Lab Exercise #5

This week, we will explore spatial neighborhood and weights which is an important step before we conduct spatial autocorrelation tests and construct regression models.

Download the “sids2” data and unzip it to your folder.

```
#R libraries we’ll use
install.packages("ctv")
library("ctv")
library(maptools)
install.packages("sf")
library(sf)
install.packages("terra")
library(terra)
install.packages("spdep")
library(spdep)
install.packages("rgdal", repos = "https://packagemanager.posit.co/cran/2023-10-13")
library(rgdal)

# Import shapefiles into R
> sids<-readShapePoly(“path to shapefile/shpfile.shp”)
> class(sids)

# Check the projection of the shapefile

>proj4string(sids)

#Projeting a Shapefile.
#If the shapefile has no .prj file associated with it, can assign a coordinate system

> proj4string(sids)<-CRS("+proj=longlat +ellps=WGS84")

#If a shapefile has a CRS String (projection/coordinate system) associated with it can be reprojected using spTransform().

>sids_NAD<-spTransform(sids, CRS("+init=epsg:2031")) #this is in NAD27 UTM Zone 17N
>sids_SP<-spTransform(sids, CRS("+proj=lcc +lat_1=34.33333333333334
+lat_2=36.16666666666666 +lat_0=33.75 +lon_0=-79 +x_0=609601.2199999999 +y_0=0
+ellps=GRS80 +datum=NAD83 +to_meter=0.3048006096012192 +no_defs"))  #this is in North
CarolinaNAD 83 State Plane
```

It is important to use an appropriate projection, especially when working at large scales (continental, global, etc.).
For the properties of projections see: http://pubs.usgs.gov/pp/1453/report.pdf
For a list of applicable CRS codes: http://www.spatialreference.org/ref/

If the file does have projection information you can import the shapefile with readOGR(). The result is the same but readOGR will read the projection information.
Now you can plot the data in three different coordinate systems.

```
>par(mfrow=c(3,1))
>plot(sids, axes=T)
>title("WGS84")
>plot(sids_NAD, axes=T)
>title("NAD27")
>plot(sids_SP, axes=T)
>title("NC State Plane")
```

Q1. Take a screen shot of your figures and describe the differences (10 points).

#Contiguity based neighbors

Counties sharing any boundary point (QUEEN) are taken as neighbors, using the poly2nb function, which accepts a SpatialPolygonsDataFrame

```
> library(spdep)
> sids_nbq<-poly2nb(sids) (QUEEN)
```

If contiguity is defined as counties sharing more than one boundary point (ROOK), the queen= argument is set to FALSE

```
> sids_nbr<-poly2nb(sids, queen=FALSE)

> coords<-coordinates(sids)
> dev.off() #clears the screen and the panels
> par(mfrow=c(1,1))
> plot(sids_nbq, coords) #This plots the QUEEN case
> plot(sids, add= T) #Add the county outlines
```

Q2. Take a screen shot of your figure (10 points).

Q3. Plot the ROOK case and take a screenshot of your figure. Briefly describe the difference between the figure from Q2 and Q3 (10 points).

#Distance based (k nearest) neighbor

```
coords<-coordinates(sids_SP)
sids_kn1<-knn2nb(knearneigh(coords, k=1))
sids_kn2<-knn2nb(knearneigh(coords, k=2))
sids_kn4<-knn2nb(knearneigh(coords, k=4))
plot(sids_SP) > plot(sids_kn1, coords, add=T)
plot(sids_SP) > plot(sids_kn2, coords, add=T)
plot(sids_SP) > plot(sids_kn4, coords, add=T)
```

Q4. Take a screen shot of your figures and briefly describe their differences (10 points).

```
#assign neighbors based on a specified distance
dist<-unlist(nbdists(sids_kn1, coords))
#Notice that we are using the State Plane version so that the distances are easier to interpret.
summary(dist)
max_k1<-max(dist)
sids_kd1<-dnearneigh(coords, d1=0, d2=0.75*max_k1)
sids_kd2<-dnearneigh(coords, d1=0, d2=1*max_k1)
sids_kd3<-dnearneigh(coords, d1=0, d2=1.5*max_k1)
par(mfrow=c(3,1))
plot(sids_SP) > plot(sids_kd1, coords, add=T)
title("dist=0.75*max_k1")
plot(sids_SP) > plot(sids_kd2, coords, add=T)
title("dist=1*max_k1")
plot(sids_SP) > plot(sids_kd3, coords, add=T)
title("dist=1.5*max_k1")
```

Q5. Take a screen shot of your figures and briefly describe their differences (10 points).

Q6. Define neighbors based on a specified distance (the mean distance of neighbors). Take a screen shot of your plot and briefly describe their differences (10 points).

```
#Row-standardized weights matrix
sids_nbq_w<- nb2listw(sids_nbq)
sids_nbq_w
sids_nbq_w$neighbours[1:5]
```

Q7. What is the result? What does this mean? (10 points)

```
sids_nbq_w$weights[1:5]
```

Q8. What is the result? What does this mean? (10 points)

```
#Binary Weights
sids_nbq_wb<-nb2listw(sids_nbq, style="B")
sids_nbq_wb
sids_nbq_wb$weights[1:5]
```

Q9. What is the result? What does this mean? (10 points)

```
# Inverse distance weighting bases weights on the distance between centroids.
dist<-nbdists(sids_nbq, coordinates(sids_SP))
idw<-lapply(dist, function(x) 1/(x/1000))
sids_nbq_idwb<-nb2listw(sids_nbq, glist=idw, style="B")
sids_nbq_idwb$weights[1:5]
```

summary(unlist(sids_nbq_idwb$weights))

Q10. What is the result? What does this mean? (10 points)

Note:

For regions with no neighbors

```
# Islands do not have neighbors in a queen's or rook's case. Use k nearest
# neighbor to rectify, or use 'zero.policy=T' as an argument when
# assigning weights:
sids_nbq_w <- nb2listw(sids_nbq, zero.policy = T)
```