Making Maps with R - Part III
Choropleth Map

Xuemao Zhang
Department of Mathematics
East Stroudsburg University

June 27, 2019

Outline

- Overview
- County Population Data
- Map Visualization of State Population Data
- Map Visualization of County Population Data

Install the following packages if you don't have them.

```
install.packages("ggplot2");
install.packages("usmap");
install.packages("dplyr");
install.packages("gridExtra");
```

Overview

- The **choropleth map** colors regions 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.
- We discuss plot of choropleth maps using usmap package and tmap package.
- 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`. Check this package if you are a fan of thematic 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/2010-2017/counties/totals/
- The data can be read from the web directly:

```
#url="https://www2.census.gov/programs-surveys/popest/datasets/2010-
#data = read.csv(url, stringsAsFactors = FALSE);
#dim(data);
```

US Population Data

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

```
data = read.csv("F:/DataCamp/data/co-est2017-alldata.csv",
                header=TRUE, sep=",");
dim(data);
```

```
## [1] 3193  132
```

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

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

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

```
str(datapop0);
```

```
## 'data.frame':    52 obs. of  17 variables:
##  $ SUMLEV          : int  40 40 40 40 40 40 40 40 40 50 ...
##  $ REGION          : int  3 4 4 3 4 4 1 3 3 3 ...
##  $ DIVISION        : int  6 9 8 7 9 8 1 5 5 5 ...
```

US Population Data

- Subset data by years

```
#construct 8 sub-data with variable year added
library(dplyr);
datayears=list();  #creat a list
Years=seq(2010,2017,1);  #All possible years
vars=c("POPESTIMATE2010","POPESTIMATE2011","POPESTIMATE2012",
       "POPESTIMATE2013","POPESTIMATE2014","POPESTIMATE2015",
       "POPESTIMATE2016","POPESTIMATE2017");
for (i in 1:length(Years))
{
subdata0=dplyr::select(datapop0, STATE,STNAME, Pop=vars[i]);
#sub dataset in Years[i] and change the variable name to 'Pop'
subdata0=dplyr::mutate(subdata0, Year=Years[i]);
#Add a variable Year
datayears[[i]]=subdata0;
}
```

US Population Data

- Row Combine all years data

```
# Then combine these data by rows
datapop=datayears[[1]];
for (i in 2:length(Years))
{
datapop=dplyr::bind_rows(datapop, datayears[[i]]);
}
datapop=dplyr::rename(datapop, fips=STATE);
#Change the variable 'STATE' to fips
str(datapop);
```

```
## 'data.frame':    416 obs. of  4 variables:
##  $ fips  : int  1 2 4 5 6 8 9 10 11 11 ...
##  $ STNAME: Factor w/ 51 levels "Alabama","Alaska",..: 1 2 3 4 5 6
##  $ Pop   : int  4785579 714015 6407002 2921737 37327690 5048029 3
##  $ Year  : num  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010
```

```
head(datapop);
```

```
##   fips     STNAME      Pop Year
## 1    1     Alabama  4785579 2010
```

Map Visualization of State Population Data

- 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 2010.

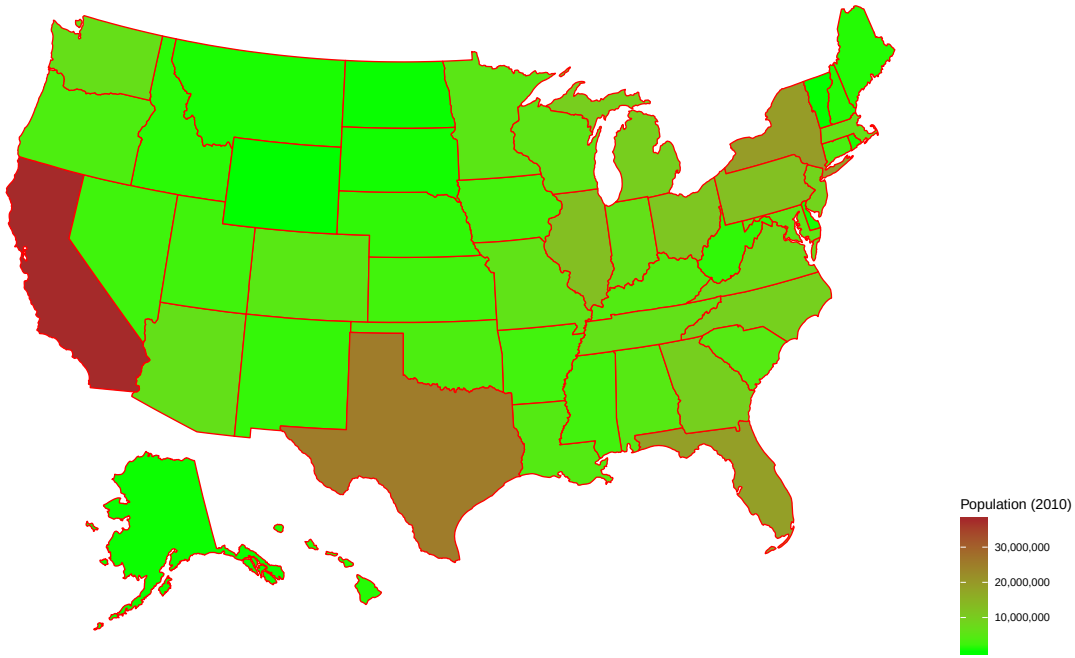
```
datapop_2010=dplyr::filter(datapop, Year == 2010);
str(datapop_2010);
```

```
## 'data.frame':    52 obs. of  4 variables:
##  $ fips  : int  1 2 4 5 6 8 9 10 11 11 ...
##  $ STNAME: Factor w/ 51 levels "Alabama","Alaska",..: 1 2 3 4 5 6
##  $ Pop   : int  4785579 714015 6407002 2921737 37327690 5048029 3
##  $ Year  : num  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010
```

Map Visualization of State Population Data

- First, we visulize the population data for the year 2010.

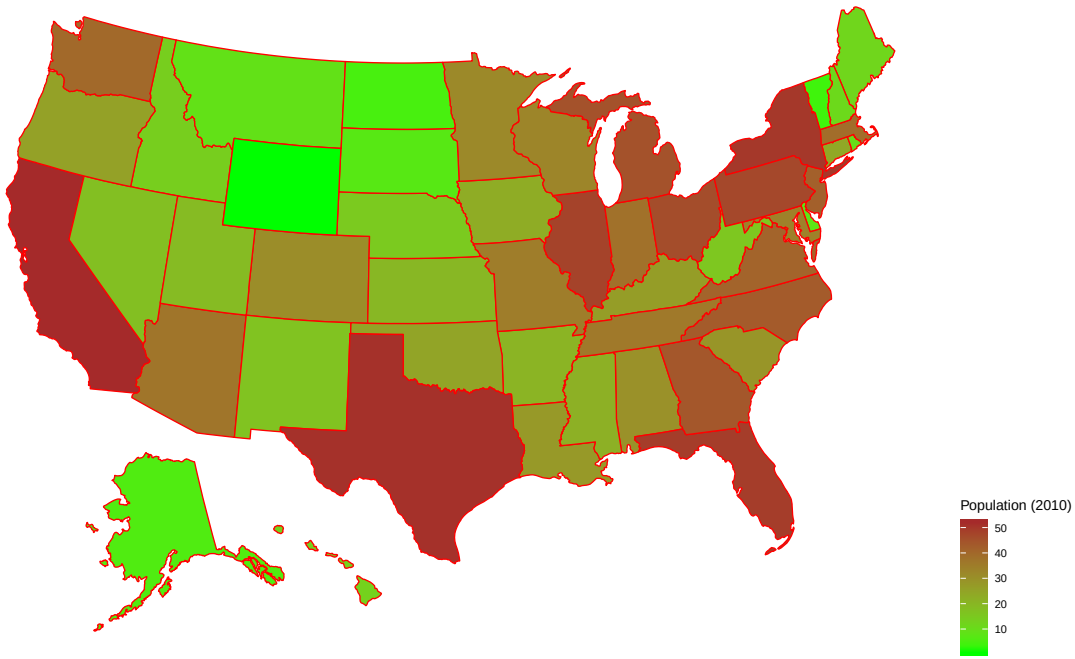
```
library(usmap); library(ggplot2);
plot_usmap(data = datapop_2010, values = "Pop", lines = "red") +
scale_fill_continuous(low = "green", high = "brown",
  name = "Population (2010)", label = scales::comma)+
theme(legend.position = "right");
```



Map Visualization of State Population Data

- 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.

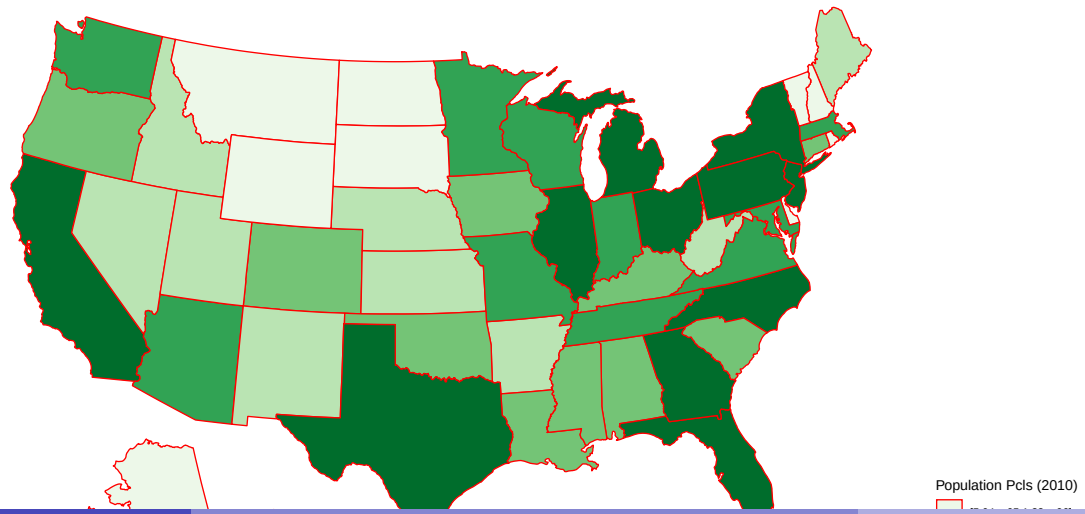
```
datapop_2010 = dplyr::mutate(datapop_2010, Rpop = rank(Pop));
plot_usmap(data = datapop_2010, values = "Rpop", lines = "red") +
scale_fill_continuous(low = "green", high = "brown",
  name = "Population (2010)", label = scales::comma)+
theme(legend.position = "right");
```



Map Visualization of State Population Data

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

```
bins=6;  # consider 6 percentitles
datapop_2010 = dplyr::mutate(datapop_2010,
pcls = cut(Pop, quantile(Pop, seq(0, 1, len = bins)),
           include.lowest = TRUE));
plot_usmap(data = datapop_2010, values = "pcls", lines = "red") +
scale_fill_brewer(palette =  "Greens",
                  name = "Population Pcls (2010)")+
theme(legend.position = "right");
```



Map Visualization of State Population Data

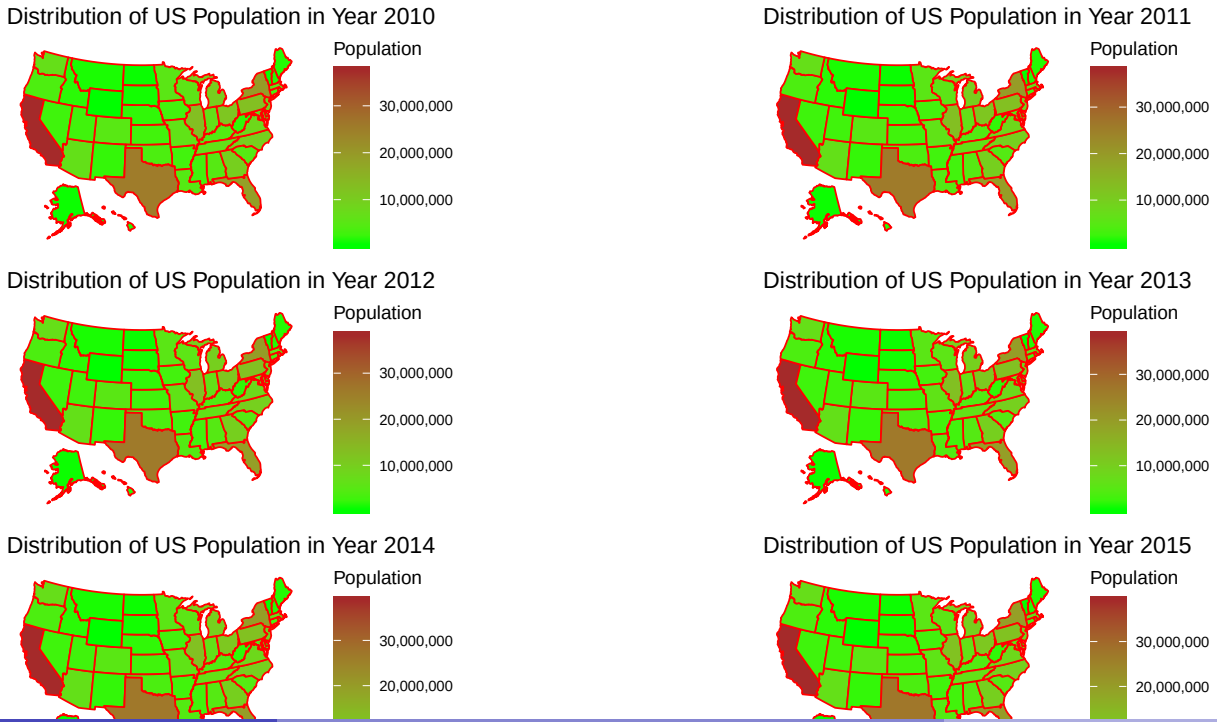
- Now we plot the population data by Year. For this, we define a function of year.

```
PlotPop= function(x)
{
datapop_year=dplyr::filter(datapop, Year == x);
# datapop_year=datapop[which(datapop$Year==x),];
plot_usmap(data = datapop_year, values = "Pop", lines = "red") +
scale_fill_continuous(low = "green", high = "brown",
  name = "Population", label = scales::comma)+
  ggtitle(paste("Distribution of US Population in Year",x))+
theme(legend.position = "right");
}
```

Map Visualization of State Population Data

- Show all 8 maps

```
library(gridExtra);
PopMaps=list();
for( i in 1:length(Years))
{
PopMaps[[i]]=PlotPop(i+2009);
}
gridExtra::grid.arrange(grobs = PopMaps,nrow=4,ncol=2);
```



Map Visualization of Population Data

- Animated Maps

```
library(magick);
```

```
## Linking to ImageMagick 6.9.9.14
## Enabled features: cairo, freetype, fftw, ghostscript, lcms, pango
## Disabled features: fontconfig, x11
```

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

```
## pdf
##   2
```

```
Popanimation = image_animate(img, fps = 1);#show(Popanimation);
image_write(Popanimation, "Popanimation.gif")
```

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

Map Visualization of County Population Data

- First, remove the rows with state population total.

```
datapop1=data[,1:17]; #select the first 17 variables
rowsremoved=which(datapop1$CTYNAME%in%datapop1$STNAME);
datapop1 = datapop1[-rowsremoved,];
dim(datapop1);  #52 rows are removed
```

```
## [1] 3141   17
```

```
str(datapop1);
```

```
## 'data.frame':    3141 obs. of  17 variables:
##  $ SUMLEV          : int  50 50 50 50 50 50 50 50 50 50 ...
##  $ REGION          : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ DIVISION        : int  6 6 6 6 6 6 6 6 6 6 ...
##  $ STATE           : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ COUNTY          : int  1 3 5 7 9 11 13 15 17 19 ...
##  $ STNAME          : Factor w/ 51 levels "Alabama","Alaska",..:
##  $ CTYNAME         : Factor w/ 1927 levels "Abbeville County",..
##  $ CENSUS2010POP   : int  54571 182265 27457 22915 57322 10914 2
##  $ ESTIMATESBASE2010: int  54571 182265 27457 22919 57324 10911 2
##  $ POPESTIMATE2010 : int  54750 183110 27332 22872 57381 10880 2
```

Map Visualization of County Population Data

- Subset data by years.
- Again, the fact that some counties with large population sizes will make other county population sizes very small. We consider quantile bins.

```
#construct 8 sub-data with variable year added
bins=6; CountyYears=list();  #creat a list
Years=seq(2010,2017,1);  #All possible years
vars=c("POPESTIMATE2010","POPESTIMATE2011","POPESTIMATE2012",
       "POPESTIMATE2013","POPESTIMATE2014","POPESTIMATE2015",
       "POPESTIMATE2016","POPESTIMATE2017");
for (i in 1:length(Years))
{
subdata1=dplyr::select(datapop1, STATE,COUNTY, Pop=vars[i]);
#sub dataset in Years[i] and change the variable name to 'Pop'
subdata1=dplyr::mutate(subdata1, Year=Years[i]);#Add the Year varial
subdata1=dplyr::mutate(subdata1,
pcls = cut(Pop, quantile(Pop, seq(0, 1, len = bins)),
include.lowest = TRUE));
CountyYears[[i]]=subdata1;
}
```

Map Visualization of County Population Data

- Row Combine all years data

```
# Then combine these data by rows
CountyPop=CountyYears[[1]];
for (i in 2:length(Years))
{
CountyPop=dplyr::bind_rows(CountyPop, CountyYears[[i]]);
}
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to

## Warning in bind_rows_(x, .id): binding character and factor vecto
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vecto
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vecto
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vecto
## coercing into character vector
```

Map Visualization of County Population Data

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

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

```
## 'data.frame':    54187 obs. of  10 variables:
##  $ long  : num  1225889 1244873 1244129 1272010 1276797 ...
##  $ lat   : 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
```

Map 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$STATE=as.character(CountyPop$STATE);
CountyPop$COUNTY=as.character(CountyPop$COUNTY);
CountyPop=mutate(CountyPop, fips=NA); #add the variable fip - county
CountyPop$fips=ifelse(nchar(CountyPop$COUNTY)==3,
            paste(CountyPop$STATE, CountyPop$COUNTY, sep=""),
            ifelse(nchar(CountyPop$COUNTY)==2,
            paste(CountyPop$STATE, CountyPop$COUNTY, sep="0"),
            paste(CountyPop$STATE, CountyPop$COUNTY, sep="00")));
str(CountyPop);
```

```
## 'data.frame':    25128 obs. of  6 variables:
##  $ STATE : chr  "1" "1" "1" "1" ...
##  $ COUNTY: chr  "1" "3" "5" "7" ...
##  $ Pop   : int  54750 183110 27332 22872 57381 10880 20944 118466
##  $ Year  : num  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010
##  $ pcls  : chr  "(3.67e+04,8.99e+04]" "(8.99e+04,9.82e+06]" "(1.8
##  $ fips  : chr  "1001" "1003" "1005" "1007" ...
```

```
head(CountyPop);
```

Map Visualization of County Population Data

- Check if there are any missing values.

```
#CountyPop$fips=as.integer(CountyPop$fips);
dim(CountyPop);
```

```
## [1] 25128     6
```

```
which(complete.cases(CountyPop)==F); #check if there are missing va
```

```
## integer(0)
```

```
str(CountyPop);
```

```
## 'data.frame':    25128 obs. of  6 variables:
##  $ STATE : chr  "1" "1" "1" "1" ...
##  $ COUNTY: chr  "1" "3" "5" "7" ...
##  $ Pop   : int  54750 183110 27332 22872 57381 10880 20944 118466
##  $ Year  : num  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010
##  $ pcls  : chr  "(3.67e+04,8.99e+04]" "(8.99e+04,9.82e+06]" "(1.8
##  $ fips  : chr  "1001" "1003" "1005" "1007" ...
```

Map Visualization of County Population Data

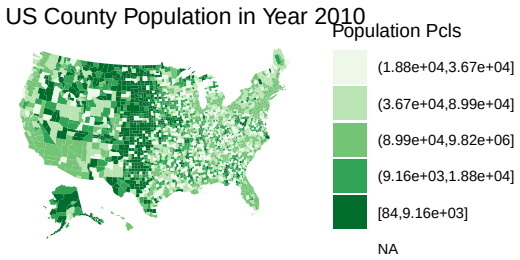
- Now we plot the population data by Year. For this, we define a function of year.

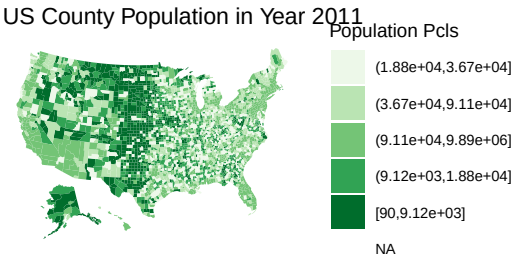
```
PlotCountyPop= function(x)
{
CountyPop_year=dplyr::filter(CountyPop, Year == x);
plot_usmap(regions="counties",data = CountyPop_year,
           values ="pcls", lines=NA) +
ggplot2::scale_fill_brewer(palette = "Greens",
                  name = "Population Pcls")+
ggtitle(paste("US County Population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

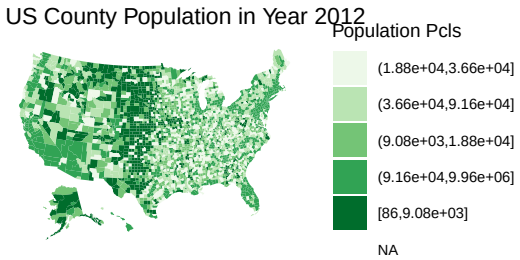
Map Visualization of County Population Data

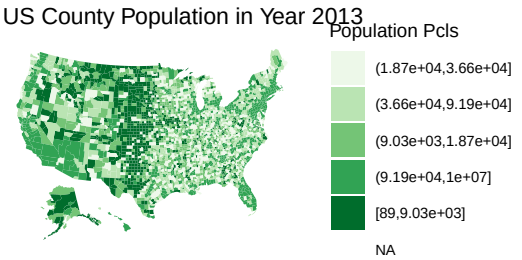
- Show all 8 maps

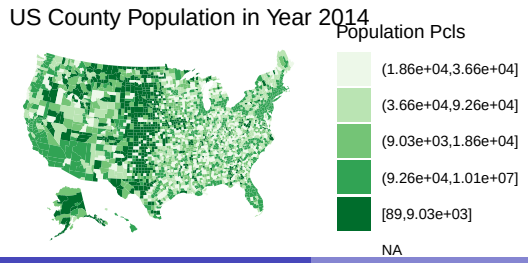
```
PopCountyMaps=list();
for( i in 1:length(Years))
{
PopCountyMaps[[i]]=PlotCountyPop(i+2009);
}
gridExtra::grid.arrange(grobs = PopCountyMaps,nrow=4,ncol=2);
```

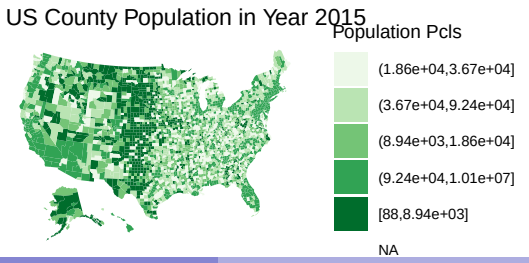












Map Visualization of County Population Data

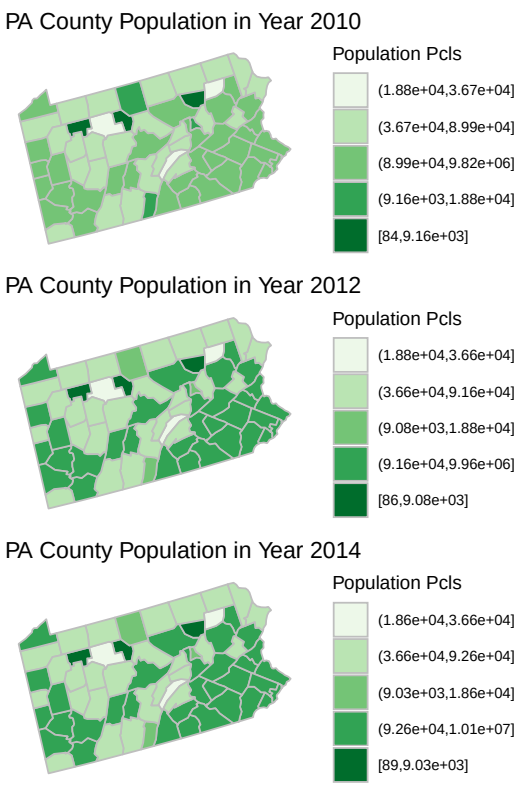
- Now let's plot the poulation distribution in certain states.

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

Map Visualization of County Population Dat

- Show all 8 maps

```
PACountyMaps=list();
for( i in 1:length(Years))
{
PACountyMaps[[i]]=PACountyPop(i+2009);
}
gridExtra::grid.arrange(grobs = PACountyMaps,nrow=4,ncol=2);
```



Map Visualization of County Population Data

- Animated Maps

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

```
## pdf
##   2
```

```
PACountyanimation = image_animate(img, fps = 1);#show(PACountyanima
image_write(PACountyanimation, "PACountyanimation.gif")
```

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

Questions?