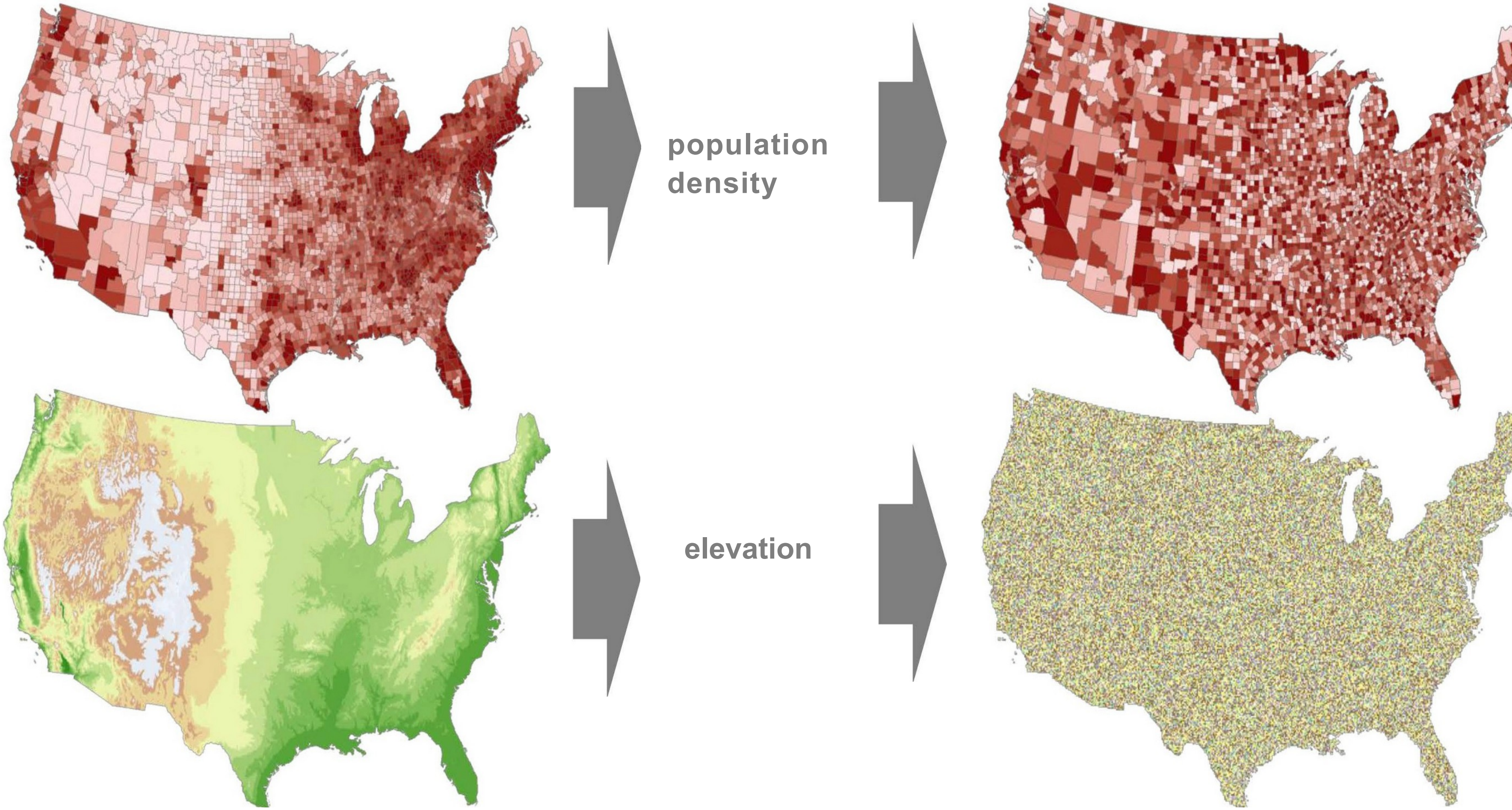
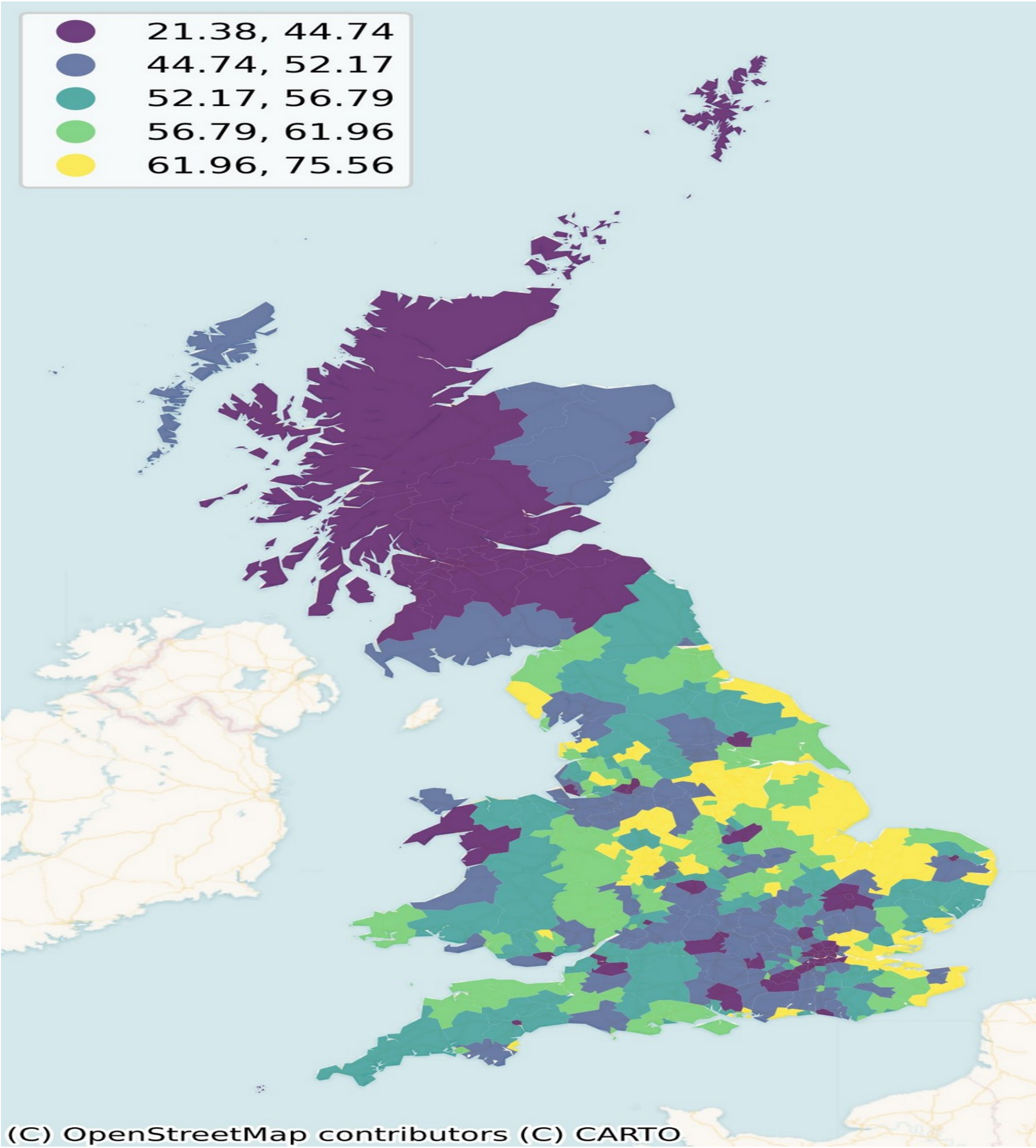
If features *were* randomly distributed





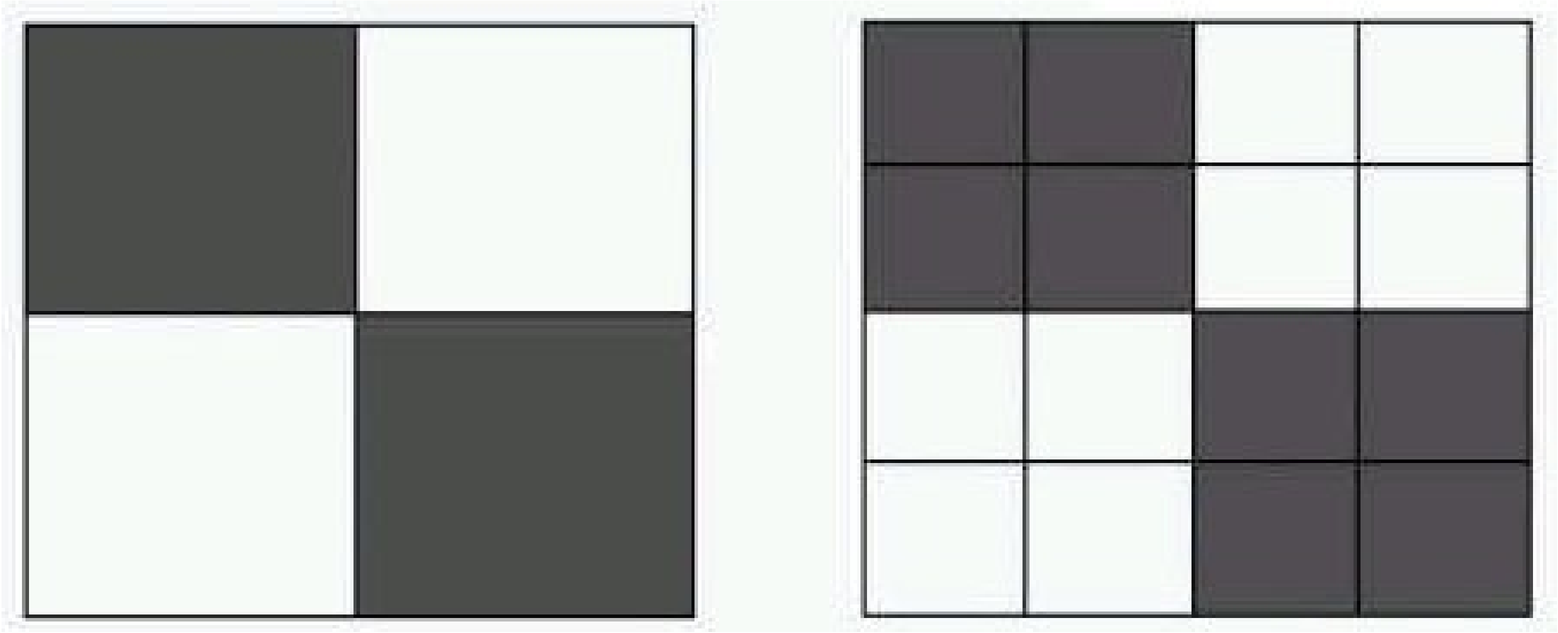
The Geography of BREXIT

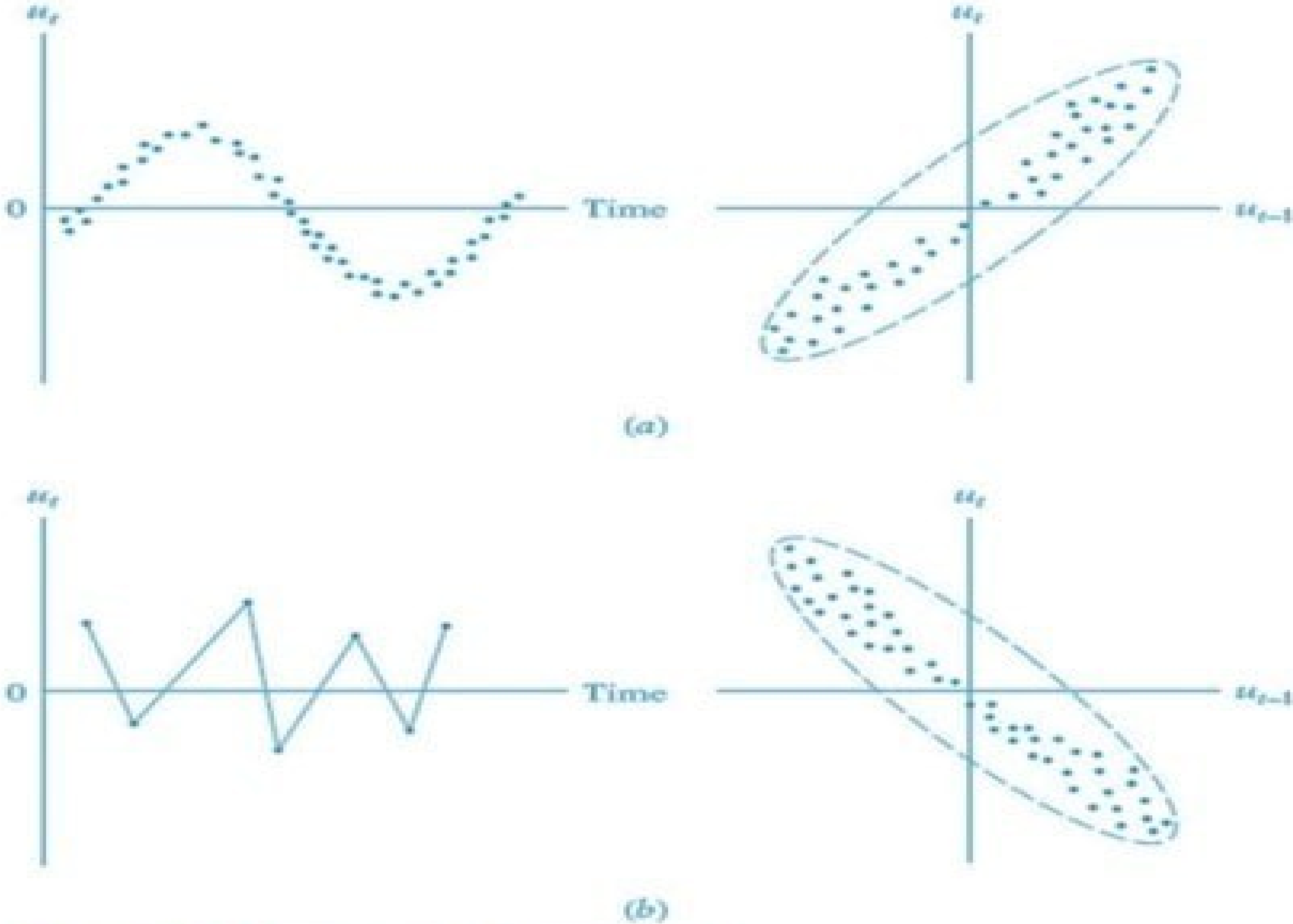
Self-correlation in Spatial Analysis

- In spatial analysis self-correlation (autocorrelation) is commonly used to describe spatial patterns.

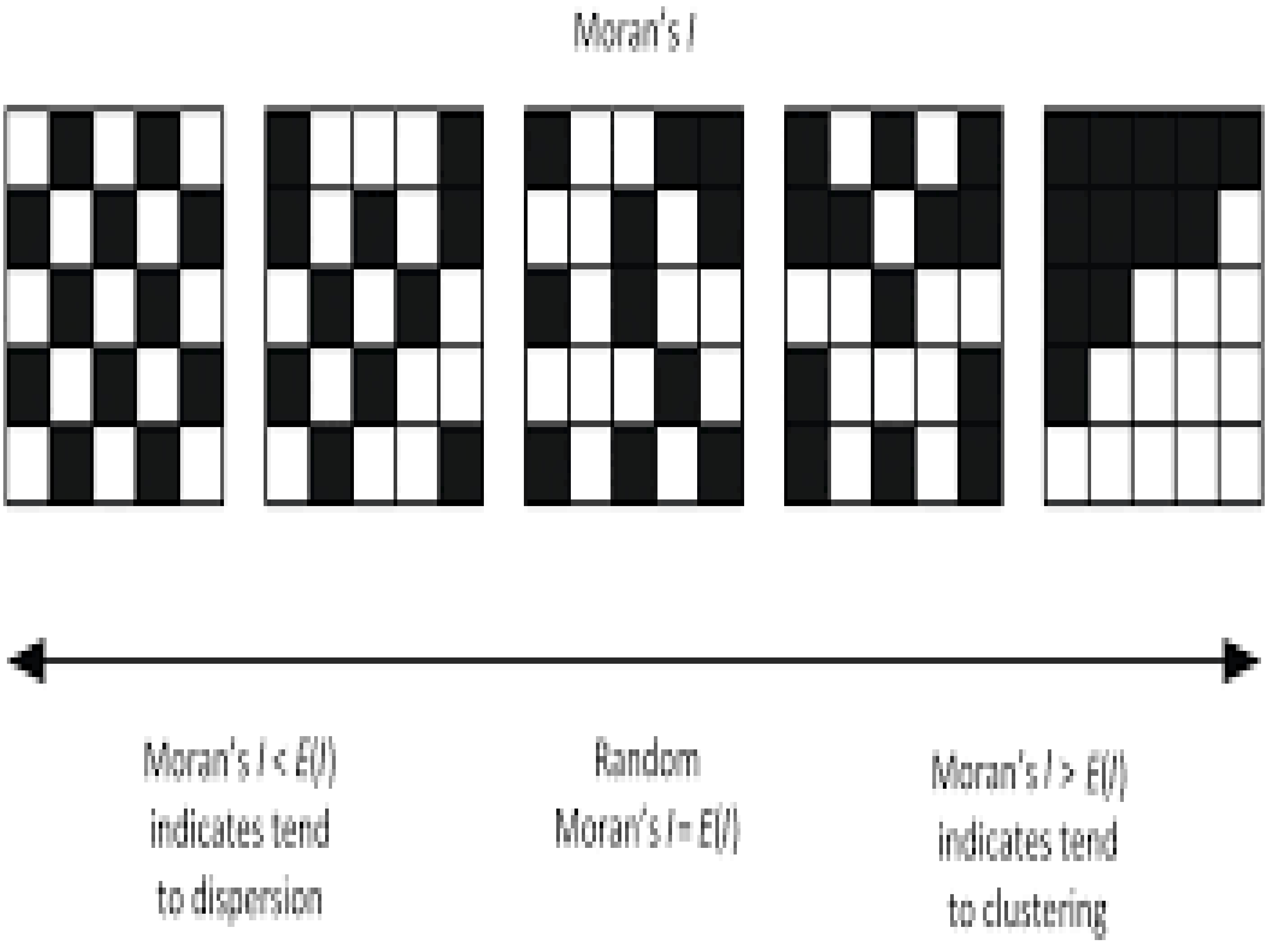
- The relationship between a value over here compared to the value over there

Spatial Autocorrelation is Scale-Dependent!





(*a*) Positive and (*b*) negative autocorrelation.


Moran's I
Moran's I < E(I)
indicates tend
to dispersion
Random
Moran's I = E(I)
Moran's I > E(I)
indicates tend
to clustering

Spatial Relationships

- To measure spatial autocorrelation we need a way to mathematically represent spatial relationships.
- Everything on a map has some kind of spatial relationship with everything else.

Spatial Weights Matrix

- We represent spatial relationships using a matrix.
- A matrix is essentially a table, like a mileage chart, that has rows and columns.

- The most common weights matrix is a ***connectivity matrix*** (also called a contiguity matrix).
- Uses an indicator/dummy variable to measure connections between areas.
- The weights matrix is how we define "near" values.

Measuring Spatial Autocorrelation

$$w_{ij} = \begin{cases} 1, & if\ regions\ i\ and\ j\ share\ a\ boundary \\ 0, & otherwise \end{cases}$$

$$sim_{ij} = similarity\ between\ i\ and\ j$$

Measuring Spatial Autocorrelation

•

Neighborness	Similarity

$$\boxed{w_{ij}}\boxed{sim_{ij}}$$

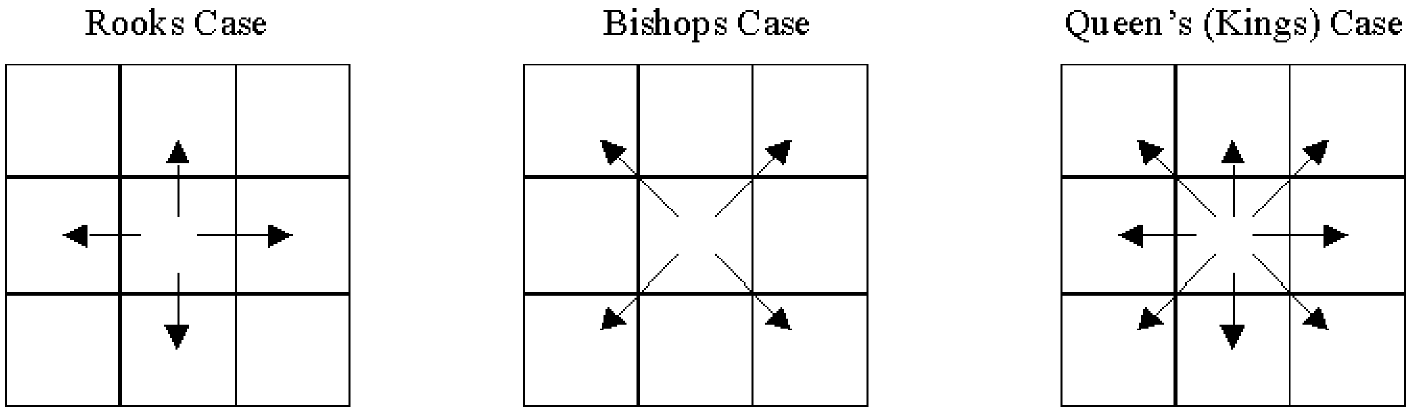
Spatial autocorrelation

Steps in determining the extent of spatial autocorrelation in your data:

1. Choose a neighbor criterion
   - Which areas are linked?
2. Assign weights to the areas that are linked
   - Create a spatial weights matrix
3. Calculate a statistic, using weights matrix, to examine spatial autocorrelation

Spatial weights matrices

- Neighbors can be defined by:
  - Contiguity (common boundary)
    - What is a “shared” boundary?
  - Distance (distance band, K-nearest neighbors)
    - How many “neighbors” to include, what distance do we use?
  - General weights (social distance, distance decay)



Common weights measures

- Most common is using *binary connectivity* based on contiguity
  - $w_{ij} = 1$ if regions i and j are contiguous, $w_{ij} = 0$ otherwise
- May also be defined as a function of the distance between i and j
  - Distance of the line connecting the centroids of two areas

STEP 1: CHOOSE A NEIGHBORHOOD CRITERION

Importing shapefiles into R and constructing neighborhood sets

relevant libraries:

```
> library(maptools)

> library(rgdal)

> library(spdep)
```

Importing a shapefile

```
> sids<-readShapePoly("path   to shapefile/
  shpfile.shp")

> class(sids)
[1] "SpatialPolygonsDataFrame"
attr(,"package")
> slotNames(sids)
[1] "data"         "polygons"    "plotOrder"   "bbox"
[5] "proj4string"
```

Projecting a shapefile

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

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

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

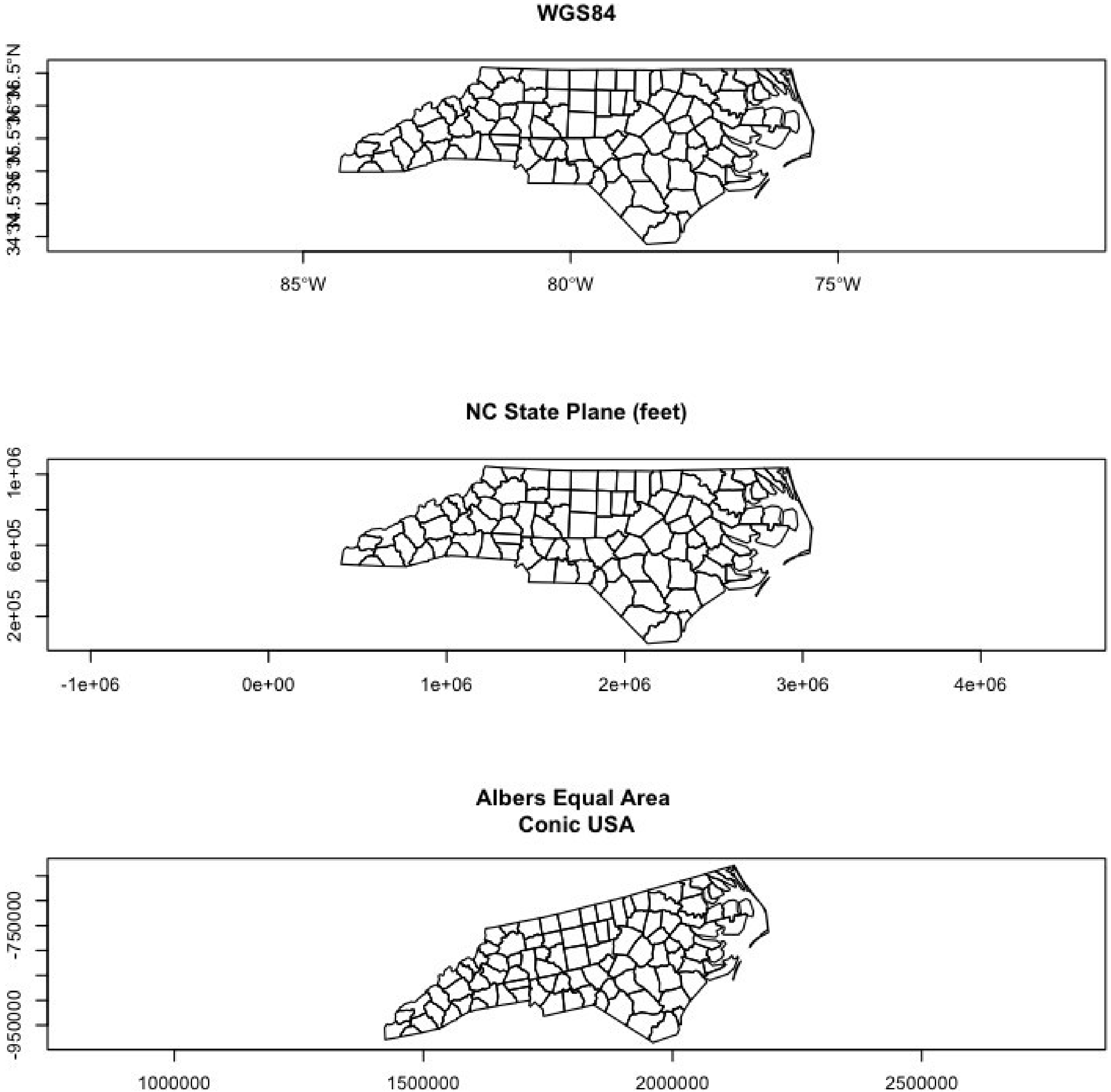
```
> library(rgdal)
> sids_Albers<-spTransform(sids,
 CRS("+init=epsg:2163")) #this    is clearly  wrong  in
  spatial  reference.
> sids_SP<-spTransform(sids,   CRS("+init=ESRI:102719"))
```

Projecting a Shapefile

- It is important to use an appropriate projection, especially when working at large scales (continental, global, etc.).

- If the file does have projection information you can import the shapefile with readOGR().

  - The result is the same but readOGR will read the projection information.




```
par(mfrow=c(3,1))
plot(sids, axes=T)
title("WGS84")
plot(sids_SP,
axes=T)
title("NC State
Plane (feet)")
plot(sids_Albers,
axes=T)
title("Albers
Equal Area
Conic USA")
```

Contiguity based neighbors

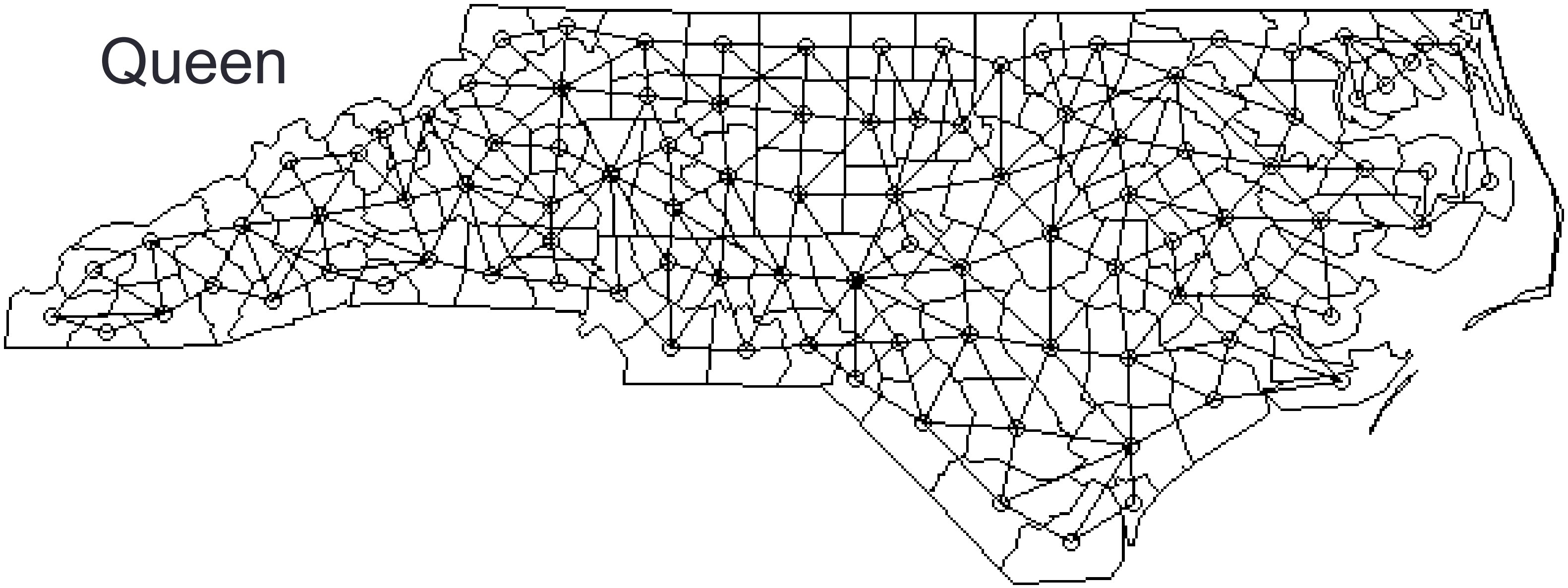
- Counties sharing any boundary point (QUEEN) are taken as neighbors, using the `poly2nb` function, which accepts a `SpatialPolygonsDataFrame`

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

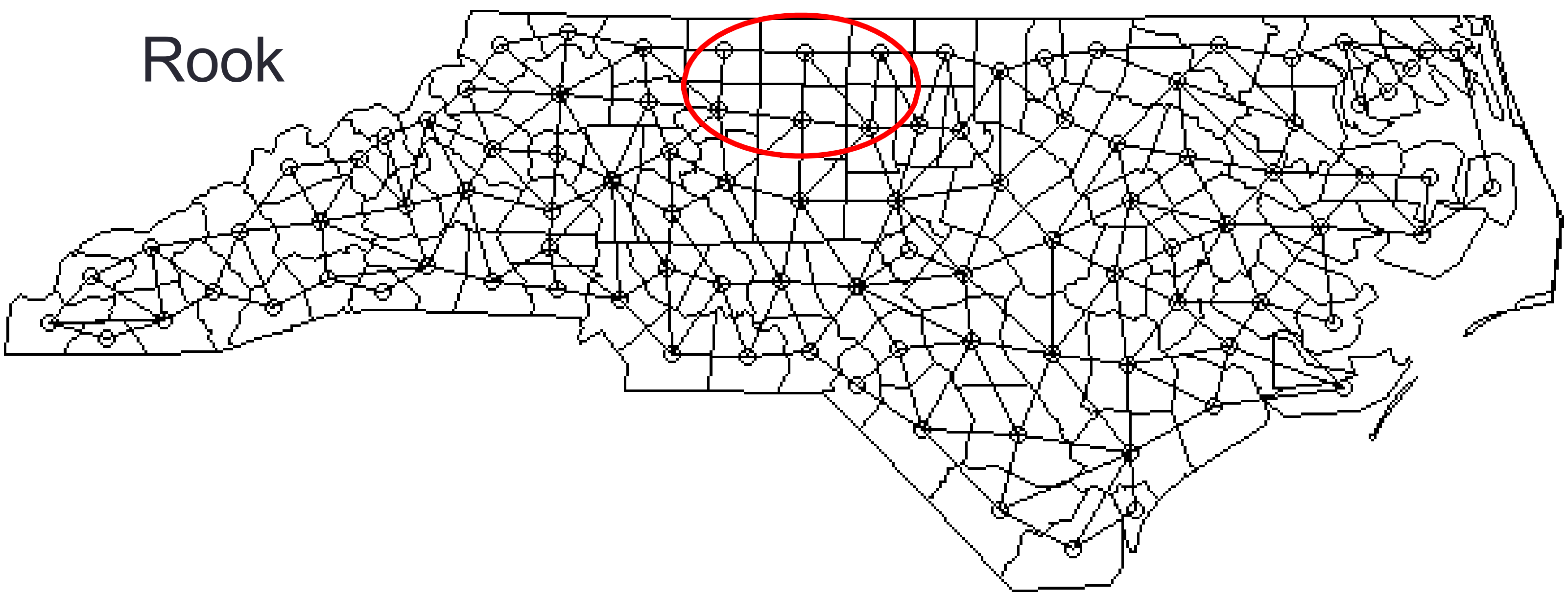
- If contiguity is defined as counties sharing more than one boundary point (ROOK), the `queen=` argument is set to FALSE

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

> coords<-coordinates(sids)
> dev.off()   #clears the screen and the panels
> plot(sids_nbq,   coords)
> plot(sids,   add= T) #Add the count outlines
```

Queen



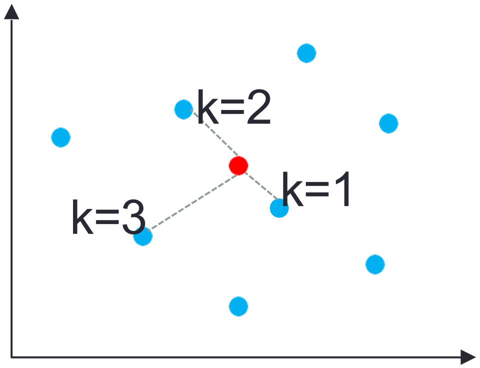
Rook



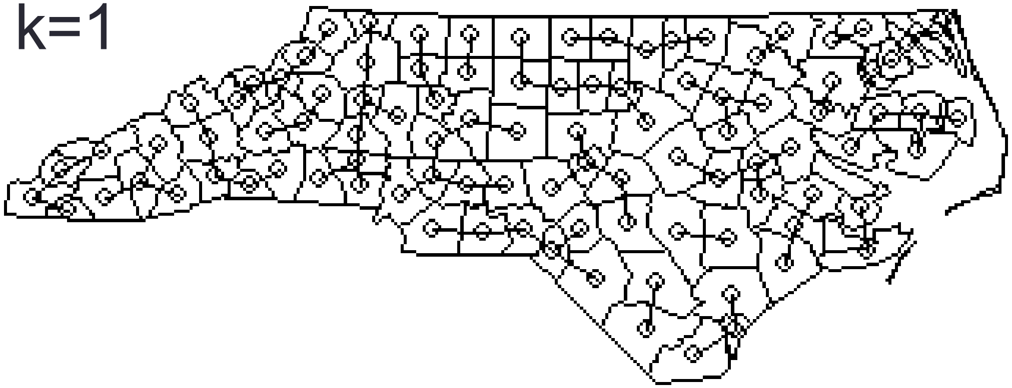
Distance based neighbors k nearest neighbors

- Can also choose the k nearest points as neighbors

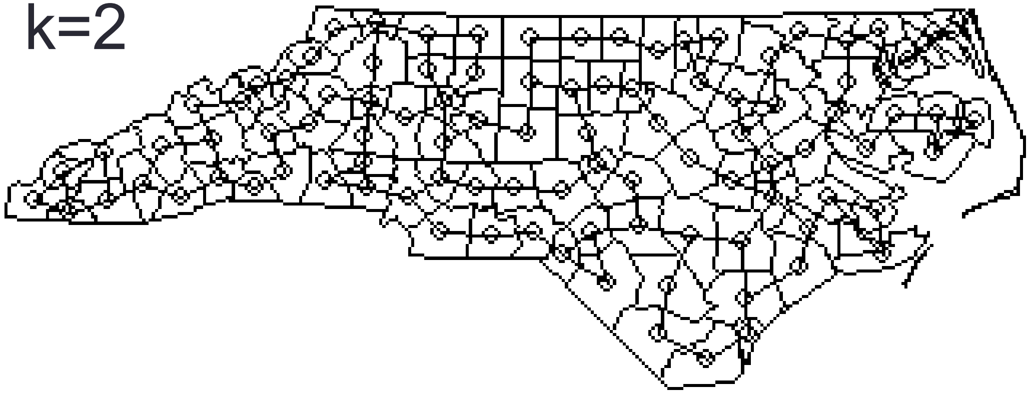
```
> coords<-coordinates(sids_SP)

> sids_kn1<-knn2nb(knearneigh(coords,   k=1))
> sids_kn2<-knn2nb(knearneigh(coords,   k=2))
> sids_kn4<-knn2nb(knearneigh(coords,   k=4))

> plot(sids_SP)
> plot(sids_kn2,   coords, add=T)
```



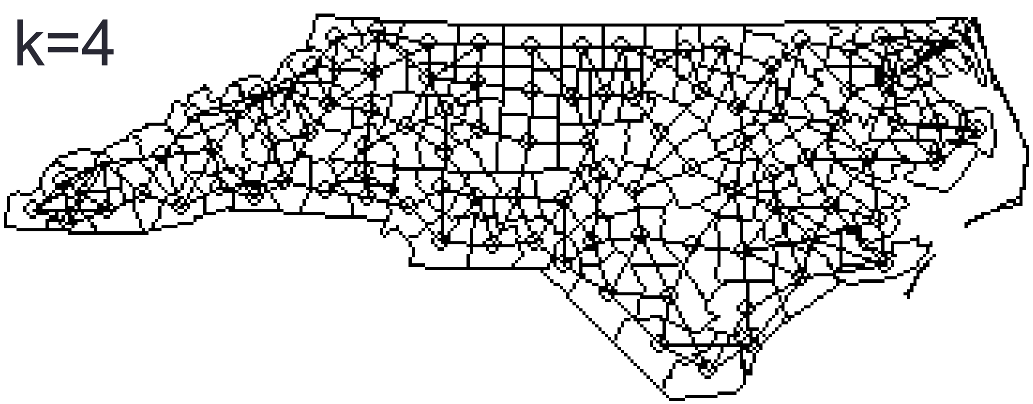
k=1



k=2



k=4



Distance based neighbors Specified distance

- Can also assign neighbors based on a specified distance

```
> dist<-unlist(nbdists(sids_kn1, coords))

#Notice that we are using the State Plane version so that the
 distances are easier to interpret

> summary(dist)
    Min. 1st Qu.    Median          Mean 3rd Qu.     Max.
  40100     89770   97640   96290  107200  134600
> max_k1<-max(dist)

> sids_kd1<-dnearneigh(coords, d1=0, d2=0.75*max_k1)
> sids_kd2<-dnearneigh(coords, d1=0, d2=1*max_k1)
> sids_kd3<-dnearneigh(coords, d1=0, d2=1.5*max_k1)
```

OR by raw distance

```
> sids_ran1<-dnearneigh(coords, d1=0, d2=134600)
```

STEP 2: ASSIGN WEIGHTS TO THE AREAS THAT ARE LINKED

Creating spatial weights matrices using neighborhood lists

Spatial weights matrices

- Once our list of neighbors has been created, we assign spatial weights to each relationship
  - Can be binary or variable
  - If we don't know much about the spatial process, try to stick with binary weights
- Even when the values are binary 0/1, the issue of what to do with no-neighbor observations arises
- Binary weighting will, for a target feature, assign a value of 1 to neighboring features and 0 to all other features
  - Used with fixed distance, k nearest neighbors, and contiguity

Row-standardized weights matrix

```
> sids_nbq_w<- nb2listw(sids_nbq)
> sids_nbq_w

Characteristics of weights list:
Neighbour list object:
Number of regions: 100
Number of nonzero links: 490
Percentage nonzero weights: 4.9
Average number of links: 4.9
…output deleted…
>
  sids_nbq_w$neighbours[1:3]
  #WHAT DOES THIS SHOW YOU?
>  sids_nbq_w$weights[1:3]
  #WHAT DOES THIS SHOW YOU?
```

- Row standardization is used to create proportional weights in cases where features have an unequal number of neighbors
  - Divide each neighbor weight for a feature by the sum of all neighbor weights
    - Obs i has 3 neighbors, each has a weight of 1/3
    - Obs j has 2 neighbors, each has a weight of 1/2

Binary weights

```
>  sids_nbq_wb<-
 nb2listw(sids_nbq, style="B")
> sids_nbq_wb

Characteristics of weights list:
Neighbour list object:
Number of regions: 100
Number of nonzero links: 490
Percentage nonzero weights: 4.9
Average number of links: 4.9

Weights style: B
Weights constants summary:
    n      nn      S0
           S1      S2
B 100 10000 490 980 10696
```

- Row-standardized weights increase the influence of links from observations with few neighbors
- Binary weights make interpretation more difficult…
- Standard to use row standardization.

Binary vs. row-standardized

- A binary weights matrix looks like:
  - Observation 1 has neighbor 2
  - Observation 2 has neighbors 3 and 4
  - Observation 3 has neighbors 1 and 2
  - Observation 4 has neighbor 2, 3 and 4

0	1	0	0
0	0	1	1
1	1	0	0
0	1	1	1

- A row-standardized matrix it looks like:

0	1	0	0
0	0	.5	.5
.5	.5	0	0
0	.33	.33	.33

- In practice R uses lists not matrices because the matrices have lots of zeros and take up a lot of space…

Regions with no neighbors

- If you ever get the following error:

```
Error in nb2listw(filename): Empty neighbor sets found
```

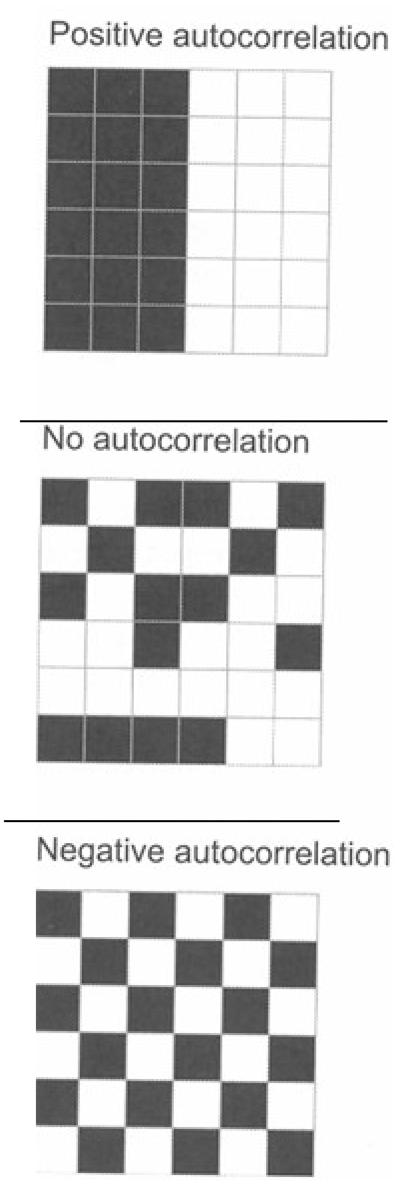
- You have some regions that have NO neighbors
  - This is most likely an artifact of your GIS data (digitizing errors, slivers, etc), which you should fix in a GIS
  - Also could have “true” islands (e.g., Hawaii, San Juans in WA)
    - May want to use k nearest neighbors
    - Or add `zero.policy=T` to the `nb2listw` call

```
> sids_nbq_w<-nb2listw(sids_nbq, zero.policy=T)
```

Weights based on IDW

```
> dist<-nbdists(sids_nbq,   coordinates(sids_SP))
> idw<-lapply(dist,   function(x) 1/(x/1000))

> sids_nbq_idwb<-nb2listw(sids_nbq,   glist=idw,
  style="B")

> summary(unlist(sids_nbq_idwb$weights))
    Min.            1st    Qu.            Median
       Mean         3rd    Qu.            Max. 0.004123
       0.006274     0.007640      0.008037
0.009268     0.024930
```

Join (or Joins or Joint) Count Statistic



Polygons only

binary (1,0) data only

- Polygon has or does not have a characteristic
- For example, a candidate won or lost an election

Based on examining polygons which share a border

- Do they have the same characteristic or not?

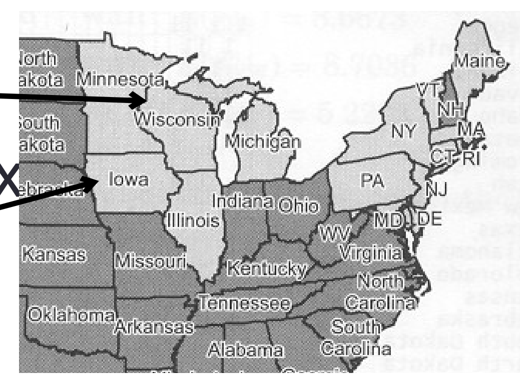
Border same

  on each side

Border <u>not</u> the same

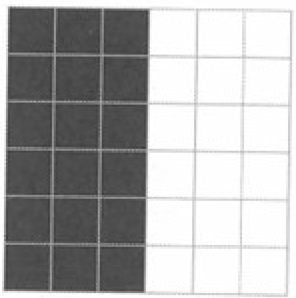
  on each side

Requires a contiguity matrix



Join (or Joint or Joins) Count Statistic

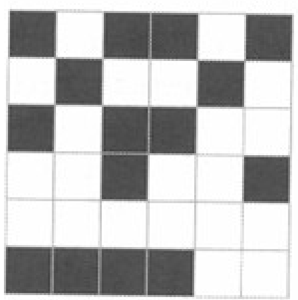
	Rook's case	Queen's case
Positive autocorrelation	$J_{BB} = 27$	$J_{BB} = 47$
	$J_{WW} = 27$	$J_{WW} = 47$
	$J_{BW} = 6$	$J_{BW} = 16$



Small number of BW joins (6 only for rook)
Large proportion of BB and WW joins

No autocorrelation

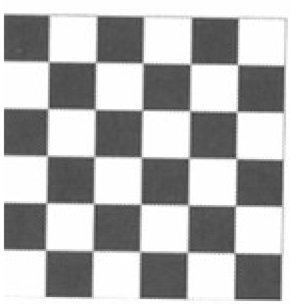
$J_{BB} = 6$	$J_{BB} = 14$
$J_{WW} = 19$	$J_{WW} = 40$
$J_{BW} = 35$	$J_{BW} = 56$



Different numbers of BW, BB and WW joins

Negative autocorrelation

$J_{BB} = 0$	$J_{BB} = 25$
$J_{WW} = 0$	$J_{WW} = 25$
$J_{BW} = 60$	$J_{BW} = 60$



Large number of BW joins
Small number of BB and WW joins

- Uses binary (1,0) data
  - Shown here as B/W (black/white)
- Measures the number of borders ("joins") of each type (1,1), (0,0), (1,0 or 0,1) relative to total number of borders
- For 6 x 6 matrix, border totals are:
  - 60 for Rook Case
  - 110 for Queen Case

Join Count: Test Statistic

Test Statistic given by: Z= $\dfrac{\text{Observed - Expected}}{\text{SD of Expected}}$

Expected = random pattern generated by tossing a coin in each cell.

Expected given by:

$$E(J_{BB}) = kp_B^2$$

$$E(J_{WW}) = kp_W^2$$

$$E(J_{BW}) = 2kp_Bp_W$$

Standard Deviation of Expected (standard error) given by:

$$E(s_{BB}) = \sqrt{kp_B^2 + 2mp_B^3 - (k+2m)p_B^4}$$

$$E(s_{WW}) = \sqrt{kp_W^2 + 2mp_W^3 - (k+2m)p_W^4}$$

$$E(s_{BW}) = \sqrt{2(k+m)p_Bp_W - 4(k+2m)p_B^2p_W^2}$$

Where: k is the total number of joins (neighbors)

p_B is the expected proportion Black, if random

p_W is the expected proportion White

m is calculated from k according to:

$$m = \frac{1}{2}\sum_{i=1}^{n} k_i(k_i - 1)$$

Note: the formulae given here are for free (normality) sampling. Those for non-free (randomization) sampling are substantially more complex. See Wong and Lee 1st ed. p. 151 compared to p. 155. Se next slide for explanation.

A Note on Sampling Assumptions:

applies to most tests for spatial autocorrelation

Test results depend on the assumption made regarding the type of sampling:

- *Free (or normality)* sampling
  - Analogous to sampling with replacement
  - After a polygon is selected for a sample, it is returned to the population set
  - The same polygon can occur more than one time in a sample
- *Non-free (or randomization)* sampling
  - Analogous to sampling without replacement
  - After a polygon is selected for a sample, it is not returned to the population set
  - The same polygon can occur only one time in a sample

The formulae used to calculate the test statistic (particularly the standard error) are different for each

- Generally, the formulae are substantially more complex for *free* sampling—unfortunately, it is also the more common situation!
- Assuming free sampling requires knowledge about larger trends from outside the region or access to additional information within the region in order to estimate parameters.

Gore/Bush Presidential Election 2000

Is there evidence of clustering by State?

Use Join Count to answer this question!

	Actual
Jbb	60
Jgg	21
Jbg	28
Total	109

Many BB joins

total number of joins = 109
= sum of neighbors/2 in the sparse contiguity matrix
= number of 1s/2 in the full contiguity matrix for US States

Sparse Contiguity Matrix for US States -- obtained from Anselin's web site (see powerpoint for link)										
Name	**Fips**	**Ncount**	**N1**	**N2**	**N3**	**N4**	**N5**	**N6**	**N7**	**N8**
Alabama	1	4	28	13	12	47				
Arizona	4	5	35	8	49	6	32			
Arkansas	5	6	22	28	48	47	40	29		
California	6	3	4	32	41					
Colorado	8	7	35	4	20	40	31	49	56	
Connecticut	9	3	44	36	25					
Delaware	10	3	24	42	34					
District of Columbia	11	2	51	24						
Florida	12	2	13	1						
Georgia	13	5	12	45	37	1	47			
Idaho	16	6	32	41	56	49	30	53		
Illinois	17	5	29	21	18	55	19			
Indiana	18	4	26	21	17	39				
Iowa	19	6	29	31	17	55	27	46		
Kansas	20	4	40	29	31	8				
Kentucky	21	7	47	29	18	39	54	51	17	
Louisiana	22	3	28	48	5					
Maine	23	1	33							
Maryland	24	5	51	10	54	42	11			
Massachusetts	25	5	44	9	36	50	33			
Michigan	26	3	18	39	55					
Minnesota	27	4	19	55	46	38				
Mississippi	28	4	22	5	1	47				
Missouri	29	8	5	40	17	21	47	20	19	31
Montana	30	4	16	56	38	46				
Nebraska	31	6	29	20	8	19	56	46		
Nevada	32	5	6	4	49	16	41			
New Hampshire	33	3	25	23	50					
New Jersey	34	3	10	36	42					
New Mexico	35	5	48	40	8	4	49			
New York	36	5	34	9	42	50	25			
North Carolina	37	4	45	13	47	51				
North Dakota	38	3	46	27	30					
Ohio	39	5	26	21	54	42	18			
Oklahoma	40	6	5	35	48	29	20	8		
Oregon	41	4	6	32	16	53				
Pennsylvania	42	6	24	54	10	39	36	34		
Rhode Island	44	2	25	9						
South Carolina	45	2	13	37						
South Dakota	46	6	56	27	19	31	38	30		
Tennessee	47	8	5	28	1	37	13	51	21	29
Texas	48	4	22	5	35	40				
Utah	49	6	4	8	35	56	32	16		
Vermont	50	3	36	25	33					
Virginia	51	6	47	37	24	54	11	21		
Washington	53	2	41	16						
West Virginia	54	5	51	21	24	39	42			
Wisconsin	55	4	26	17	19	27				
Wyoming	56	6	49	16	31	8	46	30		

Queens Case Sparse Contiguity Matrix for US States

- *Ncount* is the number of neighbors for each state
- Equals number of 1s in a row of full contiguity matrix
- Sum of Ncount is 218
- Number of common borders (joins) = $\sum$ ncount / 2 = 109

- *N1, N2…* FIPS codes for neighbors

Join Count Statistic for Gore/Bush 2000 by State

% of Votes	
in election	
Bush % (Pb)	0.49885
Gore % (Pg)	0.50115

$$E(J_{\mathrm{BB}}) = kp_B^2$$

	Actual	Expected	Stan Dev	Z-score
Jbb	60	27.125	8.667	3.7930
Jgg	21	27.375	8.704	-0.7325
Jbg	28	54.500	5.220	-5.0763
Total	109	109.000		

- The expected number of joins is calculated based on the proportion of votes each received in the election (for Bush = 109*.499*.499=27.125)
- K = 109= total number of joins
- There are far <u>more</u> Bush/Bush joins (actual = 60) than would be expected (27)
  - Since test score (3.79) is greater than the critical value (2.54 at 1%) result is statistically significant at the 99% confidence level (p <= 0.01)
  - Strong evidence of spatial autocorrelation—clustering
- There are far <u>fewer</u> Bush/Gore joins (actual = 28) than would be expected (54)
  - Since test score (-5.07) is greater than the critical value (2.54 at 1%) result is statistically significant at 99% confidence level (p <= 0.01)
  - Again, strong evidence of spatial autocorrelation—clustering

– Actual calculations available in *spatstat.xls* spreadsheet (*JC-%vote* tab)

STEP 3: EXAMINE SPATIAL AUTOCORRELATION

Using spatial weights matrices, run statistical tests of spatial autocorrelation

Moran's I

The most common measure of Spatial Autocorrelation

Use for points <u>or</u> polygons

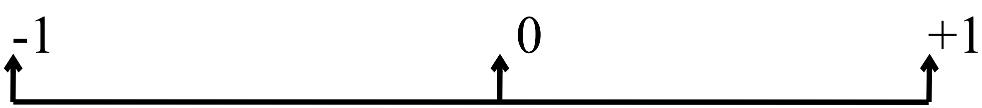
Join Count statistic only for polygons

Use for a continuous variable (any value)

Join Count statistic only for binary variable (1,0)

Varies on a scale between –1 through 0* to + 1

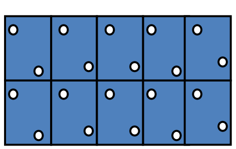
technically it is: –1/(n-1)

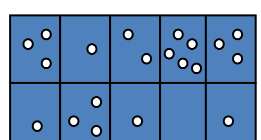


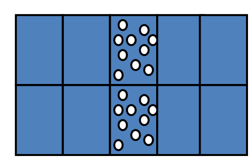
high <u>negative</u> spatial autocorrelation	<u>no</u> spatial autocorrelation*	high <u>positive</u> spatial autocorrelation

Can also use it as an index for dispersion/random/cluster patterns.

Dispersed Pattern	Random Pattern	Clustered Pattern



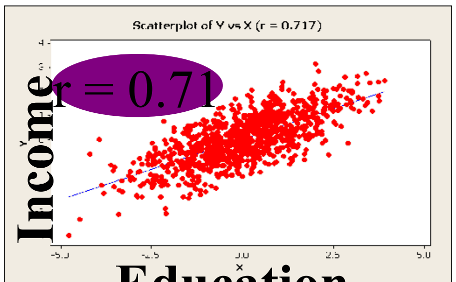




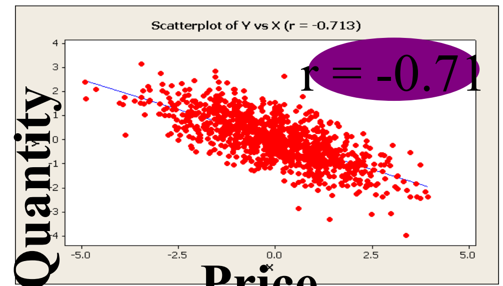
Moran's *I* and Correlation Coefficient *r*
Differences and Similarities

Correlation Coefficient *r*

Relationship between <u>two</u> variables

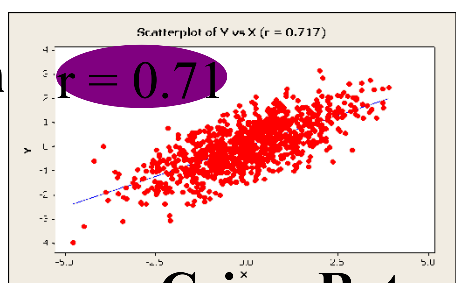


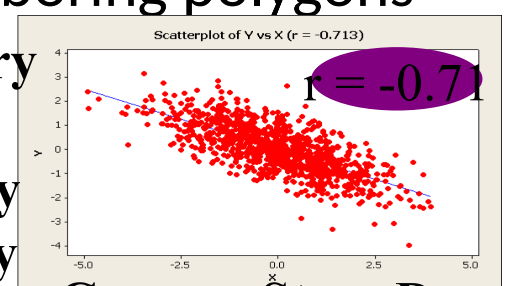
or



Moran's I

- Involves <u>one</u> variable only
- Correlation between variable, X, and the "spatial lag" of X formed by averaging all the values of X for the neighboring polygons





Moran's I

- Moran's I statistic measures spatial autocorrelation.
- Moran's I is on the same scale as the correlation coefficient.
- The Moran's I equation can be simplified to something similar to the correlation coefficient.

Moran’s I

- Similar to correlation: avg of the product of the Z-scores
- -- or ++ z-score pairs contribute to a positive correlation coefficient.
- -+ or +- z-score pairs contribute to a negative correlation coefficient.
- Different from correlation in that it only considers neighbors.

The correlation coefficient

$$r = \frac{1}{n}\sum_{i=1}^{n}\left(\boxed{\frac{X_i - \bar{X}}{s_x}} * \frac{Y_i - \bar{Y}}{s_y}\right)$$

What is this???

Moran's I, simplified

$$z = \frac{(Y_i - \bar{y})}{SD_y}$$

$$I_i = \frac{Y_i - \bar{Y}}{sd_y} \sum_{j=1}^{N} w_{ij} \frac{Y_j - \bar{Y}}{sd_y}$$

Moran's *I* vs. Pearson's *r*

Need a weights matrix

Pearson correlation coefficient

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

feasible range: -1 to +1

Moran's *I* coefficient

$$I = \frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(x_i - \bar{x})(x_j - \bar{x})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{j=1}^{n}(x_j - \bar{x})^2}}$$

feasible range: -1 to +1
(sort of)

Moran’s I

- High values near low values will lead to negative Moran’s I statistic.

- High values near high values will lead to positive Moran’s I statistic.

- When there is no pattern $E(I) = 0$.

Correlation is the …

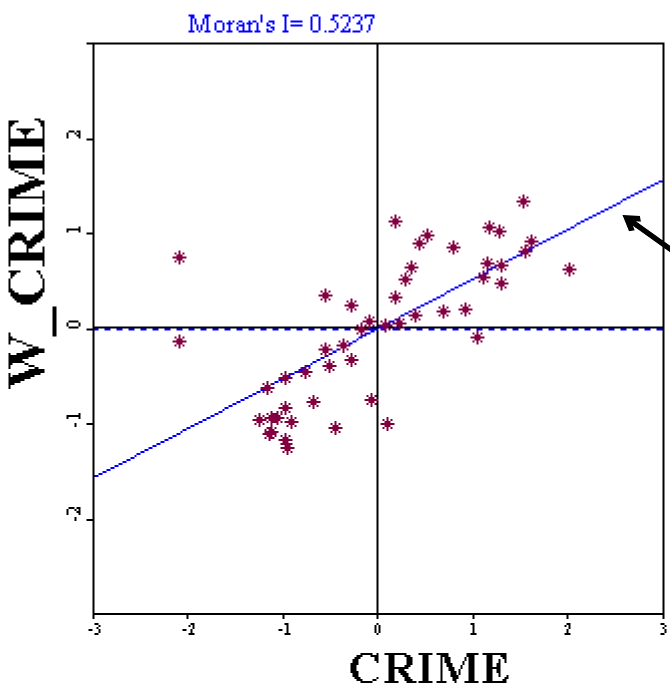
- The Pearson correlation is simply the slope of a regression line through a scatterplot of two variables.

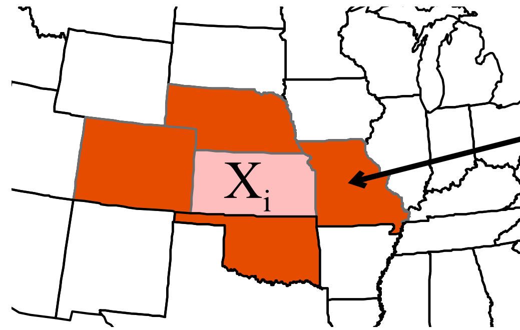
- Conceptually Moran's I is simply the correlation between each observation and its neighbors.
  - We can calculate Moran's I by fitting a regression through a scatter plot of each value and the weighted average of its neighbors.
  - We get the neighbors (and their weights) from the weights matrix.

- A "lagged variable":
  - The lag of variable X for observation *i* is $w_{ij}X_i$

Moran Scatter Plots

Moran's I can be interpreted as the correlation between variable, X, and the "spatial lag" of X formed by averaging all the values of X for the neighboring polygons
We can then draw a scatter diagram between these two variables (in standardized form): **X**   and   **lag-X** (or W_X)





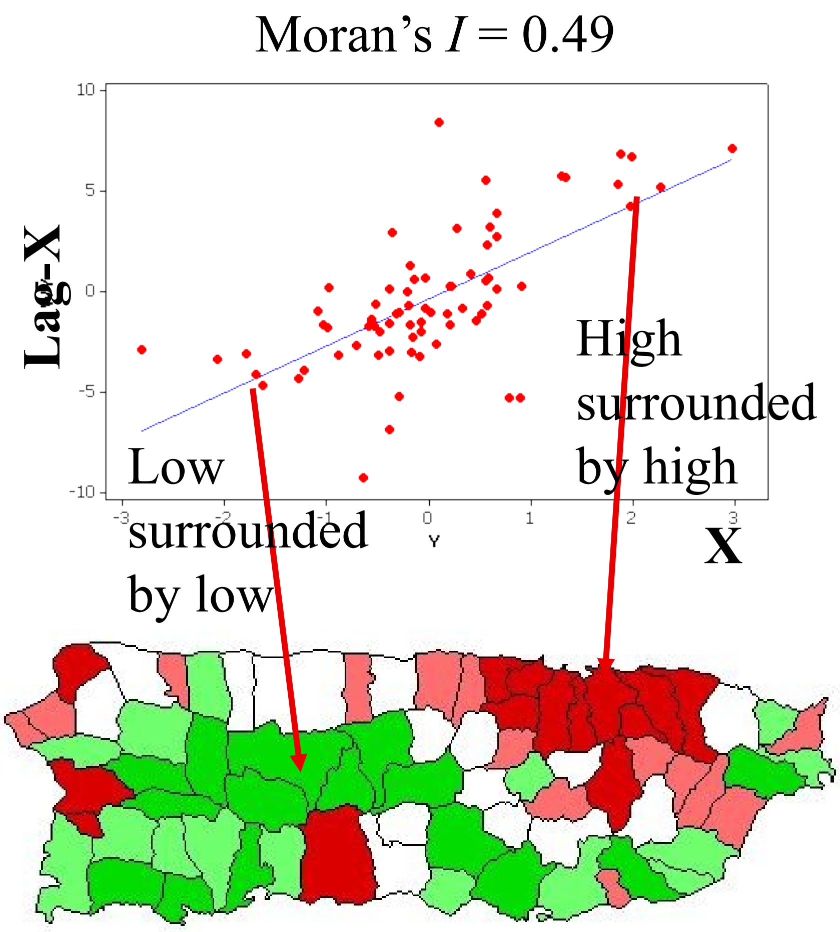
Lag X_i
is average of these

Least squares "best fit" line to the points.
The <u>slope</u> of this *regression line* is Moran's I
(will discuss Regression later)

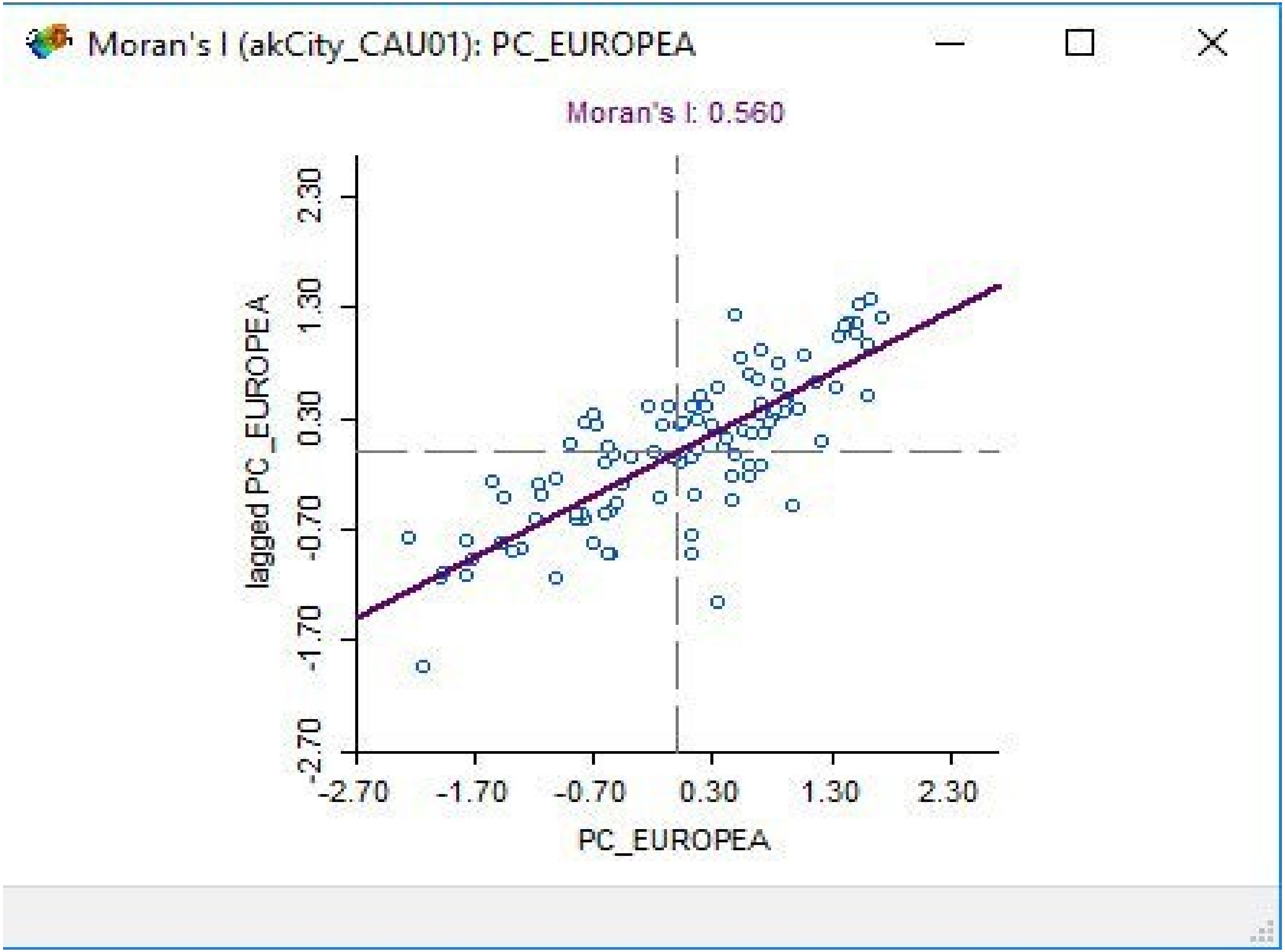
Moran Scatterplot: example

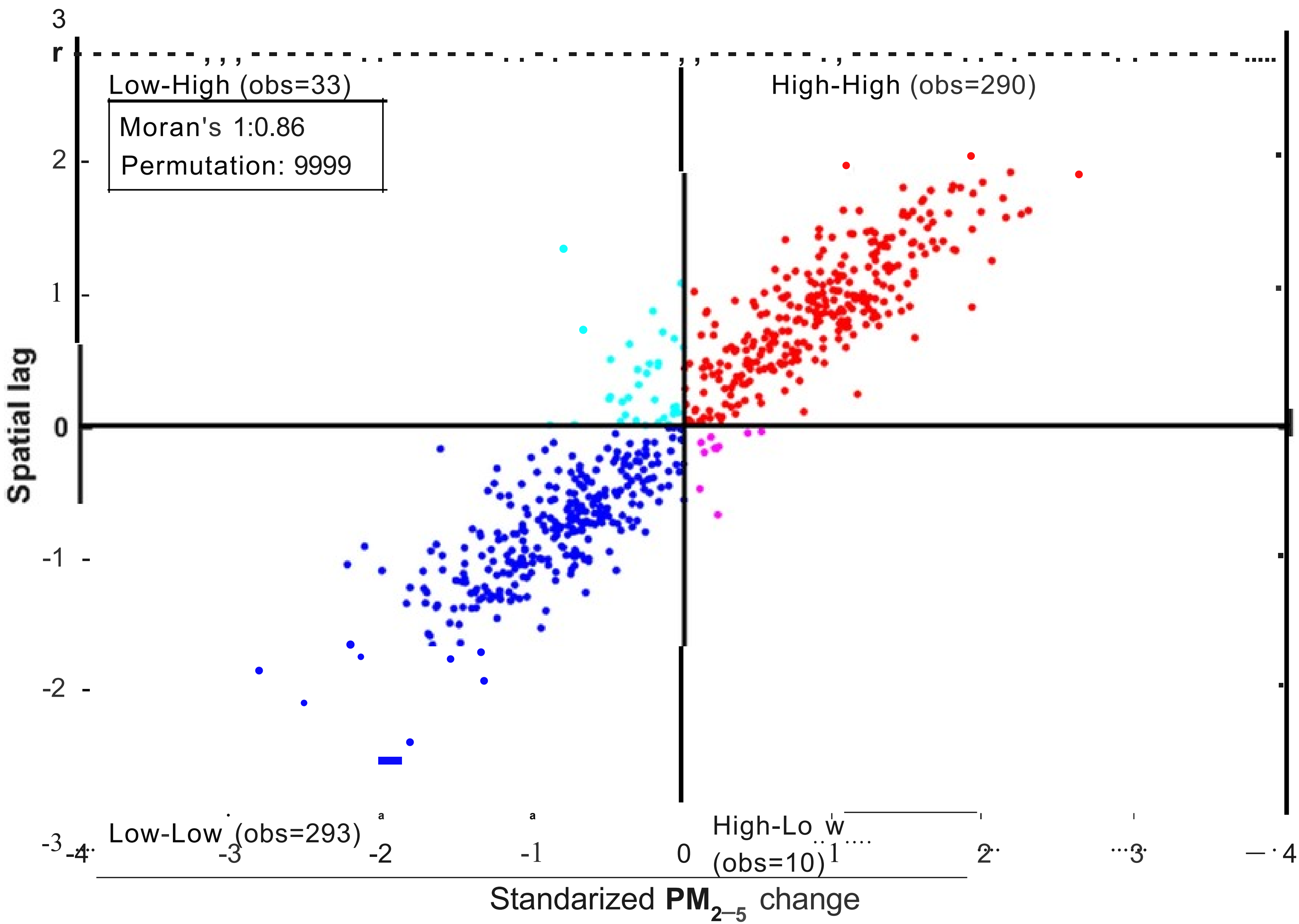
Scatterplot of **X** vs. **Lag-X**

The slope of the regression line is Moran's *I*



Population density
in Puerto Rico


Moran's I (akCity_CAU01): PC_EUROPEA
Moran's I: 0.560
lagged PC_EUROPEA
2.30
1.30
0.30
-0.70
-1.70
-2.70
-2.70
-1.70
-0.70
0.30
1.30
2.30
PC_EUROPEA


Low-High (obs=33)
Moran's 1:0.86
Permutation: 9999
High-High (obs=290)
Low-Low (obs=293)
High-Low (obs=10)
Spatial lag
Standarized PM2–5 change
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3
4

To calculate Moran's I

- We need:
  - A variable of interest
  - A new “lagged” variable which measures the same thing for each observations neighbors.
- We need to look at the correlation between these variables.
- **THIS MEASURES SPATIAL ASSOCIATION**

```
> sids$NWBIR79 #births to non-white mothers
> sids$NWBIR79_lag <- lag(sids_nbq_w, sids$NWBIR79) #lag using row standardized weights
> lm1 <-
  lm(sids$NWBIR79_lag~sids$NWBIR79)
  >lm1
Call:
lm(formula = sids$NWBIR79_lag ~ sids$NWBIR79)
Coefficients:
 (Intercept)   sids$NWBIR79
   1205.9441        0.1497
> plot(y= sids$NWBIR79_lag, x= sids$NWBIR79)
> title("The relationship between a variable \n and the lag of itself is moran’s I")
```

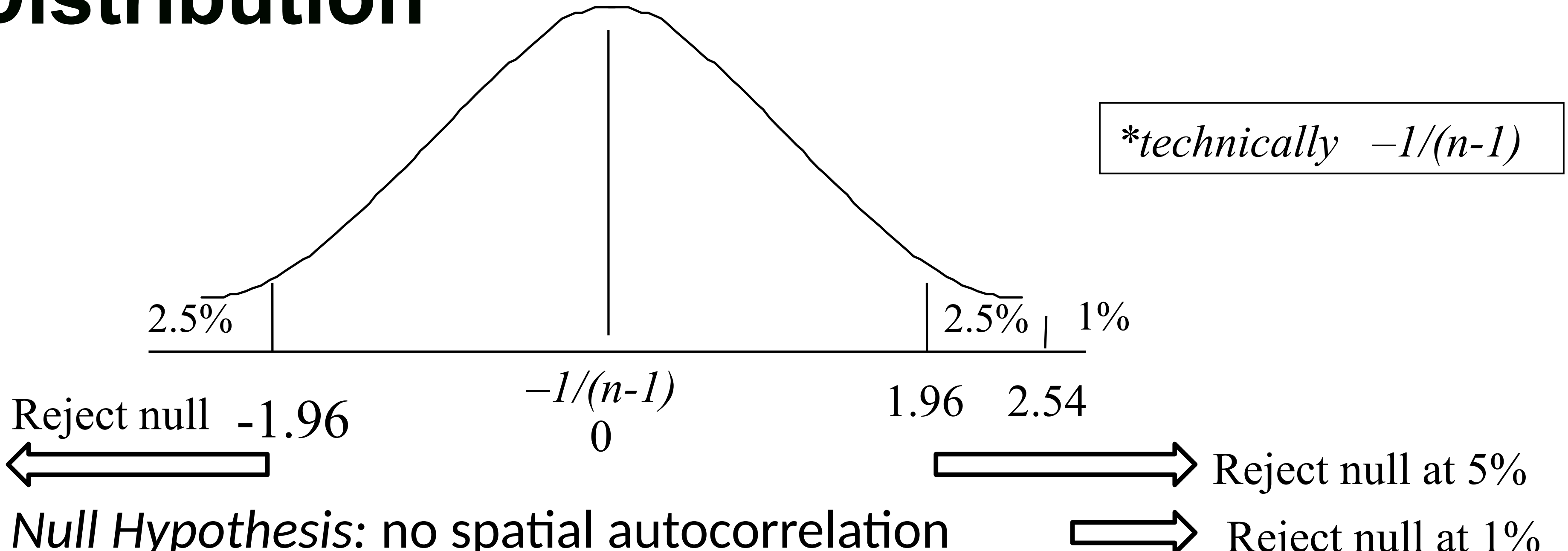
Moran's I is about the relationship between a variable and its lag…

```
>lm
1
Call:
lm(fo
rmula
= sids$
NWBI
R79_l
ag ~
sids$
NWBI
R79)
```

```
Coefficients:
 (Intercept)   sids$NWBIR79
    105.9441         0.1497
> moran.test(sids$NWBIR79, sids_nbq_w)
          Moran's I test under randomisation

data:  sids$NWBIR79
weights: sids_nbq_w

Moran I statistic standard deviate = 2.6131, p-value = 0.004487
alternative hypothesis: greater
sample estimates:
 Moran I statistic       Expectation          Variance
       0.149686037      -0.010101010       0.003739254
```

Notice that the regression coefficient is the Moran's I

Test Statistic for Normal Frequency Distribution



Null Hypothesis: no spatial autocorrelation

*Moran's I = 0

Alternative Hypothesis: spatial autocorrelation exists

*Moran's I > 0

Reject *Null Hypothesis* if Z test statistic > 1.96 (or < -1.96)

---less than a 5% chance that, in the population, there is no

spatial autocorrelation

Hypothesis Tests

Expected Moran's *I* just based on the *n* of observations, approaching 0 as *n* increases: $E(I) = -1/(n-1)$

O>E= clustering, E>O= dispersion

under assumptions of near normality, can use Z-scores to test significance

$Z^* = I - E(I)/sd$, where sd=sqrt of var(I)

I from 30 cities precip, exp I, var .00199

eg: .0822-(-.034)/.0446=2.605 (reject H_0)