Modifiable Areal Unit Problem (MAUP)

More stuff I think about with respect to spatial binning

Sarah Battersby
Tableau Research
Spatial Gloom-And-Doom Spreader

Modifiable Areal Unit Problem

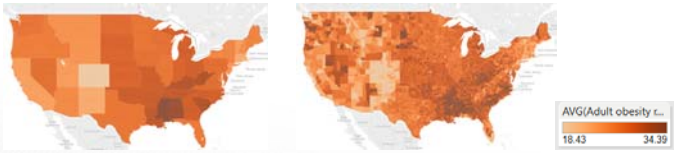
"a problem arising from the imposition of artificial units of spatial reporting on continuous geographical phenomena resulting in the generation of artificial spatial patterns" (Heywood, 1988)

Essentially – you need to think carefully about how your data will be represented when grouped into bins for analysis

Two sorts of MAUP

Scale MAUP

The scale at which you aggregate may have impact
(e.g., state vs. county vs. tract vs. block)



Zone MAUP (what I'm looking at right now…)

Adjustments to the boundaries into different zones may have impact

A simple example

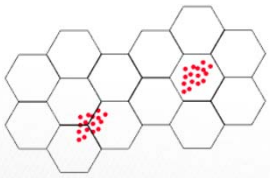
Data

- Two clusters of points
(let's just pretend we did some stats and can say they are "significantly clustered")

A simple example

Data

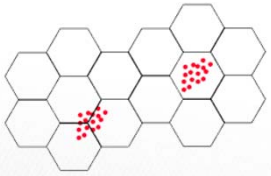
- Two clusters of points
  (let's just pretend we did some stats and can say they are "significantly clustered")
- Throw some bins on top of the data… aggregate

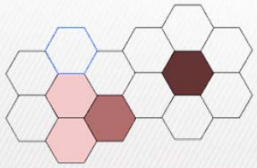


A simple example

Data

- Two clusters of points
  (let's just pretend we did some stats and can say they are "significantly clustered")
- Throw some bins on top of the data… aggregate

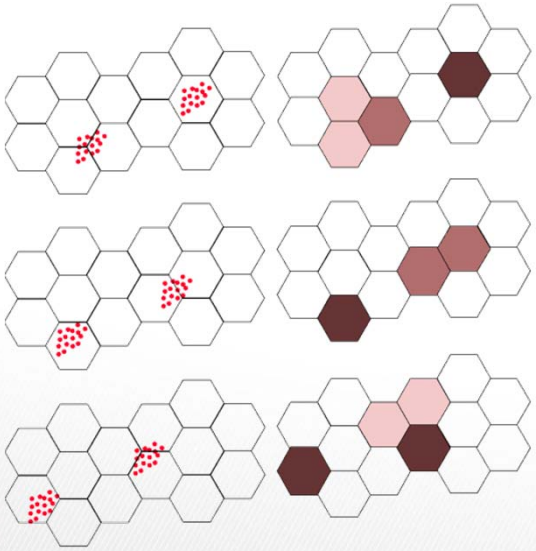




A simple example

But what impact does the bin placement have on the resulting spatial pattern

(for visual analysis, or for *spatial analytics*)???



My arena for exploration

USDA Food Deserts

> "Food deserts are defined as urban neighborhoods and rural towns without ready access to fresh, healthy, and affordable food."

http://apps.ams.usda.gov/fooddeserts/fooddeserts.aspx

How do you find a food desert?

Census tract-based. The tract has to be:

- Low income
  - Poverty rate > 20%, OR
  - Median family income at or below 80% of area mean
- Low access
  - 500+ persons and/or at least 33% of the census tract's population lives more than 1 mile from a supermarket or large grocery store (urban)
  - 10 miles (rural)

Access is an interesting issue…

How do you figure out how far every person is from a store??

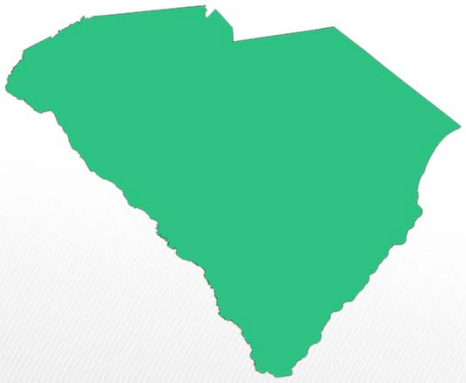
In a sort of roundabout way

1. Get population data (block level)
2. Turn this into a population grid (1/2 km grid cells)
   1. Make a grid
   2. Crack blocks with the grid
   3. Allocate population by area
   4. Dissolve back into the grid
3. Measure distance (straight line) center of grid cell to nearest store
4. Magic to turn this back into an access measure for the tracts

Get some data

Pick a state

to make the process of testing this a little less painful than doing this for the whole US…

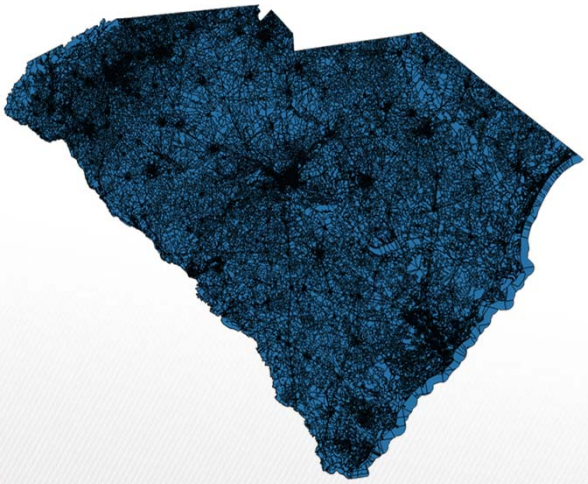


Get some data

Census blocks

With pop.

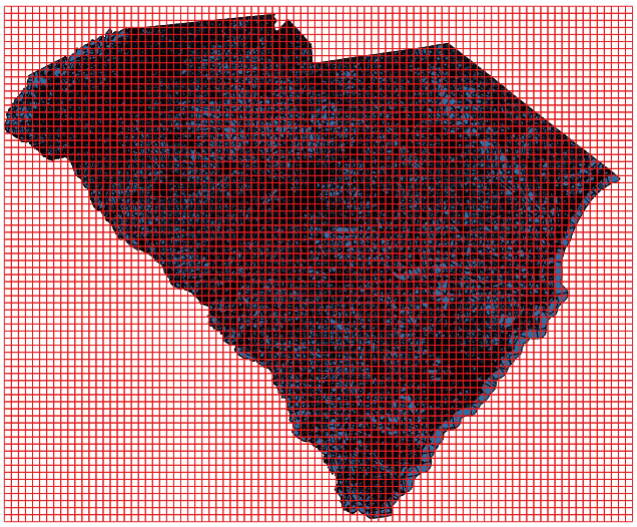
181,908 polys



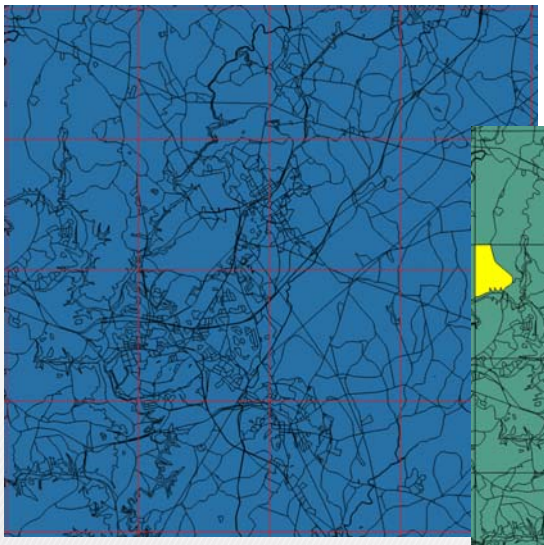
Make a grid (or, a bunch of grids)

This is 5km grid

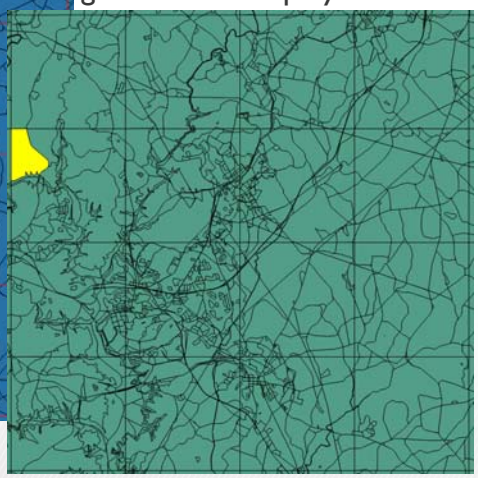
I really used 0.5km



Intersect the grid and blocks…



With the 0.5km grid this gives me ~1mil polys

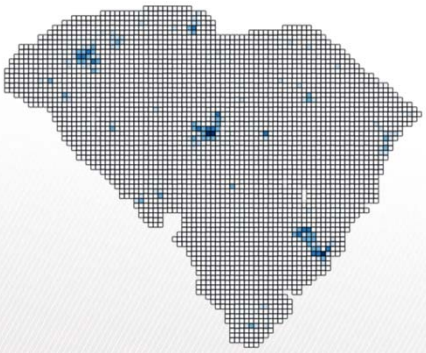


Allocate population, make back into a grid

Re-allocate the population for each little bit

- (Area of new polygon / Area of original tract) * population
- Assume population is evenly distributed and just assign based on percent of the tract that each new polygon covers

Merge back into one big grid



But, what about our friend MAUP?

I want to make a bunch of these grids…and look at how the population "shifts" based on the grid placement!

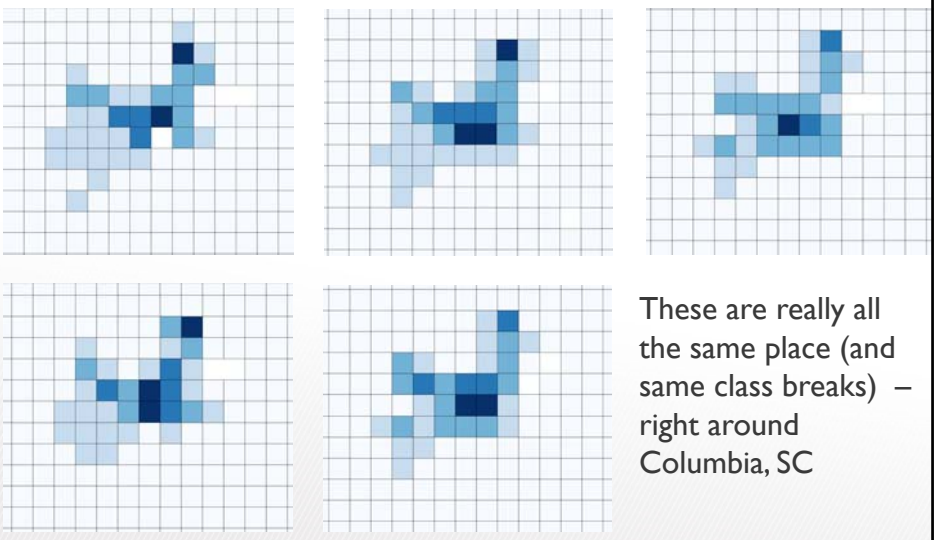
And I want you to help me figure out how to do this the most efficient way possible.

My current strategy:

- Python + QGIS to create grids with a random origin
- Same script cracks the geometries and calculates the population
- Import the new shapefiles into PostgreSQL
- Sum population by grid cell, join back into the original grid

Seemed like a fine strategy with 5 grids to test this, but (yawn) too many steps for the, say, 100 I want

There are actually some cool looking differences



These are really all the same place (and same class breaks) – right around Columbia, SC

The sticking points…Too much wasted time

I let my script run in QGIS for a few hours

Then I wander back over and load the data into PostgreSQL

Then I wander away and come back when it's imported

Then I merge the data back into grids, which takes about a nano-second, but I manually change the input tables ☺

How can I put this all together into one happy place?

Stuff I tried

Getting someone else to do this

Intersecting the geometries in PostgreSQL. I thought I might die waiting for it to finish. QGIS was way better.

Making the summary table in QGIS (instead of PostgreSQL) – totally stumped me

Files if anyone wants them for anything…

MAUP_talk_QGIS.py

- The script I run in QGIS to make all of my grids (currently set to a spacing of 0.5km and makes 5 grids)

Blocks came from the Census

Projection is Albers Equal Area Conic (EPSG: 102008)

I'll probably start varying this with the grid placement at some point