Project 1 Report

Problem

As stated in the project problem statement, “Recently a possible cancer risk in adults from nitrate (and nitrite) has emerged, but the magnitude of the risk is unknown.” The purpose of this project is to try and determine the scale of this problem by analyzing nitrate levels at water well locations and compare this information with the distribution of cancer occurrence based on Wisconsin census tracts.

Project Plan - Analysis

Since the data required for this project was readily available, I was able to quickly move along to the implementation of a solution. The analysis portion of this project basically consists of 4 major steps:

Step 1: Run Inverse Distance Weighted Interpolation on the well nitrate data to approximate nitrate levels in areas around the wells.

Step 2: Run zonal statistics on the output of the IDW to associate a unique nitrate level to the census tracts with cancer rates.

Step 3: Perform regression analysis to find the relationship between the nitrate value and cancer rates using Geographically Weighted Regression*.

Step 4: Run Moran’s I Spatial autocorrelation tool to assess the significance of the output.

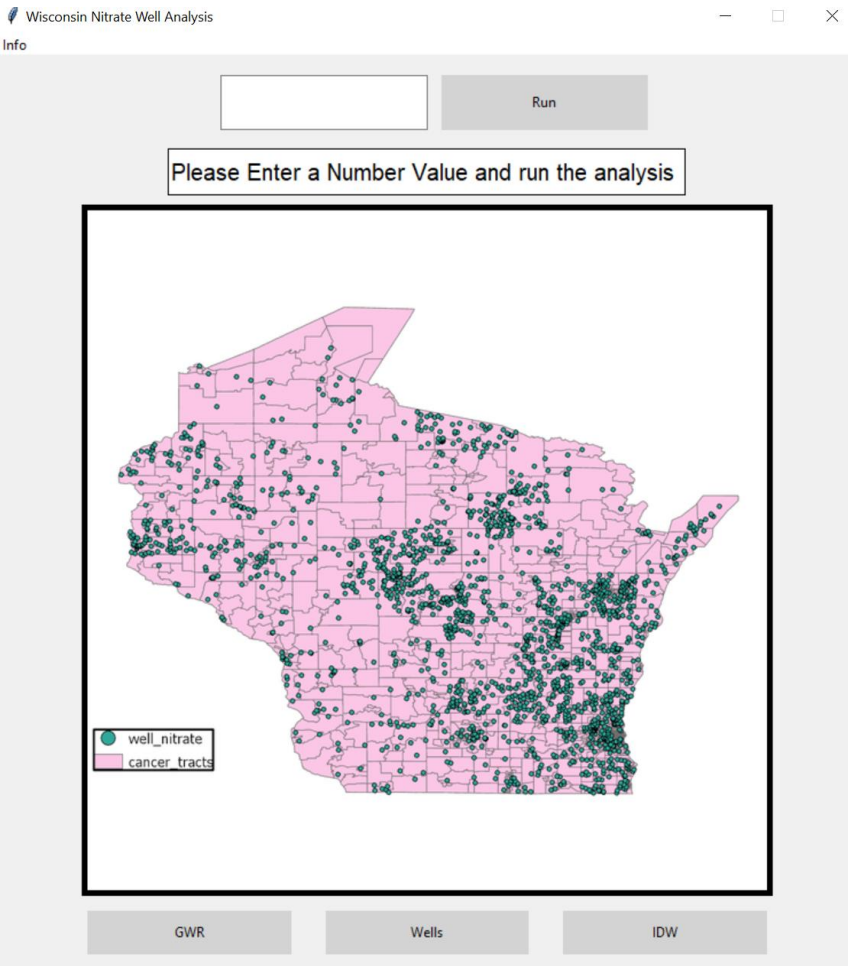
*Note: the project calls for using Ordinary Least Squares, but due to issues with the software that I couldn’t resolve GWR was chosen as a secondary solution.

When calculating the IDW the changing the K decay coefficient allowed me to test different values to see how it changed the output. All of this I accomplished using arcpy in conjunction with ArcGIS Pro.

Project Plan – Graphical User Interface

To facilitate easier analysis, I decided to create a simple GUI in python using tkinter to allow me to input different K values and view the output maps and statistical reports. This approach presented several challenges that I needed to overcome to be successful. Some of these had straight forward solution, others less so, as detailed below.

The layout of the tool needed to be relatively simple, but intuitive enough that users could figure out what to do with minimal instruction. To that end I opted for an approach that allows the user to read the GUI from the top down, as can be seen below:



This layout encourages users to first enter a K value and start running the analysis. They can then follow along with the analysis as the GUI displays update messages. Their eye then goes down to the image, with initially shows the location of the wells against the census tracts layer. After the analysis completes this will automatically update to show the GWR output, and the update message will encourage them to switch between images by clicking the buttons along the bottom. If users click the dropdown in the top right corner, they are greeted with options to open popup dialog boxes describing IDW and GWR, as well as an about box explaining the purpose of the application.

Considerations

There are several ways in which this GUI could be insufficient if the right processes and protections were not put into place, here are a few that I thought were more interesting:

-Since I didn't want users to try and switch images before running the analysis, if they click on the GWR or IDW buttons before images have been created, they're shown a message encouraging them to run the analysis first.

-K values in the IDW are not valid below 0 and are increasingly less valuable as you go higher, with an ideal somewhere between 0.5 and 3. Since I wanted the ability to play with the data I opted to allow users to input values between 1 and 10, but included verbiage in the IDW popup encouraging them to use values in that ideal range. I also needed to ensure that the value could be either a float or an integer, and that it was numerical, so that no errors would be introduced while the analysis was running.

-Clicking on the run button multiple times in succession led to python trying to run multiple instances of the analysis function at once, leading to some errors and freezing the program. To prevent this, I set the button to “disabled” after one successful click and enable it once the analysis is finished running.

-In order for the user to be able to run the analysis multiple times, I needed to ensure that they would have no problems overwriting the output from previous iterations. Arcpy has a built-in output overwrite setting, but this didn’t seem to always work when dealing with the output map images. To assist I built a function that would sweep through the output workspace and delete all previously created files each time a new K value was submitted.

-The most instructive part of the GUI is probably the output text box. Initially I thought this would be a straightforward process of updating a text field as the different functions in the code executed, but because python is a single threaded language the GUI would freeze while the arcpy analysis was processing and none of the update messages would display. To fix this in the final version of the program the “thread” of the arcpy analysis function is separate from that of the GUI. Now it can make calls to the GUI to update the text from within the analysis thread. As a bonus the GUI interface no longer freezes while the arcpy analysis is running, and you can open info dialog boxes from the top menu while it is processing.

Results

Based on the results from running this process, with a variety of K values that fall in the generally accepted good range of around 0.5-3, I believe there is a relationship between nitrate values and cancer occurrence in Wisconsin. Viewing the output on a map will show that it is not a perfect 1 to 1 relationship between the two variables, but the Moran’s I output consistently shows a high level of clustering. Of course, more robust analysis than this would be needed to verify and build on these results, but this tool should be a good starting point to investigate the relationship between nitrates in the water and cancer.