Critical Computational Geographies: Spatial Regression Analysis

Hye Ryeon Jang, Ph.D.

10/27/2025

Introduction

The United States often argues that social mobility is attainable through hard work and advertises itself as a pure meritocracy. This has been challenged throughout the United States' history and called into question the integrity of the statement. Specifically when interrogating the social position of non-white males. This session seeks to analyze how income varies between different demographic groups to understand what factors contribute to differing socioeconomic outcomes.

This lab session has been developed based on Sciabolazza (2017)'s Spatial Statistics and Spatial Econometrics and Alexander (2025)'s technical documentation on Critical Critical Computational Geographies - Measures of Segregation: Dissimilarity materials (Sciabolazza 2017) (Alexander 2025).

Research Question

How do different demographic backgrounds affect income level in Georgia?

Our independent variables are race (black and white), age, education, and immigrant status and the dependent variable is median household income. We chose to use these independent variables because, with the exception of gender, these are the most studied factors in social science when identifying discrimination. Income is used as an indicator of socioeconomic status because it reflects the level of economic capital individuals possess in the United States. While there are more factors that influence socioeconomic status, within Georgia and Atlanta income can be weighed more heavily and provide greater validity.

Necessary Packages

```
library(tidycensus)
library(tidyverse)
library(sf)
library(viridis)
library(scales)
library(mapview)
library(stargazer)
library(ggeffects)
library(gtools)
library(spdep)
library(gstat)
library(spectralGP)
```

```
library(MASS)
library(lattice)
library(nlme)
library(splancs)
library(lmtest)
library(boot)
library(RColorBrewer)
library(tmap)
library(spatialreg)
```

Census Data

We are going to collect the census data we want using the, `get_acs` function.

```
ga_tract <- get_acs(
  geography = "tract",
  variables = c(black = "B02001_003", # Black/African American population alone
                population = "B01001_001", # Total population
                immigrant = "B05002_014", # Immigrant population
                education = "B06009_005", # Individuals that have a bachelor's degree
                poverty = "B17001_001",  # Individuals whose household income
                                         # is below the poverty line
                income = "B19001_001",   # Household income
                young1 = "B01001_007",   # Male 18-19
                young2 = "B01001_008",   # Male 20
                young3 = "B01001_009",   # Male 21
                young4 = "B01001_010"    # Male 22-24


  ),
  state = "GA",
  year = 2023,
  geometry = T,
  output = "wide"
)
```

```
##   |                                                                              |
```

```
ga_tract <- na.omit(ga_tract)
head(ga_tract)
```

```
## Simple feature collection with 6 features and 22 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -84.31486 ymin: 30.5918 xmax: -82.57877 ymax: 31.9752
## Geodetic CRS:  NAD83
##         GEOID                                          NAME blackE blackM
## 1 13101880100      Census Tract 8801; Echols County; Georgia    103    120
## 2 13271950500     Census Tract 9505; Telfair County; Georgia    882    317
## 3 13185011100      Census Tract 111; Lowndes County; Georgia   1052    259
```

```
## 4 13277960700        Census Tract 9607; Tift County; Georgia   2778    511
## 5 13095001500     Census Tract 15; Dougherty County; Georgia   1652    417
## 6 13095010402 Census Tract 104.02; Dougherty County; Georgia   2483    585
##   populationE populationM immigrantE immigrantM educationE educationM povertyE
## 1        1358         189         11         17         82         47     1358
## 2        2897         348          0         14         74         63     2834
## 3        3643         436         52         60        291        103     2491
## 4        4639         586         36         58        366        175     4639
## 5        1803         404          5          9         90         71     1790
## 6        2988         584          0         14        246        133     2988
##   povertyM incomeE incomeM young1E young1M young2E young2M young3E young3M
## 1      189     500     107       0      14      27      28       0      14
## 2      347     831     152      26      40       0      14       0      14
## 3      432    1179     190     245     110      50      67      21      27
## 4      586    2019     269      33      64      14      32       0      14
## 5      405     896     162       0      14      39      45      71     112
## 6      584    1305     169     132     171       0      14       0      14
##   young4E young4M                       geometry
## 1      50      32 MULTIPOLYGON (((-83.05618 3...
## 2      25      37 MULTIPOLYGON (((-83.10079 3...
## 3     116     104 MULTIPOLYGON (((-83.30259 3...
## 4      36      36 MULTIPOLYGON (((-83.5864 31...
## 5       0      14 MULTIPOLYGON (((-84.17615 3...
## 6      10      16 MULTIPOLYGON (((-84.3132 31...
```

To control for population size we will wrangle the data so that our population variables appear as a fraction of the total population. We will include the variables:

- Proportion of Black People = blackP
- Proportion of Immigrants = immigrantP
- Proportion of people with a Bachelor's Degree = educationP
- Proportion of Young Men = youngmaleP
- Proportion of Individuals living below the Poverty line = povertyP

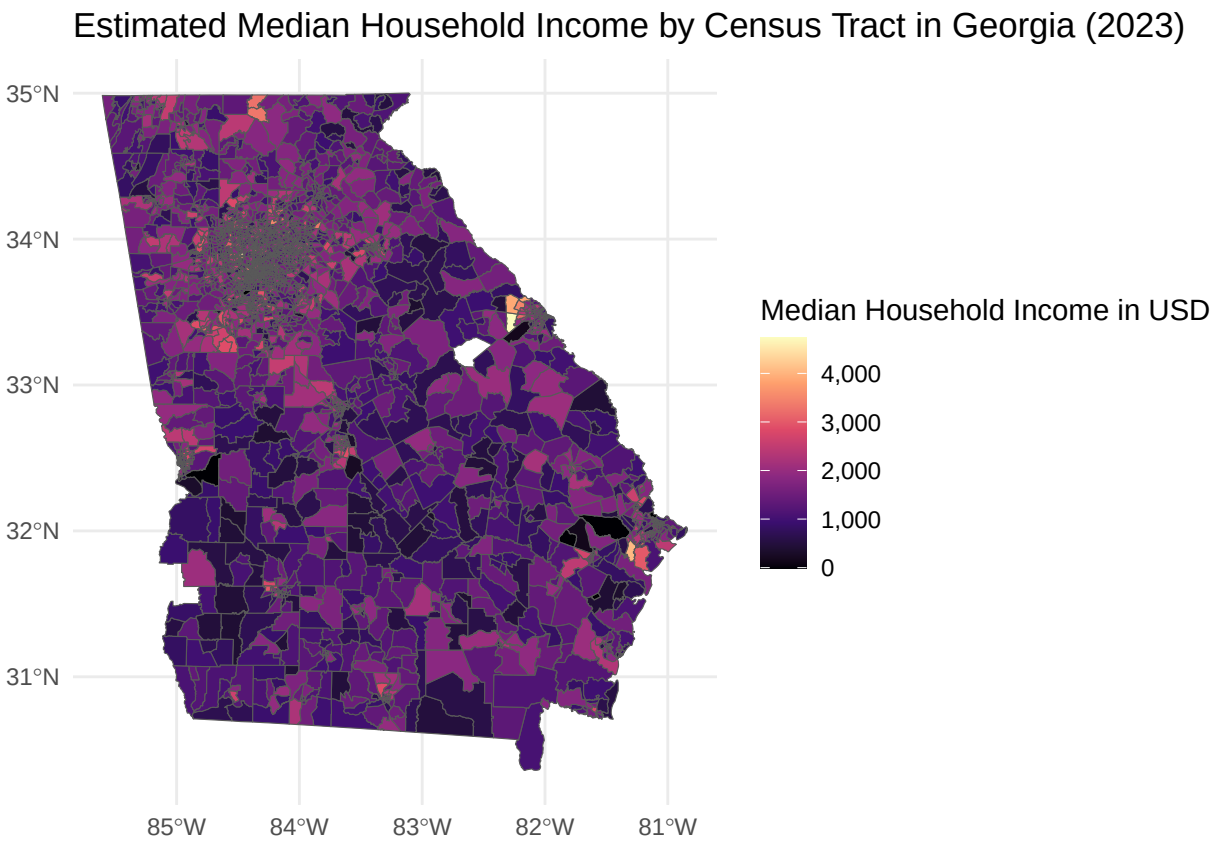
```
#Turning population data into proportions to control for population size

ga_tract <- ga_tract %>%
  mutate(youngmale = young1E + young2E + young3E + young4E) %>%
  mutate(blackP = blackE / populationE) %>%
  mutate(immigrantP = immigrantE / populationE) %>%
  mutate(educationP = educationE / populationE) %>%
  mutate(youngmaleP = youngmale / populationE) %>%
  mutate(povertyP = povertyE / populationE)
```

If you want to extract a shape file from the census data, below is the code.

```
#st_write(ga_tract, "GA_tract_income.shp")
# Dataset already exists in my working directory
```

Let's plot our median income variable per tract in Georgia.

```r
ggplot(ga_tract) +
  geom_sf(aes(fill = incomeE)) +
  scale_fill_viridis_c(option = "magma",
                       na.value = "grey50",
                       labels = comma) + # this is where we apply the scales lib
  labs(title = "Estimated Median Household Income by Census Tract in Georgia (2023)",
       fill = "Median Household Income in USD") +
  theme_minimal()
```



Spatial Data

Spatial data combine **attribute information** (e.g., name of the spatial object, population density, productivity, income, etc.) with **location information** (spatial coordinates such as latitude and longitude).

This combination allows researchers to analyze not only what is happening but also *where* it is happening.

Types of Spatial Data

- Point data: represent a single location defined by one coordinate pair (latitude and longitude).
  - Examples: houses, firms, bus stops, crime incidents, or GPS readings.
- Line data (arcs): represent paths or connections between ordered points. Each line consists of vertices connected by straight segments.
  - Examples: roads, rivers, pipelines, or migration routes.

- Polygon data: represent areas enclosed by one or more boundary lines. Each polygon has an interior and exterior.
  - Examples: countries, states, counties, campus boundaries (e.g., Morehouse College), or city limits (e.g., Atlanta, Georgia).
- Grid (raster) data: represent spatially continuous phenomena measured or estimated on a regular grid (a lattice of rectangular cells or pixels).
  - Examples: air temperature, precipitation, elevation, satellite imagery, or land cover.

Coordinate Reference Systems (CRS) and Projections

To uniquely reference a two-dimensional spatial point on Earth's surface, measure distances, and compute relative positions, we need a Coordinate Reference System (CRS) and map projection.

- A **Coordinate Reference System (CRS)** defines how locations on Earth are mapped onto a flat surface. It approximates the Earth as a sphere or ellipsoid, allowing us to use geometric properties (distance, area, direction) consistently.
- **Geographic coordinates** are expressed as pairs of numbers - latitude and longitude - usually measured in degrees (°).

Common global CRS:

- WGS 84 (EPSG:4326) - used by GPS and most web maps.
- NAD 83 - used in many U.S. datasets.
- UTM (Universal Transverse Mercator) - divides the Earth into zones for more precise local mapping.

Shapefiles and GIS Software

Information on spatial data and coordinates is stored in special file formats used by **Geographic Information Systems (GIS)** software, such as ArcGIS, QGIS, and R.

The most common format is the shapefile, which actually consists of several component files sharing the same name but different extensions:

File Type	Extension	Description
Main file	.shp	Contains the geometry (points, lines, polygons)
Attribute table	.dbf	Stores non-spatial attributes (e.g., population)
Index file	.shx	Indexes the geometry for faster access
Projection file	.prj	Stores coordinate reference system (CRS) info

Preparing for Spatial Data with Census

From the shapefile, we will extract the geographic coordinates (latitude and longitude) and the polygon identifiers associated with each spatial unit, tract. This step allows us to analyze the spatial structure of the data and link geometry information to their corresponding data attributes.

```r
# Get polygon centroids
# The centroid is a single point that represents the center of that polygon.
centroids_ga <- st_centroid(ga_tract)

# Extract centroid coordinates
coord_ga <- st_coordinates(centroids_ga)
head(coord_ga)
```

```
##              X        Y
## [1,] -82.84734 30.70399
## [2,] -82.88789 31.87441
## [3,] -83.29014 30.85291
## [4,] -83.54781 31.45300
## [5,] -84.16708 31.56617
## [6,] -84.26554 31.56828
```

```r
# Extracting polygons id
id_ga_tract <- ga_tract$GEOID
```

Distance and Proximity

In spatial analysis, understanding how locations relate to one another is essential. Distance and proximity describe the degree of closeness or connectedness between spatial units (points, lines, or polygons).

Spatial relationships can be defined using topological (contiguity-based) or distance-based concepts of neighborhood.

Topological or Contiguity-Based Neighbors

Two spatial units are considered neighbors if they share a common boundary (for polygons) or touch each other in space.

This type of relationship is discrete - either two areas are neighbors or they are not.

Common types of contiguity:

- Rook contiguity: two polygons share a common edge (like rook moves in chess).
- Bishop contiguity: two polygons share a common vertex (corner point).
- Queen contiguity: two polygons share either a boundary or a vertex (like queen moves in chess).

Example: Two census tracts in Atlanta are neighbors if they share a boundary line or corner. (Canche 2020)

Distance-Based Neighbors

Two spatial units are considered neighbors if they lie within a specified distance threshold from one another.

This relationship is continuous - the smaller the distance, the stronger the relationship.

Common approaches:

- Critical cut-off neighbors: A threshold distance (e.g., 10 km) is chosen so that each location has at least one neighbor.

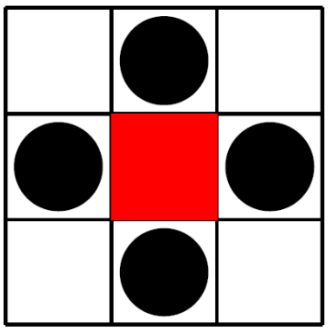


Figure 1: Rook's

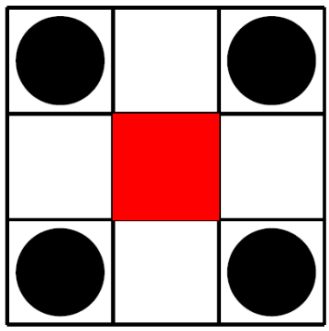


Figure 2: Bishop's

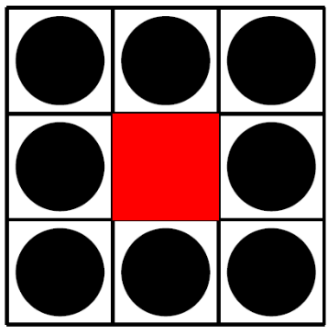


Figure 3: Queen's

Figure 1: Three Types of Contiguity

  - If the distance is too small, some locations may become "islands" with no neighbors.
- k-nearest neighbors (k-NN): Each observation has exactly k neighbors, regardless of physical distance.
  - For example, setting k = 4 means each point is connected to its four closest neighbors.

Example: In a spatial network of lithium mines, each mine can be connected to its five nearest facilities.

Why Neighborhood Definitions Matter

The way we define neighborhood structures affects:

- Spatial weights matrices (W): used in spatial regression and autocorrelation analysis (e.g., Moran's I, spatial lag models).
- Clustering outcomes: identifying hot spots or diffusion patterns.
- Interpretation of spatial dependence: whether proximity is defined by shared borders or by distance thresholds.

Binary Adjacency Matrix

A binary adjacency matrix is a mathematical way to represent which spatial units are neighbors in our dataset. It is a square matrix (N × N) where N is the number of spatial units (e.g., states, counties, tracts).

Each row and column represents a spatial unit. Each cell in the matrix (W) takes one of two values:

$$W_{ij} = 1 \quad \text{for i and j neighbors}$$

The two spatial units are neighbors (share a border or are within a threshold distance).

$$W_{ij} = 0 \quad \text{otherwise}$$

The two spatial units are not neighbors. The diagonal values (self-neighbors) are usually 0 because a location is not considered its own neighbor.

	ALABAMA	GEORGIA	QUORGNA	MISISSIPPI	SOUTH C
ALABAMA	1	1	1	1	0
FLORIDA	0	0	1	0	1
GEORGIA	1	1	0	1	0
LOUISIANA	0	1	0	1	0
MISSISSIPPI	1	0	1	0	1
SOUTH CAROLINA	0	1	0	1	0

Figure 2: Binary Adjacency Matrix with Queen Contiguity

Weighted Adjacency Matrix

A weighted adjacency matrix extends the idea of a binary adjacency matrix by not only indicating who is connected to whom, but also how strong or important each connection is.

Instead of using simple 1's and 0's, each cell in the matrix contains a weight that represents the intensity, strength, or relative influence between two spatial units.

Weights can be based on:

- Distance (closer neighbors get higher weights)
- Shared boundary length
- Population interaction or trade intensity
- Row-standardization (common in spatial econometrics)

For example, a square matrix (N × N) where N is the number of spatial units:

$$W_{ij} = 1/n_i \quad \text{i and j neighbors, where } n_i \text{ is the number of i's neighbors.}$$

$$W_{ij} = 0 \quad \text{otherwise}$$

Additional Adjacency Matrix

- k-nearest-neighbors weights matrix:

$$W_{ij} = 1 \quad \text{if the geographic center of region j is the one of the nearest k to the center of region i}$$

$$W_{ij} = 0 \quad \text{otherwise}$$

- Contiguity weights matrix:

$$W_{ij} = 1 \quad \text{if regions i and j have a common boundary}$$

$$W_{ij} = 0 \quad \text{otherwise}$$

- Distance-based binary weights matrix:

$$W_{ij} = 1 \quad \text{if the distance between regions i and j is less than a threshold cut-off distance}$$

$$W_{ij} = 0 \quad \text{otherwise}$$

Making Adjacency Matrix in R

We will make two binary distance matrices of tracts in Georgia with 10km and 50km. The code requires a complete matrix of numeric coordinates (no NAs). We will clean our coordinate data first.

```r
sum(is.na(coord_ga))
```

```
## [1] 10
```

```r
# If you find any NAs, drop them
coord_ga_clean <- coord_ga[complete.cases(coord_ga), ]
id_ga_tract_clean <- id_ga_tract[complete.cases(coord_ga)]
```

```r
# Two points are connected if they are within 10 km
dnb10 <- dnearneigh(x = coord_ga_clean, d1 = 0, d2 = 10,
                    row.names = id_ga_tract_clean, longlat = TRUE)
class(dnb10) # nb = neighbor list
```

```
## [1] "nb"
```

```r
# Two points are connected if they are within 50 km
dnb50 <- dnearneigh(x = coord_ga_clean, d1 = 0, d2 = 50,
                    row.names = id_ga_tract_clean, longlat = TRUE)
class(dnb50)
```

```
## [1] "nb"
```

We can also make a weighted matrix where every point has at least one connection.

```r
# k-nearest-neighbors weights matrix:
neighbors <- knearneigh(x = coord_ga_clean, k = 1, longlat = T)

  # point's id
  neighbors$nn
  # number of points
  neighbors$np
  # value of k (nearest neighbors)
  neighbors$k
  # coordinates (coord_ga_clean)
  neighbors$x

# Transform the knn object in a nb object:
# A list of integer vectors containing neighbor region number ids (neighbors$nn)
k1 <- knn2nb(neighbors)

# Compute link distances
k1.dist <-  nbdists(k1, coord_ga_clean, longlat = T)
# Unlist the object k1.dist
k1.dist <- unlist(k1.dist); k1.dist

# Find max link distance:
# i.e. the distance at which each point has at least one neighbor
all.linkedT <- max(k1.dist); all.linkedT

# Create an adjacency matrix where two points are connected if
# the distance between their centroids is < all.linkedT
```

```
dnb68 <- dnearneigh(x = coord_ga_clean, d1 = 0, d2 = all.linkedT,
                    row.names = id_ga_tract_clean, longlat=TRUE)
dnb68
```

Let's compare different adjacency matrices.

```
par(mfrow=c(1,2))
plot(dnb68, coord_ga_clean)
title("k-nearest-neighbors")
plot(dnb50, coord_ga_clean)
title("Distance-based matrix: 50 km")
```

Spatial Autocorrelation Statistics

Once a neighborhood structure (such as a binary or weighted adjacency matrix) has been defined, we can test whether values observed in nearby locations are spatially dependent - that is, whether similar or dissimilar values tend to occur close together in space.

Spatial autocorrelation measures the degree to which a variable is correlated with itself across space. It indicates whether the spatial distribution of values is random, clustered, or dispersed.

- Positive spatial autocorrelation: similar values cluster together in space (e.g., wealthy neighborhoods near wealthy neighborhoods).
- Negative spatial autocorrelation: dissimilar values are neighbors (e.g., high-income areas next to low-income areas).
- No spatial autocorrelation: spatial arrangement is random - location does not explain variation in the variable.

Spatial Lag Operator

The adjacency matrix can be used to compute the average value of neighboring observations, serving as a spatial lag operator that summarizes the influence of surrounding units on each observation.

$$Wy = \sum_j W_{ij} y_j = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} y_2 \\ \frac{1}{2}y_1 + \frac{1}{2}y_3 \\ \frac{1}{2}y_2 + \frac{1}{2}y_4 \\ y_3 \end{bmatrix}$$

Spatial Tests

The adjacency (or spatial weights) matrix provides the structure needed to formally test spatial dependence.

We test the null hypothesis that spatial location does not matter:

Null Hypothesis (H_0): There is no spatial autocorrelation.

- Values at one location are independent of values at neighboring locations.
- The observed pattern could be produced by random spatial arrangement (spatial randomness).
- If locations were randomly permuted, the overall spatial pattern would remain unchanged.

Alternative Hypotheses (H_1):

- Positive spatial autocorrelation: nearby observations are similar (spatial clustering).
- Negative spatial autocorrelation: nearby observations are dissimilar (spatial dispersion).

Testing Levels (Granularity)

- Global level: evaluates whether spatial autocorrelation exists across the entire study area.
  - Moran's (1950) I
  - Geary's (1954) c
  - Getis and Ord's (1992) G
- Local level: identifies clusters or outliers at specific locations.
  - Local Moran's I (Anselin,1995)
  - Local Geary's c (Anselin,1995)
  - Local G* (Getis and Ord, 1995)

Testing Global Moran's I Spatial Autocorrelation Statistic

Global Moran's (1950) I spatial autocorrelation statistic can be defined:

$$I = (\frac{N}{\sum_i \sum_j W_{ij}})(\frac{\sum_i \sum_j W_{ij}(x_i - \overline{x})(x_j - \overline{x})}{\sum_i (x_i - \overline{x})^2})$$

Note that the cross-product of the values at locations i and j is weighted by their spatial proximity - that is, by the spatial weights matrix (W). This weighting reflects how strongly each pair of locations is connected in space, similar to the spatial lag operator, where neighboring observations exert influence on each other.

The Moran's I statistic measures the degree of spatial autocorrelation, that is, the extent to which similar (or dissimilar) values cluster together in space.

- Maximum positive spatial autocorrelation: I - +1

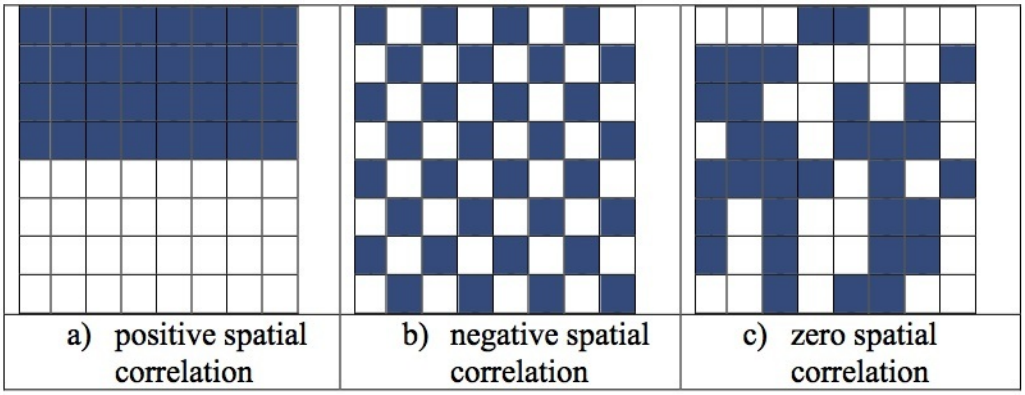


Figure 3: Spatial Autocorrelation

  - High values tend to be located near other high values, and low values near other low values.
  - Indicates strong spatial clustering or concentration of similar values.

- Maximum negative spatial autocorrelation: I - -1
  - High and low values are located near each other (a "checkerboard" pattern).
  - Indicates strong spatial dispersion or alternation of values.

- No spatial autocorrelation (spatial randomness): E[I] = -1 / (n - 1)
  - Values at each location are spatially independent; their arrangement is random.
  - The observed spatial pattern is no more clustered or dispersed than expected by chance.

```
# no. of row should be the same
ga_clean <- ga_tract[!st_is_empty(ga_tract), ]
coords   <- st_coordinates(st_centroid(ga_clean))  # one row per feature
id       <- ga_clean$GEOID
y        <- ga_clean$incomeE

dnb50 <- dnearneigh(x = coords, d1 = 0, d2 = 50,
                    row.names = id, longlat = TRUE)
class(dnb50)
```

[1] "nb"

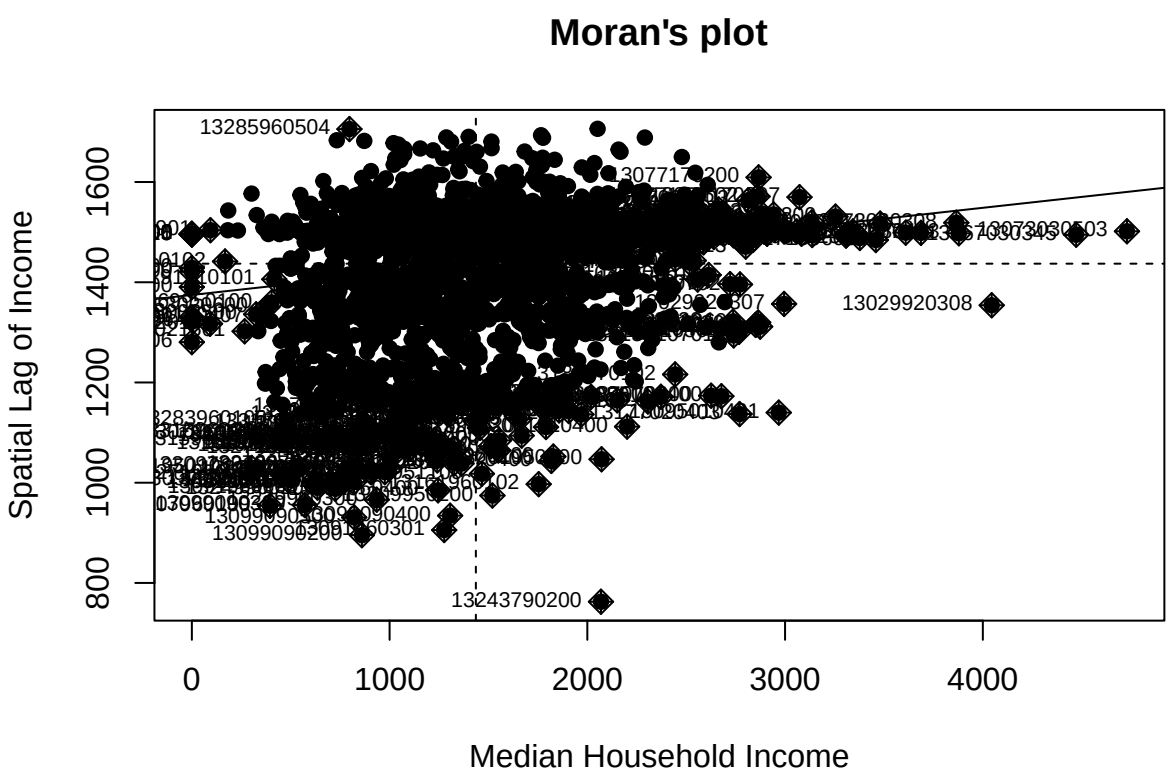
```
# Transform an nb object in a listw object
# (e.g. assign weights to the adjacency matrix)
dnb50.listw <- nb2listw(dnb50, style = "W") # Row standardized matrix
dnb50.listw
```

```
## Characteristics of weights list object:
## Neighbour list object:
## Number of regions: 2789
## Number of nonzero links: 1523882
## Percentage nonzero weights: 19.5909
## Average number of links: 546.3901
##
## Weights style: W
## Weights constants summary:
##      n      nn   S0       S1       S2
## W 2789 7778521 2789 41.55201 11247.75
```

```r
# Moran test (Moran I significantly close to 1, positive spatial autocorrelation)
# Null hypothesis: There is no spatial autocorrelation in the variable of interest
moran.test(x = y, listw = dnb50.listw, randomisation = F,
           zero.policy = F, alternative= "greater", na.action = na.fail)
```

```
##
##  Moran I test under normality
##
## data:  y
## weights: dnb50.listw
##
## Moran I statistic standard deviate = 19.447, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##      4.347511e-02     -3.586801e-04      5.080452e-06
```

```r
# Moran scatterplot
msp<- moran.plot(x = y, listw = dnb50.listw,
                 labels = id_ga_tract_clean, pch = 19,
                 main = "Moran's plot",
                 xlab = "Median Household Income",
                 ylab = "Spatial Lag of Income",
                 zero.policy = F, quiet = F)
```




```
## Potentially influential observations of
##   lm(formula = wx ~ x) :
##
```

```
##             dfb.1_ dfb.x dffit   cov.r   cook.d hat
## 13271950500 -0.07   0.06 -0.08    1.00_*  0.00   0.00
## 13277960700  0.03  -0.05 -0.06    1.00_*  0.00   0.00
## 13095010402 -0.03   0.01 -0.04    1.00_*  0.00   0.00
## 13215010401  0.05  -0.06 -0.07    1.00    0.00   0.00_*
## 13215010204  0.05  -0.06 -0.07    1.00    0.00   0.00_*
## 13121980000  0.05  -0.05  0.05    1.00_*  0.00   0.00_*
## 13051010702  0.03  -0.04 -0.04    1.00_*  0.00   0.00_*
## 13067030912  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13185010601  0.05  -0.07 -0.07    1.00    0.00   0.00_*
## 13309780200 -0.06   0.04 -0.07    0.99_*  0.00   0.00
## 13135050315  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13315960100 -0.10   0.08 -0.10_*  0.99_*  0.00   0.00
## 13215002500 -0.02   0.02 -0.02    1.00_*  0.00   0.00
## 13121006801  0.05  -0.05  0.05    1.00_*  0.00   0.00_*
## 13091960301 -0.05   0.02 -0.08_*  0.99_*  0.00   0.00
## 13127000407 -0.01  -0.01 -0.04    1.00_*  0.00   0.00
## 13063040632  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13029980000  0.02  -0.02  0.02    1.00_*  0.00   0.00_*
## 13141480400  0.00  -0.02 -0.05    1.00_*  0.00   0.00
## 13121010527  0.00   0.00  0.01    1.00_*  0.00   0.00_*
## 13121001901  0.05  -0.05  0.05    1.00_*  0.00   0.00_*
## 13039010405  0.05  -0.07 -0.07    1.00    0.00   0.00_*
## 13001950302 -0.07   0.05 -0.07    1.00_*  0.00   0.00
## 13089023115  0.05  -0.05  0.05    1.00_*  0.00   0.00_*
## 13095000501  0.05  -0.08 -0.09_*  1.00_*  0.00   0.00
## 13121001102  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13121001206  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13279970202 -0.04   0.03 -0.06    1.00_*  0.00   0.00
## 13095000400  0.08  -0.11 -0.12_*  1.00    0.01   0.00
## 13089980000  0.05  -0.05  0.05    1.00_*  0.00   0.00_*
## 13175950400 -0.01  -0.01 -0.06    0.99_*  0.00   0.00
## 13121009607  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13177020403  0.10  -0.13 -0.14_*  1.00_*  0.01   0.00_*
## 13067030356  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13063040519  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13201950200 -0.08   0.06 -0.08    1.00_*  0.00   0.00
## 13205090100 -0.01  -0.01 -0.04    1.00_*  0.00   0.00
## 13069010600  0.04  -0.06 -0.08    1.00_*  0.00   0.00
## 13111050200 -0.01   0.01  0.01    1.00_*  0.00   0.00_*
## 13059150800  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13099090400 -0.04   0.02 -0.08    0.99_*  0.00   0.00
## 13099090300 -0.12   0.10 -0.12_*  0.99_*  0.01   0.00
## 13001950500 -0.02   0.00 -0.05    1.00_*  0.00   0.00
## 13009970400  0.00  -0.02 -0.05    1.00_*  0.00   0.00
## 13013180501  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13077170200 -0.03   0.04  0.05    1.00_*  0.00   0.00_*
## 13209950300 -0.09   0.07 -0.10_*  0.99_*  0.00   0.00
## 13175950700 -0.04   0.03 -0.06    1.00_*  0.00   0.00
## 13063040609  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13309780100 -0.10   0.09 -0.11_*  1.00_*  0.01   0.00
## 13113140404  0.00  -0.01 -0.01    1.00_*  0.00   0.00_*
## 13057091007 -0.01   0.01  0.01    1.00_*  0.00   0.00_*
## 13073030503  0.06  -0.07 -0.07    1.01_*  0.00   0.01_*
```

```
## 13121011419  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13069010801  0.01  -0.02 -0.05    1.00_*  0.00   0.00
## 13077170304 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13077170502 -0.02   0.03  0.03    1.00_*  0.00   0.00_*
## 13031110407 -0.02   0.02 -0.02    1.00_*  0.00   0.00
## 13121010212  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13097080304 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13175950201  0.01  -0.03 -0.06    1.00_*  0.00   0.00
## 13191980000  0.01  -0.01  0.01    1.00_*  0.00   0.00_*
## 13297110400  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13315960300 -0.09   0.08 -0.10_*  1.00_*  0.00   0.00
## 13315960400 -0.06   0.04 -0.06    1.00_*  0.00   0.00
## 13223120301 -0.01   0.01  0.01    1.00_*  0.00   0.00_*
## 13047030100 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13117130301 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13155950201 -0.02   0.01 -0.04    1.00_*  0.00   0.00
## 13161960101 -0.07   0.06 -0.08    1.00_*  0.00   0.00
## 13161960201 -0.03   0.01 -0.06    0.99_*  0.00   0.00
## 13183970102  0.05  -0.07 -0.08_*  1.00    0.00   0.00
## 13219030104 -0.01   0.01  0.01    1.00_*  0.00   0.00_*
## 13091960302 -0.05   0.03 -0.06    1.00_*  0.00   0.00
## 13223120108 -0.01   0.01  0.01    1.00_*  0.00   0.00_*
## 13057090807 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13175951002 -0.03   0.01 -0.06    0.99_*  0.00   0.00
## 13151070313  0.01  -0.01 -0.01    1.00_*  0.00   0.00_*
## 13051010704  0.03  -0.03 -0.04    1.00_*  0.00   0.00_*
## 13051010703  0.02  -0.02 -0.03    1.00_*  0.00   0.00
## 13067030248  0.00  -0.01 -0.01    1.00_*  0.00   0.00_*
## 13089021222  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13095011000 -0.03   0.01 -0.04    1.00_*  0.00   0.00
## 13095010401  0.12  -0.15 -0.16_*  1.00_*  0.01   0.00_*
## 13095010602 -0.01   0.00 -0.04    1.00_*  0.00   0.00
## 13201950100 -0.08   0.07 -0.09_*  0.99_*  0.00   0.00
## 13201950300 -0.07   0.05 -0.08    0.99_*  0.00   0.00
## 13253200100 -0.04   0.02 -0.06    1.00_*  0.00   0.00
## 13017960300  0.00  -0.02 -0.05    1.00_*  0.00   0.00
## 13017960400 -0.01  -0.01 -0.06    1.00_*  0.00   0.00
## 13069010802  0.02  -0.04 -0.06    1.00_*  0.00   0.00
## 13069010100 -0.03   0.02 -0.06    1.00_*  0.00   0.00
## 13271950200 -0.11   0.10 -0.12_*  1.00_*  0.01   0.00
## 13087970600  0.08  -0.10 -0.11_*  1.00    0.01   0.00
## 13247060305  0.01  -0.01 -0.01    1.00_*  0.00   0.00
## 13089021502  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13223120603 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13099090200 -0.10   0.08 -0.11_*  0.99_*  0.01   0.00
## 13099090100 -0.12   0.10 -0.12_*  0.99_*  0.01   0.00
## 13091960500 -0.10   0.09 -0.11_*  0.99_*  0.01   0.00
## 13091960400 -0.05   0.02 -0.07    0.99_*  0.00   0.00
## 13175950900 -0.05   0.03 -0.06    1.00_*  0.00   0.00
## 13175951100 -0.02   0.00 -0.06    0.99_*  0.00   0.00
## 13215010701  0.05  -0.07 -0.08    1.00    0.00   0.00_*
## 13209950200 -0.02  -0.01 -0.07    0.99_*  0.00   0.00
## 13209950100 -0.07   0.06 -0.08    1.00_*  0.00   0.00
## 13175951400 -0.07   0.05 -0.08    0.99_*  0.00   0.00
```

```
## 13283960100 -0.08   0.07 -0.08_*  1.00    0.00   0.00
## 13283960200  0.02  -0.04 -0.08    0.99_*  0.00   0.00
## 13279970400  0.02  -0.04 -0.08    0.99_*  0.00   0.00
## 13279970600 -0.05   0.03 -0.06    1.00_*  0.00   0.00
## 13161960300 -0.08   0.07 -0.08_*  1.00_*  0.00   0.00
## 13103030301  0.02  -0.02 -0.03    1.00_*  0.00   0.00
## 13081010400  0.05  -0.07 -0.09_*  1.00_*  0.00   0.00
## 13243790200  0.07  -0.12 -0.16_*  0.98_*  0.01   0.00
## 13259950400 -0.07   0.05 -0.07    1.00_*  0.00   0.00
## 13239960300 -0.08   0.06 -0.09_*  0.99_*  0.00   0.00
## 13063040410  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13009970600 -0.02   0.00 -0.04    1.00_*  0.00   0.00
## 13121000100  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13249960100 -0.07   0.06 -0.08_*  0.99_*  0.00   0.00
## 13261950300  0.04  -0.07 -0.10_*  0.99_*  0.00   0.00
## 13067030602  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13067030413  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13107970300  0.00  -0.01 -0.05    1.00_*  0.00   0.00
## 13067030505  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13077170501 -0.01   0.01  0.02    1.00_*  0.00   0.00
## 13113140207  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13117130607  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13117130302  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13117130307 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13153020600 -0.01   0.01 -0.01    1.00_*  0.00   0.00
## 13077170503 -0.01   0.01  0.02    1.00_*  0.00   0.00_*
## 13133950301 -0.01   0.00 -0.04    1.00_*  0.00   0.00
## 13017960502  0.00  -0.02 -0.06    1.00_*  0.00   0.00
## 13151070114  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13307960200 -0.09   0.08 -0.09_*  1.00    0.00   0.00
## 13135050542  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13217100600  0.01  -0.01 -0.01    1.00_*  0.00   0.00
## 13179010102  0.02  -0.02  0.02    1.00_*  0.00   0.00
## 13183980000 -0.02   0.02 -0.02    1.00_*  0.00   0.00_*
## 13067031314 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13313001200  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13305970202  0.02  -0.03 -0.05    1.00_*  0.00   0.00
## 13153021501 -0.03   0.03 -0.03    1.00_*  0.00   0.00
## 13153021115  0.06  -0.07 -0.08    1.00    0.00   0.00_*
## 13151070119  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13063040420  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13001950301 -0.04   0.03 -0.06    1.00_*  0.00   0.00
## 13279970301 -0.05   0.03 -0.06    1.00_*  0.00   0.00
## 13279970102 -0.07   0.05 -0.07    1.00_*  0.00   0.00
## 13271950102 -0.12   0.11 -0.12_*  1.00_*  0.01   0.00
## 13113140210  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13029920308  0.12  -0.14 -0.14_*  1.01_*  0.01   0.01_*
## 13071970902 -0.03   0.01 -0.05    1.00_*  0.00   0.00
## 13107970602 -0.04   0.03 -0.06    1.00_*  0.00   0.00
## 13095001100  0.05  -0.07 -0.09_*  1.00_*  0.00   0.00
## 13191110101  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13223120111 -0.02   0.02  0.02    1.00_*  0.00   0.00
## 13161960202 -0.09   0.07 -0.09_*  0.99_*  0.00   0.00
## 13135050223  0.01  -0.02 -0.02    1.01_*  0.00   0.00_*
```

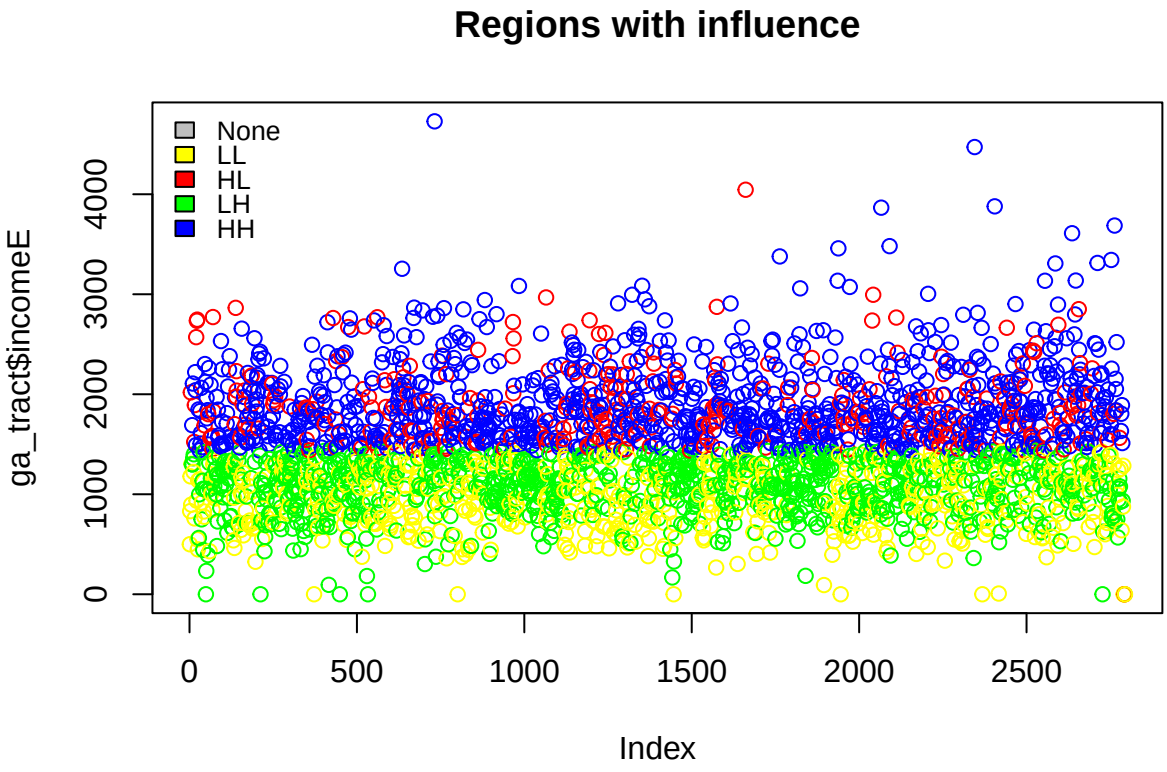
```
## 13069010500  0.02  -0.04 -0.06    1.00_*  0.00   0.00
## 13121001702  0.00   0.00  0.01    1.00_*  0.00   0.00
## 13121011643  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13121000501  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13121010129  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13067030603  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13135050631  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13053020201 -0.02   0.02 -0.02    1.00_*  0.00   0.00_*
## 13307960100 -0.14   0.12 -0.14_*  0.99_*  0.01   0.00
## 13271950103 -0.10   0.09 -0.11_*  0.99_*  0.01   0.00
## 13139001501  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13117130410  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13117130603  0.02  -0.02 -0.02    1.01_*  0.00   0.00_*
## 13009970302 -0.02   0.00 -0.04    1.00_*  0.00   0.00
## 13249960200 -0.10   0.08 -0.10_*  0.99_*  0.01   0.00
## 13051980000  0.02  -0.02  0.02    1.00_*  0.00   0.00_*
## 13237960101  0.01  -0.03 -0.05    1.00_*  0.00   0.00
## 13237960104 -0.02   0.00 -0.05    1.00_*  0.00   0.00
## 13077170307 -0.02   0.03  0.03    1.00_*  0.00   0.00_*
## 13279970201 -0.05   0.03 -0.06    1.00_*  0.00   0.00
## 13259950100 -0.06   0.05 -0.07    1.00_*  0.00   0.00
## 13279970500 -0.04   0.02 -0.06    0.99_*  0.00   0.00
## 13153021204  0.05  -0.06 -0.07    1.00    0.00   0.00_*
## 13029920307  0.05  -0.06 -0.07    1.00_*  0.00   0.00_*
## 13279970302 -0.08   0.06 -0.08_*  0.99_*  0.00   0.00
## 13285960504  0.06  -0.05  0.07    1.00_*  0.00   0.00
## 13073030308  0.01  -0.02 -0.02    1.01_*  0.00   0.01_*
## 13057090906  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13175950800 -0.04   0.02 -0.06    0.99_*  0.00   0.00
## 13073030304  0.00   0.00 -0.01    1.01_*  0.00   0.01_*
## 13081010100 -0.02   0.00 -0.05    1.00_*  0.00   0.00
## 13215010501  0.05  -0.07 -0.08    1.00    0.00   0.00_*
## 13087970400  0.01  -0.03 -0.06    1.00_*  0.00   0.00
## 13273120400 -0.04   0.03 -0.05    1.00_*  0.00   0.00
## 13059130400  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13269950100 -0.01   0.01 -0.01    1.00_*  0.00   0.00
## 13089021208  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13099090500 -0.10   0.08 -0.11_*  0.99_*  0.01   0.00
## 13261950200  0.02  -0.04 -0.06    1.00_*  0.00   0.00
## 13317010101 -0.07   0.06 -0.08    1.00_*  0.00   0.00
## 13175950202  0.03  -0.05 -0.07    1.00_*  0.00   0.00
## 13315960200 -0.09   0.08 -0.09_*  1.00    0.00   0.00
## 13067031004  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13067031101  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13091960200 -0.04   0.02 -0.06    0.99_*  0.00   0.00
## 13171970100 -0.02   0.02  0.02    1.00_*  0.00   0.00
## 13081010300 -0.03   0.01 -0.05    1.00_*  0.00   0.00
## 13255160500 -0.01   0.02  0.02    1.00_*  0.00   0.00
## 13303950500 -0.01  -0.01 -0.04    1.00_*  0.00   0.00
## 13135050571  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13253200200 -0.03   0.01 -0.06    0.99_*  0.00   0.00
## 13091960600 -0.11   0.09 -0.11_*  0.99_*  0.01   0.00
## 13175950100 -0.01  -0.01 -0.05    1.00_*  0.00   0.00
## 13067030405  0.00   0.00  0.00    1.00_*  0.00   0.00_*
```

```
## 13067030345  0.05  -0.06 -0.06    1.01_*  0.00   0.01_*
## 13057090303 -0.02   0.02  0.03    1.00_*  0.00   0.00_*
## 13001950202 -0.06   0.05 -0.07    1.00_*  0.00   0.00
## 13223120110 -0.01   0.02  0.02    1.00_*  0.00   0.00
## 13053020206 -0.04   0.04 -0.04    1.00_*  0.00   0.00_*
## 13113140309  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13073030403  0.03  -0.03 -0.03    1.01_*  0.00   0.01_*
## 13215010606 -0.02   0.02 -0.02    1.00_*  0.00   0.00_*
## 13077170408  0.00   0.00  0.01    1.00_*  0.00   0.00_*
## 13313001500  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13105000502  0.01  -0.03 -0.06    1.00_*  0.00   0.00
## 13017960100 -0.05   0.04 -0.06    1.00_*  0.00   0.00
## 13271950101 -0.07   0.06 -0.08_*  0.99_*  0.00   0.00
## 13067031511  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13057091001  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13073030505  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13073030306 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13161960102  0.01  -0.04 -0.08    0.99_*  0.00   0.00
## 13135050744  0.00   0.00  0.00    1.00_*  0.00   0.00
## 13135050617  0.01  -0.01 -0.01    1.00_*  0.00   0.00_*
## 13117130406 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13175951001 -0.04   0.02 -0.06    1.00_*  0.00   0.00
## 13077170405  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13037950200 -0.05   0.03 -0.06    1.00_*  0.00   0.00
## 13067031215  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13279970101 -0.04   0.03 -0.05    1.00_*  0.00   0.00
## 13067031114  0.02  -0.02 -0.02    1.01_*  0.00   0.01_*
## 13223120107 -0.01   0.01  0.01    1.00_*  0.00   0.00
## 13121010006  0.00   0.00  0.00    1.00_*  0.00   0.00_*
## 13121010133  0.01  -0.01 -0.01    1.00_*  0.00   0.00_*
## 13303950400  0.01  -0.03 -0.05    1.00_*  0.00   0.00
## 13185010204  0.06  -0.07 -0.08    1.00    0.00   0.00_*
## 13067031112  0.00   0.01  0.01    1.00_*  0.00   0.00
## 13087970200 -0.04   0.02 -0.05    1.00_*  0.00   0.00
## 13253200300 -0.04   0.03 -0.06    1.00_*  0.00   0.00
## 13175950500 -0.01  -0.01 -0.06    0.99_*  0.00   0.00
## 13217100301  0.01  -0.01 -0.01    1.00_*  0.00   0.00
## 13069010200 -0.04   0.02 -0.05    1.00_*  0.00   0.00
## 13111050400  0.01  -0.01 -0.01    1.00_*  0.00   0.00_*
## 13217100903  0.01  -0.01 -0.01    1.00_*  0.00   0.00
## 13063980000  0.05  -0.05  0.05    1.00_*  0.00   0.00_*
## 13121008905  0.01  -0.01 -0.01    1.01_*  0.00   0.00_*
## 13067031207  0.02  -0.02 -0.02    1.01_*  0.00   0.01_*
## 13089021224  0.00   0.00  0.01    1.00_*  0.00   0.00
```

```
#msp
```

```
# Select outliers
infl <- any(msp$is_inf)

# Identify low and high values
# e.g. lower or higher then the mean
lhx <- cut(y, breaks=c(min(y), mean(y), max(y)),
           labels=c("L", "H"), include.lowest=TRUE)
```

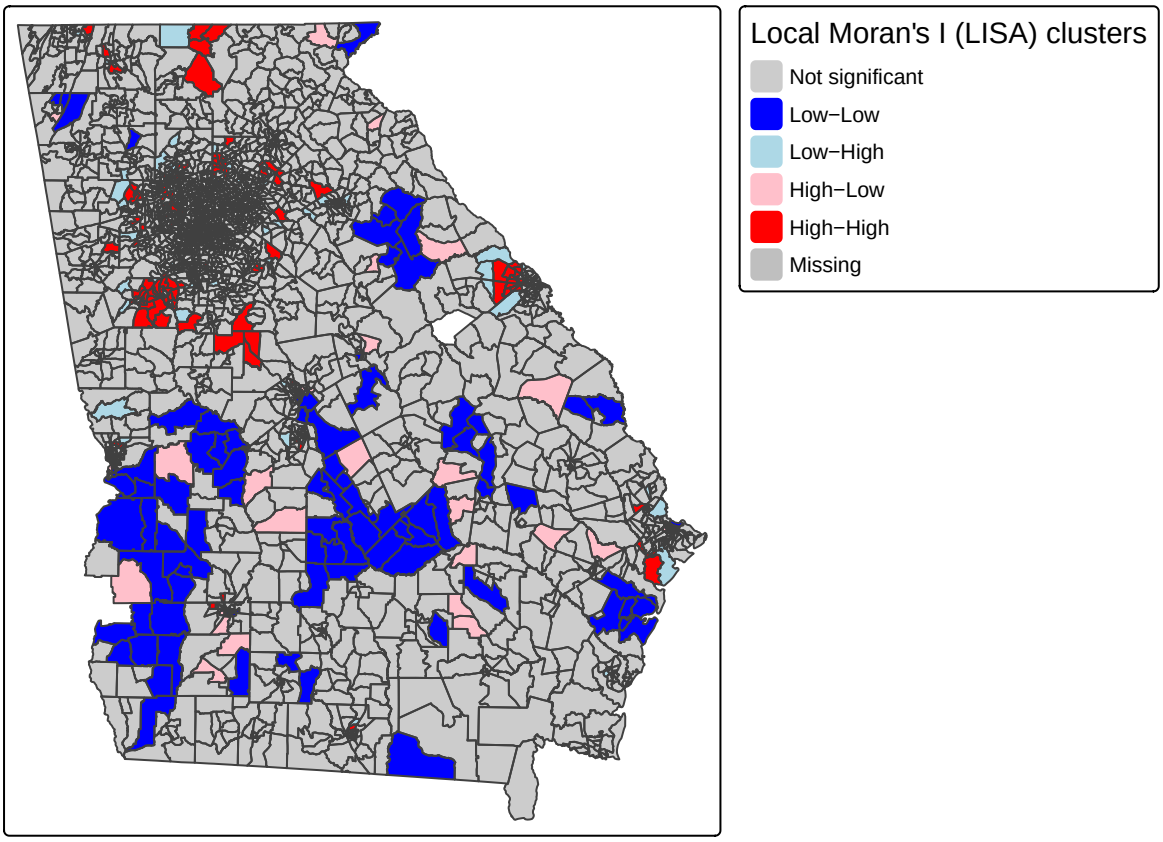
```r
# Find low and High values in the lagged independent variable
# Create lagged variable (Gy)
# # set zero.policy if you have isolates
wx <- spdep::lag.listw(dnb50.listw, y, zero.policy = TRUE)

# Now make L/H bins
lhx  <- cut(y, breaks = c(min(y, na.rm=TRUE), mean(y, na.rm=TRUE),
                          max(y, na.rm=TRUE)), labels = c("L","H"),
                          include.lowest = TRUE, right = TRUE)

lhwx <- cut(wx, breaks = c(min(wx, na.rm=TRUE), mean(wx, na.rm=TRUE),
                           max(wx, na.rm=TRUE)), labels = c("L","H"),
                           include.lowest = TRUE, right = TRUE)

# Find neighborhoods with similar values (H-H, L-L, etc)
# Note that outliers are considered a separate category
# (H-H-Outlier, H-H-Non_Outlier)
lhlh <- interaction(lhx, lhwx, infl, drop=TRUE)

# Transform lhlh in a numerical vector
cols <- rep(1, length(lhlh))
cols[lhlh == "L.L.TRUE"] <- 2
cols[lhlh == "H.L.TRUE"] <- 3
cols[lhlh == "L.H.TRUE"] <- 4
cols[lhlh == "H.H.TRUE"] <- 5

plot(ga_tract$incomeE, col=c("gray", "yellow", "red", "green", "blue")[cols],axes=T)
  legend("topleft", legend=c("None", "LL", "HL", "LH", "HH"),
         fill=c("gray", "yellow", "red", "green", "blue"), bty="n", cex=0.8,
         y.intersp=0.8)
  title("Regions with influence")
```



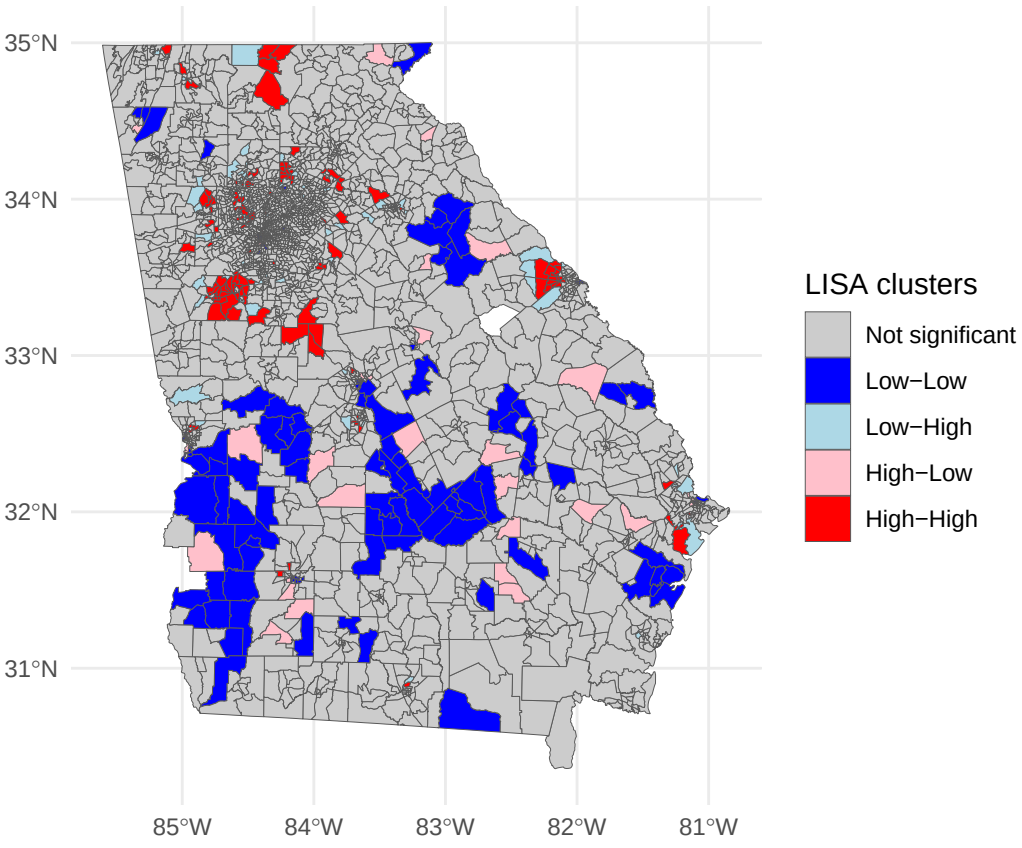
Testing Local Moran's I Spatial Autocorrelation Statistic

```r
# Choose contiguity type:
#    - queen = TRUE  : shares edge OR vertex (more neighbors)
#    - queen = FALSE : shares edge only (rook contiguity)
queen_contiguity <- TRUE

# Stable IDs for safety (use our tract id if available)
id_vec <- if ("GEOID" %in% names(ga_clean)) ga_clean$GEOID else as.character(seq_len(nrow(ga_clean)))

# Build contiguity neighbors
nb <- spdep::poly2nb(ga_clean, queen = queen_contiguity, row.names = id_vec)

# Convert to weights; allow isolates (zero neighbors) to avoid errors downstream
lw <- spdep::nb2listw(nb, style = "W", zero.policy = TRUE)

# Target vector
y   <- ga_clean$incomeE

# Local Moran (returns Ii, E.Ii, Var.Ii, Z.Ii, Pr(z > 0))
li <- spdep::localmoran(y, lw, zero.policy = TRUE)

# Spatial lag
wy <- spdep::lag.listw(lw, y, zero.policy = TRUE)

# Center variables to define quadrants
y_c  <- y  - mean(y,  na.rm = TRUE)
```

```r
wy_c <- wy - mean(wy, na.rm = TRUE)

alpha <- 0.05
sig   <- li[, "Pr(z != E(Ii))"] < alpha   # significance mask

cluster <- ifelse(!sig, "Not significant",
           ifelse(y_c > 0 & wy_c > 0, "High-High",
           ifelse(y_c < 0 & wy_c < 0, "Low-Low",
           ifelse(y_c > 0 & wy_c < 0, "High-Low",
           ifelse(y_c < 0 & wy_c > 0, "Low-High", "Not significant")))))

# Attach to sf for mapping
ga_clean <- ga_clean %>%
  mutate(
    Ii        = li[, "Ii"],
    Ii_pval   = li[, "Pr(z != E(Ii))"],
    lisa_cl   = factor(cluster, levels = c("Not significant","Low-Low",
                                           "Low-High","High-Low","High-High"))
  )

# Use tmap
tmap_mode("plot")
tm_shape(ga_clean) +
  tm_fill("lisa_cl",
          title = "Local Moran's I (LISA) clusters",
          palette = c("grey80", "blue", "lightblue", "pink", "red")) +
  tm_borders() +
  tm_layout(legend.outside = TRUE)
```




```
# Use ggplot2
ggplot(ga_clean) +
  geom_sf(aes(fill = lisa_cl), linewidth = 0.1) +
  scale_fill_manual(values = c("grey80","blue","lightblue","pink","red"),
                    name = "LISA clusters") +
  theme_minimal()
```



Model Specification

The Ordinary Least Square (OLS) linear model can be presented in both forms: (1) Vector (2) Matrix

$$y_i = \alpha + \sum_k \beta_k x_{ik} + \epsilon_i \quad \epsilon_i \sim i.i.d.(0, \sigma^2)$$

$$y = X\beta + \epsilon \quad E[\epsilon] = 0 \quad E[\epsilon\epsilon'] = \sigma^2 l_n$$

Consequences of the independence assumptions, there should be no spatial diffusion of idiosyncratic shocks and there should be no spatial indirect (interaction or spillover) effects. How to test if the independence assumption holds? Moran's I test of spatial autocorrelation in OLS residuals.

Given our practice from Lab 3, we will keep using four independent variables to explain variation in median household income per tract in Georgia by using multiple OLS linear regression. Substantive hypotheses:

- As the proportion of Black population increases, the median household income decreases.
- As the proportion of immigrant population increases, the median household income increases.
- As the proportion of bachelor's degree holders increases, the median household income increases.
- As the young male population increases, the median household income decreases.

```
# Clean the data
dat <- ga_clean |>
  sf::st_make_valid() |>
  dplyr::select(incomeE, blackP, immigrantP, educationP, youngmaleP, geometry) |>
```

```r
  tidyr::drop_na(incomeE, blackP, immigrantP, educationP, youngmaleP)

# Creates a neighbors list based on polygon contiguity
nb <- poly2nb(dat, queen = TRUE)
# Converts a neighbor list into a spatial weights object
lw <- nb2listw(nb, style = "W", zero.policy = TRUE)

# Running the OLS linear model with the clean data
model1 <- lm(incomeE ~ blackP + immigrantP + educationP + youngmaleP,
             data = dat, x = TRUE)
summary(model1)
```

```
##
## Call:
## lm(formula = incomeE ~ blackP + immigrantP + educationP + youngmaleP,
##     data = dat, x = TRUE)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1312.5  -412.4   -52.5   345.0  3310.8
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1313.421     29.469  44.570  < 2e-16 ***
## blackP         1.065     39.807   0.027 0.978653
## immigrantP   545.059    213.040   2.558 0.010566 *
## educationP   990.441    134.567   7.360 2.41e-13 ***
## youngmaleP  -774.481    225.540  -3.434 0.000604 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 552.4 on 2777 degrees of freedom
## Multiple R-squared:  0.0415, Adjusted R-squared:  0.04012
## F-statistic: 30.06 on 4 and 2777 DF,  p-value: < 2.2e-16
```

Let's test whether independence assumption holds based on Moran's I test on residuals.

```r
# Parametric test
# Relies on theoretical (normal) distribution assumptions for the test statistic
lm.morantest(model = model1, listw = lw)
```

```
##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = incomeE ~ blackP + immigrantP + educationP +
## youngmaleP, data = dat, x = TRUE)
## weights: lw
##
## Moran I statistic standard deviate = 16.74, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
```

```
## Observed Moran I      Expectation         Variance
##     0.1825428946    -0.0011363107     0.0001203943
```

```r
# Non-parametric test: Monte Carlo experiment
set.seed(123)
res <- residuals(model1)
mc  <- spdep::moran.mc(res, listw = lw, nsim = 999, zero.policy = TRUE)
mc
```

```
##
##  Monte-Carlo simulation of Moran I
##
## data:  res
## weights: lw
## number of simulations + 1: 1000
##
## statistic = 0.18254, observed rank = 1000, p-value = 0.001
## alternative hypothesis: greater
```

If Moran's I is higher than expected and statistically significant, it indicates the presence of spatial autocorrelation in the residuals. In this case, the standard OLS model is misspecified, and we should consider fitting a spatial regression model - such as the Spatial Error Model (SEM), Spatial Lag Model (SAR), or the more general Spatial Durbin Model (SDM) - to properly account for spatial dependence.

Spaital Error Model (SEM)

This estimates a Spatial Error Model (SEM), sometimes called the Spatial Autocorrelation Model, where spatial dependence is in the error term, not in the dependent variable. In other words, it is used when there is no specific theoretical expectation about how effects spill over across spatial units. For example, one might assume that the median household income level in a given tract is influenced by the characteristics of neighboring tracts, without specifying a particular direction or pattern of influence.

Structural form:

$$y_i = \alpha + \sum_k \beta_k x_{ik} + \epsilon_i \quad \epsilon_i = \delta \sum_j^n W_{ij} \epsilon j + \upsilon_i$$

$$|\delta| < 1 = \text{Spatial Autoregressive Parameter}$$

Reduced form:

$$y = X\beta + \beta_n \upsilon \quad \beta_n = (I_n - \delta W_n)^{-1}$$

```r
SEM1 <- errorsarlm(formula = formula(model1),listw = lw,
                   method = "eigen", data = dat)
summary(SEM1)
```

```
##
## Call:errorsarlm(formula = formula(model1), data = dat, listw = lw,
##     method = "eigen")
##
```

```
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1268.518  -386.964   -40.195   336.639  3118.751
##
## Type: error
## Coefficients: (asymptotic standard errors)
##             Estimate Std. Error z value  Pr(>|z|)
## (Intercept) 1258.177     38.028 33.0853 < 2.2e-16
## blackP       139.003     51.847  2.6810 0.0073401
## immigrantP   612.732    250.410  2.4469 0.0144086
## educationP   941.413    162.392  5.7972 6.744e-09
## youngmaleP  -826.778    225.391 -3.6682 0.0002443
##
## Lambda: 0.39935, LR test value: 214.85, p-value: < 2.22e-16
## Asymptotic standard error: 0.026098
##     z-value: 15.302, p-value: < 2.22e-16
## Wald statistic: 234.14, p-value: < 2.22e-16
##
## Log likelihood: -21403.85 for error model
## ML residual variance (sigma squared): 273870, (sigma: 523.33)
## Number of observations: 2782
## Number of parameters estimated: 7
## AIC: 42822, (AIC for lm: 43035)
```

Let's interpret the results. Did you find any difference from OLS model?

We should focus on Lambda:

- It measures the strength of spatial autocorrelation in the model's error term.
- A lambda (λ) of 0.399 means that about 40% of the residual variation in one area is systematically related to the residuals in neighboring areas (as defined by our spatial weights matrix).
- Since it is positive and highly significant ($p < 0.001$), you can conclude that unmodeled factors are spatially correlated - meaning there is still some spatial structure in the error term that OLS would have ignored.

Spatial Durbin Model (SDM)

The Spatial Durbin Model (SDM) extends traditional regression models by allowing spatial spillovers in both the dependent and independent variables. In other words, the outcome in a given unit (i) is influenced not only by the characteristics of that unit itself but also by the characteristics of neighboring units - as defined by the spatial weights matrix (W).

Structural form:

$$y = \alpha + X\beta + WX\gamma + \epsilon \quad \epsilon_i \sim i.i.d.(0, \sigma^2)$$

γ is the spatial autoregressive parameter.

```
SDM1 <- lagsarlm(formula = formula(model1), listw = lw, type = "Durbin",
                 method = "eigen", data = dat)
summary(SDM1)
```

```
##
```

```
## Call:
## lagsarlm(formula = formula(model1), data = dat, listw = lw, type = "Durbin",
##     method = "eigen")
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1246.250  -383.045   -41.924   322.563  3132.620
##
## Type: mixed
## Coefficients: (asymptotic standard errors)
##                Estimate Std. Error z value  Pr(>|z|)
## (Intercept)     823.550     49.888 16.5080 < 2.2e-16
## blackP          463.128     73.793  6.2760 3.473e-10
## immigrantP      773.436    293.620  2.6341 0.0084351
## educationP      938.486    201.789  4.6508 3.306e-06
## youngmaleP     -767.702    226.668 -3.3869 0.0007069
## lag.blackP     -562.865     86.020 -6.5434 6.014e-11
## lag.immigrantP -593.969    372.608 -1.5941 0.1109165
## lag.educationP -310.063    252.158 -1.2296 0.2188338
## lag.youngmaleP  292.414    415.220  0.7042 0.4812839
##
## Rho: 0.39091, LR test value: 215.14, p-value: < 2.22e-16
## Asymptotic standard error: 0.026221
##     z-value: 14.908, p-value: < 2.22e-16
## Wald statistic: 222.25, p-value: < 2.22e-16
##
## Log likelihood: -21384.22 for mixed model
## ML residual variance (sigma squared): 270390, (sigma: 519.99)
## Number of observations: 2782
## Number of parameters estimated: 11
## AIC: 42790, (AIC for lm: 43004)
## LM test for residual autocorrelation
## test value: 32.628, p-value: 1.116e-08
```

ρ is 0.391 ($p < 0.001$), which indicates strong positive spatial dependence in the outcome. After controlling for our covariates (both local and neighbors'), income in a tract tends to move with income in neighboring tracts. If neighbors' incomes rise, the focal tract's income is pulled up via spatial feedback loops, and vice versa.

Among neighbor (WX) spillover coefficients, spatially lagged proportion of Black population is statistically significant ($p < 0.001$) which coefficient is -562.9. This indicates that higher neighboring Black share is associated with lower focal income, holding focal characteristics constant.

Spatial Autoregressive Model (SAR)

Use Spatial Autoregressive Model (SAR) when the outcome in each unit is influenced by neighbors' outcomes (endogenous spatial dependence). Think diffusion/contagion/peer effects: outcomes spill over across space.

Structural form:

$$y = \alpha + X\beta + Wy\rho + \epsilon \quad \epsilon_i \sim i.i.d.(0, \sigma^2)$$

ρ is the spatial autoregressive parameter.

Reduced form:

$$y = A_n\alpha + A_nX\beta + A_n\epsilon \quad A_n = (I_n - \rho W_n)^{-1}$$

The matrix A_n, sometimes called the *spatial multiplier* or *spatial propagation matrix*, captures how the effect of a shock or change propagates through the spatial network.

```
SAR1 <- lagsarlm(formula = formula(model1), listw = lw, type = "lag",
                 method = "eigen", data = dat)
summary(SAR1)
```

```
##
## Call:lagsarlm(formula = formula(model1), data = dat, listw = lw, type = "lag",
##     method = "eigen")
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1274.562  -388.888   -35.518   333.909  3141.988
##
## Type: lag
## Coefficients: (asymptotic standard errors)
##             Estimate Std. Error z value  Pr(>|z|)
## (Intercept)  789.534     43.663 18.0826 < 2.2e-16
## blackP        46.221     37.819  1.2221 0.2216555
## immigrantP   353.621    202.822  1.7435 0.0812460
## educationP   688.186    129.419  5.3175 1.052e-07
## youngmaleP  -808.572    214.212 -3.7746 0.0001602
##
## Rho: 0.38359, LR test value: 208.46, p-value: < 2.22e-16
## Asymptotic standard error: 0.026143
##     z-value: 14.672, p-value: < 2.22e-16
## Wald statistic: 215.28, p-value: < 2.22e-16
##
## Log likelihood: -21407.05 for lag model
## ML residual variance (sigma squared): 275170, (sigma: 524.57)
## Number of observations: 2782
## Number of parameters estimated: 7
## AIC: 42828, (AIC for lm: 43035)
## LM test for residual autocorrelation
## test value: 15.18, p-value: 9.7716e-05
```

Endogenous spillover (ρ) is 0.384 ($p < 0.001$), which indicates that strong positive spatial dependence in the outcome: tracts' incomes move with neighboring tracts' incomes even after controlling for covariates.

Regression Tables

We will learn how to export our regression table in a fancy way. You need to install a new package, `stargazer` (`install.packages("stargazer")`) and load it to our current R session. Within the parentheses, we can list any model objects (in this case, `model1` and `model2`) that we want to include. Afterwards, we specify our `type = ""`, which in this case is `html`. (The other option is `latex` if we use LaTex, which is an advanced document processor common in political science.) If you want to import the table in a MS Word file, we use `out = ""` to specify the output's file name with extension - the new file to which we want to write our tables. For Word documents, this needs to have a `.doc` extension (NOT `docx`). Make sure to give it a new

name, not the name of an existing document you already have. It will overwrite any pre-existing file with that name. This will print the html code for a table in R;s console and create a new Word document in our working directory.

```
stargazer(model1, SEM1, SDM1, SAR1,
          type = "latex",
          title = "OLS VS. Spatial Linear Models",
          column.labels = c("OLS", "SEM", "SDM", "SAR"),
          colnames = F,
          model.numbers = F,
          dep.var.caption = "",
          dep.var.labels = "Median Household Income",
          covariate.labels = c("Black Population", "Immigrants", "Bachelor Holders",
                               "Young Male Population", "Lagged Black Population",
                               "Lagged Immigrants", "Lagged Bachelor Holders",
                               "Lagged Young Male"),
          keep.stat = c("rsq", "f"),
          notes.align = "l")
```

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Mon, Oct 27, 2025 - 5:09:49 AM

This table presents everything that readers will expect to see in research papers for regression model output. Now you do not need to manually make a regression table anymore. Make sure to change the key elements:

- **title**: Table's title
- **column.labels**: Model numbers
- **dep.var.labels**: Dependent variable's name
- **covariate.labels**: Independent variables' names

Goodness of Fit

Since we have multiple ways to model spatial dependence - specifically the Spatial Error Model (SEM), the Spatial Autoregressive Model (SAR), and the Spatial Durbin Model (SDM) - it is important to evaluate which specification provides the best overall fit to the data.

We can compare these models using two complementary goodness-of-fit criteria: the ANOVA likelihood ratio test and the Akaike Information Criterion (AIC).

The ANOVA test (implemented via `anova.sarlm()` in the `spatialreg` package) compares the log-likelihoods of nested spatial models. It tests whether a more complex model (e.g., SDM) provides a statistically significant improvement in model fit compared to a simpler one (e.g., SEM or SAR). A significant likelihood-ratio statistic (with a small p-value) indicates that the more complex model fits the data better.

```
anova(SEM1, SDM1)
```

```
##      Model df   AIC logLik Test L.Ratio    p-value
## SEM1     1  7 42822 -21404    1
## SDM1     2 11 42790 -21384    2  39.266 6.1379e-08
```

```
anova(SEM1, SAR1)
```

Table 2: OLS VS. Spatial Linear Models

	Median Household Income		model1	
	OLS	*spatial error*	*spatial autoregressive*	
	OLS	SEM	SDM	SAR
Black Population	1.065	139.003***	463.128***	46.221
	(39.807)	(51.847)	(73.793)	(37.819)
Immigrants	545.059**	612.732**	773.436***	353.621*
	(213.040)	(250.410)	(293.620)	(202.822)
Bachelor Holders	990.441***	941.413***	938.486***	688.186***
	(134.567)	(162.392)	(201.789)	(129.419)
Young Male Population	−774.481***	−826.778***	−767.702***	−808.572***
	(225.540)	(225.391)	(226.668)	(214.212)
Lagged Black Population			−562.865***	
			(86.020)	
Lagged Immigrants			−593.969	
			(372.608)	
Lagged Bachelor Holders			−310.063	
			(252.158)	
Lagged Young Male			292.414	
			(415.220)	
Constant	1,313.421***	1,258.177***	823.550***	789.534***
	(29.469)	(38.028)	(49.888)	(43.663)
R^2	0.041			
F Statistic	30.056*** (df = 4; 2777)			
Note:	*p<0.1; **p<0.05; ***p<0.01			

```
##      Model df   AIC logLik
## SEM1     1  7 42822 -21404
## SAR1     2  7 42828 -21407
```

```
anova(SDM1, SAR1)
```

```
##      Model df   AIC logLik Test L.Ratio    p-value
## SDM1     1 11 42790 -21384    1
## SAR1     2  7 42828 -21407    2  45.665 2.8917e-09
```

According to the ANOVA test, SDM fits significantly better than SEM, suggesting that incorporating both spatially lagged dependent and independent variables captures additional spatial structure in the data.

Based on the AIC values, the SEM model fits slightly better than SAR, while the SDM model has the lowest AIC among all three, indicating the best trade-off between goodness of fit and model complexity.

Both the likelihood-ratio tests and AIC comparisons indicate that SDM is the best-fitting model for these data, outperforming both the SEM and SAR specifications.

However, we should always consider the theoretical logic behind spatial dependence before choosing the "best" model. Model selection should balance empirical fit with the substantive theory of how spatial processes operate in the real world.

Acknowledgement

I gratefully acknowledge the contributions of my teaching assistants, Kade Davis and Myles Ndiritu whose professionalism and commitment have played an essential role in developing and delivering this lab.

References

Alexander, Nathan. 2025. *Critical Computational Geographies - Measures of Segregation: Dissimilarity.*
Canche, Manuel S. Gonzalez. 2020. *Matrices of Influence.* https://rpubs.com/msgc/matrices_influence.
Sciabolazza, Valerio Leone. 2017. *Spatial Statistics and Spatial Econometrics.*