SVI Toolkit - Exercise 2

Explore Spatial Autocorrelation with the SVI

Learning Objectives:

1. Subset geographic data in R
2. Explore the spatial autocorrelation of the Social Vulnerability Index using the **spdep** package in R
3. Create maps showing spatial autocorrelation within the Social Vulnerability Index

Important note for this exercise:

This exercise is intended for people who have some familiarity of geospatial mapping and spatial concepts such as spatial autocorrelation. This exercise is not intended as a teaching tool for spatial concepts, but rather a way to use the SVI to apply these spatial concepts. Users of this exercise are encouraged to have prior foundational knowledge of geospatial mapping and spatial epidemiology before completing this exercise.

Defining Spatial Autocorrelation:

For this exercise, we are going to run some basic spatial models and explore the concept of **spatial autocorrelation** using the **spdep** package. For assistance using basic mapping in R, please see SVI Academic Toolkit Exercise 1.

Spatial autocorrelation indicates the occurrence of systematic spatial variability in the mapped variable of interest. In other words, it describes the degree to which a spatial variable (e.g., the SVI) is correlated with itself through space and the strength of this association (Gangodagamage et al, 2008). For example, areas that are ranked as having high social vulnerability may be geographically clustered nearby other areas that are similarly ranked as having high social vulnerability. The scale of the measurement of autocorrelation can be **global** (measured over the entire study area) or **local** (measured at specific locations) (Mendez, 2020). Spatial autocorrelation can be either positive or negative, with values between -1 to 1. If the autocorrelation is close to 1, this would mean the area is very tightly clustered with all geographic units being near each other having very similar values (e.g., high SVI areas are near high SVI areas). A “-1” would mean all geographic units are perfectly distributed with no similar values near each other (e.g., high SVI areas are near low SVI areas).

The concept of spatial autocorrelation in the context of area-level data is relevant to spatial epidemiology because it can indicate likely patterns that can be explained by the presence of a another variable or determinant, measurement error, spillover effects, issues with the statistical model, interaction, or other forces of dispersion (Agovino et al, 2018). We can quantify spatial autocorrelation using statistics such as Moran’s I, used for area-level or aggregated spatial data. Point pattern or geostatistical data have other corresponding statistics to measure autocorrelation, but we will demonstrate using Moran’s I later in this exercise. Further, we can also categorize spatial autocorrelation in several ways.

The concept of spatial autocorrelation is relevant to spatial epidemiology because it is a clue that there is likely a pattern in the data that can be explained by the presence of measurement error, spillover effects, issues with the statistical model, interaction, or other forces of dispersion. We can quantify spatial autocorrelation using statistics such as Moran's I and will demonstrate later in this exercise. Further, we can also categorize spatial autocorrelation in several ways:

- **Spatial Dependence -** the values of a variable in one location influence or are influenced by nearby values. This creates clusters and patterns within the spatial data.
- **Spatial Independence -** the values of a variable in one location are not related to the values of other locations and there are no spatial or geographic patterns in the data.
- **Negative Spatial Autocorrelation -** geographies (e.g., counties, census tracts, etc.) that are near to each other have *very different or contrasting* values, are usually *negatively spatially correlated*, and correlation values will be below zero.
- **Positive Spatial Autocorrelation** - geographies near to each other with more *similar* values rather than different are generally *positively spatially correlated* and correlation values will be above zero.
- **No spatial autocorrelation -** there is no pattern of values across geographies close to each other and the correlation value is zero.

As noted at the start of this exercise, it is important to have prior knowledge of the concept of spatial autocorrelation to help interpret findings. For more information on spatial autocorrelation, we have provided several references at the end of this document.

Part 1: Getting Started

Create a designated folder for your data files.

Download your files. Download the exercise and the **"SVI_2024_analytic.RDS"** file. Save these files in a folder on your computer's desktop named "**SVI Project**". Be sure that this folder is located on your local computer's drive and not on any type of cloud service to avoid issues with loading your data and saving your files. This is the folder you will set as your working directory for each of these exercises.

Set up your working directory.

```
#Here is an example of how your code may look. NOTE: you must make sure all of the "\"
#symbols are converted to "/" if you copy and paste your file path from your computer.

#This is especially important for PC users.

#Problem: Uh oh! This one won't run! Check direction of slashes.
  setwd("C:\Users\janedoe\Desktop\SVI Project")

#Solution:
  setwd("C:/Users/janedoe/Desktop/SVI Project")

#Now you try!
 setwd("Your file path here")
```

Load your R packages.

R Packages are containers for collections of R code that have a specific purpose or use. Many R packages are available and the ones you use will depend on what you are working on in R.

TIPS:

1. You only need to install packages once after downloading R and RStudio, but you do need to load them each time you use RStudio with the "**`library()`**" function.
   - To install the packages, run the first block of code below.
   - If you have installed the packages previously and only need to load them, run the second block of code below.
2. When you update R and RStudio on your computer, you *will* need to install your packages again.
3. Type **`>?nameofthepackage`** in the console to see a description and key information about the functions of the packages.

For this exercise, we will need the following packages:

```
#Use the code below to install the packages you need for this exercise.
#You only need to perform this task once after installing or updating R and RStudio.
install.packages("tidyverse")
install.packages("sf")
install.packages("tmap")
install.packages("tmaptools")
install.packages("RColorBrewer")
install.packages("spdep")
install.packages("rgeos")
install.packages("spgwr")
install.packages("gridExtra")
install.packages("rio")
install.packages("bispdep")
install.packages("rgeoda")

Use the code below to load each of the packages you need for this exercise. You need to
perform this task each time you use R and RStudio.
library(tidyverse)
library(sf)
library(tigris)
library(tmap)
library(tmaptools)
library(RColorBrewer)
library(spdep)
library(spdep)
library(rgeos)
library(gridExtra)
library(rio)
library(bispdep)
library(rgeoda)
```

Load your datasets into your R environment.

```
data <- readRDS(file = "SVI_2024_analytic_file.RDS")

#You can type in your state of interest in the quotation marks of the code below to
subset your state of interest from the entire United States dataset. Make sure that you
write the full name of the state and spell it correctly with sentence case
capitalization. For Washington D.C., write "District of Columbia".

#For this exercise, we will be using Georgia as an example.
```

```
my_state <- data[data$STATE==" Georgia",]

#This created a new dataset called “my_state” that only contains records within Georgia.
#Use the head function to check that your state of choice is listed in the STATE column.

head(my_state)


#Next, we will clean our dataset. First, subset a specific county of interest from the
my_state dataset that you have created. We will also take the variable "depression" out
of the data frame since we do not need it for this exercise, and we do not want missing
values for the depression variable to impact our analysis. To finish our data cleaning,
we will omit any observations with missing values so our subsequent code functions. Then,
use the head function to confirm that the dataset columns correspond to your state and
county of interest.

my_county <- my_state[my_state$COUNTY==" Fulton",]
my_county <- my_county %>%
                select(-c(depression))
my_county <- na.omit(my_county) #This code removes missing polygons
head(my_county)
```

Part 2: Defining Neighbors

Next, we need to define neighboring observations as polygons with assigned values. This means that we need to tell R which geographies (counties, census tracts) are next to each other.

First, we will be determining which polygons neighbor each other using contiguity.

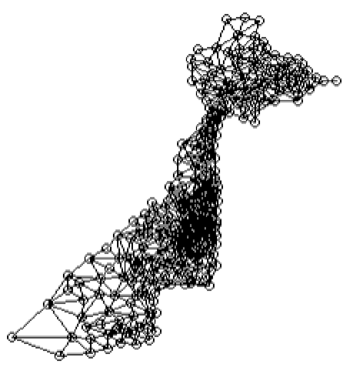
Contiguity is a method to define neighbors based on sharing a **border** or **boundary**. Similar to the pieces of a chess game, the *Queen's contiguity* method assigns neighbors by using both the edges and corners of the boundary. The *Rook contiguity* method assigns neighbors using only the shared boundary edges. Identifying where geographies share boundaries is important because they are places where there may be connections of spillover effects between neighboring populations. As an example of spill over in the context of the SVI, if an area with low social vulnerability attracts new businesses and investments, neighboring counties or census tracts might also indirectly experience economic benefits such as increase job opportunities or improved infrastructure. Such improvements could reduce the social vulnerability of these nearby communities even if those regions did not directly benefit from the initial investments.

In this example, we will be using census tracts (indicated by FIPS codes) as the unit of observation within your county of choice. For subsequent spatial autocorrelation analysis to proceed, we need to determine which polygons (i.e., census tracts) neighbor each other. Start by assigning neighbors using the Queen’s contiguity method. The Queen’s contiguity method assigns neighbors by both edges and corners of polygons. The code below creates a data file assigning neighbors to every census tract in the county dataset you created (the example below is called “my_county”) with SVI values only within your county of interest.

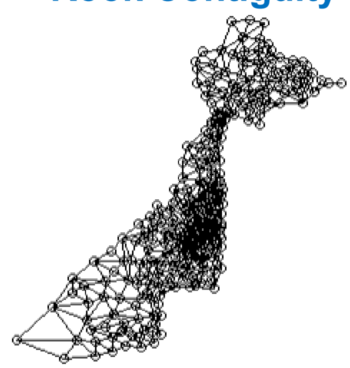
```
neigh_queen <- poly2nb(my_county) #Make queen neighbors list for every census tract in
#your county
neigh_queen

#Make queen neighbors plot.
plot.nb(neigh_queen, st_centroid(st_geometry(my_county)))

#Now run the same code but using the Rook configuration.

#Calculate rook case neighbors. Note, this code includes "queen = F" to switch from a
broader definition (Queen's, considers neighbors that share both an edge and a corner) to
a stricter one (Rook's, considers neighbors that share just an edge).
neigh_rook <- poly2nb(my_county, queen = F)
neigh_rook

#Visually compare queen versus rook contiguity by running the code below.

#rook
plot.nb(neigh_queen, st_centroid(st_geometry(my_county)))
#queen
plot.nb(neigh_rook, st_centroid(st_geometry(my_county)))
```

After running the code above, a map plotting the relationship between neighboring counties based on your selected contiguity method will be generated in your R Studio viewer. Below is an example of this plot using queen and rook contiguity, respectively.

Queen's Contiguity



Rook Contiguity



Examine the plots you have created with the two contiguity methods. Describe the visual differences, if any, you see between two plots. Similarities? What do you hypothesize may explain these differences/similarities?

Note: Depending on the county, the contiguity plots could be more similar or different compared to the example of Fulton County, GA.

Part 3: Assessing for spatial autocorrelation between neighbors

To test for spatial autocorrelation, we will use the **Moran's I statistic**. This statistical test can be used globally, with a single test statistic that describes autocorrelation over the entire map, or locally such that there is a test statistic for each location. For more on global and local spatial autocorrelation, please see the references at the end of this document.

We will also be using **weights** for neighbors because geographies that are near each other are often more similar than geographies that are not near to each other. Weights help us consider the similarity or dissimilarity between neighbors. In this exercise, weights between a pair of census tracts equal one if the two census tracts are neighbors and zero otherwise.

Use the code below to convert the neighbor data to a **listw** object. This will be used to determine how neighbors will be weighted in the statistical model. For this example, use the queen contiguity method from above with the **neigh_queen** variable.

```
listw <- nb2listw(neigh_queen, zero.policy = TRUE)
listw
```

Run the Moran's I test of spatial autocorrelation.

The value for the correlation will be between -1 and 1 similar to correlation coefficients that you may have been introduced to in other math and statistical courses.

- A value of 1 indicates perfect positive spatial autocorrelation (similar values are near each other).
- A value of -1 indicates perfect negative spatial autocorrelation (dissimilar values are near each other).
- A value of 0 indicates no spatial autocorrelation (no observed pattern).

Use the code below to generate a test of **global spatial autocorrelation** for the overall SVI.

```
moran.test(my_county$svi_overall, listw)

#Use the code below to generate a test of LOCAL spatial autocorrelation for overall SVI
using the local Moran’s I.
#Note, we are recreating the queen weights list using the rgeoda function queen_weights.
#As above, weights are one for a pair of census tracts that are neighbors and zero.

queenw <- queen_weights(my_county)
localmoran_svi <- local_moran(queenw, st_drop_geometry(my_county["svi_overall"]))

#Use the code below to print the p-values indicating the statistical significance of the
local autocorrelation.
print(localmoran_svi)
```

Test the significance of the relationships between the Local Moran statistic values using visualization.

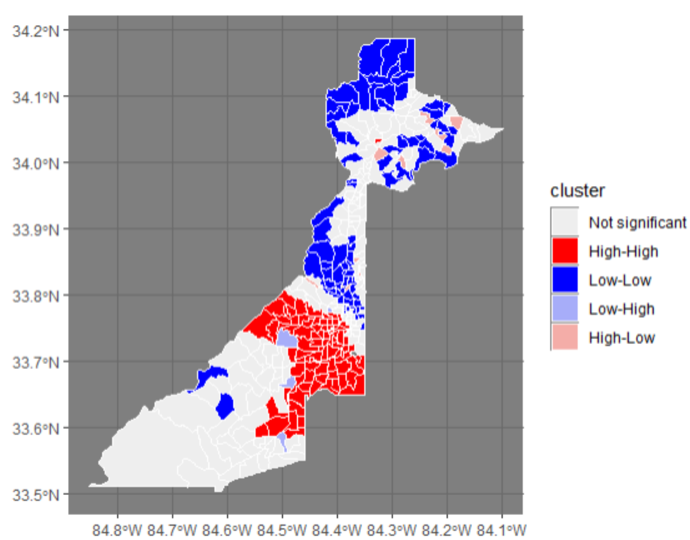
We will now leverage the object, “localmoran_svi”, created in the code above to create a visualization of spatial autocorrelation. With respect to spatial autocorrelation, we are interested in knowing:

- Are census tracts with a high SVI near census tracts with a high SVI?
- Are census tracts with a low SVI near census tracts with a low SVI?
- Are census tracts with a low SVI near census tracts with a high SVI?

- Are these relationships (e.g., spatial dependencies and clustering) statistically significant?

Use the code below to address these questions. Create the Local Indicator of Spatial Autocorrelation (LISA) test statistic cluster map to address these more specific questions. The code for this exercise is adapted from: https://uk.sagepub.com/en-gb/eur/an-introduction-to-r-for-spatial-analysis-and-mapping/book241031

```
#The code below assigns labels to different spatial autocorrelation categories (e.g.,
“High-High”, “Low-High”, etc.) These labels indicate which areas exhibit clustering
(e.g., areas with high values surrounded by other high values or low values surrounded by
high values, respectively).

moran_lbls <- lisa_labels(localmoran_svi)

#The code below assigns colors to each type of spatial cluster and maps the colors to
correspond to the labels created with the code above.

moran_colors <- setNames(lisa_colors(localmoran_svi), moran_lbls)

#The code below prepares a new clustered dataset that will be used to create the maps.

mycounty_clustered <- my_county %>%
  st_drop_geometry() %>%
  select(FIPS) %>%
  mutate(cluster_num = lisa_clusters(localmoran_svi) + 1,
         cluster = factor(moran_lbls[cluster_num], levels = moran_lbls)) %>%
  right_join(my_county, by = "FIPS") %>%
  st_as_sf()

#The code below creates the visualization of the clusters with the ggplot package. This
code will generate a map using the code above.

ggplot(mycounty_clustered, aes(fill = cluster)) +
  geom_sf(color = "white", size = 0) +
  scale_fill_manual(values = moran_colors, na.value = "green") +
 theme_dark()fill=colors,bty="n")
```



Interpret spatial autocorrelation results.

Now that we have our results for the Moran's I LISA statistical test that used Queen's contiguity weights, interpret the findings based on the typology described below:

- **High-High** regions indicate that counties with relatively high SVI values are surrounded by counties with relatively high SVI values. These regions are commonly described as "hot spots" and the regions are considered similar to one another (e.g., the individual county has a high SVI value, and the neighbors also have a high SVI value).
- **Low-Low** regions indicate that counties with relatively low SVI values are surrounded by counties with relatively low SVI values. These regions are commonly described as "cold spots" and the regions are considered similar to one another (e.g., the individual county has a low SVI value, and the neighbors also have a low SVI value).
- **Low-High/High-Low** regions: Low-high regions indicate that counties with relatively low SVI values are surrounded by counties with relatively high SVI values. High-Low regions indicate that counties with relatively high SVI values are surrounded by counties with relatively low SVI values. These **Low-High** and **High-Low** regions are commonly described as "spatial outliers" and the regions are considered opposite to one another (e.g., the individual county has a low SVI value, but the neighbors have high SVI).
- **Non-significant** regions indicate that there is no statistically significant spatial clustering of high and low LISA levels.This does not indicate there is no heterogeneity of SVI values across these counties. This test just tells us that there is no clustering or spatial autocorrelation based on the contiguity method chosen in this analysis.

References

Agovino, M., Aprile, M.C., Garofalo, A., Mariani, A., Cancer mortality rates and spillover effects among different areas: A case study in Campania (southern Italy), *Social Science & Medicine*, Volume 204, 2018, Pages 67-83, ISSN 0277-9536, https://doi.org/10.1016/j.socscimed.2018.03.027

Anselin, Luc. "Local indicators of spatial association—LISA." Geographical analysis 27.2 (1995): 93-115.

Chaney RA, Rojas-Guyler L. Spatial Analysis Methods for Health Promotion and Education. Health Promot Pract. 2016 May;17(3):408-15. doi: 10.1177/1524839915602438.

Gangodagamage, C., Zhou, X., Lin, H. (2008). Autocorrelation, Spatial. In: Shekhar, S., Xiong, H. (eds) *Encyclopedia of GIS*. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-35973-1_83

Haining R. *Spatial Data Analysis: Theory and Practice*. Cambridge University Press; 2003.

Mendez C. (2020). Spatial autocorrelation analysis in R. R Studio/RPubs. https://rpubs.com/quarcs-lab/spatial-autocorrelation