Intro\_R\_Spatial\_Tools

RMM

2023-09-22

### R Markdown

Both R and Python have libraries that can process files referred to as “shape files”. Details on the definition and development of the format can be read at [wikipedia] (<https://en.wikipedia.org/wiki/Shapefile>). An important thing to know is that whereas shape file refers to data that is used to represent geographical features, it needs 3 files to be operational.

### .shp

where coordinates of geographical features are kept.

### .shx

an indexing file for geometric features.

### .dbf

data that is associated with the geometric features. The folder we are going to download exists within the US Census Bureau website. From the website we navigate to where the [shape files for all the counties in USA are located](https://www2.census.gov/geo/tiger/TIGER2021/COUNTY/)

As you can see from the download, there are additional files present besides the .shp, .shx and .dbf. These are files that are not required for R and Python to create informative maps but we will keep them in our folder. In this project we make use of a multitude of spatial tools and functions for statistical analysis but for now we will focus only on the mapping aspects and the simple features, [sf package](https://r-spatial.github.io/sf/).

# Loading the library, if you are using Windows you will have to make sure you have Rtools before being able to use the library -->  
library(sf)

## Linking to GEOS 3.11.2, GDAL 3.6.2, PROJ 9.2.0; sf\_use\_s2() is TRUE

#We downloaded the shape zip folder, unzipped all the files in it to the data folder and read it in with the read\_sf command-->   
shape <- read\_sf(dsn = "C:/Users/rm84/Desktop/research/HMM/data/tl\_2021\_us\_county.shp")  
#as can be seen from the dim command you have 3234 observations (counties) and 18 features associated with each county  
dim(shape)

## [1] 3234 18

#In R, objects have attributes. The shape object has 5 of these  
length(attributes(shape))

## [1] 5

#We will look at only the first attribute's values  
attributes(shape)[1]

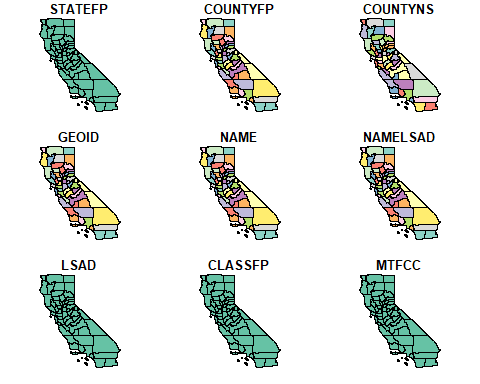
## $names  
## [1] "STATEFP" "COUNTYFP" "COUNTYNS" "GEOID" "NAME" "NAMELSAD"  
## [7] "LSAD" "CLASSFP" "MTFCC" "CSAFP" "CBSAFP" "METDIVFP"  
## [13] "FUNCSTAT" "ALAND" "AWATER" "INTPTLAT" "INTPTLON" "geometry"

#Names function will give us the values in the first attribute   
names(shape)

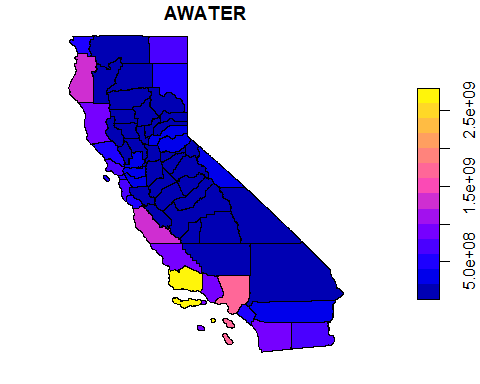
## [1] "STATEFP" "COUNTYFP" "COUNTYNS" "GEOID" "NAME" "NAMELSAD"  
## [7] "LSAD" "CLASSFP" "MTFCC" "CSAFP" "CBSAFP" "METDIVFP"  
## [13] "FUNCSTAT" "ALAND" "AWATER" "INTPTLAT" "INTPTLON" "geometry"

#We will be working with only the Californian state   
#selecting California via FIPS state code as you can see selection is done via base R via the %in% statement  
shape=shape[(shape$STATEFP %in% '06'),]  
  
#We could look at a very simple set of plots  
plot(shape)

## Warning: plotting the first 9 out of 17 attributes; use max.plot = 17 to plot  
## all



#The shape function has 17 attributes and R chooses to plot a smaller number of them to fit the screen.   
  
  
#We would rather do something a bit more useful and plot one of the attributes we are interested in  
plot(shape[15])



#Note that if you wrote plot(shape$AWATER) you get a scatterplot.

A good [vignette](https://r-spatial.github.io/sf/articles/) exists to explain the sf package.

Before we introduce better looking plots we need to understand how to merge data to a shapefile.

We created a dataframe named by wrangling through a multitude of datasets which will be discussed on a later post. It is important to look at some of the output from this dataset.

library(sf)  
library(ggplot2)  
library(ggspatial)  
library(ggpubr)  
library(cowplot)

##   
## Attaching package: 'cowplot'

## The following object is masked from 'package:ggpubr':  
##   
## get\_legend

library(viridis)

## Loading required package: viridisLite

data\_for\_spatial=read.table(header=TRUE,sep=',','C:/Users/rm84/Documents/data\_for\_spatial.csv')  
poverty\_for\_spatial=read.table(header=TRUE,sep=',','C:/Users/rm84/Documents/poverty\_for\_spatial.csv')  
#Each of the 58 counties of California in alphabetical and FIPS order have 164 columns of information associated with it. The last 154 columns represent 2 sets of variables across a 77 biweek time period. Mortality and Vaccinations Rates.  
dim(data\_for\_spatial)

## [1] 58 244

#First 10 values in the data.frame object  
#  
names(data\_for\_spatial)[1:10]

## [1] "X" "ID" "Counties" "Poverty20" "Poverty21" "Income20"   
## [7] "Income21" "Density" "Gini20" "Gini21"

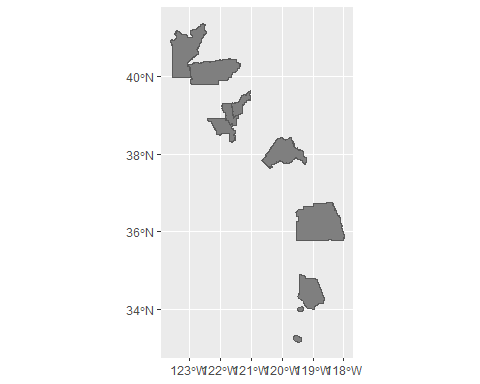
#identify the unique keys in each object. shape and #CASUmmary has 58 unique GEOID and ID values representing #the counties in California They are 3 digit FIPS codes.  
#001 003 etc...  
#it is important to first write the shape file.  
shapeanddata=merge(shape,data\_for\_spatial,by.y="ID",by.x="COUNTYFP")  
meanpov=mean(poverty\_for\_spatial[,4])

## Warning in mean.default(poverty\_for\_spatial[, 4]): argument is not numeric or  
## logical: returning NA

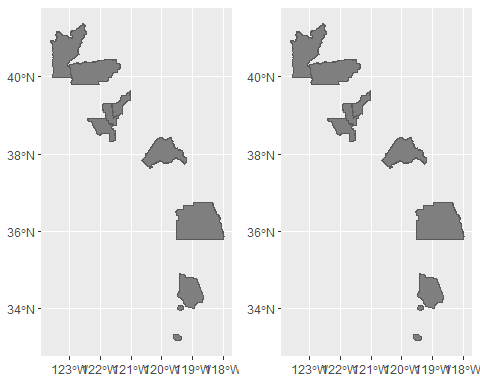
sdpov=sd(poverty\_for\_spatial[,4])

## Warning in var(if (is.vector(x) || is.factor(x)) x else as.double(x), na.rm =  
## na.rm): NAs introduced by coercion

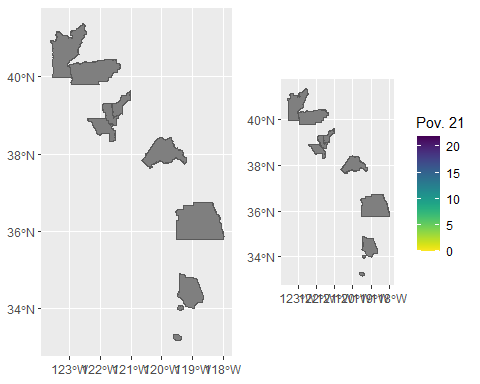
#\_spatial options require ggspatial package  
#operations on data are allowed. In this instance we used it  
#to unnormalize the poverty variable  
map\_Pov20=ggplot() +  
 annotation\_spatial(shapeanddata) +  
 layer\_spatial(shapeanddata, aes(fill = (Poverty20\*sdpov+meanpov)))+  
 labs(fill = "Pov. 20")  
  
map\_Pov20



map\_Pov21=ggplot() +  
 annotation\_spatial(shapeanddata) +  
 layer\_spatial(shapeanddata, aes(fill = (Poverty21\*sdpov+meanpov)))+  
 labs(fill = "Pov. 21")  
  
#Different scales we need to put them on the same scale  
plot\_grid(map\_Pov20,map\_Pov21)



#########################################  
#Let us hide the legend and use a common scale for these two years  
map\_Pov20=ggplot() +  
 annotation\_spatial(shapeanddata) +  
 layer\_spatial(shapeanddata, aes(fill = (Poverty20\*sdpov+meanpov)))+  
 labs(fill = "Pov. 20")+scale\_fill\_viridis(limits = c(0,22),direction=-1)+  
 theme(legend.position = "none",plot.title =element\_blank())  
  
  
map\_Pov21=ggplot() +  
 annotation\_spatial(shapeanddata) +  
 layer\_spatial(shapeanddata, aes(fill = (Poverty21\*sdpov+meanpov)))+  
 labs(fill = "Pov. 21")+scale\_fill\_viridis(limits = c(0,22),direction=-1)  
  
#This has obvious scale on the output issues   
plot\_grid(map\_Pov20,map\_Pov21)



map\_Pov20=ggplot() +  
 annotation\_spatial(shapeanddata) +  
 layer\_spatial(shapeanddata, aes(fill = (Poverty20\*sdpov+meanpov)))+  
 theme(legend.title= element\_blank())+  
 labs(fill = "Pov. 20")+scale\_fill\_viridis(limits = c(0,22),direction=-1)  
   
  
  
map\_Pov21=ggplot() +  
 annotation\_spatial(shapeanddata) +  
 layer\_spatial(shapeanddata, aes(fill = (Poverty21\*sdpov+meanpov)))+annotate("text", x = -118, y =40 , label = "Merced")+  
 annotate("segment", color="red", x=-118, xend = -120.5,   
 y=39.8,   
 yend=37.2, arrow=arrow(length=unit(0.2,"cm")))+  
 labs(fill = "Pov. 21")+scale\_fill\_viridis(limits = c(0,22),direction=-1)  
  
  
  
map\_Pov2021 <- ggarrange(labels=c("Poverty 2020","Poverty 2021"),map\_Pov20, map\_Pov21, nrow = 1, align = "h", common.legend = TRUE)  
annotate\_figure(fig.lab.face="bold",fig.lab.size=14,fig.lab.pos="top.left",map\_Pov2021, fig.lab = "Maps of Poverty Percentages")

