Determining County Area Using Python

Programming for Geospatial Science and Technology

GISC 3200K

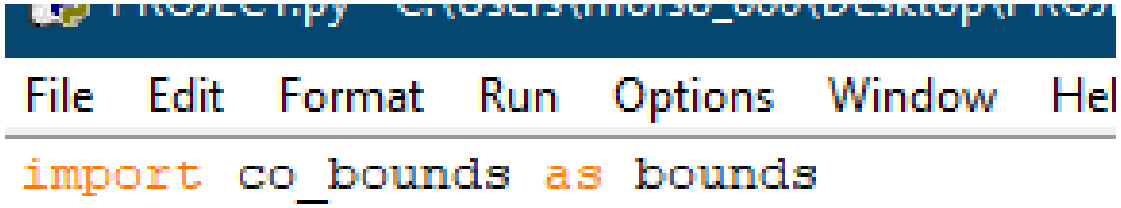
Tyler Henderson

Finding a Method

- I used google and typed in “How to find the area of a irregular polygon” and found that you can use what is known as the Shoelace Formula to calculate this type of area from an article on Wikipedia. (https://en.wikipedia.org/wiki/Shoelace_formula) I watched a youtube video about the formula as well.
- The rest of the methodology for creating the script was taken from past exercises in class.

First Steps

- First we have to get the module imported.



- Then we create a way to present the data once a function is created.

```
for i in bounds.co_bounds:
    counties = bounds.co_bounds[i]
    print(i,":",countyarea(counties))
```

Creating the Function

- You start off by defining the function.

```
def countyarea(counties):
```

- Then figure out what kind of math you need. The formula is actually pretty simple (almost like the accumulation functions we did in early exercises)

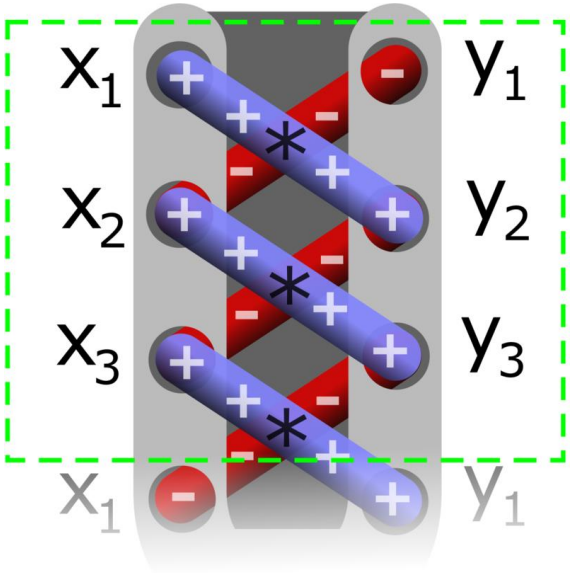
The formula can be represented by the expression

$$\mathbf{A} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$= \frac{1}{2}\left|x_1 y_2 + x_2 y_3 + \cdots + x_{n-1} y_n + x_n y_1 - x_2 y_1 - x_3 y_2 - \cdots - x_n y_{n-1} - x_1 y_n\right|$$

where

- A is the area of the polygon,
- n is the number of sides of the polygon, and
- (x_i, y_i), $i = 1, 2, \ldots, n$ are the ordered vertices (or "corners") of the polygon.



- Set values to 0 (because it's summation) INSIDE THE FUNCTION
- Unpack the individual tuples for x and y and the next x and y
- Define first and last coordinates for purposes of the formula. (It loops back around.)
- Write the equation INSIDE the for loop.
- Area is in absolute value because some values will turn out negative, there's no negative areas.

```
def countyarea(counties):
    value1 = 0
    value2 = 0
    n = len(counties)
    for p in range (0,n-1):
        x1, y1 = counties[p]
        x2, y2 = counties[p + 1]
        finx1, finy1 = counties[0]
        finx2, finy2 = counties[n-1]

        value1 = value1 + x1 * y2
        value2 = value2 + x1 * x2
    value1 = value1 + finx2 * finy1
    value2 = value2 + finx1 * finy2

    area = abs(value1 - value2) / 2

    return area
```

- My output looked something like this. I had no idea about the units so I couldn’t check myself really.
- I think my math may be a bit off so I’ll tweak my code before final submission. (The shoelace formula requires counterclockwise cartesian points, and I’m not entirely sure the co_bounds module was set up that way.)

```
Jefferson : 14954213.757761145
Seminole : 10973504.689542627
Camden : 20209393.339191124
Glynn : 10590025.21455178
Polk : 2977523.4160943325
Morgan : 20143662.663310718
Talbot : 8768235.018441077
Schley : 2473234.3055490814
Union : 13058864.020841584
Wilcox : 10449585.139939696
Ware : 12995810.1892654
Clarke : 8142169.3365723975
Jones : 6814893.924603334
Rockdale : 13438171.594187248
Dade : 2639179.719817712
Murray : 24763148.22222629
Cobb : 10438037.031688454
Columbia : 8268738.108358417
Forsyth : 15400687.823691443
Baldwin : 16875350.121892266
Walker : 5843028.096434588
Emanuel : 29936769.307773698
Gilmer : 9556296.727900796
Warren : 28391780.58112532
Elbert : 15060356.925295576
Clayton : 11410265.441911645
Brantley : 19213790.849161785
Chatham : 11529596.11331167
Peach : 8613396.521782503
Echols : 11741519.898832738
Evans : 12343932.852545975
Quitman : 2935172.9086514153
Irwin : 8102242.748633424
Burke : 25580827.146184884
Henry : 20303848.31331513
Liberty : 21210609.59619119
Madison : 16172640.773686212
Toombs : 19503334.38241914
Monroe : 6412560.92977823
```

If you somewhat enjoyed this class

- Consider taking app dev. I signed up for it.
- It could potentially help in higher skilled jobs and will especially help if you want to pursue research of any kind, because a lot of research institutions won’t shell out money for ArcGIS licenses and you have to create your own tools for analysis.