Exploratory Spatial Data Analysis in R

Hi, there.

- Research Analyst at The NPD Group
- MS in Urban Informatics at Northeastern
- BA Sociology & Anthropology @ Plymouth State, NH
  - Certificate in GIS
  - Minor in Math
    - I got a D in calc II though :/ Trig sub? Really?

Today's Goal

- Introduce the fundamentals of:
  - Spatial data
  - The first law of geography
  - The objective of ESDA
  - {sfdep} R package

Exploratory Data Analysis (EDA)

A brief review

"Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to:

1. *maximize insight into a data set;*
2. *uncover underlying structure;*
3. *extract important variables;*
4. *detect outliers and anomalies…"*

– *NIST Handbook*

❝

"EDA is not a formal process with a strict set of rules. More than anything, EDA is a state of mind."

– *R for Data Science, Hadley Wickham*

Exploratory Data Analysis

- Goal: “discover potentially explicable patterns” (Good, 1983)
- Emphasis on data visualization
- Use of descriptive statistics
- Discovering outliers

Exploratory Spatial Data Analysis (ESDA)

Extending EDA to spatial data

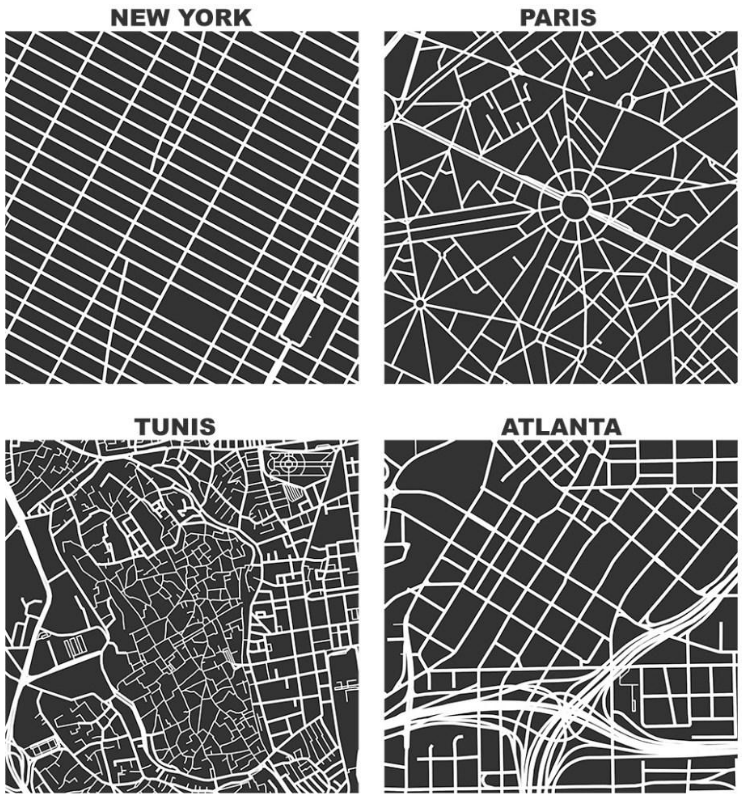
First, spatial data.

First, spatial data

- Many sources and domains:
  - Transportation
  - Ecology
  - Environmental Sciences
  - Demography
- Let’s explore a few

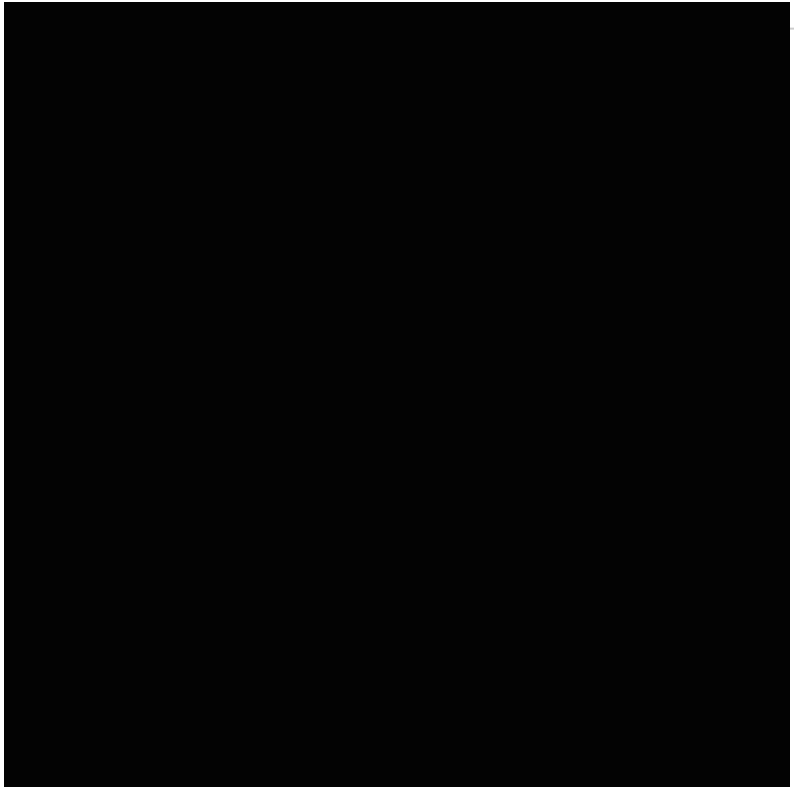
Road Networks

- Square Mile Street Network, Geoff Boeing
- [Source]



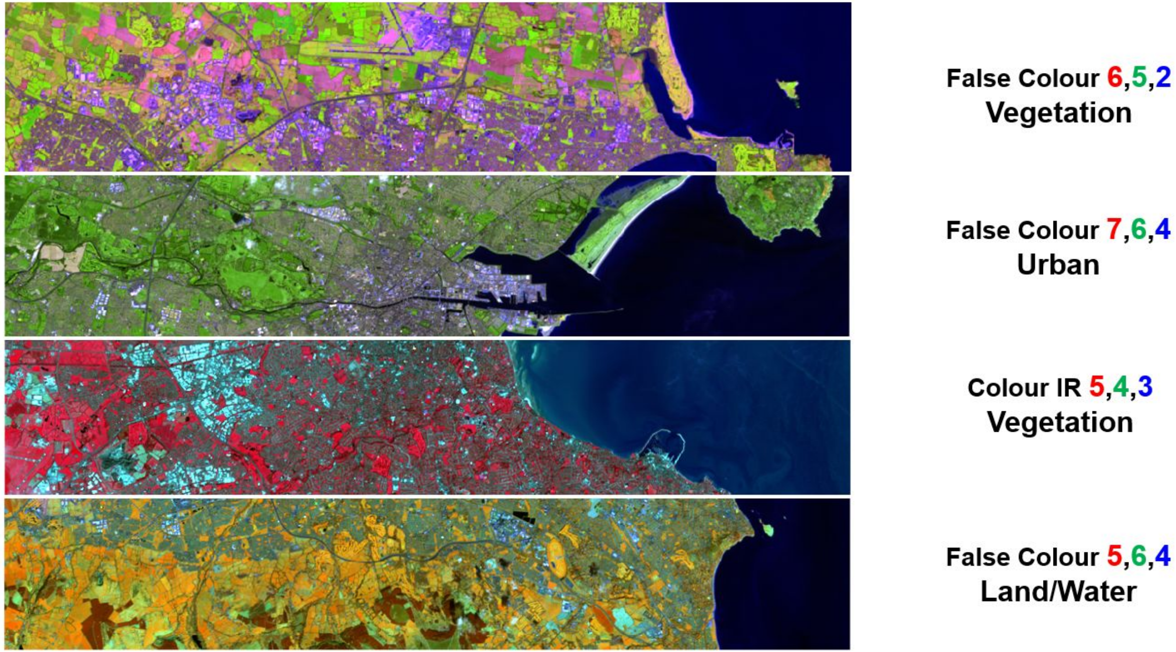
Plant Species

- Flower colors in Pune, India, Aseem Deodhar
- Source



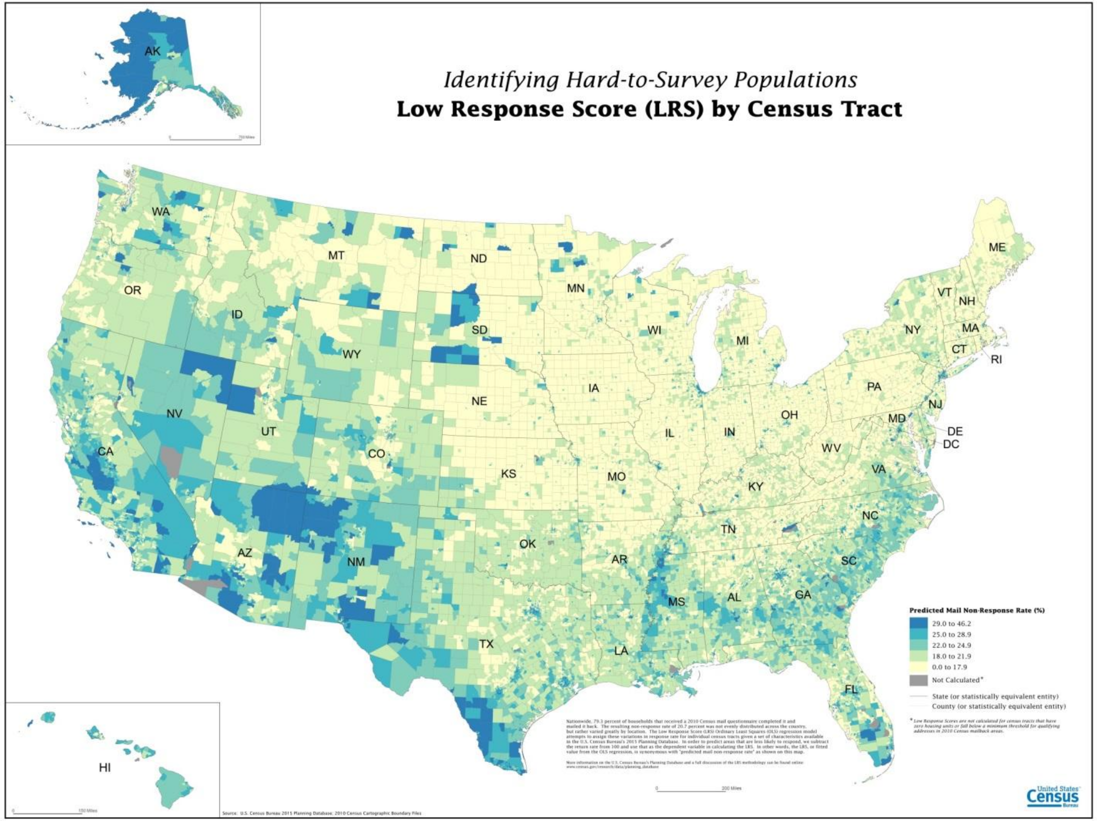
Satellite Imagery

- Landsat Color Band Combinations, David Harbor
- [Source]



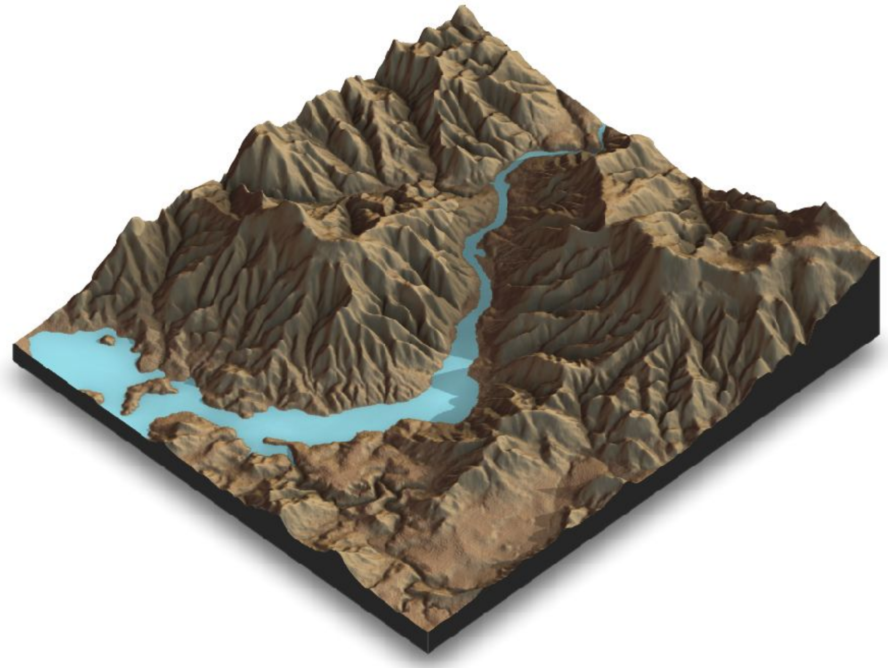
Regions

- Identifying Hard-to-Survey populations, Census
- [Source]



Digital Elevation Models

- {rayshader}, Tyler Morgan-Wall
- Source



BIG Spatial Data

- Big Data is often Big Spatial Data
- ~80% of Big Data have a spatial component
  - Huang & Wang, 2020
- Sources:
  - LiDAR (Light detection and ranging)
  - Point-of-Sale (POS) data
  - Point-of-interest (POI) data
  - IoT devices
  - 311 services
  - Road network data

LiDAR

- How Autonomous Cars Map the Environment
- Source

POS Data

- E.g. The NPD Group collects POS data from 600k+ retailers
- Every sales transaction is recorded
- Sales are tracker at a per-store level
- Stores are in a physical location
- Used to track sales performance

NPD Retail Tracking

POI Data

- One of the fastest growing data markets
- Tracks any place:
  - Restaurants (Yelp)
  - Housing (Zillow, Trulia, etc)
  - Rentals (Airbnb)
- Vendors like SafeGraph and Carto

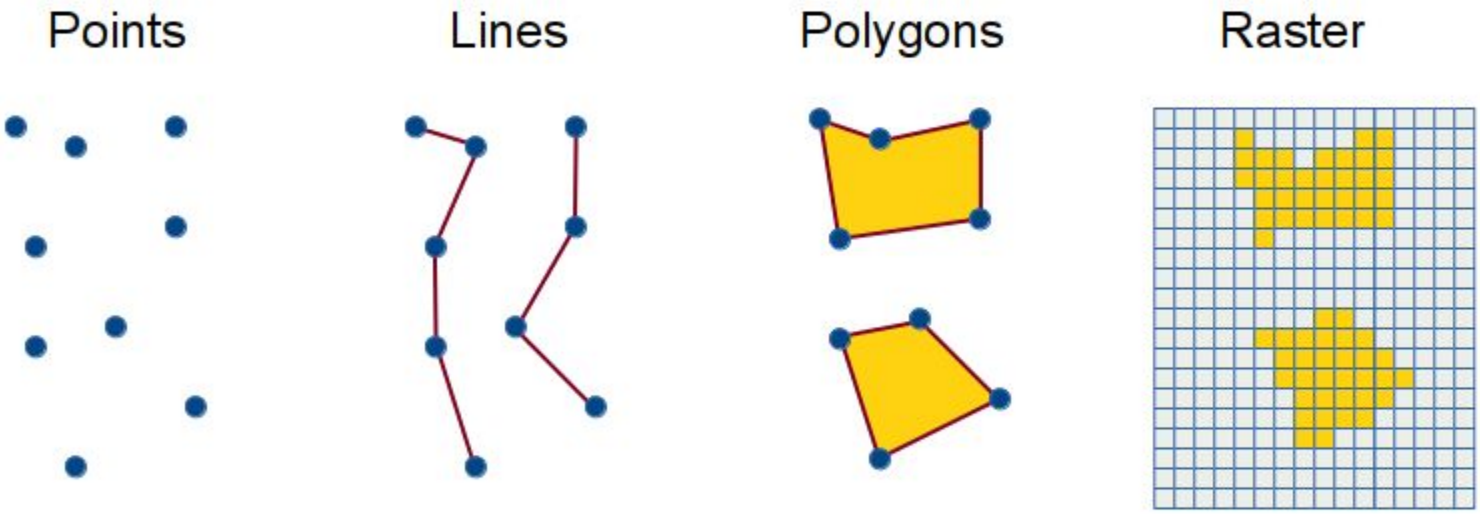
Types of Spatial Data

- **Vector:**
  - Coordinate based data
  - Points, lines, and polygons
  - Most commonly seen
- **Raster:**
  - A grid of cells (pixels) where each cell has a value

Vector Data

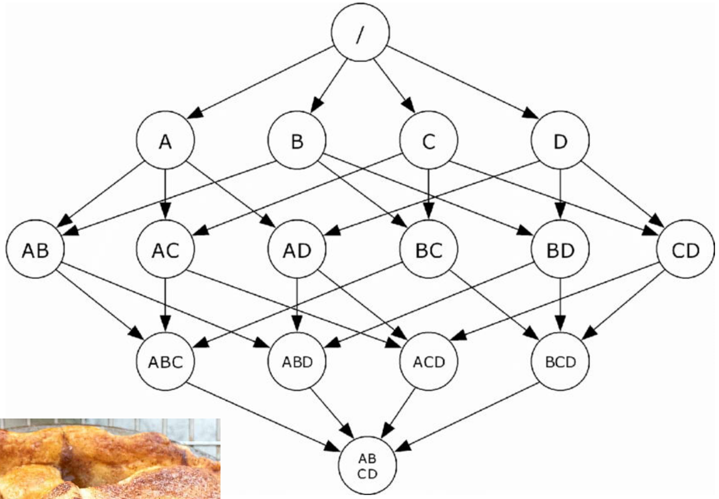
- **Points**: a (long, lat) coordinate
  - Sometimes Z (altitude)
- **Lines**: two or more connected points
- **Polygons**: four connected points that create a closed shape

Vector Data



Some terms you may see

- **Point Pattern**:
  - Refers to analysis of points
- **Lattice**:
  - Refers to analysis of polygons
  - Much of ESDA fall into the lattice data category
  - We'll focus on lattice



Some nice lattices

“Everything is related to everything else, but near things are more related than distant things.”

(Waldo R. Tobler, 1970)

Exploratory Spatial Data Analysis (ESDA)

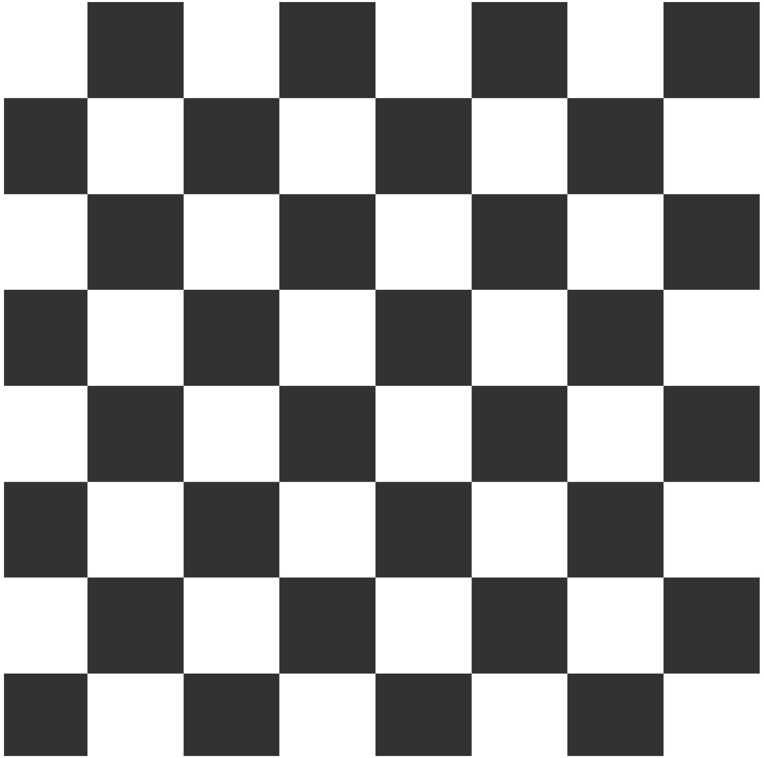
- Extends EDA to spatial relationships
- Are things randomly distributed?
- Are there spatial outliers?
- Focus on:
  - spatial autocorrelation
    - Are values related to their neighbors?
  - Spatial heterogeneity
    - Is there an uneven distribution of values?

ESDA

- EDA compares a part to the whole
- ESDA compares a part to its neighboring parts
- In ESDA we evaluate a location to its *neighborhood*
- How do we define a neighborhood?

Spatial Neighbors

- Take a chess board
- Each square is a polygon
- What is the neighborhood of each square?

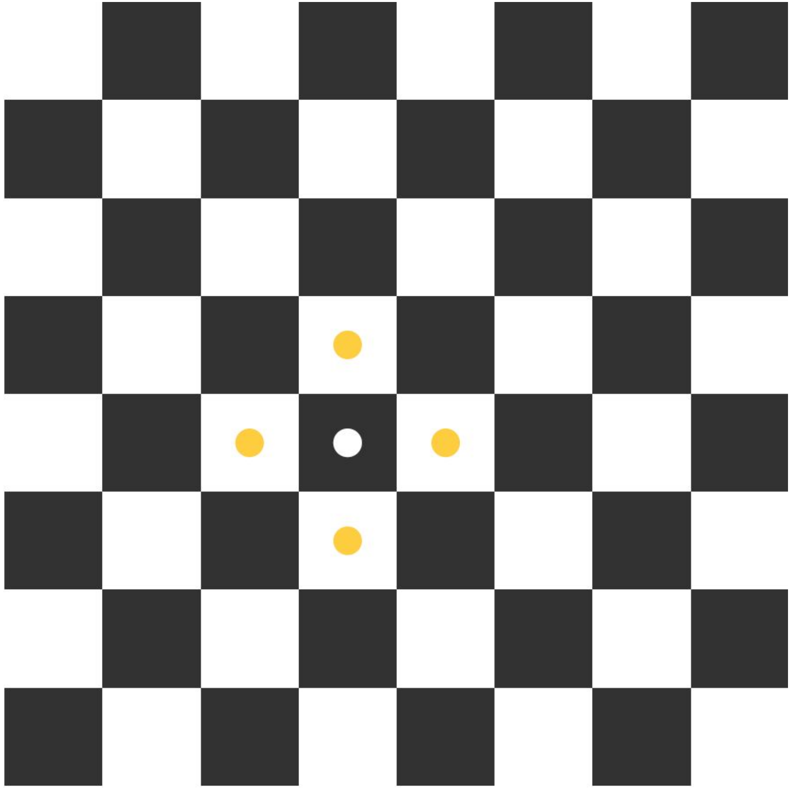


Defining Spatial Neighbors

- Lattice data uses *contiguity*
- **Contiguity:**
  - "touching or being next to something."
- Two main types:
  - Rook (edges) contiguity:
    - touching by sides
  - Queen (edges and vertices) contiguity:
    - Touching by sides and corners

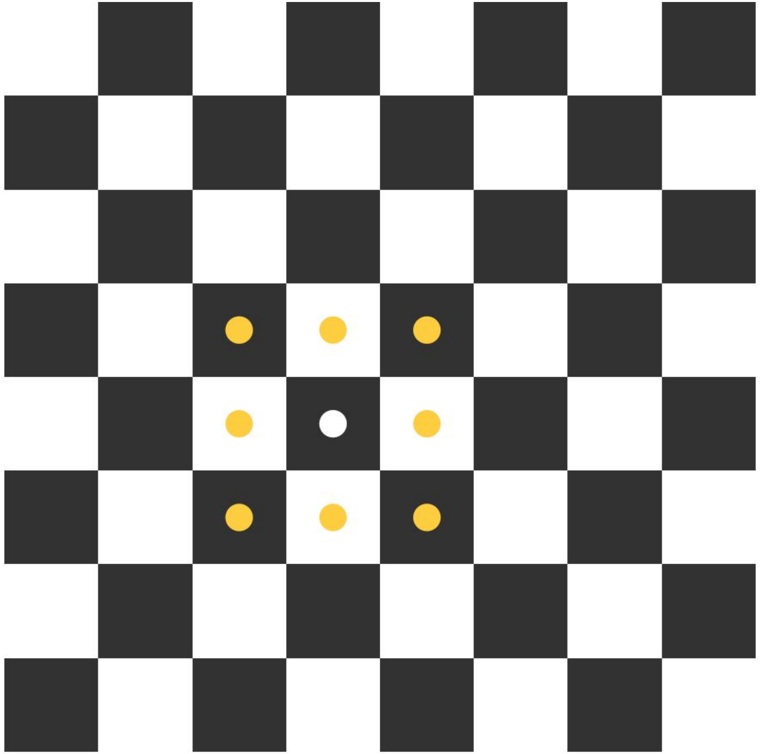
Queen's Gambit: Rook Contiguity

- 1. d4 ..
- Edges (sides) touch:
  - c4
  - d3, d5
  - e4
- 4 total neighbors

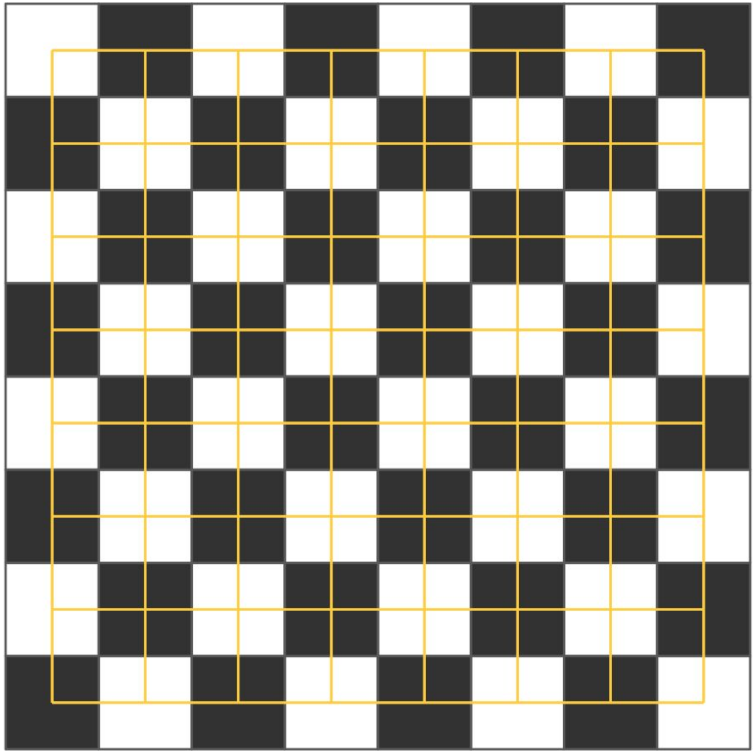


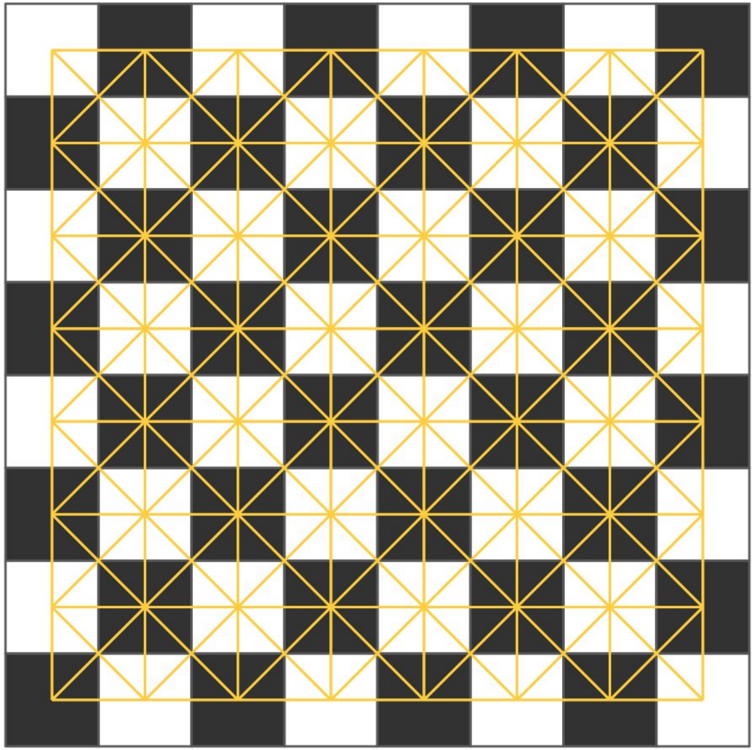
Queen's Gambit: Queen Contiguity

- Neighbors:
  - c3, c4, c5
  - d3, d5
  - e3, e4, e5
- 8 total neighbors



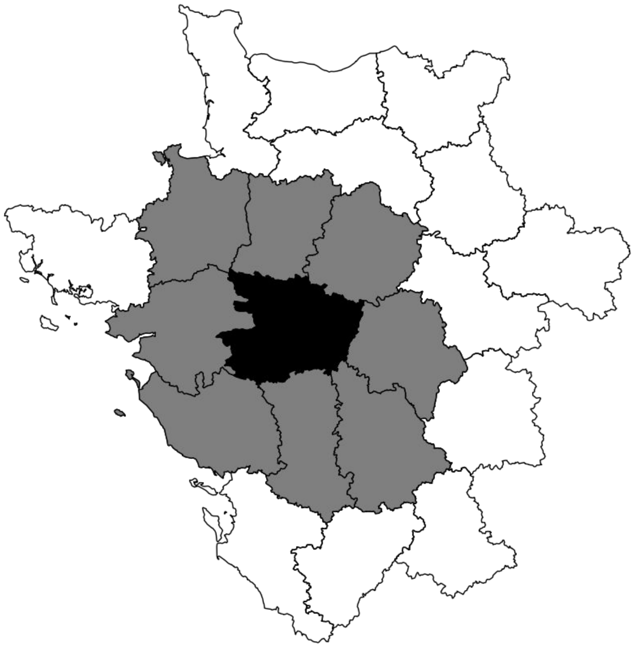
Rook & Queen Contiguity





Spatial Neighbors in Practice

- Vertices and edges aren't so simple in reality
- A region in France and its neighbors



How do we evaluate the neighborhood?

- We compare a point value to the neighborhood’s point values
- Calculate the *spatial lag*
  - The average value for a variable in a neighborhood
  - More on this in a bit
- But how important should each neighbor be?

Spatial Weights

- Based on Tobler's first law, closer neighbors should have higher weights
- No effective distance measure for polygons
- Each polygon gets an equal weight
  - Called *row standardized* weights
  - Defined as $w_{ij(s)} = w_{ij} / \sum_j w_{ij}.$
  - More simply: `1 / (# of neighbors)`
  - E.g. 4 neighbors with weight 0.25

The spatial lag

What is the *average* value of a neighborhood?

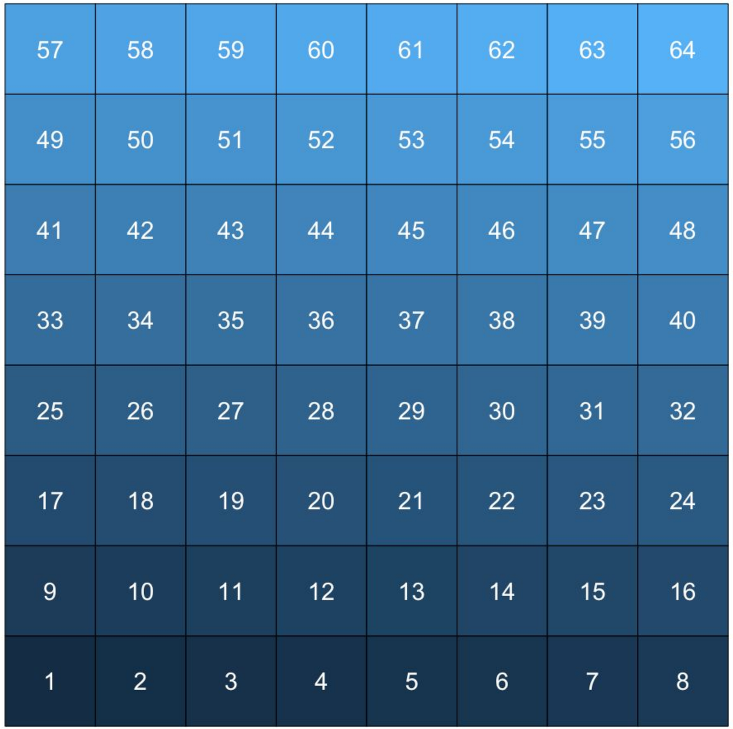
Spatial Lag

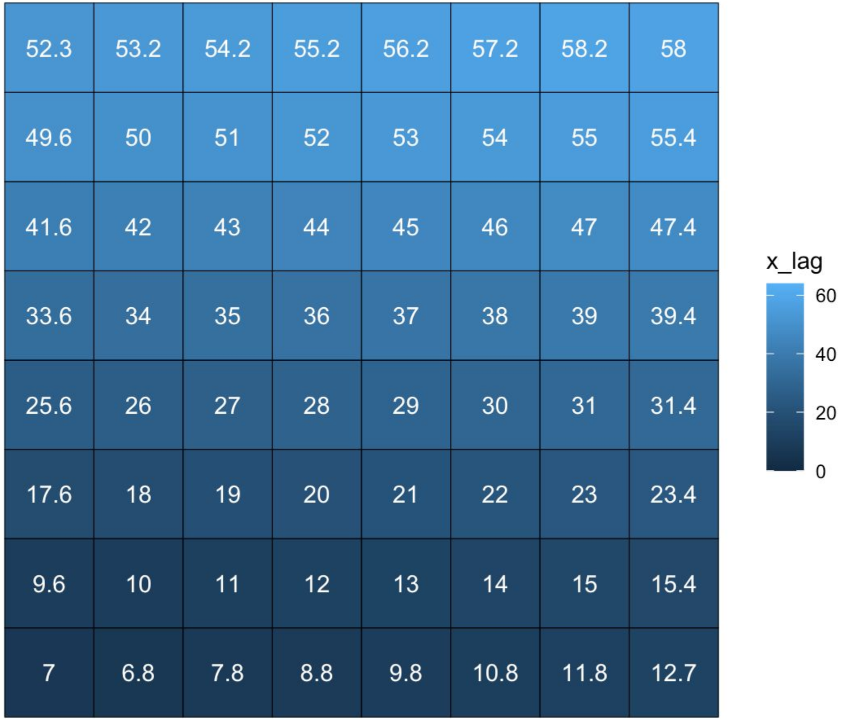
- In time-series a lag compares a value to itself after an amount of time
- In spatial analysis, compare a value to its neighbours
- The expected value of the neighborhood (excluding the observed point)
- Given by $[Wy]_i = \sum_{j=1}^{n} w_{ij} y_j$

Spatial Lag Example

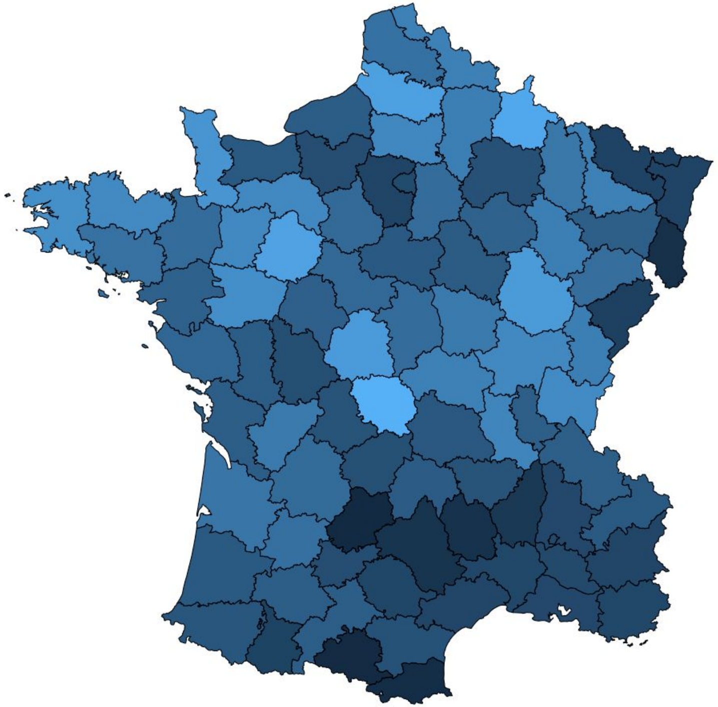
```
xi <- 100
xj <- c(5, 40, 100, 35)
wj <- 1 / length(xj)
sum(xj * wj)
#> [1] 45
```

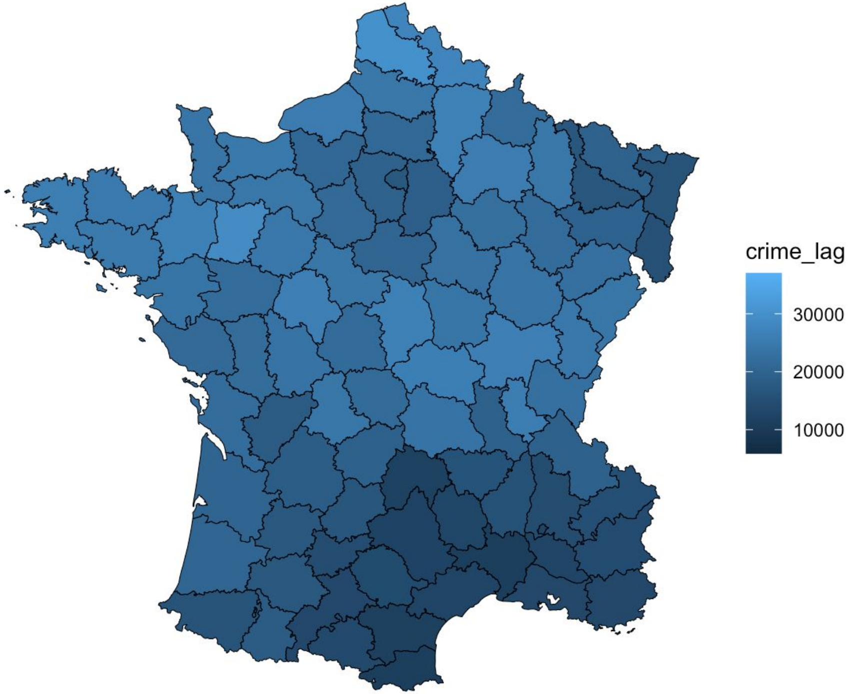
Chess Board Spatial Lag





Crime in 1830's France



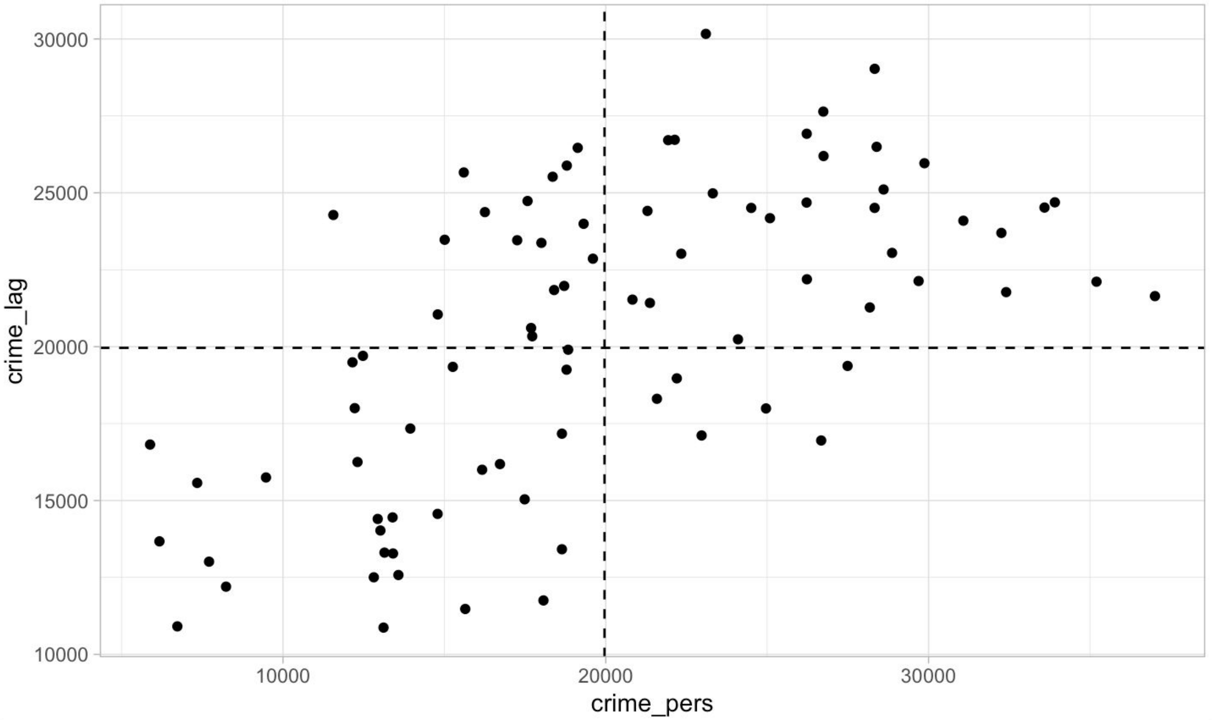


Plotting the Spatial Lag

- Use a scatter plot
  - AKA Moran Plot
- X axis: original variable
- Y axis: the lagged variable
- Slope of the regression line Y ~ X is the autocorrelation
- Can use the plot to classify clusters

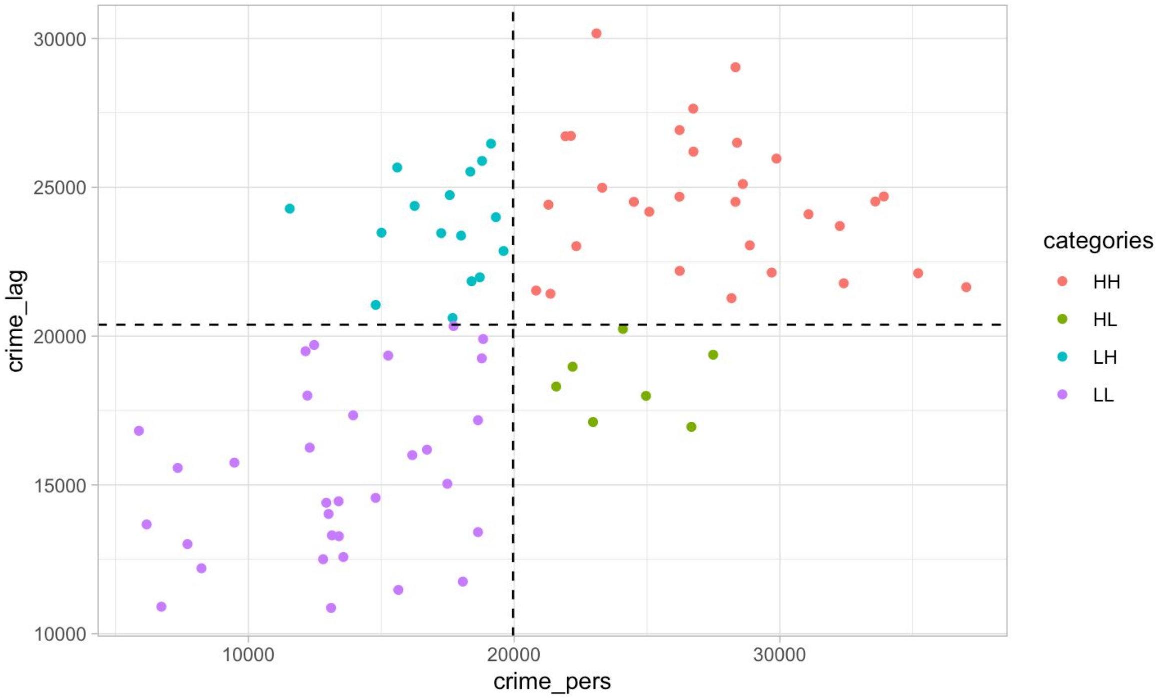
Moran Plot

- Intercepts at mean
  - Centers at 0
- Effectively plotting Z-scores
- Quadrants identify cluster types



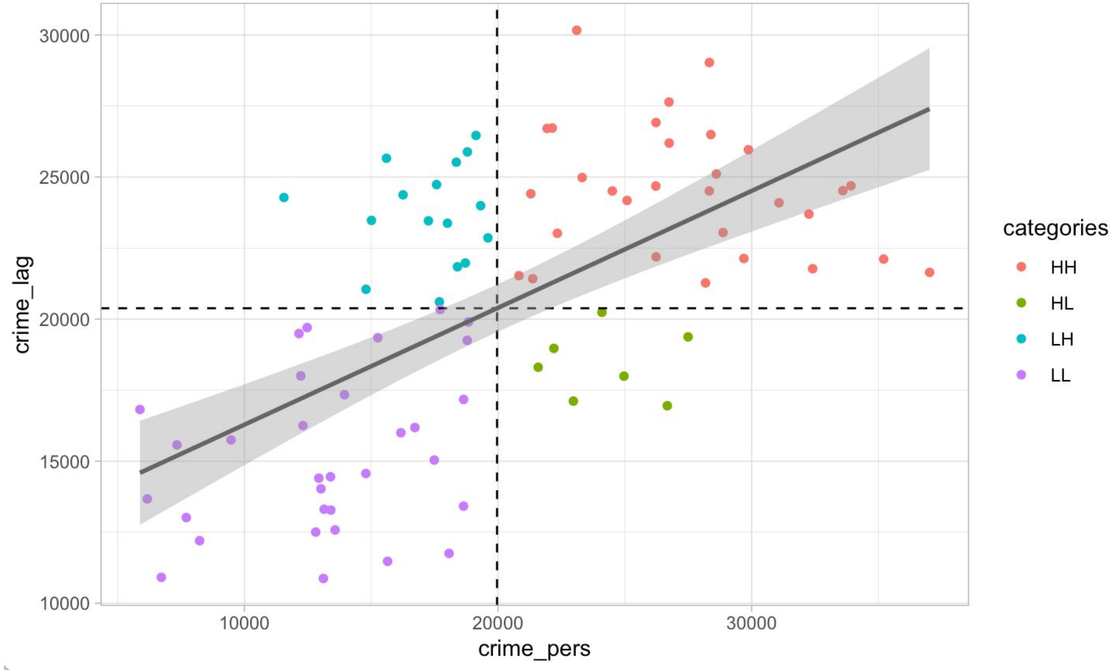
Moran Plot

- High-High
  - Large value with large value neighbors
- High-Low
  - Large value with low value neighbors
- Low-Low
  - Small value with low value neighbors
- Low-High
  - Small value with high value neighbors



Moran Plot

- Slope represents spatial autocorrelation



Spatial Autocorrelation

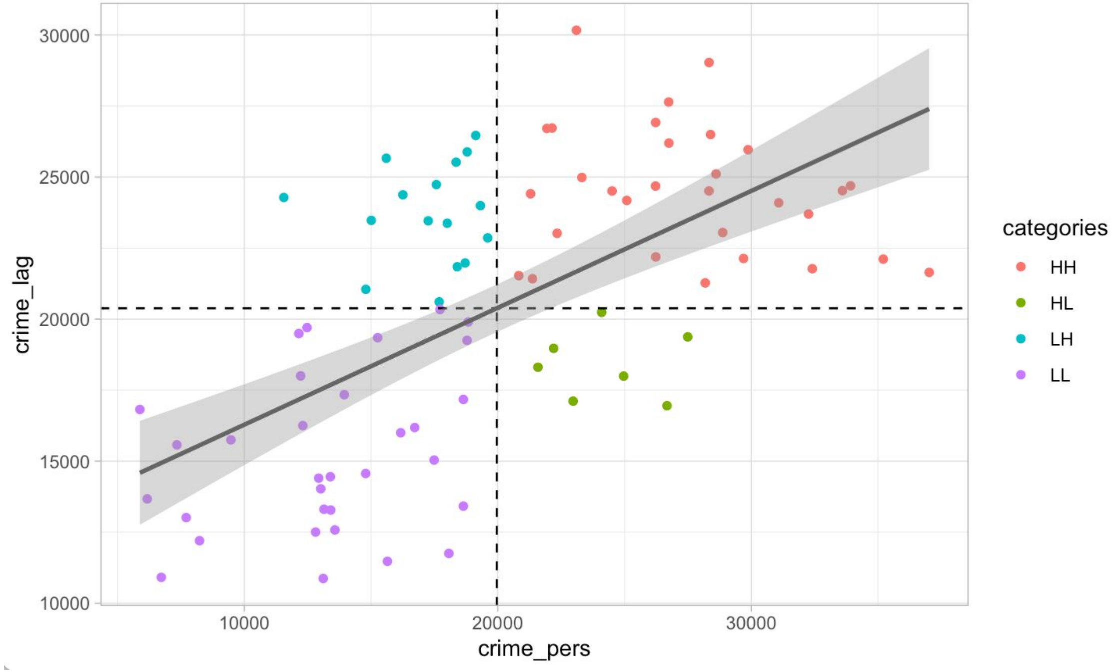
- The tendency of like values to cluster in space
- Based on the assumption of spatial randomness
  - Values are randomly dispersed through a landscape
- Positive: when like values cluster together
- Negative: when dissimilar values cluster together

Global Moran's I

- Most common measure of autocorrelation
- Like Pearson's R, ranges from [–1, 1]
- –1 perfect negative autocorrelation
- 1 perfect positive autocorrelation
- Defined as $$\frac{\sum_i \left(z_i \times \sum_j w_{ij} z_j\right)}{\sum_i z_i^2}$$

Moran Plot

- Moran's I is 0.411 here



Inference

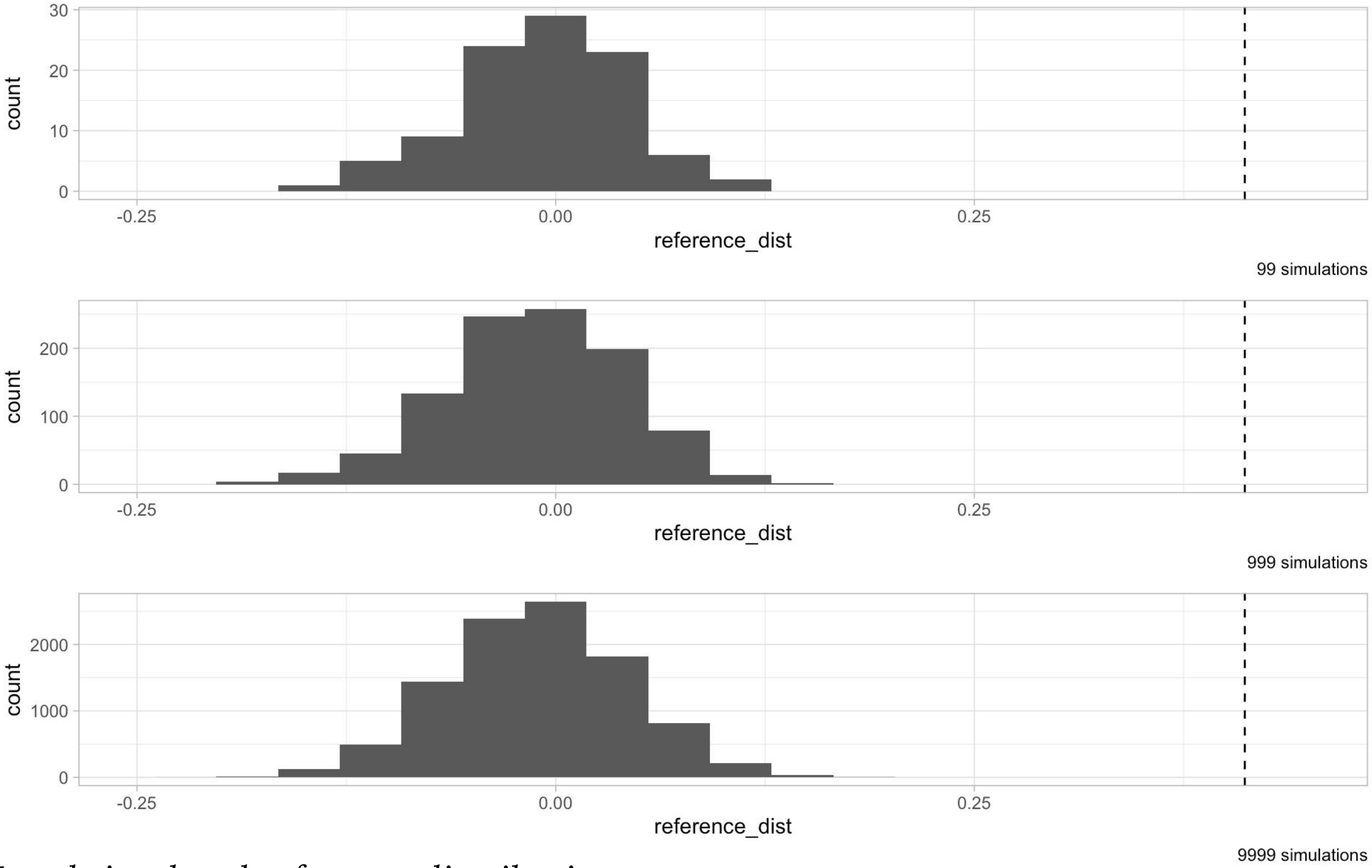
- Null hypothesis: spatial randomness
- Traditional hypothesis testing uses assumptions of normality
- Normality is rarely observed in spatial data
- Permutation based inference is used
- Permutations create a random reference distribution
- $p = (R + 1) / (M + 1)$

Inference

- Uses conditional permutation
  - Neighbors are randomly sampled at point i
  - Point i can never be its own neighbor
  - Simulates spatial randomness
- Calculate Moran's I with randomized neighbors
- M is the number of permutations
- R is the number of times the random Moran's I is greater than the observed Moran's I

Inference

- Simulated p-value is dependent upon how many simulations you run
- Be extra conservative!
- Typically a p-value of 0.01 or 0.001 is used for significance cut-offs



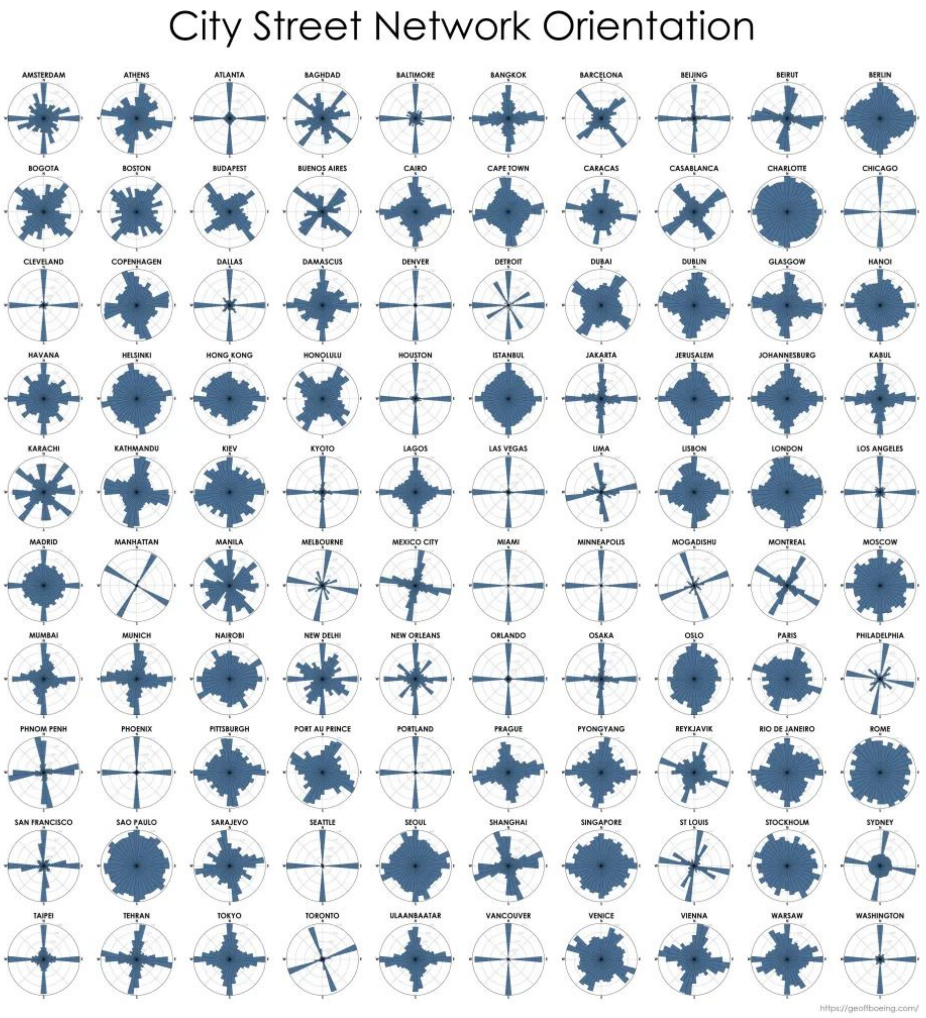
Moran's I and simulated reference distributions

{sfdep}

A tidy interface to spdep for spatial dependence.

Motivation

- PPUA 7237: Advanced Spatial Analytics w/ Geoff Boeing, 2019
- Used python and Pysal library
- Forever miffed if I have to use python
- Mind blown by spatial regression
- An R tool for ESDA sat in the back of my mind



Mid-pandemic 2021

- Somehow got the idea to relearn spatial stats
- Discovered r-spatial/spdep
- spdep felt idiosyncratic as a tidyverse-stan
- spdep *almost* tidyverse compatible
- Resultant objects lacked "list" class and could not be included in a tibble
- Issue 59 insightful and motivational

{sfdep} precursor

- In May, 2021 developed {sfweight}
  - Too heavy reliance on dplyr
  - Ignored advice from Issue 59
- Added functionality to spdep and learned a lot
- Decided on a new interface to spdep

{sfdep} principles

- Always use sf objects for geometry
- Always return dplyr friendly objects
  - Vectors, lists, data frames
- Functionality is not dependent upon dplyr
- Minimal light dependencies
- All functionality is implemented on spdep objects when possible
  - nb and listw class objects

Making neighbors

- `st_contiguities(geometry, queen = TRUE)`
- Takes an sfc class object
  - the geometry column of an sf object
- Returns a nb class object (list)

Making neighbors

```
library(sf)
library(sfdep)
library(tidyverse)
st_contiguity(st_geometry(guerry))
#> Neighbour list object:
#> Number of regions: 85
#> Number of nonzero links: 420
#> Percentage nonzero weights: 5.813149
#> Average number of links: 4.941176
```

Making neighbors

```
guerry |>
  transmute(nb = st_contiguity(geometry))
#> Simple feature collection with 85 features and 1 field
#> Geometry type: MULTIPOLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 47680 ymin: 1703258 xmax: 1031401 ymax: 2677441
#> CRS:           NA
#> # A tibble: 85 × 2
#>    nb                                                             geometry
#>  * <nb>                                                     <MULTIPOLYGON>
#>  1 <int [4]> (((801150 2092615, 800669 2093190, 800688 2095430, 800780 2095795,…
#>  2 <int [6]> (((729326 2521619, 729320 2521230, 729280 2518544, 728751 2517520,…
#>  3 <int [6]> (((710830 2137350, 711746 2136617, 712430 2135212, 712070 2134132,…
```

The neighbor list

- Each element is an integer vector
- Elements contain row position of neighbors

```
nb <- st_contiguity(st_geometry(guerry))
nb[1:3]
#> [[1]]
#> [1] 36 37 67 69
#> [[2]]
#> [1]  7 49 57 58 73 76
#> [[3]]
#> [1] 17 21 40 56 61 69
```

Spatial Weights

- `st_weights(nb)` requires a nb object
  - Row standardized by default
  - Each weight is the same
- Each element is a numeric vector
  - Contains weight for each indexed observation in nb

Spatial Weights

```
guerry_nb <- guerry_nb |>
  mutate(nb = st_contiguity(geometry),
         wt = st_weights(nb))

pull(guerry_nb, "wt")[1:2]
#> [[1]]
#> [1] 0.25 0.25 0.25 0.25
#>
#> [[2]]
#> [1] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
```

Spatial Lag

- st_lag(x, nb, wt) calculates the spatial lag for x
  - X is a numeric vector
  - nb is a neighbor object
  - wt is a weights list
- Use cases:
  - Moran Plot
  - Identify neighborhood averages
  - Useful in regression

Spatial Weights

```
guerry_lag <- guerry_nb |>
  mutate(crime_lag = st_lag(crime_pers, nb, wt))

select(guerry_lag, crime_pers, crime_lag) |>
  slice(1:3)
```

```
#> # A tibble: 3 × 3
#>   crime_pers crime_lag                                                  geometry
#>        <int>     <dbl>                                        <MULTIPOLYGON>
#> 1      28870    23048. (((801150 2092615, 800669 2093190, 800688 2095430, 80078…
#> 2      26226    26920. (((729326 2521619, 729320 2521230, 729280 2518544, 72875…
#> 3      26747    26195. (((710830 2137350, 711746 2136617, 712430 2135212, 71207…
```

Classifying observations

- Utilize dplyr to classify observations into clusters
- Use to clusters to create a Moran Plot

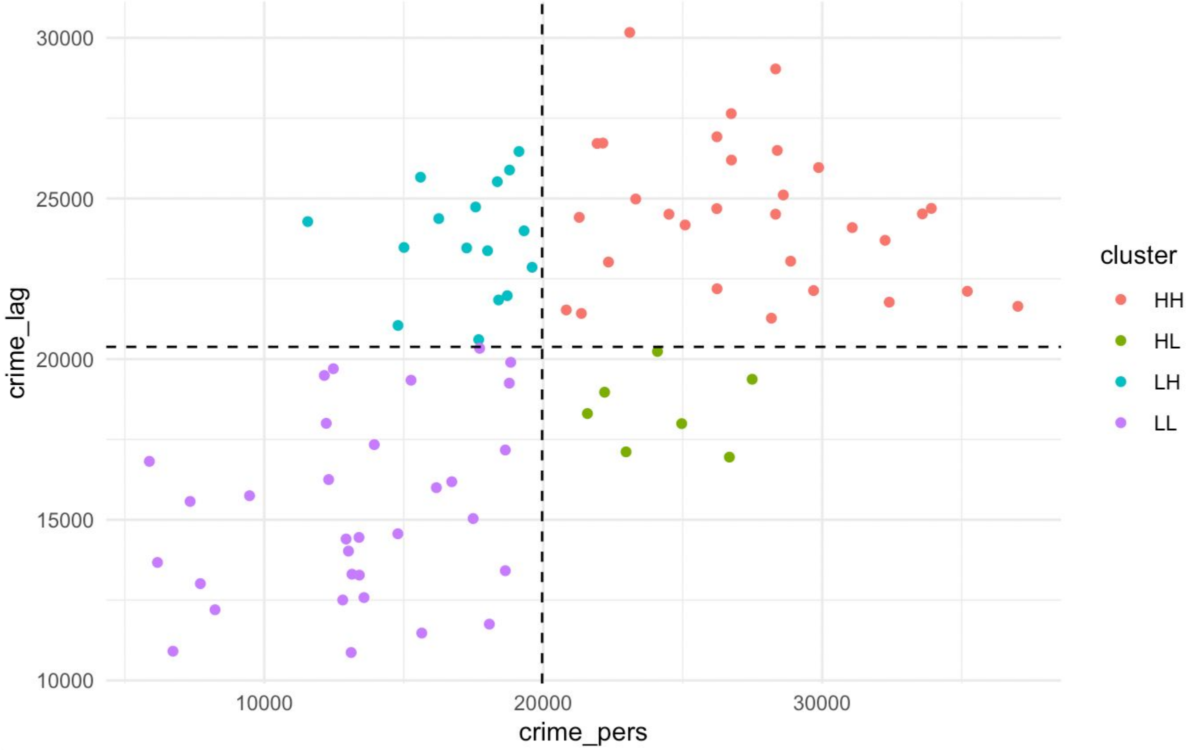
```
guerry_clusters <- guerry_lag |>
  mutate(cluster = case_when(
  crime_pers > mean(crime_pers) & crime_lag > mean(crime_lag) ~ "HH",
  crime_pers > mean(crime_pers) & crime_lag < mean(crime_lag) ~ "HL",
  crime_pers < mean(crime_pers) & crime_lag < mean(crime_lag) ~ "LL",
  crime_pers < mean(crime_pers) & crime_lag > mean(crime_lag) ~ "LH"
  ))
```

Creating a Moran Plot

- Create a basic scatter plot
- Add intercepts to delineate clusters

```
guerry_clusters |>
  ggplot(aes(crime_pers, crime_lag, color = cluster)) +
  geom_point() +
  geom_vline(aes(xintercept = mean(crime_pers)), lty = 2) +
  geom_hline(aes(yintercept = mean(crime_lag)), lty = 2) +
  theme_minimal()
```

Creating a Moran Plot



Local Indicators of Spatial Association (LISAs)

- LISAs evaluate a statistic at the neighborhood
- Moran's I can be calculated for each region
  - Called the Local Moran
  - The OG LISA
- Local Moran evaluates autocorrelation within clusters
- Uses conditional permutation for significance

Local Moran

- `local_moran(x, nb, wt)`
  - Calculates Moran's I at each point
  - Identifies clusters (High-High, Low-Low, etc.)
  - Calculates multiple p-values
    - Assumption of normality
    - Simulated rank-based p-values
    - Simulated p-values using (R + 1) / (M + 1)
  - Returns a data frame
    - tidyr to the rescue

Local Moran

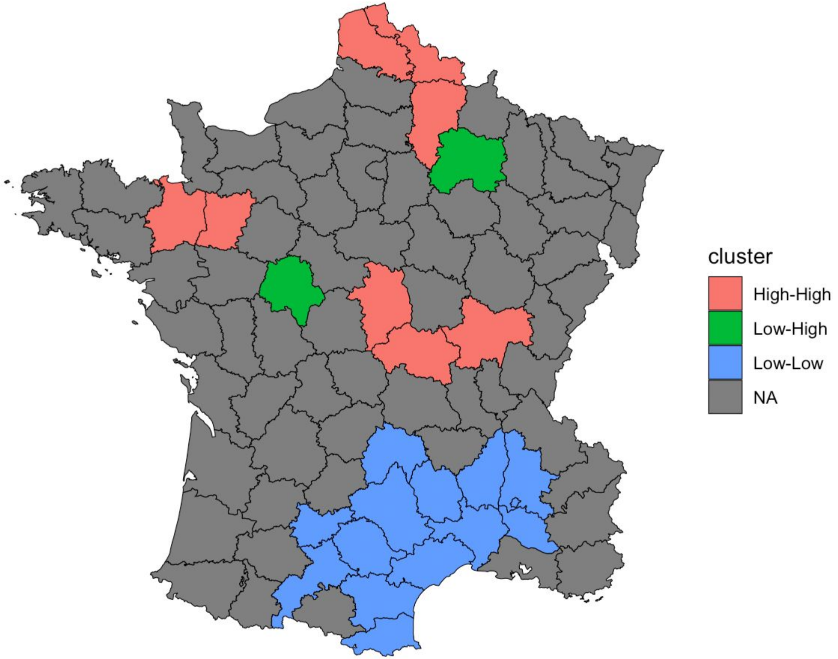
```
# calculate local moran
guerry_lisa <- guerry_nb |>
  mutate(lisa = local_moran(crime_pers, nb, wt)) |>
  unnest(lisa)

# preview rows
guerry_lisa |>
  st_drop_geometry() |>
  select(last_col(11):last_col()) |>
  glimpse()
```

Local Moran

```
#> Rows: 85
#> Columns: 12
#> $ ii          <dbl> 0.52226452, 0.82801651, 0.80353997, 0.74188966, 0.2311871…
#> $ eii         <dbl> -0.0231299954, -0.0357342570, 0.0256200595, -0.0504730086…
#> $ var_ii      <dbl> 0.3347282713, 0.1231900508, 0.1486139225, 0.2244847022, 0…
#> $ z_ii        <dbl> 0.9426806, 2.4609378, 2.0179256, 1.6723633, 1.1733057, 1.…
#> $ p_ii        <dbl> 0.345844300, 0.013857441, 0.043599016, 0.094452704, 0.240…
#> $ p_ii_sim    <dbl> 0.360, 0.008, 0.052, 0.108, 0.240, 0.108, 0.572, 0.104, 0…
#> $ p_folded_sim <dbl> 0.180, 0.004, 0.026, 0.054, 0.120, 0.054, 0.286, 0.052, 0…
#> $ skewness    <dbl> -0.08509936, -0.07472420, -0.07216080, -0.02028766, -0.21…
#> $ kurtosis    <dbl> -0.399030265, -0.109908940, 0.221181963, -0.033306492, -0…
#> $ mean        <fct> High-High, High-High, High-High, Low-Low, Low-Low, Low-Lo…
#> $ median      <fct> High-High, High-High, High-High, Low-Low, Low-Low, Low-Lo…
#> $ pysal       <fct> High-High, High-High, High-High, Low-Low, Low-Low, Low-Lo…
```

Local Moran Clusters

```
guerry_lisa |>
  mutate(cluster = ifelse(
    p_folded_sim <= 0.05,
    as.character(mean), NA)) |>
  ggplot(aes(fill = cluster)) +
  geom_sf(lwd = 0.2, color = "black") +
  theme_void()
```



Worked Example

Educational Attainment in Boston
To RStudio we go.

What else is there?

It can’t just be this one statistic…

Topics to explore

- Point pattern analysis
- Different neighbors:
  - k-NN, distance band, lagged neighbors
- Different weighting:
  - Distance based
  - Kernel decay
  - Inverse distance
- Spatial regressionx

Some other measures

- Join Counts
  - Evaluates binary variables
- Neighbor match test
  - Continuous variables
  - Compare neighbors in physical and attribute distance
- Colocation
  - Categorical variables
  - Checks for asymmetric relationships between points

Thank you!