**Guided Project 2 –**

Problem Statement: Google Earth Engine (GEE) is an expansive API resource for data, as well as a GIS workspace for geospatial analysis. GEE users can play the role of developer by using this platform for developing API apps or sharing code with other programmers for collaborative purposes. The problem is that many who are familiar with a graphic user interface (GUI) of conventional GIS software may be limited in using the data sources of GEE or relating these valuable data sources to their own collected data. Therefore, to solve this problem my main objectives were:

* 1. Use python code to read a .csv file and create a shapefile (shp) based on the data in the .csv
  2. Use python code to access the “USGS 3DEP 10m” elevation map and extract the elevation values from this map to be written to the points of the created shp file.
  3. Create a GUI tool that can execute the previous objectives based on a user’s entered parameters.

I began by making sure that all appropriate packages were installed for access to GEE. Anaconda prompt was used to install and authorize GEE in the same environment that my python libraries exist for in ArcGIS pro. I then created a jupyter notebook so that I could test small chunks of code and be sure that I had all the necessary packages installed. The coding elements that I tested for, and the cell outputs can be viewed in “Project2.ipynb”. The summary of which is as follows:

* The first cell was code written to open and read the “boundary.csv” file. The output displayed the structure of the “boundary.csv” so that I could understand what data was included in the .csv file.
* The next cell was used to determine the spatial reference that we will work from going forward. By using describe, we could find the spatial reference set to an image of the Area of Interest (AOI) “flood\_2class.tif”. The output returned that the spatial reference for this image is set as NAD\_1983\_StatePlane\_North\_Carolina\_FIPS\_3200 with the WKID of 32119
* I could then create a shapefile based on the data from the “boundary.csv” using xy table to point. The output of this cell was a vector point shapefile called “boundary\_pnts.shp” based on the data from “boundary.csv” and set to the spatial reference WKID 32119.
* The next three cells were used to 1.) import the Google Earth Engine module 2.) Access the “USGS/3DEP/10 m” digital elevation model (dem) from GEE 3.) Get information about the dem.
* I imported a basemap from GEE to the notebook to view the dem on top of it.
* I imported the dem to the GEE basemap

I opened my anaconda command terminal to install some important packages that were necessary for accessing the geometrical data necessary to include in our feature class. These packages were pandas (pd), geopandas (gpd), and shapely.

* I imported the pd, gpd, shapely modules to the jupyter notebook and used them to access the x,y geometry from the “boundary.csv”.
* This geometry data was used to define and create a feature class, and this feature class was added to the GEE map.
* I defined sample feature collection dictionary (sample\_fc)
* I defined and viewed sample feature collection information (sampled\_info)
* I viewed sampled\_info features
* I defined and viewed original information dictionary(origin\_info)
* I used a for loop to enumerate and update the sampled dictionary with features from the original dictionary so that the feature class that I created going forward could contain the elevation data that I accessed from the GEE dem.
* I imported operating system (os) and used arcpy.management to name and create the feature class “pnt\_elev.shp”
* I added the field called “elevation” and set the data type to float
* In the final cell of the notebook, I used insert cursor to iterate through the feature class attribute table and insert the elevation data value from the sampled feature class (dem (USGS 3DEP)).

The additive result of these steps accomplished our first two objectives. It was time for me to move on to accomplishing my third objective of creating a GUI tool that can execute the previous objectives based on a user’s entered parameters. I began by creating a python script that could be used as a “tool” in ArcGIS pro. This script can be viewed/edited and is included with this report (“Project2.py”). This script consists of three parts that allow the tool to be used by someone not familiar with coding. The first part loads all necessary packages (os, ee, arcpy, pd) (Figure 1). The second section of code defines the function for getting the GEE elevation data and writing it to a feature class shapefile created from the user’s data input (e.g. “boundary.csv”) (Figure 2). The third section of code executes the main function according to the defined parameter inputs. These defined parameters are workspace, input csv, output file name, and spatial reference (Figure 3).

A black background with white text

Description automatically generated

**Figure 1: Packages necessary for getGeeelevation script tool**

A screen shot of a computer program

Description automatically generated

**Figure 2: getGeeelevation function**

**A black background with a line

Description automatically generated**

**Figure 3: Main function**

This script was successful in creating a GUI tool that can be added to an ArcGIS pro project’s geodatabase (GDB). This script is written in a way that can be shared and applied to a different dataset (csv) with XY data. After the tool is added to the GDB, the user can run the tool by filling in the parameters in the GUI (Figure 4).

A screenshot of a computer

Description automatically generated

**Figure 4: Graphic User Interface tool “Get Elevation”**

1. Begin by selecting the workspace in which the csv file can be found for the “Workspace” parameter.
2. Write in the CSV file that is found in the user’s workspace in the parameter field called “CSVfile”. This should be the user’s XY data that the feature class will be created from, and the elevation data will be added to the feature class.
3. Determine the file name for the output shapefile and enter this information in the parameter field “Output Shape File”.
4. The last parameter field should be the WKID of the spatial reference geographic coordinate system. In my efforts in accomplishing our first objective, I determined the spatial reference to be associated with the geographic X Y data in the “boundary.csv” was WKID:32119. However, this tool is designed so that the user can input the most appropriate WKID for their own CSV data.

After running the tool, the user should see the appropriate messages notifying that the tool was executed with success (Figure 5).

A screenshot of a computer

Description automatically generated

**Figure 5: Result of a successful run of “Get Elevation” tool.**

This tool makes it possible to extract elevation values from the USGS 3DEP 10m map available on GEE and join them to a CSV XY table to produce a shapefile with elevation data for the CSV dataset (Figure 6). This tool can be used for any data in the United States because the USGS 3DEP 10m map is confined to the 50 US states. The tool is valuable to those who need to amend their geometry with elevation data for creating a shapefile. For many ArcGIS pro users who are not familiar with python, this tool can enable these users to automate work that would otherwise require utilization of multiple geoprocessing tools or more knowledge of python coding. Further, this tool can be used for data preparation and visualization for application in ecological, civil, military, geographic research and project development.

A bird's eye view of a pond

Description automatically generated

**Figure 6: Resulting shape created from the “Get Elevation” tool.**

Works Cited

United States Geological Survey. (2024). *USGS 3DEP 10m* [Data set]. Google Earth Engine. <https://developers.google.com/earth-engine/datasets>

Wang, L. (2024, November 22). Project2.mkv[Video]. GIS Programming GEOG4057, Louisiana State University. <https://lsu.app.box.com/s/voll3bxiwkqhpyme31bqrw55m5d7ttdg/file/1707534486808>