**Introduction**

The purpose of this documentation is to outline how to obtain water quality data as well as spatial tile drainage and land usage data in order to determine the effects of land use on water quality. A Python script was generated to run this analysis; however, some work was done on QGIS in order to perform some more complex spatial operations. This workflow discusses all of the steps that were taken. The steps outliened here align with the headings of the associated .ipynb file.

**1. Importing Water Quality Portal Data**

**1.1** Required parameters were entered in [the portal](https://www.waterqualitydata.us/#advanced=true) and the ‘Query URL’ was generated

**1.2** Used this link to create a python function **get\_data()** to download data from the Water Quality portal using any parameters. The inputs of this function can be modified such that the required data can be outputted.

**1.2.1** Alternatively, one can download the data directly from the portal by selecting ‘Sample Results (physical/chemical metadata)’ and then ‘download.’ This file can then be imported directly into python using pandas

**1.2.2** In this case, only surface water was needed, and measurements with units ug/L were not required, so these modifications to the dataset were made in python after the data was imported.

**1.3** It is important to note that the latitude and longitude data for these points are not included in this output. One can download location the data directly from the portal by selecting ‘Site Data Only’ under data profiles and then ‘download.’ This file can be directly imported into python using pandas.

**1.4** the location data and the chemical data are then combined using pd.merge()

**2. Importing CSI Data**

**2.1** CSI data was downloaded directly from [here](http://www.database.communityscience.org/queries) based on the required parameters (similar to step 1.1) This data is important because there are many samples here that were taken in the finger lakes region, where a large amount of tile drainage is.

**2.2** Then, location data for all of the CSI sampling locations was obtained through email correspondence with the CSI

**2.3** Using python the formatting of the CSI data was modified to match that of the WQP and the location CSI data and then chemical CSI data were merged together and then merged with the WQP data

**2.4** In this case, we only were looking at nitrate data in upstate NY in the 2000s. So, any data that was south of latitude 41 and before 01-01-2000, was removed.

**3. Modifying Units of Data**

**3.1** The nitrate data was reported with different units based on the providers in the WQP. Some nitrate data was reported in units of mg/L as N or mg/L as NO3. In order to do proper analyses, all data needed to be converted to mg/L as N. Additionally, there were many cases where there were multiple results from the same sampling location and date.

3.2 The Python function **process\_samples()** was created in order to reformat the data quickly and efficiently. The length of the dataframe went from lengths 71205 to 43731 after going through this function

**4. Preliminary Understanding of Nitrogen Data over time**

**4.1** Nitrate data collected from 2000 to present was grouped by decade (2000s, 2010s, and 2020s).

**4.2** These data were then run through the python function **loc\_stats()** which groups each decade dataframe by its sampling location and then finds the following:

the first sample date, most recent sample date, average nitrate concentration, the coordinates of that location, and the number of samples taken at this location

**4.2.1** Outliers were removed for plotting purposes only. Any value at or above the 97th percentile of average value or sample count for a sampling location was changed to the 97th percentile value.

**4.3** Each decade of data was plotted using geopandas on the basis of sample counts and average nitrogen concentrations. These plots revealed high numbers of samples taken in the Adirondacks region and an increasing number of samples taken in the finger lakes region as time went on after 2000. Additionally, higher nitrate concentrations were seen in the fingerlakes region in all three decades just based on visual observations of the plots.

**5. Obtaining Watershed delineations**

**5.1** A dataframe containing all of the sampling points from 2000 to present was created by running the dataset output from **step 3.2** (united\_adjusted\_df in the python code) through loc\_stats to get the average sample value and total number of samples at each point.

**5.1.1** This file was then converted to a GDF so that it can be easily plotted

**5.2** This df of sampling points (points\_gdf) was then broken down into 250-point sub-gdfs and a zip file was created for each sub GDF.

**5.3** Each of these zip files were run through the [stream stats batch processor](https://streamstats.usgs.gov/ss/?BP=submitBatch).

**5.3.1** New York State was entered under ‘State’. ‘Monitoring’ was entered as the ID Feild.

**5.4** It may take a day or two for the processor to run these watersheds. Once completed, the processor will output multiple .gbd files (one for each zip file entered in step **5.3**). The function load\_gdf() will read these gbd files which give both the points entered as well as their respective watersheds (polygons). For this case, on the watershed polygons will need be added to the gdf

**5.4.1** These polygons were merged with the points\_gdf from step **5.1.1** based on the ‘ID Feild’ column specified in step **5.3.1** and converted to a GeoDataFrame (points\_gdf\_shed)

**6. Getting crop and tile drainage spatial data through QGIS**

**6.1** NYS HUC12 shape file was downloaded from [here](https://opdgig.dos.ny.gov/datasets/NYSDOS::watershed-boundary-huc-12/about). Note, this interface had some issues so it might need to be obtained through someone else who has it already downloaded.

**6.1.1**. This was then QGIS was imported for clipping purposes. Default CRS of EPSG 4263 was kept.

**6.1.2** Note, any shapefile that outlines New York State would work instead of the HUC12 shapefile.

**6.2** Land Use from 2016 data (.img raster file) was downloaded from [here](https://www.mrlc.gov/data/nlcd-2016-land-cover-conus) (click the download button)

This specific land use dataset was used as it was the one used in developing the [tile drainage model](https://www.nature.com/articles/s41597-020-00596-x)

**6.2.1** This is a very large dataset and thus is very slow to process and thus modifying it in QGIS and then transferring it to python was the optimal choice.

**6.2.2** The default CRS of this file is a custom CRS called ‘Albers Conical Equal Area’, but spatially lines up with the HUC12 shapefile in EPSG 4326. If it is converted in QGIS, it will become misaligned, so the two were left with different CRS.

**6.2.3** The crop raster has about 200 legend entries, but only raster legend number 82 (cultivated crops) is needed (as referenced in the [legend](https://www.mrlc.gov/data/legends/national-land-cover-database-class-legend-and-description)). ‘Raster Calculator’ in the geoprocessing toolbox was used to remove the rest of the legend entries.

**6.2.3.1** In Raster Calculator, the expression "nlcd\_2016\_land\_cover\_l48\_20210604@1"=82 was entered. The reference layer of the same Landuse\_data layer is needed. The cell size should be set to 100. This is to lower the resolution of the raster, which makes future operations much quicker. The default resolution of 30m is higher than is needed for this analysis. The output (us\_crops.tif) was saved to a file on the computer.

**6.2.4** The us\_crops file was then clipped to the HUC12 data to only get data relevant to NYS. ‘Clip Raster by Mask Layer’ under ‘GDAL’ in the geoprocessing toolbox was used to do this with the input layer being the us\_crops.tif and mask layer being the HUC12 polygon shape file.

**6.2.5** Before finding the percentage of crop cover, the crop raster must be converted from raster pixels to points. ‘Raster Pixels to Points’ under ‘Vector Creation’ in the geoprocessing toolbox was used. The clipped crop\_raster was entered into the input file, and the output was saved as nys\_crops\_points.shp on the computer.

**6.2.5.1** The output of this file will be a dataset with a point for each cell of the raster. Each point will be assigned a value of either 0 or 1 (yes or no tile drainage coverage).

**6.3** Tile drainage data (.tif raster file) was downloaded from [here](https://figshare.com/articles/dataset/AgTile-US/11825742). More information on how this model was generated is written in [this paper](https://www.nature.com/articles/s41597-020-00596-x).

**6.3.1** The default CRS of this file is ESRI: 102003, but spatially lines up with the HUC12 shapefile in EPSG 4326. If it is converted in QGIS, it will become misaligned, so the two were left with different CRS.

**6.3.2** The crop raster has three legend entries (0,1, and 255), but only raster legend number 1 (tile drainage) is needed. ‘Raster Calculator’ in the geoprocessing toolbox was used to remove the rest of the legend entries.

**6.3.2.1** In Raster Calculator, the expression "Agtile-US@1"=1 was entered. The reference layer of the same Agtile-US is needed. The cell size should be set to 100. This is to lower the resolution of the raster, which makes future operations much quicker. The default resolution of 30m is higher than is needed for this analysis. The output (us-tile.tif) was saved to a file on the computer.

**6.3.3** The us\_tile file was then clipped to the HUC12 data to only get data relevant to NYS. ‘Clip Raster by Mask Layer’ under ‘GDAL’ in the geoprocessing toolbox was used to do this with the input layer being the us\_tile.tif and mask layer being the HUC12 polygon shape file.

**6.3.4** Before finding the percentage of tile drainage cover, the tile drainage raster must be converted from raster pixels to points. ‘Raster Pixels to Points’ under ‘Vector Creation’ in the geoprocessing toolbox was used. The clipped tile drainage raster was entered into the input file, and the HUC12 shape file was set as the mask layer. The output was saved as nys\_tile\_points.shp on the computer.

**6.3.4.1** The output of this file will be a dataset with a point for each cell of the raster. Each point will be assigned a value of either 0 or 1 (yes or no tile drainage coverage).

**7. Formatting Spatial Data in Python**

**7.1** The tile drainage points and crop coverage points shape files were loaded into python using geopandas and converted into geodataframes. Geodataframes along with the df in step **5.4.1** (points\_gdf\_shed) were then converted to CRS EPSG: 3857. This CRS was chosen because it is in units of meters which is helpful for potential area calculations in the future.

**7.1.1** The loading in of these datasets into python and converting their CRS take about 2 to 3 hours.

**7.2** The percent\_in\_poly() function in python was made to find the percent within each polygon (in this case, it is watersheds) that is covered by either tile drainage or crop cover. This is done by first finding the total points (which represent pixels) in each polygon, and then finding the number of points with value 1 (representing tile drainage or crop coverage on the pixel that the point represents as mentioned in 6.3.4.1 and 6.2.5.1)

**7.3.1** The percent crop and tile coverage in each polygon is done by doing the following calculation: (points\_with\_value\_1 / total\_points)\*100

**7.3** The function **plot\_polys()** was created to more efficiently plot watersheds based on any parameter.

**7.4** Another parameter, tile\_proportion was created, which is the proportion of cropland that is tile drained in each HUC12.

**7.5** Each of these parameters were plotted using the plot\_polys() function created in step 7.3.

**8. Statistical Analysis**

**8.1** The parameters of study were first outlined and formatted for proper statistical analysis.

**8.1.1** The independent variables are percent crop cover of each watershed, percent tile drained area in each watershed, and the proportion of crop cover that is tile drained in each watershed. All of these data are each converted to 2D arrays and the nan values are dropped

**8.1.2** The independent variable is the average nitrogen concentration of each sampling point.

**8.1.2.1** Note that each watershed has at least one sampling point associated with it, however not all of these watersheds are guaranteed to have any crop cover or tile drainage.

**8.2** The first statistical analysis that was conducted was a weighted multiple linear regression involving percent tile drainage cover.

**8.2.1** The general form of the multiple linear regression model is

y=β0+β1x1+β2x2

Where y = Nitrogen Concentration, x1 = crop cover, and x2 = tile drained area. The weights (w) of this linear regression were the number of samples taken per sampling location.

**8.2.2** The stats models WLS package was used to plot the nitrogen data against the independent variables and outputted values for β0-2

**8.3** The second statistical analysis that was conducted was a weighted multiple linear regression involving the proportion of crop cover that is tile drained.

**8.3.1** The general form of the multiple linear regression model is

y=β0+β1x1+β2x2

Where y = Nitrogen Concentration, x1 = crop cover, and x2 = proportion of crop cover that is tile drained. The weights (w) of this linear regression were the number of samples taken per sampling location.

**8.3.2** The stats models WLS package was used to plot the nitrogen data against the independent variables and outputted values for β0-2