```

```
## 'data.frame':    15537 obs. of  6 variables:
##  $ long     : num  -87.5 -87.5 -87.5 -87.5 -87.6 ...
##  $ lat      : num  30.4 30.4 30.4 30.3 30.3 ...
##  $ group    : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ order    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ region   : chr  "alabama" "alabama" "alabama" "alabama" ...
##  $ subregion: chr  NA NA NA NA ...
```

Plotting locations with maps

Approximate Centroids

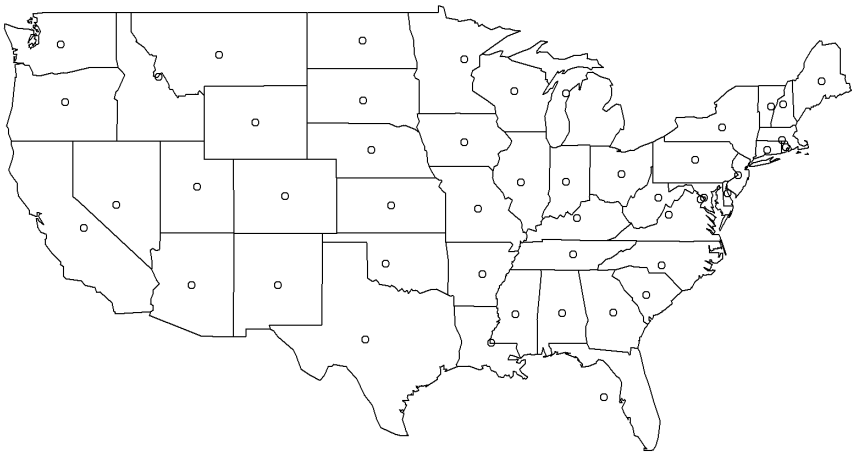
```
state_centroids = dplyr::summarize(group_by(usmap, region),
          x = mean(range(long)), y = mean(range(lat)));
names(state_centroids)[1] = "state"
head(state_centroids)
```

```
## # A tibble: 6 × 3
##   state            x     y
##   <chr>        <dbl> <dbl>
## 1 alabama      -86.7  32.6
## 2 arizona     -112.   34.2
## 3 arkansas     -92.1  34.8
## 4 california  -119.   37.3
## 5 colorado    -106.   39.0
## 6 connecticut  -72.8  41.5
```

Plotting locations with `maps`

- Adding points to a map drawn by the `maps::map` function with the default projection can be added using the function `points()`, treating longitude as your x-coordinate and latitude as your y-coordinate.

```
maps::map("state");
with(state_centroids, points(x, y));
```

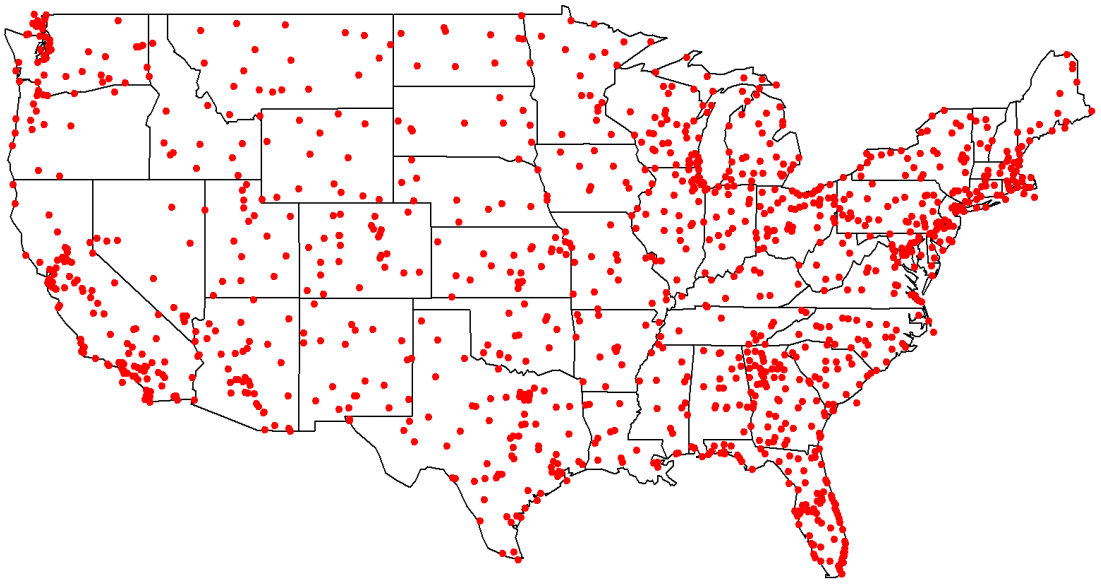


```
# with() function evaluates an R expression in an
# environment constructed from data, original or modified
# or: points(state_centroids$x, state_centroids$y)
```

Plotting locations with `maps`

- Example: Plot all airports in the **airports** data.

```
maps::map("state");
with(airports, points(lon, lat,pch=20,col='red'));
```



Plotting locations with tmap

```
library(tigris)
us_geo <- tigris::states(class = "sf")
contiguous_states <- us_geo%>% filter(REGION != 9)%>%shift_geometry()
nrow(contiguous_states) #51 rows
```

- We first convert the data to an `sf` object using the function `st_as_sf`

```
library(sf)
nrow(airports) # 1458 rows
```

```
## [1] 1458
```

```
airports2=sf::st_as_sf(airports, coords = c("lon", "lat"), crs =4326 )  #Convert
         foreign object to an sf object
```

Plotting locations with `tmap`

- Note that some airports are not in US

```
tm_shape(airports2) +
  tm_dots(size=0.01, col="red")
```

- We can consider airports with tzone starting with `America` only

```
dplyr::filter(airports, grepl('America', tzone))
#grepl returns a logical vector (match or not for each element).
```

Plotting locations with tmap

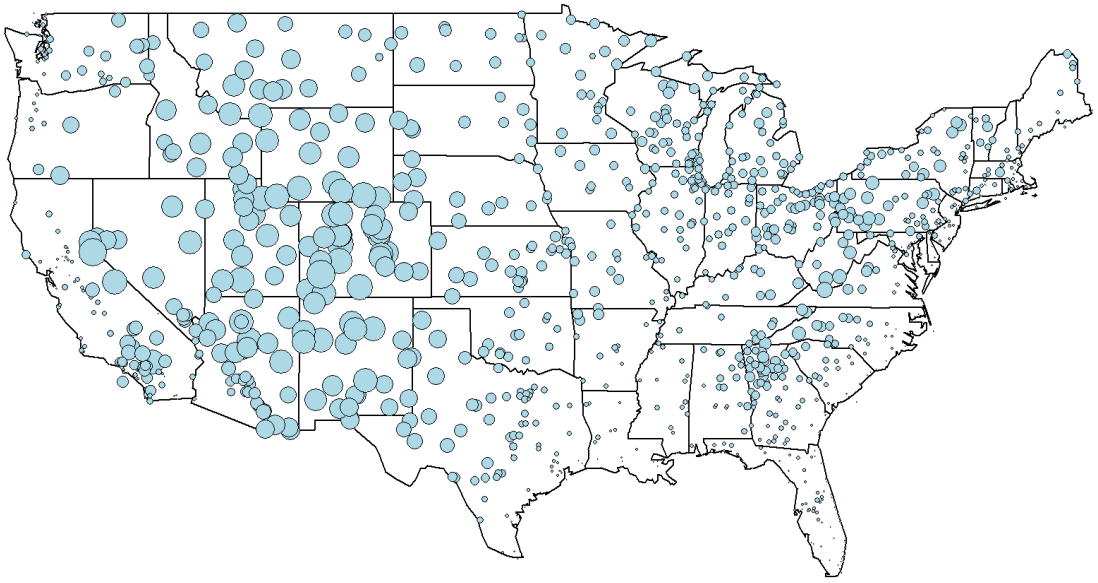
```
airports_copy=dplyr::filter(airports, grepl('America', tzone))
airports3=sf::st_as_sf(airports_copy, coords = c("lon", "lat"), crs =4326 )
tm_shape(airports3) +
  tm_dots(size=0.01, col="red")
```

Plotting symbols

- Points only represent location.
- Symbols (scaled shapes) both represent location and a second metric that corresponds to the location.
- For example, the function `symbols()` can be used to draw symbols.
- In the following example,
  - *Set `add` to `TRUE`, so that the circles are added to the current map instead of drawing a new plot.*
  - *Set `inches` to `FALSE` so that you can size circles by the current coordinate system instead of by inches.*
  - *Set `bg` to whatever `color` you want.*
  - *`circles` is a vector giving the radii of the circles.*
  - *Circle size is set to the absolute value of `alt` (altitude) of each airport.*
  - *The 0.008 multiplier sizes all the circles to fit how you want on the screen.*

Plotting symbols

```
maps::map("state")
with(airports, symbols(lon, lat, circles=.008*sqrt(abs(alt)),
     add=TRUE,inches=FALSE,bg="lightblue",lwd=0.5))
```



Plotting symbols with tmap

```
airports_copy$alt2=0.001*sqrt(abs(airports_copy$alt))
airports4=sf::st_as_sf(airports_copy, coords = c("lon", "lat"), crs =4326 )
tm_shape(airports4) +
  tm_dots(size='alt2', col="red")
```


```
#Legend for symbol sizes not available in view mode
```

Plotting choropleth maps

US Population Data:

- The census bureau provides estimates of populations of US counties.
- Estimates are available in several formats, including CSV.
- The CSV file is avaiable at https://www2.census.gov/programs-surveys/popest/datasets/2020-2021/counties/totals/
- The data can be read from the web directly:

```
library(readr)
url="https://www2.census.gov/programs-surveys/popest/datasets/2020-
         2021/counties/totals/co-est2021-alldata.csv";
data = read_csv(url)
#read.csv(url, stringsAsFactors = FALSE)
```

Plotting choropleth maps

- If you already downloaded the data, import from your local disk:

```
library(readr)
data = read_csv("../data/co-est2021-alldata.csv")
dim(data)
```

```
## [1] 3194   35
```

```
str(data)
```

```
## spec_tbl_df [3,194 × 35] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ SUMLEV               : chr [1:3194] "040" "050" "050" "050" ...
##  $ REGION               : num [1:3194] 3 3 3 3 3 3 3 3 3 ...
##  $ DIVISION             : num [1:3194] 6 6 6 6 6 6 6 6 6 ...
##  $ STATE                : chr [1:3194] "01" "01" "01" "01" ...
##  $ COUNTY               : chr [1:3194] "000" "001" "003" "005" ...
##  $ STNAME               : chr [1:3194] "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ CTYNAME              : chr [1:3194] "Alabama" "Autauga County" "Baldwin County" "Barbour County"
...
##  $ ESTIMATESBASE2020    : num [1:3194] 5024279 58805 231767 25223 22293 ...
##  $ POPESTIMATE2020      : num [1:3194] 5024803 58877 233140 25180 22223 ...
##  $ POPESTIMATE2021      : num [1:3194] 5039877 59095 239294 24964 22477 ...
##  $ NPOPCHG2020          : num [1:3194] 524 72 1373 -43 -70 ...
##  $ NPOPCHG2021          : num [1:3194] 15074 218 6154 -216 254 ...
##  $ BIRTHS2020           : num [1:3194] 13410 143 527 64 62 ...
##  $ BIRTHS2021           : num [1:3194] 56320 649 2260 274 226 ...
##  $ DEATHS2020           : num [1:3194] 16148 168 661 109 90 ...
##  $ DEATHS2021           : num [1:3194] 64868 681 2867 394 282 ...
##  $ NATURALCHG2020       : num [1:3194] -2738 -25 -134 -45 -28 ...
##  $ NATURALCHG2021       : num [1:3194] -8548 -32 -607 -120 -56 ...
##  $ INTERNATIONALMIG2020 : num [1:3194] -2 0 -1 0 0 0 0 0 1 0 ...
##  $ INTERNATIONALMIG2021 : num [1:3194] 1244 5 63 1 2 ...
##  $ DOMESTICMIG2020      : num [1:3194] 3339 97 1516 3 -42 ...
##  $ DOMESTICMIG2021      : num [1:3194] 22136 237 6780 -98 309 ...
##  $ NETMIG2020           : num [1:3194] 3337 97 1515 3 -42 ...
##  $ NETMIG2021           : num [1:3194] 23380 242 6843 -97 311 ...
##  $ RESIDUAL2020         : num [1:3194] -75 0 -8 -1 0 -2 0 -2 -4 -3 ...
##  $ RESIDUAL2021         : num [1:3194] 242 8 -82 1 -1 9 2 0 24 6 ...
##  $ GQESTIMATESBASE2020  : num [1:3194] 114572 442 2177 2789 2062 ...
##  $ GQESTIMATES2020      : num [1:3194] 114572 442 2177 2789 2062 ...
##  $ GQESTIMATES2021      : num [1:3194] 114572 442 2177 2789 2062 ...
##  $ RBIRTH2021           : num [1:3194] 11.19 11 9.57 10.93 10.11 ...
##  $ RDEATH2021           : num [1:3194] 12.9 11.5 12.1 15.7 12.6 ...
##  $ RNATURALCHG2021      : num [1:3194] -1.699 -0.543 -2.57 -4.786 -2.506 ...
##  $ RINTERNATIONALMIG2021: num [1:3194] 0.2472 0.0848 0.2667 0.0399 0.0895 ...
##  $ RDOMESTICMIG2021     : num [1:3194] 4.4 4.02 28.7 -3.91 13.83 ...
##  $ RNETMIG2021          : num [1:3194] 4.65 4.1 28.97 -3.87 13.91 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   SUMLEV = col_character(),
##   ..   REGION = col_double(),
##   ..   DIVISION = col_double(),
##   ..   STATE = col_character(),
##   ..   COUNTY = col_character(),
##   ..   STNAME = col_character(),
##   ..   CTYNAME = col_character(),
##   ..   ESTIMATESBASE2020 = col_double(),
##   ..   POPESTIMATE2020 = col_double(),
##   ..   POPESTIMATE2021 = col_double(),
##   ..   NPOPCHG2020 = col_double(),
```

```
##   ..   NPOPCHG2021 = col_double(),
##   ..   BIRTHS2020 = col_double(),
##   ..   BIRTHS2021 = col_double(),
##   ..   DEATHS2020 = col_double(),
##   ..   DEATHS2021 = col_double(),
##   ..   NATURALCHG2020 = col_double(),
##   ..   NATURALCHG2021 = col_double(),
##   ..   INTERNATIONALMIG2020 = col_double(),
##   ..   INTERNATIONALMIG2021 = col_double(),
##   ..   DOMESTICMIG2020 = col_double(),
##   ..   DOMESTICMIG2021 = col_double(),
##   ..   NETMIG2020 = col_double(),
##   ..   NETMIG2021 = col_double(),
##   ..   RESIDUAL2020 = col_double(),
##   ..   RESIDUAL2021 = col_double(),
##   ..   GQESTIMATESBASE2020 = col_double(),
##   ..   GQESTIMATES2020 = col_double(),
##   ..   GQESTIMATES2021 = col_double(),
##   ..   RBIRTH2021 = col_double(),
##   ..   RDEATH2021 = col_double(),
##   ..   RNATURALCHG2021 = col_double(),
##   ..   RINTERNATIONALMIG2021 = col_double(),
##   ..   RDOMESTICMIG2021 = col_double(),
##   ..   RNETMIG2021 = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Plotting choropleth maps

- We first consider the **state** population total data.

```
library(dplyr);
datapop0=data[,1:17];  #select the first 17 variables
datapop0=datapop0%>%filter(CTYNAME%in%STNAME);
#choose the state pop total data
dim(datapop0); # 52 by 17
```

```
## [1] 52 17
```

```
str(datapop0);
```

```
## tibble [52 × 17] (S3: tbl_df/tbl/data.frame)
##  $ SUMLEV           : chr [1:52] "040" "040" "040" "040" ...
##  $ REGION           : num [1:52] 3 4 4 3 4 4 1 3 3 3 ...
##  $ DIVISION         : num [1:52] 6 9 8 7 9 8 1 5 5 5 ...
##  $ STATE            : chr [1:52] "01" "02" "04" "05" ...
##  $ COUNTY           : chr [1:52] "000" "000" "000" "000" ...
##  $ STNAME           : chr [1:52] "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ CTYNAME          : chr [1:52] "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ ESTIMATESBASE2020: num [1:52] 5024279 733391 7151502 3011524 39538223 ...
##  $ POPESTIMATE2020  : num [1:52] 5024803 732441 7177986 3012232 39499738 ...
##  $ POPESTIMATE2021  : num [1:52] 5039877 732673 7276316 3025891 39237836 ...
##  $ NPOPCHG2020      : num [1:52] 524 -950 26484 708 -38485 ...
##  $ NPOPCHG2021      : num [1:52] 15074 232 98330 13659 -261902 ...
##  $ BIRTHS2020       : num [1:52] 13410 2407 18036 8554 104957 ...
##  $ BIRTHS2021       : num [1:52] 56320 9280 76497 35021 424333 ...
##  $ DEATHS2020       : num [1:52] 16148 1384 18340 9449 81738 ...
##  $ DEATHS2021       : num [1:52] 64868 5641 75665 38257 332337 ...
##  $ NATURALCHG2020   : num [1:52] -2738 1023 -304 -895 23219 ...
```

Plotting choropleth maps

- The data is wide in terms of years. We change the names of the variables and then convert the data to long form.

```
library(tidyr)
vars=c("POPESTIMATE2020","POPESTIMATE2021")
datapop0%>%select(STATE,STNAME,vars)%>%
pivot_longer(cols=vars, names_to = "Year", values_to = "Pop") -> datapop
datapop
```

```
## # A tibble: 104 × 4
##    STATE STNAME     Year                Pop
##    <chr> <chr>      <chr>             <dbl>
##  1 01    Alabama    POPESTIMATE2020  5024803
##  2 01    Alabama    POPESTIMATE2021  5039877
##  3 02    Alaska     POPESTIMATE2020   732441
##  4 02    Alaska     POPESTIMATE2021   732673
##  5 04    Arizona    POPESTIMATE2020  7177986
##  6 04    Arizona    POPESTIMATE2021  7276316
##  7 05    Arkansas   POPESTIMATE2020  3012232
##  8 05    Arkansas   POPESTIMATE2021  3025891
##  9 06    California POPESTIMATE2020 39499738
## 10 06    California POPESTIMATE2021 39237836
## # … with 94 more rows
```

Plotting choropleth maps

```
library(stringr)
datapop$Year= str_sub(datapop$Year, start=-4, end=-1)
datapop=dplyr::rename(datapop, fips=STATE)
#Change the variable 'STATE' to fips
datapop
```

```
## # A tibble: 104 × 4
##    fips  STNAME      Year       Pop
##    <chr> <chr>      <chr>    <dbl>
##  1 01    Alabama    2020   5024803
##  2 01    Alabama    2021   5039877
##  3 02    Alaska     2020    732441
##  4 02    Alaska     2021    732673
##  5 04    Arizona    2020   7177986
##  6 04    Arizona    2021   7276316
##  7 05    Arkansas   2020   3012232
##  8 05    Arkansas   2021   3025891
##  9 06    California 2020  39499738
## 10 06    California 2021  39237836
## # … with 94 more rows
```

Plotting choropleth maps

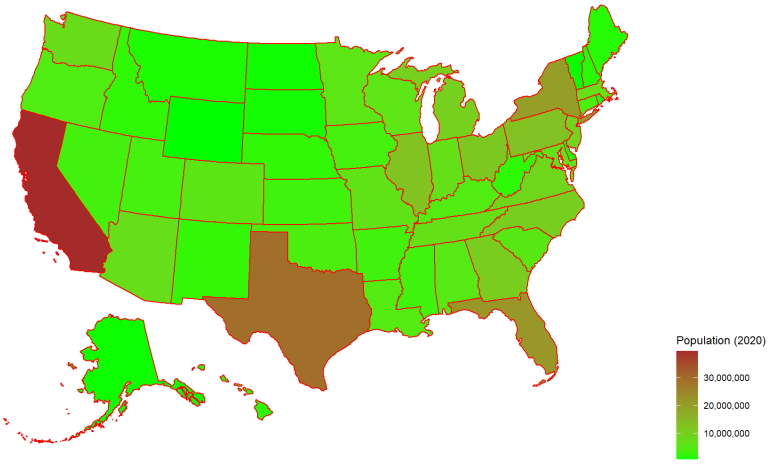
- A choropleth map needs to have the information for coloring all the pieces of a region. We use the `Pop` variable.
- First, we'll visualize the population data for the year 2020.

```
datapop_2020=dplyr::filter(datapop, Year == 2020)
str(datapop_2020)
```

```
## tibble [52 × 4] (S3: tbl_df/tbl/data.frame)
##  $ fips  : chr [1:52] "01" "02" "04" "05" ...
##  $ STNAME: chr [1:52] "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ Year  : chr [1:52] "2020" "2020" "2020" "2020" ...
##  $ Pop   : num [1:52] 5024803 732441 7177986 3012232 39499738 ...
```

Plotting choropleth maps

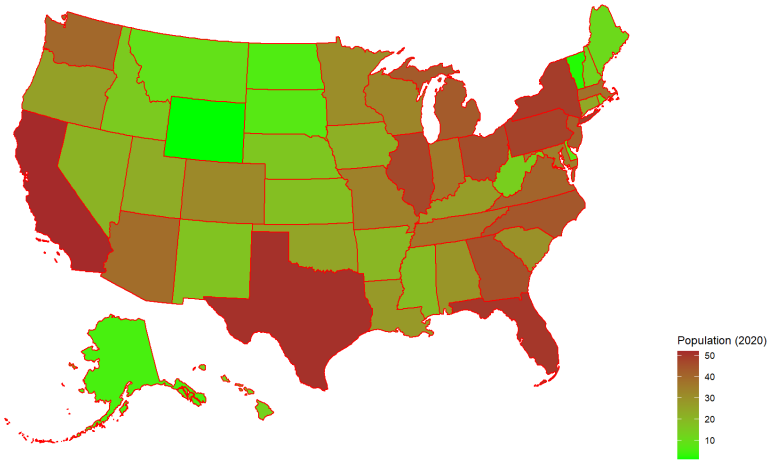
```
library(usmap)
library(ggplot2)
plot_usmap(data = datapop_2020, values = "Pop", color = "red") +
scale_fill_continuous(low = "green", high = "brown",
  name = "Population (2020)", label = scales::comma)+
theme(legend.position = "right")
```



Plotting choropleth maps

- This image is dominated by the fact that most state populations are small.
- First, showing population **ranks** can help see the variation a bit better.

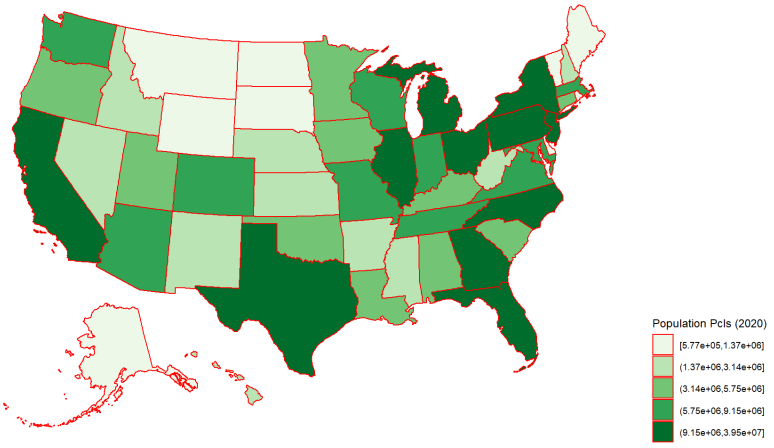
```
datapop_2020 = dplyr::mutate(datapop_2020, Rpop = rank(Pop))
plot_usmap(data = datapop_2020, values = "Rpop", color = "red") +
scale_fill_continuous(low = "green", high = "brown",
  name = "Population (2020)", label = scales::comma)+
theme(legend.position = "right")
```



Plotting choropleth maps

- Second, using quantile bins instead of a continuous scale can help see the variation better.

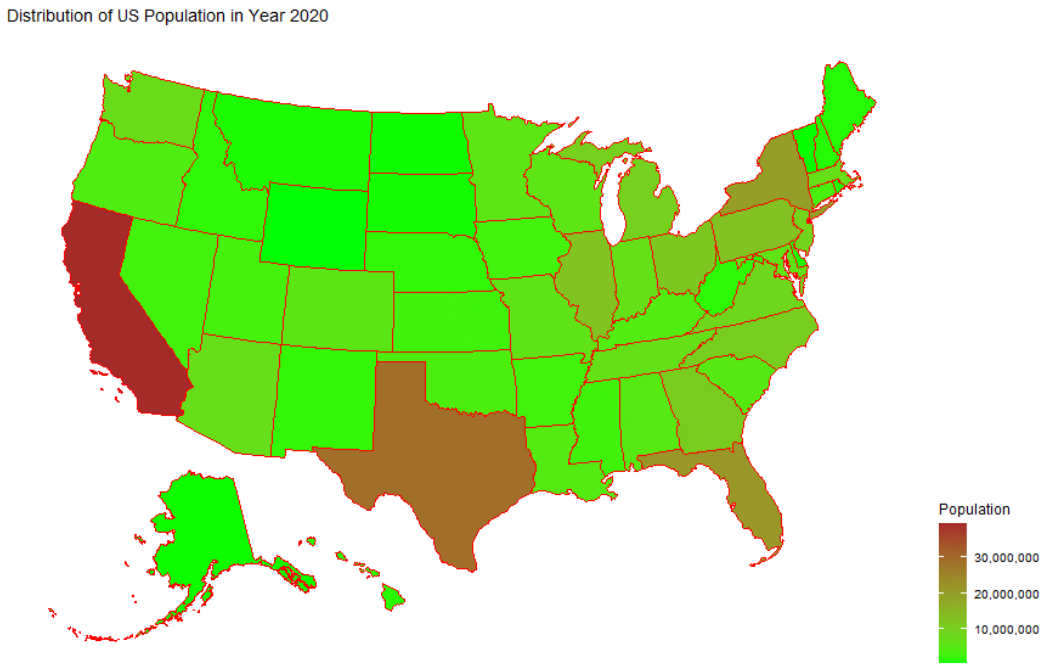
```
bins=6  # consider 6 percentiles
datapop_2020 = dplyr::mutate(datapop_2020,
pcls = cut(Pop, quantile(Pop, seq(0, 1, len = bins)),
           include.lowest = TRUE));
plot_usmap(data = datapop_2020, values = "pcls", color = "red") +
scale_fill_brewer(palette =  "Greens",
                  name = "Population Pcls (2020)")+
theme(legend.position = "right")
```



Plotting choropleth maps

- Animated Maps by `Year` using `gganimate`
  - *You can update the `values` following the above two data sets*

```
library(gganimate)
statepop_ani=plot_usmap(data = datapop, values = "Pop", color = "red") +
scale_fill_continuous(low = "green", high = "brown",
name = "Population", label = scales::comma)+
  transition_manual(Year, cumulative = FALSE)+
ggtitle(paste("Distribution of US Population in Year", '{current_frame}'))+
theme(legend.position = "right");
animate(statepop_ani,fps = 100)
```




```
#anim_save("Popanimation.gif")
```

Plotting choropleth maps

Visualization of County Population Data

- First, remove the rows with state population total.

```
datapop1=data[,1:17]; #select the first 17 variables
datapop1 = datapop1%>%  filter(!CTYNAME %in% STNAME);
dim(datapop1)   #52 rows are removed
```

```
## [1] 3142   17
```

```
str(datapop1)
```

```
## tibble [3,142 × 17] (S3: tbl_df/tbl/data.frame)
##  $ SUMLEV           : chr [1:3142] "050" "050" "050" "050" ...
##  $ REGION           : num [1:3142] 3 3 3 3 3 3 3 3 3 ...
##  $ DIVISION         : num [1:3142] 6 6 6 6 6 6 6 6 6 ...
##  $ STATE            : chr [1:3142] "01" "01" "01" "01" ...
##  $ COUNTY           : chr [1:3142] "001" "003" "005" "007" ...
##  $ STNAME           : chr [1:3142] "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ CTYNAME          : chr [1:3142] "Autauga County" "Baldwin County" "Barbour County" "Bibb County"
...
##  $ ESTIMATESBASE2020: num [1:3142] 58805 231767 25223 22293 59134 ...
##  $ POPESTIMATE2020  : num [1:3142] 58877 233140 25180 22223 59081 ...
##  $ POPESTIMATE2021  : num [1:3142] 59095 239294 24964 22477 59041 ...
##  $ NPOPCHG2020      : num [1:3142] 72 1373 -43 -70 -53 ...
##  $ NPOPCHG2021      : num [1:3142] 218 6154 -216 254 -40 ...
##  $ BIRTHS2020       : num [1:3142] 143 527 64 62 152 33 40 311 85 55 ...
##  $ BIRTHS2021       : num [1:3142] 649 2260 274 226 629 130 214 1320 356 238 ...
##  $ DEATHS2020       : num [1:3142] 168 661 109 90 220 37 58 409 121 114 ...
##  $ DEATHS2021       : num [1:3142] 681 2867 394 282 820 ...
##  $ NATURALCHG2020   : num [1:3142] -25 -134 -45 -28 -68 -4 -18 -98 -36 -59 ...
```

```
head(datapop1)
```

```
## # A tibble: 6 × 17
##   SUMLEV REGION DIVISION STATE COUNTY STNAME  CTYNAME        ESTIMATESBASE2020
##   <chr>   <dbl>    <dbl> <chr> <chr>  <chr>   <chr>                      <dbl>
## 1 050         3        6 01    001    Alabama Autauga County             58805
## 2 050         3        6 01    003    Alabama Baldwin County            231767
## 3 050         3        6 01    005    Alabama Barbour County             25223
## 4 050         3        6 01    007    Alabama Bibb County                22293
## 5 050         3        6 01    009    Alabama Blount County              59134
## 6 050         3        6 01    011    Alabama Bullock County             10357
## # … with 9 more variables: POPESTIMATE2020 <dbl>, POPESTIMATE2021 <dbl>,
## #   NPOPCHG2020 <dbl>, NPOPCHG2021 <dbl>, BIRTHS2020 <dbl>, BIRTHS2021 <dbl>,
## #   DEATHS2020 <dbl>, DEATHS2021 <dbl>, NATURALCHG2020 <dbl>
```

Plotting choropleth maps

Visualization of County Population Data

- Again, the data is wide in terms of years. We change the names of the variables and then convert the data to long form.

```
library(tidyr)
vars=c("POPESTIMATE2020","POPESTIMATE2021")
datapop1%>%select(STATE,COUNTY,vars)%>%
pivot_longer(cols=vars, names_to = "Year", values_to = "Pop") -> CountyPop
head(CountyPop)
```

```
## # A tibble: 6 × 4
##   STATE COUNTY Year              Pop
##   <chr> <chr>  <chr>           <dbl>
## 1 01    001    POPESTIMATE2020  58877
## 2 01    001    POPESTIMATE2021  59095
## 3 01    003    POPESTIMATE2020 233140
## 4 01    003    POPESTIMATE2021 239294
## 5 01    005    POPESTIMATE2020  25180
## 6 01    005    POPESTIMATE2021  24964
```

```
library(stringr)
CountyPop$Year= str_sub(CountyPop$Year, start=-4, end=-1)
head(CountyPop)
```

```
## # A tibble: 6 × 4
##   STATE COUNTY Year     Pop
##   <chr> <chr>  <chr>  <dbl>
## 1 01    001    2020   58877
## 2 01    001    2021   59095
## 3 01    003    2020  233140
## 4 01    003    2021  239294
## 5 01    005    2020   25180
## 6 01    005    2021   24964
```

Plotting choropleth maps

Visualization of County Population Data

- To use the `usmap::plot_usmap()` function, we need the county fips information.

```
str(usmap::us_map(regions = "counties"))
```

```
## 'data.frame':    55211 obs. of  10 variables:
##  $ x     : num  1225889 1244873 1244129 1272010 1276797 ...
##  $ y     : num  -1275020 -1272331 -1267515 -1262889 -1295514 ...
##  $ order : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ hole  : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
##  $ piece : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ group : chr  "01001.1" "01001.1" "01001.1" "01001.1" ...
##  $ fips  : chr  "01001" "01001" "01001" "01001" ...
##  $ abbr  : chr  "AL" "AL" "AL" "AL" ...
##  $ full  : chr  "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ county: chr  "Autauga County" "Autauga County" "Autauga County" "Autauga County" ...
```

Plotting choropleth maps

Visualization of County Population Data

- Next, add the variable 'fips' for county fips.
- The function `nchar()` counts the number of characters in a string.

```
CountyPop=mutate(CountyPop, fips=
case_when(
  nchar(COUNTY)==3 ~ paste(STATE, COUNTY, sep=""),
  nchar(COUNTY)==2 ~ paste(STATE, COUNTY, sep="0"),
  nchar(COUNTY)==1 ~ paste(STATE, COUNTY, sep="00")
        )
);  #this is more human readable compared to ifelse()
str(CountyPop)
```

```
## tibble [6,284 × 5] (S3: tbl_df/tbl/data.frame)
##  $ STATE : chr [1:6284] "01" "01" "01" "01" ...
##  $ COUNTY: chr [1:6284] "001" "001" "003" "003" ...
##  $ Year  : chr [1:6284] "2020" "2021" "2020" "2021" ...
##  $ Pop   : num [1:6284] 58877 59095 233140 239294 25180 ...
##  $ fips  : chr [1:6284] "01001" "01001" "01003" "01003" ...
```

```
head(CountyPop)
```

```
## # A tibble: 6 × 5
##   STATE COUNTY Year     Pop fips
##   <chr> <chr>  <chr>  <dbl> <chr>
## 1 01    001    2020   58877 01001
## 2 01    001    2021   59095 01001
## 3 01    003    2020  233140 01003
## 4 01    003    2021  239294 01003
## 5 01    005    2020   25180 01005
## 6 01    005    2021   24964 01005
```

Plotting choropleth maps

Visualization of County Population Data

- Check if there are any missing values.

```
dim(CountyPop)
```

```
## [1] 6284    5
```

```
anyNA(CountyPop)
```

```
## [1] FALSE
```

```
any(complete.cases(CountyPop)==F)
```

```
## [1] FALSE
```

```
#check if there are missing values
str(CountyPop)
```

```
## tibble [6,284 × 5] (S3: tbl_df/tbl/data.frame)
##  $ STATE : chr [1:6284] "01" "01" "01" "01" ...
##  $ COUNTY: chr [1:6284] "001" "001" "003" "003" ...
##  $ Year  : chr [1:6284] "2020" "2021" "2020" "2021" ...
##  $ Pop   : num [1:6284] 58877 59095 233140 239294 25180 ...
##  $ fips  : chr [1:6284] "01001" "01001" "01003" "01003" ...
```

Plotting choropleth maps

Visualization of County Population Data

- Again, the fact that some counties with large population sizes will make other county population sizes very small. We consider quantile bins.
- Let's add a variable `pcls` to the data in each year and then row bind the two years' data

```
bins=6
CountyYears=list()  #creat a list
Years=c(2020,2021)
for (i in 1:2)
{
subdata=CountyPop%>%filter(Year==Years[i])%>%
  mutate( pcls = cut(Pop, quantile(Pop, seq(0, 1, len = bins)),
include.lowest = TRUE))
CountyYears[[i]]=subdata
}
```

Plotting choropleth maps

Visualization of County Population Data

```
##Then row combine the 2 data sets
CountyPop2=dplyr::bind_rows(CountyYears[[1]], CountyYears[[2]])
str(CountyPop2)
```

```
## tibble [6,284 × 6] (S3: tbl_df/tbl/data.frame)
##  $ STATE : chr [1:6284] "01" "01" "01" "01" ...
##  $ COUNTY: chr [1:6284] "001" "003" "005" "007" ...
##  $ Year  : chr [1:6284] "2020" "2020" "2020" "2020" ...
##  $ Pop   : num [1:6284] 58877 233140 25180 22223 59081 ...
##  $ fips  : chr [1:6284] "01001" "01003" "01005" "01007" ...
##  $ pcls  : Factor w/ 10 levels "[67,8.66e+03]",..: 4 5 3 3 4 2 3 5 3 3 ...
```

```
head(CountyPop2)
```

```
## # A tibble: 6 × 6
##   STATE COUNTY Year     Pop fips  pcls
##   <chr> <chr>  <chr>  <dbl> <chr> <fct>
## 1 01    001    2020   58877 01001 (3.67e+04,9.53e+04]
## 2 01    003    2020  233140 01003 (9.53e+04,9.99e+06]
## 3 01    005    2020   25180 01005 (1.86e+04,3.67e+04]
## 4 01    007    2020   22223 01007 (1.86e+04,3.67e+04]
## 5 01    009    2020   59081 01009 (3.67e+04,9.53e+04]
## 6 01    011    2020   10309 01011 (8.66e+03,1.86e+04]
```

Plotting choropleth maps

Visualization of County Population Data

- Now we plot the population data by Year.
  - *For this, we can define a function of year.*

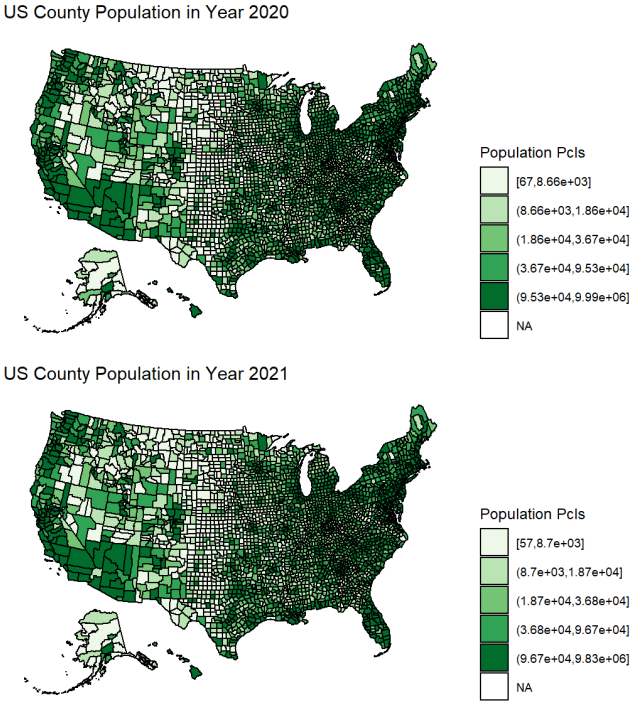
```
PlotCountyPop= function(x)
{
CountyPop_year=dplyr::filter(CountyPop2, Year == x);
plot_usmap(regions="counties",data = CountyPop_year,
           values ="pcls") +
ggplot2::scale_fill_brewer(palette = "Greens",
                  name = "Population Pcls")+
ggtitle(paste("US County Population in Year",x))+
theme(legend.position = "right");
}
```

Plotting choropleth maps

Visualization of County Population Data

- Show two maps

```
PopCountyMaps=list();
for( i in 1:length(Years)) #Length(Years)=2
{
PopCountyMaps[[i]]=PlotCountyPop(i+2019);
}
gridExtra::grid.arrange(grobs = PopCountyMaps,nrow=2,ncol=1);
```



Plotting choropleth maps

Visualization of County Population Data

- Now let's plot the poulation distribution in certain states.

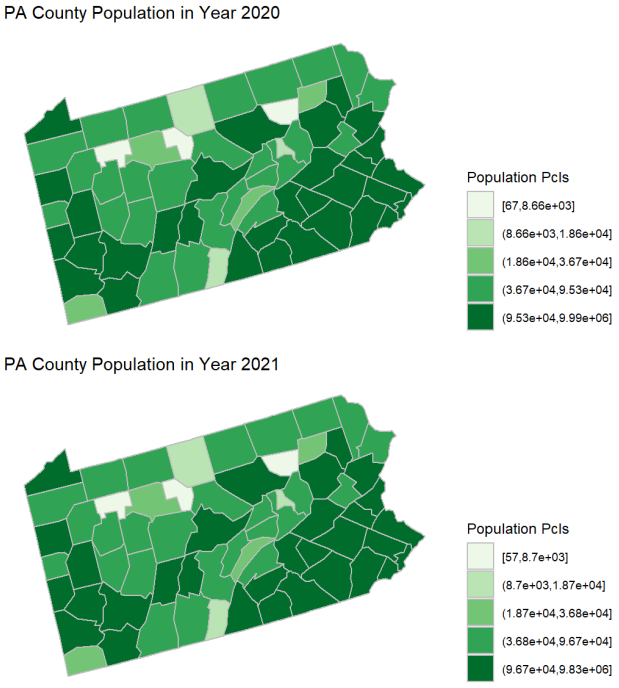
```
PA=c("PA"); #You can add more states
PACountyPop= function(x)
{
CountyPop_year=dplyr::filter(CountyPop2, Year == x);
plot_usmap(regions="counties",data = CountyPop_year,
    include = PA, values = "pcls", color= "grey") +
ggplot2::scale_fill_brewer(palette = "Greens",
                  name = "Population Pcls")+
ggtitle(paste("PA County Population in Year",x))+
theme(legend.position = "right");
}
```

Plotting choropleth maps

Visualization of County Population Data

- Show the two maps

```
PACountyMaps=list();
for( i in 1:length(Years))
{
PACountyMaps[[i]]=PACountyPop(i+2019);
}
gridExtra::grid.arrange(grobs = PACountyMaps,nrow=2,ncol=1);
```



Plotting choropleth maps

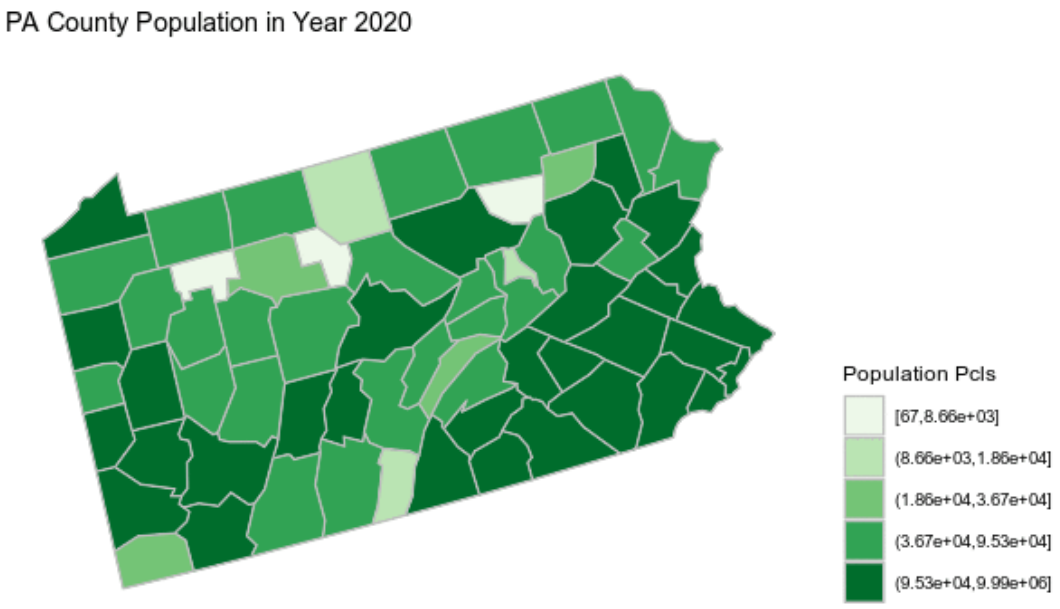
Visualization of County Population Data

- Animated Maps using `magick`

```
library(magick);
img=image_graph(width=600,height=400,res=96);
for( i in Years) print(PACountyPop(i));
dev.off();
```

```
## png
##   2
```

```
PACountyanimation = image_animate(img, fps = 1);
PACountyanimation
```




```
image_write(PACountyanimation, "PACountyanimation.gif")
```

- You can open the animated map PACountyanimation.gif in any browser.

Plotting choropleth maps with `tmap`

- Let's use package `tmap` plot the above state population data `datapop`
  - *For more, see https://map-rfun.library.duke.edu/031_thematic_mapping.html*

```
str(datapop)
```

```
## tibble [104 × 4] (S3: tbl_df/tbl/data.frame)
##  $ fips  : chr [1:104] "01" "01" "02" "02" ...
##  $ STNAME: chr [1:104] "Alabama" "Alabama" "Alaska" "Alaska" ...
##  $ Year  : chr [1:104] "2020" "2021" "2020" "2021" ...
##  $ Pop   : num [1:104] 5024803 5039877 732441 732673 7177986 ...
```

```
library(tigris)
us_geo <- tigris::states(class = "sf")
contiguous_states <- us_geo%>% filter(REGION != 9)%>%shift_geometry()
str(contiguous_states)
```

```
## Classes 'sf' and 'data.frame':   51 obs. of  15 variables:
##  $ REGION  : chr  "3" "3" "2" "2" ...
##  $ DIVISION: chr  "5" "5" "3" "4" ...
##  $ STATEFP : chr  "54" "12" "17" "27" ...
##  $ STATENS : chr  "01779805" "00294478" "01779784" "00662849" ...
##  $ GEOID   : chr  "54" "12" "17" "27" ...
##  $ STUSPS  : chr  "WV" "FL" "IL" "MN" ...
##  $ NAME    : chr  "West Virginia" "Florida" "Illinois" "Minnesota" ...
##  $ LSAD    : chr  "00" "00" "00" "00" ...
##  $ MTFCC   : chr  "G4000" "G4000" "G4000" "G4000" ...
##  $ FUNCSTAT: chr  "A" "A" "A" "A" ...
##  $ ALAND   : num  6.23e+10 1.39e+11 1.44e+11 2.06e+11 2.52e+10 ...
##  $ AWATER  : num  4.89e+08 4.60e+10 6.22e+09 1.89e+10 6.98e+09 ...
##  $ INTPTLAT: chr  "+38.6472854" "+28.3989775" "+40.1028754" "+46.3159573" ...
##  $ INTPTLON: chr  "-080.6183274" "-082.5143005" "-089.1526108" "-094.1996043" ...
##  $ geometry:sfc_MULTIPOLYGON of length 51; first list element: List of 1
##   ..$ :List of 1
##   .. ..$ : num [1:35239, 1:2] 1216736 1216792 1216795 1216811 1216823 ...
##   ..- attr(*, "class")= chr [1:3] "XY" "MULTIPOLYGON" "sfg"
##  - attr(*, "sf_column")= chr "geometry"
##  - attr(*, "agr")= Factor w/ 3 levels "constant","aggregate",..: NA NA NA NA NA NA NA NA NA NA ...
##   ..- attr(*, "names")= chr [1:14] "REGION" "DIVISION" "STATEFP" "STATENS" ...
```

- To use the `tmap` package, we need to merge the two data sets by the state names and convert to `sf` data type

```
datapop$NAME =datapop$STNAME
pop_2020=datapop%>%filter(Year==2020)
state_pop_2020=left_join(pop_2020, contiguous_states, by='NAME')
dim(state_pop_2020)
```

```
## [1] 52 19
```

```
state_pop_2020=st_sf(state_pop_2020)
```

Plotting choropleth maps with `tmap`

```
tm_shape(state_pop_2020) + tm_polygons("Pop", id = "NAME")
```

Resources

- **Analyzing US Census Data: Methods, Maps, and Models in R** https://walker-data.com/census-r/index.html
- **Geocomputation with R** https://geocompr.robinlovelace.net/index.html
- **Geospatial Health Data: Modeling and Visualization with R-INLA and Shiny** https://www.paulamoraga.com/book-geospatial/index.html
- **Interactive Maps in R** https://bhaskarvk.github.io/user2017.geodataviz/notebooks/03-Interactive-Maps.nb.html#using_tmap
- **Making Maps with R** https://bookdown.org/nicohahn/making_maps_with_r5/docs/introduction.ht
- **Mapping in R** https://map-rfun.library.duke.edu/
- **Maps in R** https://www.emilyburchfield.org/courses/eds/making_maps_in_r
- **Mastering Software Development in R** https://bookdown.org/rdpeng/RProgDA/
- **tmap: get started!** https://cran.r-project.org/web/packages/tmap/vignettes/tmap-getstarted.html
- **tmap : the easy way to plot thematic maps and show them interactively in R** https://tlorusso.github.io/geodata_workshop/tmap_package
- **Mapping the US with `usmap`** https://cran.r-project.org/web/packages/usmap/vignettes/mapping.html

License