Automation and Flexibility with ArcPy: A Stand-Alone Python Tool for Retrieving and Analyzing Natural Hazard Data

GISC 6389 - GIS Master's Project

Student: David Lindsey

Advisor: Dr. Fang Qiu

2018 April 30th

Introduction

As a Geographic Information Systems (GIS) developer, automating and simplifying a GIS user's workflow is a worthwhile task. These workflows can have considerable impacts on time management and productivity. If redundancy and manual processes can be prevented, they should be prevented. This Master's Project focuses on improving a user's workflow by automating many time-intensive, laborious tasks into a streamlined user experience.

For this project, a stand-alone Python application with ArcPy integration was chosen and designed to retrieve various forms of natural hazard data, all while giving the user the flexibility to conduct varying analyses of that data within user-specified areas of the United States. User-specified areas include a) all 50 United States and Washington, D.C., b) state-level only, as well as c) county-level only. Natural hazard data would be selectable across specific timeframes, specifics magnitude event ranges, with various analysis options at the user's disposal. The user also has the option to output a customized set of summary statistics to a local comma-separated values (CSV) file. The information provided within the CSV files can answer numerous questions for a researcher. How many earthquakes occurred within California from 1988 to 1990, and what was the minimum magnitude, mean magnitude, and maximum magnitude of those events? How many EF-5 tornadoes occurred within Dallas County, Texas from 1970 to 2000? How many states endured hurricanes in 2005, and what were the names of those storms? Valuable information such as this can quickly be attained by using this application.

The intended recipients of such an application include those who work with natural hazard data specifically (e.g. emergency management personnel, insurance companies, researchers focusing on hazard studies, among many others). The natural hazard options currently available for users within this initial version of the application encompass Earthquake, Hail, Hurricane, Tornado, and Wind.

The aforementioned data are provided for public use by the United States Geological Survey (USGS) as well as the National Oceanic and Atmospheric Administration (NOAA). These agencies provide website links for users to access and download the raw data as needed. The raw data may be available in CSV files or Esri Shapefiles, depending on the hazard selected. In order to utilize many of the raw hazard data types (especially data in CSV file format), the user may be required to download a zipped folder containing the file, extract the file from the zipped folder, and then perform visual checks on the data values within the file to remove erroneous values and to ensure row headers are present. Once these data reviews have been accomplished, the user must convert the CSV file to a GIS feature class. If the user wishes to analyze a small geographic area within the raw dataset that was downloaded (e.g. Collin County, Texas only), they would need to manually restrict (or clip) the data to that specific polygon boundary. In order to accomplish this, the user needs U.S. county boundary data. If they don't already have boundary polygons, the user would likely need to access an official government website such as the U.S. Census Bureau to acquire it. On top of all of this, the user *still* has numerous types of analyses to conduct before achieving any results. As this scenario demonstrates, there are many benefits to automating and improving the time management of this workflow for users.

Literature Review

This project's origin stems from a previous application that I had developed during the Summer 2017 semester while attending the University of Texas at Dallas. I referred to it as an "ArcGIS for Desktop Add-In Toolset for Retrieving and Displaying Natural Hazard Data". It was a simple, VB.NET-based application incorporating ArcObjects. The premise of the tool was similar to this Master's project, in that it could obtain four natural hazard types (Earthquake, Hail, Hurricane, and Tornado) and display them with proper symbology within ArcGIS for Desktop. The application did not accomplish anything beyond that, however. At the time of its creation, I knew that it could be enhanced in various ways. From personal experience, VB.NET and ArcObjects had limited documentation with a dwindling user base as years have progressed. Adding to that, in 2017, Microsoft's program manager announced a new strategy moving forward that would see C# take priority over VB.NET in future development, given its richer ecosystem (Torgersen, 2017). Any enhancements to that previous project would be better suited in a more versatile, popular programming language such as C# or Python with its access to ArcPy geoprocessing capabilities. Additionally, the four hazard types could easily be expanded to include various other options for the user to consider. The add-in tool did not allow for custom timespan or custom magnitude options for users to query refined data. There was no means to perform analysis on the data, nor was there a means to restrict the hazard data to U.S., state, or county level polygons. All of these enhancements will be addressed within this Master's project application. To be clear, this is not a reproduction of the same application. It has been completely redesigned, reformatted, and enhanced to better accommodate users. The underlying motivation has been then, and will be now, to provide ArcGIS users a more robust way to handle time-intensive data and achieve faster analysis results.

Having said that, research currently conducted indicates that there is not a basic, streamlined application to accommodate all of the aforementioned hazard data types within a central location. Additionally, the applications that exist tend to have numerous requirements and/or limitations for the general public. For example, FEMA has a very sophisticated ArcGIS-based tool known as Hazards US (Hazus). It is designed to allow users the ability to analyze and produce loss estimates for earthquakes, floods, hurricanes, and tsunamis (FEMA, 2017). It is widely popular, free of charge, and exceptionally useful for the above situations. However, it is narrow in scope (e.g. no hail, no tornadoes, among others). It is also a very large, complex piece of software. To download the latest version of the application (Hazus 4.2) within a zipped folder requires roughly four gigabytes of hard drive space. This expands to nearly six gigabytes in size once unzipped. Also, this installation does not include any state data. Those data are separately downloaded, and to possess the entire U.S. dataset requires approximately *70 gigabytes* of hard drive space according to FEMA. It is also not a stand-alone application. ArcGIS Desktop is required, and the user's computer must have the exact same version installed in which the application is designed. Hazus 4.2 is only supported on ArcGIS Desktop 10.5.x, with Hazus 4.0 operating on ArcGIS 10.4.x. The overwhelming size of this application would be enough to deter most people from using it for simplistic data retrieval purposes. This Master's project takes up minimal hard drive space, and will only download as much data as the user requests and retrieves. The only requirements from the user will be a modern internet connection, ArcGIS Pro installed with its native Conda-based Python 3.x libraries, and an "Advanced" ArcGIS license for some of the application's analysis options (additional details on these requirements within the Methodology section).

Another similar application exists, but only for earthquake data. It is an ArcGIS Desktop add-in tool described within a research paper titled “SeismoGIS: A Tool for the Visualization of Earthquake Data” (Williams, Weskamm, Baaser, Hinzen, & Bareth, 2008). Within the paper, the authors described an ArcGIS Add-In tool that was programmed with .NET and ArcObjects. Its purpose was to allow European scientists the ability to quickly load earthquake data and provide analysis of the data by programmatically extracting all required, routine inputs from a raw ASCII file. For these scientists’ purposes, this was a very effective tool. However, the tool was only useful for those users who had access to a proprietary software called SEISAN. SEISAN is essentially a software package that processes seismic activity from around the world (Ottemaller, 2014). Without the SEISAN software installed, the Add-In tool would not function.

Given the nature of occasional “human error”, performing quality checks on data is imperative. An analysis of NOAA’s storm events database yielded proof that errors and inconsistencies abound within the data, and that users should take appropriate precautions when using it (Santos, 2016). With that said, ongoing research has found multiple instances where this Master’s project application would have been highly useful for those researchers. One report entitled “An Analysis of Severe Weather Data: 2000 – 2015” (Madhavan, 2015) has an entire section devoted to manually cleaning up CSV storm data from NOAA’s website. In another report, “Visualization of Tornado Data” (Joy, Chheda, Chy and Sun, 2014), researchers spend time explaining how NOAA’s occasionally flawed tornado data needed to be prepared within Microsoft Excel, then imported into Microsoft Access for quality checks. These processes were conducted without any mention of automation. By automating many of these redundant processes within this Master’s project application, users will be able to spend more time on analysis and interpretation, with less time spent on data preparation.

Study Area and Data Sources

The study area for this Python application will focus primarily on the United States (all 50 states plus Washington, D.C.). However, the user will be able to extract global scale data for the earthquake and hurricane hazard types. One flexible aspect of this application is that it will allow the user to designate the study area as they see fit. If a user wants to focus on the entire United States, they will have that option. If the user wants to focus only on the state of Texas, the user will have that option. To provide even more granularity, the user will be able to focus on a specific county within a specified state if they elect to do so. This level of freedom is possible in large part due to the U.S. Census Bureau having public access to the latest U.S. boundary shapefiles. For this project, the application will extract the entire US polygon boundary set, a state polygon boundary set, or a county polygon boundary from the same Shapefile (at the time of this writing, all of the aforementioned polygons can be downloaded as one single polygon Shapefile from 2017[1]).

For the various natural hazard types, two U.S. federal agency websites are utilized to obtain data. Hail, hurricane, tornado, and wind data are accessible from NOAA. Earthquake data is accessible from USGS. The majority of the raw data is specific to the United States, while some hazard types represent a global footprint. As previously mentioned, earthquakes and hurricanes are two examples allowing for global datasets to be downloaded.

With the exception of hurricane data, all hazard types will be extracted as point features within this application. Hurricane data will be comprised of polylines. Depending on the feature type, this will determine what types of analysis options are available to the user (e.g. point

[1] http://www2.census.gov/geo/tiger/GENZ2017/shp/ cb_2017_us_county_500k.zip

density analysis versus line density analysis). The analysis options will be explained further within the Methodology section.

The user will be able to access data from varying timespans, depending on which hazard type is selected. For example, the USGS allows users to search back to 1900 for earthquake data, while NOAA allows users to search back to 1955 for hail and wind, but 1950 for tornadoes. For hurricanes, NOAA allows users to query data all the way back to 1842. Each hazard type will likely have its own unique timespan with no clear timeframe alignments shared among all hazards.

Methodology

Prior to running this Python application, several software prerequisites must be addressed. It is assumed that GIS users seeking to utilize this application will already be ArcGIS users with ArcGIS Pro and its Conda-based Python 3 libraries installed on their computers. Further guidance is now provided for those who do not currently fall within that category.

The user must have a stable internet connection in order to retrieve the data from the aforementioned websites. As some of these datasets can be quite large in file size, faster internet speeds will allow for faster downloads. Additionally, check Esri's system requirements to ensure your computer is compatible with ArcGIS Pro[2] and Windows. As of the date of this report, testing successfully shows that ArcGIS Pro versions 1.4 to 2.1.2 work without issue on Windows

[2] http://pro.arcgis.com/en/pro-app/get-started/arcgis-pro-system-requirements.htm

10. Testing for computers running Windows 7 Enterprise encountered repeated issues and is not recommended for use[3].

The user will be required to have Esri licensing. The licensing level should preferably be an “Advanced” license with access to the Spatial Analyst extension, as some analysis options require this licensing level. A minimum of Basic licensing is required for the application to convert CSV files to feature classes. However, if no Esri license exists and Python 3.x is installed, the application is still able to access the web URLs, download the data, and check for quality issues within the CSV files.

Once the user has confirmed that they have ArcGIS Pro installed with proper licensing, one of three options must be fulfilled in order for this application to perform correctly when launched. Because it is utilizing ArcGIS Pro’s version of ArcPy, users attempting to execute a stand-alone Python 3.x application with ArcPy are required to either 1) have the ArcGIS Pro software open at the time of application launch, 2) have the ArcGIS Pro software settings set to “Authorized to work offline”, or 3) have the “Sign me in automatically” checked when initially signing into ArcGIS Pro after installation. Per Esri, if one of the above conditions is not met, you will encounter a “RuntimeError: Not Initialized” exception when the application attempts to import ArcPy[4]. That process is how the Esri license is identified and determined. Check with Esri’s online ArcGIS Pro FAQ site for further assistance[5].

Esri has conveniently begun packaging ArcGIS Pro together with any ArcGIS Desktop purchase. If a user has previously installed ArcGIS Desktop without ArcGIS Pro, and if that user

---

[3] Per Microsoft, Windows 7 support is slated to be discontinued in January 2020, so utilizing a higher version should be encouraged.
[4] http://pro.arcgis.com/en/pro-app/arcpy/get-started/installing-python-for-arcgis-pro.htm
[5] http://pro.arcgis.com/en/pro-app/get-started/faq.htm

still currently possesses an Esri “maintenance” subscription, ArcGIS Pro can be easily downloaded. They simply need to login to their Esri accounts to access the software for download. Again, the ArcGIS Pro FAQ site provides further assistance if necessary.

Once ArcGIS Pro has been installed and properly set up, the hazard application should be ready to use. Please note: ArcGIS Desktop’s ArcMap is designed to run on, and is packaged with, Python 2.7. ArcGIS Pro’s Conda-based Python 3.x dependent applications will not natively run on Python 2.7 due to the syntax changes in Python 3.x. As a result, this application is not currently compatible with ArcMap-only execution.

Once all software prerequisites have been completed, the user may now utilize the hazard application. Although ArcGIS Pro must be installed to properly use Esri’s Python 3 libraries, ArcGIS Pro itself is not directly utilized directly by this application. It is designed to operate as a stand-alone application, with no true allegiance to ArcMap or ArcGIS Pro. The reason for this is due to the current state of ArcGIS. ArcMap retains a very large group of GIS users who have not migrated to ArcGIS Pro’s layout, but Esri has made clear that ArcGIS Pro represents the future platform and will eventually replace ArcMap (Deindorfer, 2017). The vast majority of new development taking place within ArcGIS is happening within ArcGIS Pro. As a result, the best balance with this hazard application is to be “forward-thinking”, but with the caveat of not alienating everyone who remains loyal to ArcMap. This approach allows the user to run the application without having to choose an ArcGIS allegiance. They can run the stand-alone application, extract the data from the internet, and run analyses to create the necessary feature class outputs. From there, the user can *choose* to manually open an ArcMap session or an ArcGIS Pro project to import the data created by the hazard application.

In order to develop the application as envisioned, large amounts of "Pythonic" research were required. Numerous physical and online reading resources were helpful and, ultimately, necessary to accomplish all desired objectives[6]. For the purposes of this project, the goal was to create a script as naturally Pythonic as possible. This means that, aside from ArcPy, all modules, functions, and methods are strictly "stock" Python. All graphical user interface (GUI) windows are programmed in tkinter, for example. From a maintenance perspective, the user should not need to worry if third-party extensions like wxPython or PyQt are properly installed alongside Python. As long as the user has ArcGIS Pro and its associated Python libraries installed, this application will function as desired.

The programming scripts that were constructed to create the overall application were prioritized, structured, and implemented as follows:

1. An application script "driver" from which all functionality begins. This Python script is called "GUI_ApplicationDriver.py". Its purpose is to initialize the application from the user's execution of this file. To execute the application from this script as a stand-alone application on Windows 10, the user will need to right-click on the file, choose "Open with…", select "Choose another app", and navigate to "c:\Program Files\ArcGIS\Pro\bin\Python\scripts\propy.bat" (this Program Files path represents the default storage location for ArcGIS Pro-related items). According to Esri, the "propy.bat" file determines the application's active conda environment and activates it in your stand-alone script[7]. It governs the configuration settings for where the GUI

[6] Please see Reference section for specific sources utilized.
[7] http://pro.arcgis.com/en/pro-app/arcpy/get-started/using-conda-with-arcgis-pro.htm

will launch and display on the user's computer screen. Alternatively, the user can utilize the "pythonw.exe" found here: "C:\Program Files\ArcGIS\Pro\bin\Python\envs\arcgispro-py3\pythonw.exe"[8]. Upon initialization, the application immediately imports, and proceeds to, the next module called "GUI_HazardsMenu.py". It consists of ~80 lines of code

2. The GUI_HazardsMenu.py module is the first piece of the application that interacts with the user. This module provides the user with a GUI window layout that instructs the user to select which hazard to work with (See **Appendix – Figure 1**). These hazards are displayed to the user via a read-only dropdown list (See **Appendix – Figure 2**). Once the user selects the appropriate hazard type and clicks the Next button, the GUI window will change size and layout to reflect the user's hazard choice. It consists of ~140 lines of code.

3. Each hazard has been programmed as its own separate module, with these modules being executed only when the user selects that specific hazard option. The naming conventions for each of these hazards is "GUI_[HazardNameGoesHere]Options.py". In total, there are five hazard modules. When the user selects a hazard type from the GUI_ApplicationDriver and clicks Next, the hazard-specific module will execute and display the appropriate GUI layout. Within each layout, the user will also have the ability to click a Back button (to the GUI_HazardsMenu module), as well as an Exit button to exit the application entirely. See **Appendix – Figures 3 through 7** to view the various GUI hazard option layouts. Each module consists of ~6,500 lines of code.

[8] Utilizing the "pythonw.exe" path over the "propy.bat" path will allow the application to launch without the Windows Command Prompt launching as well.

Each of these hazard layouts, along with how they function, will be discussed later within this report.

4. A Python module named "GUI_FrameLifts.py", which acts as a behind-the-scenes mechanism for switching between GUI window frame layouts. The Python application may appear to be switching between windows for the user, but it is actually the *same* window being *modified* through various layout alterations. These alterations are able to occur due to the GUI_FrameLifts module *lifting* different GUI layout views into view within the GUI itself, depending on user selection. For example, the main GUI window from Figure 1 represents a single GUI layout. When a user selects a hazard and clicks the Next button, that *same* window updates to reflect the layout of whichever hazard was chosen. Here's another way to visualize the concept: The GUI is a classic jukebox, with each hazard layout representing a vinyl record within the jukebox. When someone selects a specific record, the jukebox will load and play that record. Utilizing the GUI_FrameLifts module prevents the need for Python to open and maintain separate, individual GUI windows. This script contains ~20 lines of code.
5. Since the application will be supporting state and county clipping options, a dictionary/tuple module called "GUI_CountiesPerState.py" is required to maintain this information. If a user selects a specific state, all counties associated with that state will need to populate within a drop-down list for the user to select from. All of this functionality is reliant upon this GUI_CountiesPerState module, as it possesses all county names affiliated with each state. All state and county names within this module were derived from the same U.S. Census Bureau Shapefile used for clipping

hazard data, so no erroneous spelling discrepancies should presently exist. This module consists of ~900 lines of code. More information on this functionality to be provided further within this section.

6. Several image icons are used during different stages of the GUI application. This includes a logo of the University of Texas at Dallas, stored as "Icon_UTD.gif"[9]. Each hazard option displays an image icon specific to that hazard type. These are stored as "Icon_Earthquake.gif", "Icon_Hail.gif", "Icon_Hurricane.gif", "Icon_Tornado.gif", and "Icon_Wind.gif"[10].
7. Script requirements: In order for the application to launch successfully and perform as desired, the user must ensure that all Python scripts and image icons are stored within the same folder upon application execution (See **Appendix – Figure 8**). Otherwise, Python will not be able to locate and import the different modules as necessary, causing the application to fail.

Operation-specifics for all Hazard GUI Layouts:

Overall, each hazard GUI layout will appear similar in design and function. All hazard options will allow the user to select a timespan range, a severity range[11], optional analyses, as well as the ability to clip output data to U.S., state, or county polygons. The timespan values and severity values of each hazard are correlated to the timespan range and magnitude range available on that hazard's agency website. The purpose of each GUI hazard layout is to select various parameters that will be used to create a web URL for retrieving data. As of the date of

---

[9] UT-Dallas image obtained from the University of Texas at Dallas.
[10] Hazard icons obtained from two sources: www.clipart-library.com & www.iconfinder.com
[11] Severity type may differ per hazard: Earthquake = Magnitude, Hail = Diameter, Hurricane = Intensity, Tornado = Intensity, and Wind = Speed.

this publication, the following timespan options and severity options are available for each hazard:

For **Earthquakes**:

- USGS **Timespan** (UTC[12]) choices are:
    - Past Hour, Past Day, Past 7 Days, Past 30 Days, and Custom From/To Range with Month/Year (1900 to Present)
- USGS **Magnitude** (Moment Magnitude Scale) choices are:
    - All, 1.0+, 2.5+, 4.5+, and Custom Min/Max Range (-1.0 to 10.0)

For **Hail**:

- NOAA **Timespan** (CST[13])choices are:
    - All | 1955-2017, 2017, 2016, 2015, 2014, 2013, 2012, 2011, 2010, 2009, 2008, 2005-2007, 2000-2004, 90-99, 80-89, 70-79, 60-69, 55-59, and Custom From/To Range with Month/Year (1955 to 2017)
- NOAA **Diameter** (in inches) choices are:
    - All, 0.5" – 0.99", 1.0" – 1.99", 2.0" – 2.99", 3.0"+, and Custom Min/Max Range (0.0" to 10.0")

[12] All USGS earthquake timespans and NOAA hurricane timespans are created with "Coordinated Universal Time", or UTC, set as the default value.
[13] NOAA has chosen to publish the timestamp information in "Central Standard Time", or CST, for the Hail, Tornado, and Wind CSV files. http://www.spc.noaa.gov/wcm/data/SPC_severe_database_description.pdf

For **Hurricanes**:

- NOAA **Timespan** (UTC) choices are:
    - All | 1842-2017, every individual year from 1842 to 2017, and Custom From/To Range with Month/Year (1842 to 2017)
- NOAA **Intensity** (Saffir-Simpson Hurricane Wind Scale) choices are:
    - All, Tropical Depression, Tropical Storm, Category 1, Category 2, Category 3, Category 4, Category 5, and Custom Min/Max Wind Speed Range (0 to 200 knots per hour)

For **Tornadoes**:

- NOAA **Timespan** (CST) choices are:
    - All | 1950-2017, 2017, 2016, 2015, 2014, 2013, 2012, 2011, 2010, 2009, 2008, 2005-2007, 2000-2004, 90-99, 80-89, 70-79, 60-69, 50-59, and Custom From/To Range with Month/Year (1950 to 2017)
- NOAA **Intensity** (Enhanced Fujita Scale) choices are:
    - All, EF-0, EF-1, EF-2, EF-3, EF-4, EF-5, and Custom Min/Max EF Size Range (0 to 5)

For **Wind**:

- NOAA **Timespan** (CST) choices are:
  - All | 1955-2017, 2017, 2016, 2015, 2014, 2013, 2012, 2011, 2010, 2009, 2008, 2005-2007, 2000-2004, 90-99, 80-89, 70-79, 60-69, 55-59, and Custom From/To Range with Month/Year (1955 to 2017)
- NOAA **Speed** (in Knots) choices are:
  - All, 30 - 60, 60 - 90, 90 - 120, 120+, and Custom Min/Max Range (30 to 200 knots per hour)

Each hazard type has a partially unique, **static** web URL where all data is accessed from. Certain parameters within the URL string can be changed, via the timeframe and severity drop-down lists. The timespan/severity choices bulleted above are assigned to **dynamic** variables within the application. Each **dynamic** variable is then concatenated within the **static** URL for whichever website possesses the CSV/Shapefile data. To ensure stability, each drop-down list is read-only. The user is not allowed to manually enter timespan or severity values. The following are example web URL strings for each hazard type:

For **Earthquakes** (these will be downloaded as CSV files):

- An example web URL string for retrieving earthquakes having Magnitude **2.5+** during **Past Hour**:
  - **https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/2.5_hour.csv**
- An example web URL string for earthquakes having Magnitude **2.5+** with a **Custom** Timespan:
  - **https://earthquake.usgs.gov/fdsnws/event/1/query?format=csv&starttime=2017-1-01&endtime=2017-3-31&minmagnitude=2.5**
- Note: The USGS earthquake data will be the only hazard type utilizing a unique web URL string specifically for Custom Timespan/Magnitude queries. If the user selects a custom magnitude that spans 30 days or less, the application incorporates UTC conversion formulas to create the proper start/end times that are needed to populate within the query. The other hazard types automatically download and extract any custom timespans/severity values from within the CSV file representing all occurrences.

For **Hail** (these will be downloaded as CSV files):

- An example web URL string for retrieving **All** hail data:
  - **http://www.spc.noaa.gov/wcm/data/1955-2017_hail.csv.zip**
- An example web URL string for retrieving hail data for **2005-2007**:
  - **http://www.spc.noaa.gov/wcm/data/2005-2007_hail.csv**

For **Hurricanes** (these will be downloaded as Shapefiles):

- An example web URL string for retrieving **All** hurricane data:
  - **ftp://eclipse.ncdc.noaa.gov/pub/ibtracs/v03r10/all/shp/Allstorms.ibtracs_all_lines.v03r10.zip**
- An example web URL string for retrieving hurricane data from **1989**:
  - **ftp://eclipse.ncdc.noaa.gov/pub/ibtracs/v03r10/all/shp/year/Year.1989.ibtracs_all_lines.v03r10.zip**

For **Tornadoes** (these will be downloaded as CSV files):

- An example web URL string for retrieving **All** tornado data:
  - **http://www.spc.noaa.gov/wcm/data/1950-2017_torn.csv.zip**
- An example web URL string for retrieving tornado data for **1960-1969**:
  - **http://www.spc.noaa.gov/wcm/data/60-69_torn.csv**

For **Wind** (these will be downloaded as CSV files):

- An example web URL string for retrieving **All** wind data:
  - **http://www.spc.noaa.gov/wcm/data/1955-2017_wind.csv.zip**
- An example web URL string for retrieving tornado data for **2010**:
  - **http://www.spc.noaa.gov/wcm/data/2010_wind.csv**

To allow the user the ability to review the URL string being created, and to troubleshoot if necessary, a read-only textbox has been created within each hazard's GUI layout. This textbox will automatically populate the full web URL string once all drop-down list parameters have been selected *(*See **Appendix - Figure 9***)*. The next parameter for the user to select involves setting the output workspace folder. All downloaded files, processes, and output will be stored within this location. Setting the workspace requires the user to click a Browse button that launches a dialog box to select an output folder location. Once the folder location has been chosen, the file path will automatically display within another read-only textbox (See **Appendix – Figure 10**).

The user also has the ability to select more options prior to executing the application. If the user wishes to "clip" data to a specific geographic area, or if the user wishes to output various analysis results, they have that choice. In order to access the clipping options and analysis options, the user needs to select a checkbox labelled "More Options". Once clicked, the GUI layout will modify itself to display all clipping options and analysis options (See **Appendix – Figure 11**).

The "Clipping Options" section of the expanded GUI consists of three options: USA, State, and County. All three of these options are reliant on the U.S. Census Bureau Shapefile mentioned earlier. Depending on the user's clipping selection, the application will extract only that portion of the Shapefile and use it to clip the hazard data that falls within it. If the user selects the USA option, all hazard data that falls within the 50 U.S. states and Washington, D.C. will be clipped and extracted. If the user selects the State option, a dropdown list will appear, allowing the user to select the appropriate state (See **Appendix – Figure 12**) boundary to clip the hazard features. A similar situation applies for the County option. The user must first select the state of choice and then select the appropriate county within a county dropdown list (See **Appendix – Figure 13**). The state and county dropdown lists populate by referencing the "GUI_CountiesPerState" module discussed earlier. These dropdown lists are entirely dependent on this module. Additionally, if the “More Options” checkbox is selected, a clipping option must be chosen by the user. No analysis options are permitted without a defined clipping option.

The “Analysis Options” section provides the user with the ability to conduct varying types of analysis on the downloaded, clipped data. For point-based vector data, this section of the expanded GUI consists of 10 different analysis tools to work with. The user has the ability to calculate Inverse Distance Weight (IDW)[14], Kernel Density[15], Kriging[16], Natural Neighbor[17], Optimized Hot Spot Analysis[18], Point Density[19], Spline[20], Thiessen polygons[21], Trend[22], as well as an option to output the hazard counts to a CSV file with customized summary statistics for the specified clipping option.

For polyline-based vector data, the options dwindle to Hot Spot Analysis (with Inverse Distance Weight)[23], Kernel Density, Line Density[24], and the option to output hazard counts to CSV file (this shorter list of options is due to the incompatibility between point-based analysis options and polyline data).

These analysis tools (minus the hazard count CSV output) are the same tools that can be found within ArcGIS Desktop and ArcGIS Pro. For these analysis options, the severity field of each hazard type will be used as a parameter by default, where applicable. All other “optional”

---

14 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/idw.htm
15 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/kernel-density.htm
16 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/kriging.htm
17 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/natural-neighbor.htm
18 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-statistics/optimized-hot-spot-analysis.htm
19 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/point-density.htm
20 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/spline.htm
21 http://pro.arcgis.com/en/pro-app/tool-reference/analysis/create-thiessen-polygons.htm
22 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/trend.htm
23 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-statistics/hot-spot-analysis.htm
24 http://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/line-density.htm

fields found within the tools will be set to Esri prescribed default values. These default values can be viewed within the ArcGIS Pro reference links previously included.

The option to “Output Count Details to CSV File” is a custom operation that was created specifically for this application. It will output the hazard counts based on the user’s timespan selection, severity selection, and the selected clipping option (See **Appendix – Figure 14**). For example, if a user selects earthquakes in *California* within the *Past 30 Days* with magnitudes *2.5+*, the CSV count details will display:

- The global earthquake count within the past 30 days possessing 2.5+ magnitude.
- The U.S. earthquake count within the past 30 days possessing 2.5+ magnitude.
- The California earthquake count within the past 30 days possessing 2.5+ magnitude.
- Statewide summary statistics for California including Minimum Magnitude, Maximum Magnitude, Mean Magnitude, Median Magnitude, and Mode Magnitude.
- A list with every county within California that received earthquakes within the past 30 days possessing 2.5+ magnitude.
- In addition, the CSV file will display every county within that state that received earthquakes within the set parameters, the number of earthquakes per county, along with county-specific summary statistics.

For all options within the “Analysis Options” section, the user has the freedom to select all of them or select none of them. The analysis is completely optional. The purpose of these analysis choices is to provide the user with quickly derived analysis output, if so desired. If the

Esri default settings are not accurate and/or specific enough for the user's purposes, they can elect to perform separate, more sophisticated analysis through ArcGIS with parameters of their choice.

Once all input parameters have been set, the user is now ready to click the OK button and begin the process of obtaining final output data. Prior to any Python processes fully commencing, the application will first ensure that all necessary input parameters have been filled in properly within the hazard's GUI. If a timespan option, severity option, or output workspace folder have not been set, error messages will appear to warn the user of these discrepancies. The user will then be allowed to correctly set missing parameters before clicking OK once more. For custom timespan ranges, if the From date exceeds the To date, an error message will appear as well. The same situation will occur if the custom severity range has a minimum severity exceeding a maximum severity within the search range. Additionally, if the user has selected the "More Options" checkbox, a Clipping Option must be selected. Another error message will alert the user is a clipping option has not been selected. This application has numerous fail-safes programmed within itself to account for as many scenarios where human error may occur.

Assuming all input parameters have been properly set by the user, the OK button-click will trigger a cascade of various Python functions and operations. First, all selectable buttons, checkboxes, and dropdown lists become inactive. This is to prevent the user from attempting to modify input parameters and execute the code while it is already operational. The Exit button changes to a Cancel button, and remains active for the user to click in case they wish to terminate the program prematurely.

At the same time, a GUI layout alteration will occur, causing the window to expand downward to reveal a scroll box with processing messages, a status bar, and a progress bar (See

Appendix – Figure 15). As the application progresses through each step, the user will be able to follow the progress of all activities from within this scroll box. Regular messages will appear in black text, processing messages will appear in blue text, warning messages will appear in orange text, and error messages will appear in red text. An internal timer begins as well, as this will keep track of the elapsed time the application takes to complete all processes. The user will be notified of the elapsed time within the scroll box at the conclusion of all script operations.

The status bar and the progress bar share identical appearances, but function differently. The status bar is intended to constantly increment from 0% to 100% over the course of a few seconds. Once at 100%, the status bar will start over and repeat the same process. This will continue until the application has ceased all operations. This has been implemented so as to inform the user if the application has "frozen". Some processing tasks may take considerable time, which may cause the progress bar and application to appear non-functional. The status bar alleviates this concern for users. The progress bar is positioned just below the status bar, and is set to increment as tasks are completed. Once the application has finished all processing operations, the progress bar will display 100%.

After the GUI expansion occurs, the application attempts to create a folder within the output workspace set by the user. The naming convention of the folder will be created as follows:

{Agency Name}_{Hazard Type}_{Severity}_{Timespan}__{Date of Retrieval}_{Optional Clipping Type}

- Agency Name: Federal agency where the data was acquired.
- Hazard Type, Severity, and Timespan: Based on user selections from the GUI options.
- Date of Retrieval: The date the user clicked OK and ran the processing tasks.
- Optional Clipping Type: If selected, the USA/State/County selection.

Various example formatting for each hazard type:

- **Earthquakes**:
  - Timespan = Past Day, Magnitude = 2.5+, Clipping = USA
    - **usgs_quake_2_5_day__20180425_USA_50_and_DC_only**
  - Timespan = Custom: 201601 to 201604, Magnitude = Custom: 2.0 to 7.0, Clipping = Alaska
    - **usgs_quake_2_0_to_7_0_201601_to_201604__20180425_state_only_AK**
- **Hail**:
  - Timespan = All | 1955-2017, Diameter = All, Clipping = Denton County, TX
    - **noaa_hail_all_1955_to_2017__20180425_county_only_Denton_TX**
  - Timespan = 2015, Diameter = 2.0 to 2.99, Clipping = Texas
    - **noaa_hail_2_0_to_2_99_2015__20180425_state_only_TX**
- **Hurricanes**:
  - Timespan = 1993, Intensity = Category 1, Clipping = None
    - **noaa_hurr_cat1_1993__20180425**
  - Timespan = 2017, Intensity = Custom: 40 to 90, Clipping = USA
    - **noaa_hurr_40_to_90_2017__20180425_USA_50_and_DC_only**

- **Tornadoes**:
    - Timespan = 60 - 69, Intensity = Custom: 3 to 5, Clipping = None
        - **noaa_torn_3_to_5_60_to_69__20180425**
    - Timespan = Custom: 201401 to 201412, Intensity = EF-5, Clipping = Kansas
        - **noaa_torn_ef5_201401_to_201412__20180425_state_only_KS**

- **Wind**:
    - Timespan = All | 1955-2017, Speed = 60 to 90, Clipping = Miami-Dade County, FL
        - **noaa_wind_60_to_90_1955_to_2017__20180425_county_only_MiamiDade_FL**
    - Timespan = 2000 to 2004, Speed = All, Clipping = Oregon
        - **noaa_wind_all_2000_to_2004__20180425_state_only_OR**

Any decimal values (e.g. 2.5) will have the decimal converted to an underscore within the folder naming convention. Any negative intensity values will have the negative sign converted to "**neg_**" within the naming convention. Any spaces, dashes, or apostrophes within state/county names will be removed as well. These changes are necessary to prevent ArcPy geoprocessing scripts from potentially failing.

Once the parent folder has been created, the data download process commences. However, the steps differ slightly among hazard types. Earthquakes, Hail, Tornadoes, and Wind data are downloaded as CSV files, so a CSV subfolder (named "CSV_Folder") must be created prior to this occurring. Hurricane data is downloaded as a Shapefile, so a subfolder named "GIS_Folder" is created for its data. Any CSV files or GIS data downloaded from the web and created within the application will be automatically stored within these subfolders. With the subfolders created, the application attempts to download the CSV or Shapefile data from the URL path constructed from the user parameters. If the webpage is not accessible, the script will attempt the download one more time. If still unsuccessful, the user will be notified within the scroll box. If the data downloads successfully, it will be stored with a similar naming convention as the overall parent folder.

If the user is attempting to download earthquake data with a custom timespan or a custom magnitude, the download process changes drastically. USGS has a separate web portal for any data requests that aren't obtained from the "Past Hour", "Past Day", "Past 7 Days", or "Past 30 Days" live-feed URLs. Due to download rules set in place, a person is only be able to extract a certain number of records for each CSV download at one time. If utilizing a wide custom timespan, this can create problems. USGS only allows visitors the ability to download up to 20,000 earthquake records at a time. This equates to approximately two months of data only. Any record count above 20,000 will yield an error message from USGS (See **Appendix – Figure 16**). Manually downloading these records for a timespan consisting of multiple years would be extremely difficult and time-intensive. However, this application has been programmed to account for this problem.

If the user selects a custom timespan exceeding one calendar month, the Python scripting will recognize this and modify the download process accordingly. For example, if a user wants earthquake data from 01/2010 to 12/2010, the code will download the 01/2010 earthquakes to a CSV, then the 02/2010 earthquakes to a CSV file, and so on and so forth until all CSV files are downloaded. After downloading all CSV files, the code will then loop through all of them, appending them together into a single CSV file. The application also accounts for the excess headers present within all of these individual CSV files and prevents them from being duplicated within the appended CSV file. Each monthly CSV file that is downloaded will have "monthlyCSV" within the naming convention, along with its specific monthly range listed. The final appended CSV file will include "appended" within the file naming convention.

Once the application has downloaded the raw data, the Python scripting goes through several quality checks of the CSV data. There are specific column fields that are critical to future processing tasks, such as latitude, longitude, magnitude, event type, and date. If there are latitude/longitude coordinates of 0, 0 within the data, those will be removed. Invalid latitude/longitude values not between -90/90 and -180/180 are removed as well. If a magnitude value is blank, it will be removed. If there are hazard types within the earthquake CSV file that do not match, those will be excluded. For example, USGS includes many man-made seismic activities (such as explosions) within earthquake spreadsheets. As these are not naturally occurring earthquakes, the code removes any non-earthquake event types from the CSV file. Additionally, if any CSV files are downloaded without header rows included, those will be reconstructed and added to the file. Once the CSV file is checked, it has "_checked" added to the end of the naming convention.

Although not identical in some quality control checks, the previous process is accomplished similarly for hurricane Shapefile data. The application copies the raw Shapefile as a feature class into an empty file geodatabase within the "GIS_Folder". Once copied, the feature class undergoes data quality checks via ArcPy "UpdateCursor" operations. Any polyline features less than 1600 meters in length (approximately one mile) are removed. If no features exist within the feature class after all data quality checks are completed, the user will be notified and the application will terminate further operations.

After completing the quality checks of the downloaded files, the application proceeds to convert the applicable CSV files into GIS data. Prior to conversion, one final check of the CSV file is conducted to ensure that feature rows actually exist within the file. Similar to how the hurricane feature class was checked, if no features are present within the CSV file, then there is no GIS data to create. A message will appear for the user, stating that no features exist. The application will stop at that point, if this scenario occurs. However, if hazard data does exist, the code will proceed to create a GIS subfolder called “GIS_Folder”. Inside this subfolder, the script creates a file geodatabase where the CSV file is converted into a feature class and stored with a WGS84 geographic coordinate system. For the purposes of this project, all hazard data derived from CSV files are converted into point feature classes, specifically.

After each hazard’s feature class has been created, a projected coordinate system is assigned. For the purposes of this project, WGS1984 Web Mercator Auxiliary Sphere was chosen with a Central Meridian offset of -30.0 degrees. There is a valid reason for adjusting the Central Meridian, and it has to do with the U.S. Census Bureau Shapefile that is utilized for clipping purposes (there will be more details on this within the Results and Discussion section).

If the user did not check the “More Options” box, the script has succeeded and is finished. The scroll box will display the elapsed time it took for the application to execute all operations. Immediately following this notification, all text within the scroll box will automatically save to a text log file within the output workspace folder. It will be titled “ProcessingHistory_{Workspace Folder Name}.txt”. This allows the user to review all messages, errors, etc. that the application produced. However, if the check box is selected, additional tasks proceed once the feature classes have been created.

The next step involves downloading the U.S. Census Bureau's Shapefile. It is downloaded within a zipped folder, which is automatically extracted by the application after download. The raw Shapefile has a geographic coordinate system of NAD83, and consists of all 50 U.S. states, all U.S. counties, and all U.S. territories (e.g. Puerto Rico, Guam, among others). Since these extra territorial polygons are not needed, some data modifications are necessary to extract the required polygons.

To accomplish this, the script converts the entire Shapefile into a feature class and copies it into the same file geodatabase with the hazard data. Next, the feature class is references against a Python dictionary containing each state and its respective FIPS code designation. Depending on the user's choice, the feature class is updated to remove all polygon FIPS codes not affiliated with the user's clipping selection. If the USA clipping option is selected, all FIPS codes for the 50 states and Washington, D.C. are extracted. If a state is selected, only that state's FIPS code is extracted. If a county is selected, another Python dictionary is utilized. This dictionary matches all county names to each state. When the user selects a specific county within a specific state, the application first matches the state information before matching the county name information. The processing is done this way due to multiple states having identical county names (e.g. Washington County exists in 31 states).

Once the area of interest has been extracted, the WGS1984 Web Mercator Auxiliary Sphere projected coordinate system with a Central Meridian offset of -30.0 degrees is added to these polygon features and the extent recalculated. The application then proceeds to clip the point-based hazard features to the polygon clipping choice selected by the user. The clipped output is saved as a new feature class within the same file geodatabase, with "clipped_" added to the naming convention of the new feature class. If no features exist within the clipped output, the user will be notified and the application will finish. This could occur for some hazard types in areas where that hazard data may not exist (e.g. earthquakes for last 7 days in Florida, hurricane data for 2017 in North Dakota).

The clipping operations are conducted differently for the polyline hurricane data. A hurricane's polyline path is not sufficient enough to accurately represent a hurricane's impact on geographic areas. The eye of a hurricane, on average, spans between 20 and 40 miles in diameter, with the surrounding eyewall possessing the most intense wind speeds within the storm (NOAA, 1999). With this information, the decision was made to include any polylines within 50 miles of the user-specified clipping option. In the same fashion, any geographic locations within 50 miles of the hurricane polyline would be impacted by the storm. This would reflect a more accurate (although not exact) representation of hurricane impact. This distance would account for the eye of the storm, as well as the surrounding eyewall. For example, if a hurricane appeared within 50 miles of the South Carolina shoreline, it would be included with the clipped hurricane data for South Carolina. In order to accomplish this, the application dissolves the user-specified clipping extent and then creates a 50 mile buffer around that polygon. The output buffer polygon feature class is stored within the same file geodatabase with "buffer_" included within the naming convention. This buffer feature class is then used to clip the hurricane polyline data.

With the hazard features clipped, any analysis options selected by the user may now proceed. Some analysis options have limitations concerning sample size requirements, with some requiring a *minimum* number of sample points to operate. Others may not be able to function *beyond* a certain sample size. Per Esri, the Natural Neighbors analysis will fail if the sample points approach 15,000,000 in total. Optimized Hot Spot Analysis requires a minimum of 30 sample points and 60 sample points, depending on the type of input parameters set. Error messages will notify the user that the analysis has been skipped if these situations occurs. Adding to that, if any analysis fails for any reason, that particular analysis option will be skipped without breaking the application. It will continue to proceed through all selected analysis options, regardless if they fail or not. The user will be notified of all successes and failures within the scroll box messages.

During each analysis, the geoprocessing tasks will mask the data to the extent of the user-selected clipping option. Then, the output will be masked to the clipping option again. This will allow the output feature class to appear more presentable to the user once they import the results into a map.

If the analysis options are successful, all output data are saved as feature classes within the same file geodatabase with the Census polygons and hazard data. Each output will have the analysis name added to the beginning of the feature class naming convention. For example, an IDW output will have "idw_" added to the beginning of the feature class name.

As previously discussed, the "Output Count Details to CSV File" is an analysis option customized exclusively for this application. The count totals are formulated with, and reliant upon, many of the previous processes conducted within the application. For the point-based feature classes, several Spatial Joins are conducted in order to obtain the various count totals for each hazard type (e.g. Worldwide count – if applicable, nationwide count, statewide count, county counts). For the polyline Hurricane data, a "union" is conducted between the user-defined clipping polygons and the buffered polygons created from the polylines[25]. A Search Cursor is then utilized to extract the count totals of those overlapping regions within the union.

After all analysis options have concluded, the code calculates the length of time it took for the entire application to run and populates that elapsed time within the scroll box. Immediately following this notification, the processing history text file will save as well. The status bar will reset to 0% and stop, and the progress bar will update to reflect 100%.

The application does not automatically close or terminate upon completion. It stays open and active in case the user wishes to perform additional analysis on other data. All buttons, checkboxes and dropdown lists re-activate for the user to interact with once more upon completion. The Cancel button also returns to its original function as an Exit button (See **Appendix – Figure 17**). If no further action is required, the user can click the Exit button to close the application.

Results and Discussion

Depending on the parameters provided by the user within this application, the final output results for all hazard data and analyses will vary. For example, if a user performs all analysis options on all 2017 hail data within the state of Kansas, there will be a total of 13 feature classes within the file geodatabase located inside the GIS_Folder. In addition, there will be three CSV files observed inside the CSV_Folder. The processing history text file is present within the hazard folder alongside the GIS_Folder and CSV_Folder (See **Appendix – Figure 18**). Output results for each feature class within the aforementioned example can be observed in **Appendix –**

[25] A spatial join was first attempted, but the output was providing values that were not 100% accurate. Testing revealed a "bug" with the Python 3.x spatial join geoprocessing operation. Regardless if "INTERSECT" was utilized as the tool parameter's Match Option, results would still reflect "HAVE_THEIR_CENTER_IN" results. As a result, the lengthier "union" approach had to be used to ensure complete accuracy.

Figures 19 to 31[26]. As these figures show, valuable information can be extracted quickly and with ease. Earthquake, tornado, and wind output will be structured similarly to the hail output. Hurricane output will have a modified structuring. If a user performs all analysis options on 2005 hurricane data in the United States, there will be a total of seven feature classes within the file geodatabase located inside the GIS_Folder, with only one CSV file saved within the CSV_Folder. A processing history text file is also stored alongside the GIS_Folder and CSV_Folder (See **Appendix – Figure 32**).

Testing has shown that the length of time it takes for the application to complete all processes is dependent on three key variables: computer specifications, internet speed, and the size of data being processed. In the Kansas example above, all processes were completed in just over two minutes on a Windows 10 computer having 8 gigabytes of RAM with a 3rd Generation Intel i7 processor, while utilizing a 30 megabits-per-second download speed. Similar elapsed times occur for most situations the user may select. However, a few scenarios may require considerable time. For example, if a user attempts to retrieve all hurricane data from 1842 to present, with all analysis options selected, testing has shown this to take approximately three hours to complete. Even with such an increase in time, three hours is remarkably short considering how much data is being processed, cleaned, converted, and analyzed.

Several unique functionalities had to be incorporated and/or modified within this application due to certain “anomalies” discovered during testing. The first major obstacle encountered involved the hail, tornado, and wind hazards. All possess starting coordinates and ending coordinates within the CSV files, allowing for lines to be created. However, numerous

[26] Please note: The output results will not come with pre-defined symbology as these figures will show. Symbology has been manually added to each output feature class within ArcMap in order to provide better visualization.

data reviews have revealed that a large portion of the CSV files contain many missing ending coordinates. This inadvertently caused the geoprocessing function “XY to Line” tool to create many hazard polylines that would begin inside the United States and end at the 0, 0 coordinates on the west coast of Africa (See **Appendix – Figure 33**). This output would produce unusable data, often times causing the software to crash with the larger datasets. Due to the abundance of these discrepancies within these three hazards, this application was programmed to ignore the ending coordinate fields and only extract the starting coordinate values. This is why these features are created as point features instead of polyline features.

Additionally, coordinate system issues were encountered when extracting the Census Bureau’s Shapefile. The default WGS1984 Web Mercator Auxiliary Sphere was chosen initially, but due to extent issues caused by Alaska being divided along the International Date Line (See **Appendix – Figure 34**), a -30.0 degree shift along the Central Meridian was necessary for Alaska polygon selections (See **Appendix – Figure 35**). Otherwise, geoprocessing failures would occur during some analyses due to these abnormal extent issues. This occurred most frequently with the Kriging and Spline analysis options.

After programming the Optimized Hot Spot Analysis option, failures would occur at regular intervals in specific situations. Upon further investigation, only this geoprocessing tool would retain a feature class lock on the data, even after the application had concluded all operations. This would cause problems if the user attempted to overwrite, move, or rename the data. It was determined that including the optional “Analysis Field” parameter was the direct culprit. If this field was not assigned, the “frozen” feature class lock would not occur. Further testing revealed that, if this geoprocessing tool was executed with the optional analysis field parameter, and then re-executed without that parameter, the abnormal feature class lock would not remain. As a result, two version of the Optimized Hot Spot Analysis are included. When this analysis tool is utilized for the point-based hazards, an Optimized Hot Spot Analysis feature class with “magAnalysisField” and another with “noAnalysisField” are included with the output results.

Conclusion and Future Study

With this application, the user has the ability to easily retrieve and analyze data from five natural hazards. Being able to quickly retrieve hazard data, provide data quality checks, and immediately conduct various analysis options on geographic locations of the user’s choosing, all of this allows the user to focus more effort on data analysis and less effort on the time-intensive processes involved with preparing the data.

Additional hazard types could be added to this application to enhance its usefulness. Drought data, flood data, lightning data, blizzard data, wildfire data, among others could easily build upon the foundations of this Master’s project. More analysis options could be included to expand the versatility of the software. Additional enhancements as a Python toolbox could be included to allow the software to automatically map the output results, as well as provide options to publish online or print for those needing hardcopy options. The possibilities are numerous, and that is the beauty of programming: The user-friendliness can always be improved. These enhancements would only contribute to the greater good for all researchers or emergency disaster personnel needing to easily access these types of hazards and make sense of the information as quickly as possible.

References

Burkhard, Meier (2017, May). *Python GUI Programming Cookbook Second Edition*. Birmingham, UK: Packt Publishing Ltd.

Chaudhary, Bhaskar (2015, November). *Tkinter GUI Application Development Blueprints*. Birmingham, UK: Packt Publishing Ltd.

Deindorfer, Hannah (2017, August). *Esri's ArcGIS Blog*. "ArcGIS Pro Tips: Top 10 Questions from the 2017 User Conference". Retrieved from: https://blogs.esri.com/esri/arcgis/2017/08/15/arcgis-pro-tips-top-10-questions-from-the-2017-user-conference/

Federal Emergency Management Agency (2017, November). *Hazus in Detail*. Retrieved from: https://www.fema.gov/hazus-detail

Joy, Chheda, Chy, and Sun (2014). *Visualization of Tornado Data*. Retrieved from: http://cluster.ischool.drexel.edu/~cchen/courses/INFO633/13-14/joy.pdf

Lindsey, David (2017, August). *ArcGIS for Desktop Add-In Toolset for Retrieving and Displaying Natural Hazard Data*. Retrieved from: https://www.dropbox.com/s/yp5t6ihnz7xqdnz/ArcObjectsExample_DavidLindsey_Aug2017.zip?dl=0

Lubanovic, Bill (2016, February). *Introducing Python*. Sebastopol, CA: O'Reilly Media, Inc.

Lutz, Mark (2016, May). *Learning Python 5th Edition*. Sebastopol, CA: O'Reilly Media, Inc.

Madhavan, Renjith (2015, October). *An Analysis of Severe Weather Data: 2000 – 2015*. Retrieved from: https://www.researchgate.net/profile/Renjith_Madhavan/publication/283329659_Severe_Weather_Advisory_Based_on_Weather_Data_Analysis_2000_-2015/links/56340a9908aeb786b7013dbe.pdf

National Oceanic and Atmospheric Administration (1999, May). *Hurricane Basics*. Retrieved from: https://www.hsdl.org/?view&did=34038

Ottemaller, Lars (2014, October). The SEISAN Earthquake Analysis Software. *Incorporated Research Institutions for Seismology*. Retrieved from: https://www.iris.edu/hq/webinar/2014/10/the_seisan_earthquake_analysis_software

Pimpler, Eric (2015, July). *Programming ArcGIS with Python Cookbook Second Edition*. Birmingham, UK: Packt Publishing Ltd.

Pimpler, Eric (2015, December). *ArcGIS Blueprints*. Birmingham, UK: Packt Publishing Ltd.

Santos, Renato (2016, June). *Some Comments on the Reliability of NOAA's Storm Events Database*. Retrieved from: https://arxiv.org/ftp/arxiv/papers/1606/1606.06973.pdf

Toms, Silas, and O'Beirne, Dara (2017, June). *ArcPy and ArcGIS Second Edition*. Birmingham, UK: Packt Publishing Ltd.

Torgersen, Mads (2017, February). *The .NET Language Strategy*. Retrieved from: https://blogs.msdn.microsoft.com/dotnet/2017/02/01/the-net-language-strategy/

Williams, Weskamm, Baaser, Hinzen, & Bareth (2008, July). SeismoGIS: A Tool for the Visualization of Earthquake Data. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXVII. Part B8. Beijing 2008.* Retrieved from: http://www.isprs.org/proceedings/XXXVII/congress/8_pdf/12_WG-VIII-12/11.pdf

Zandbergen, Paul (2013). *Python Scripting for ArcGIS*. Redlands, CA: Esri Press.

Appendix

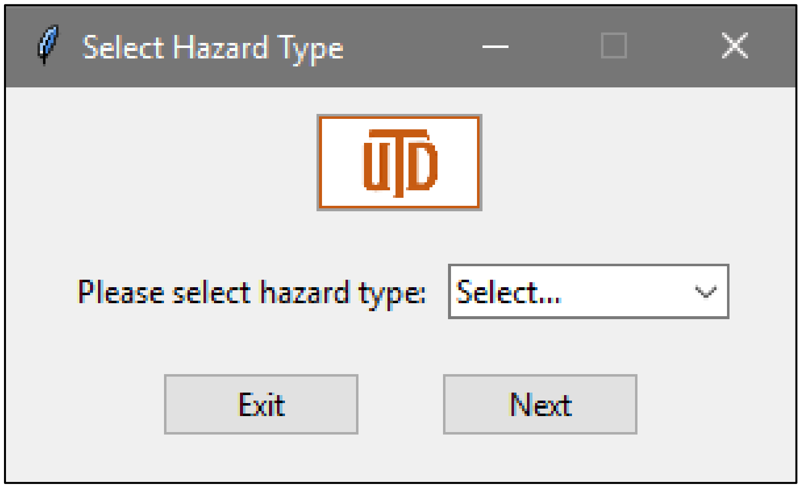

Figure 1

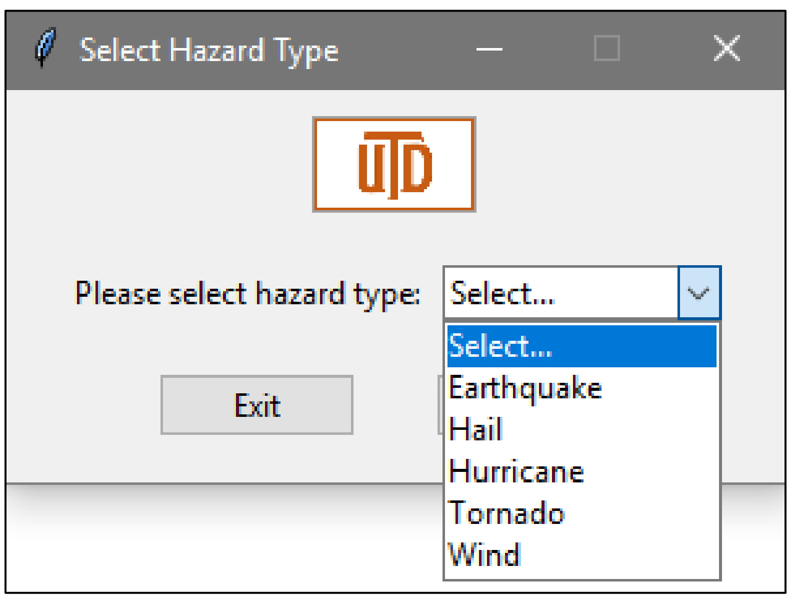

Figure 2

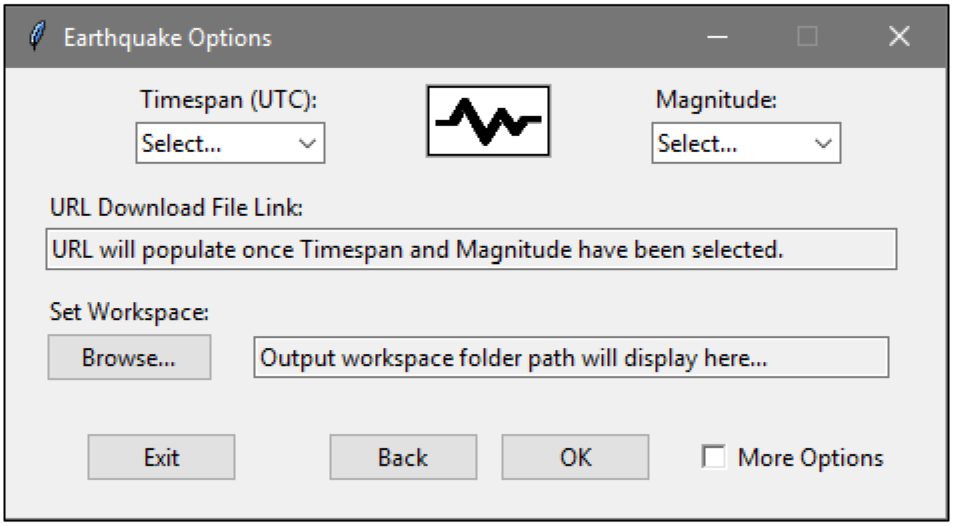

Figure 3

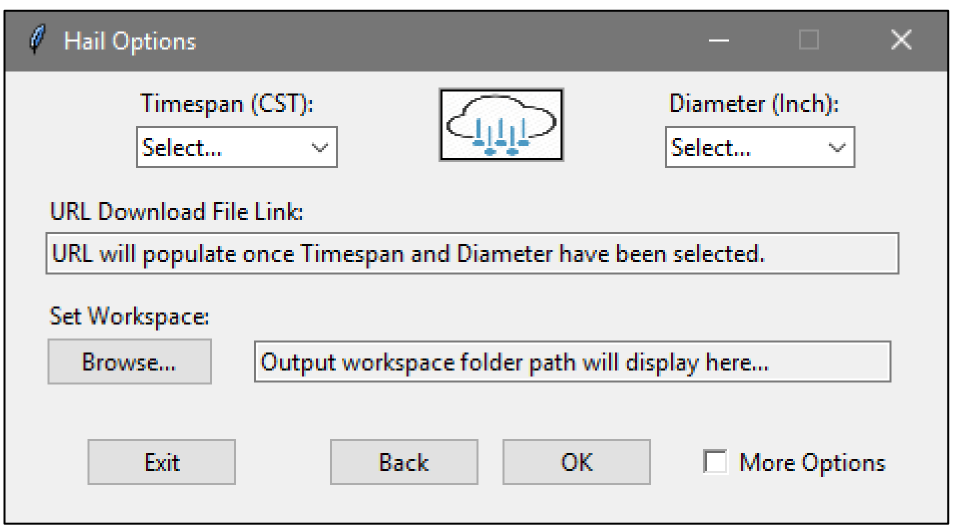

Figure 4

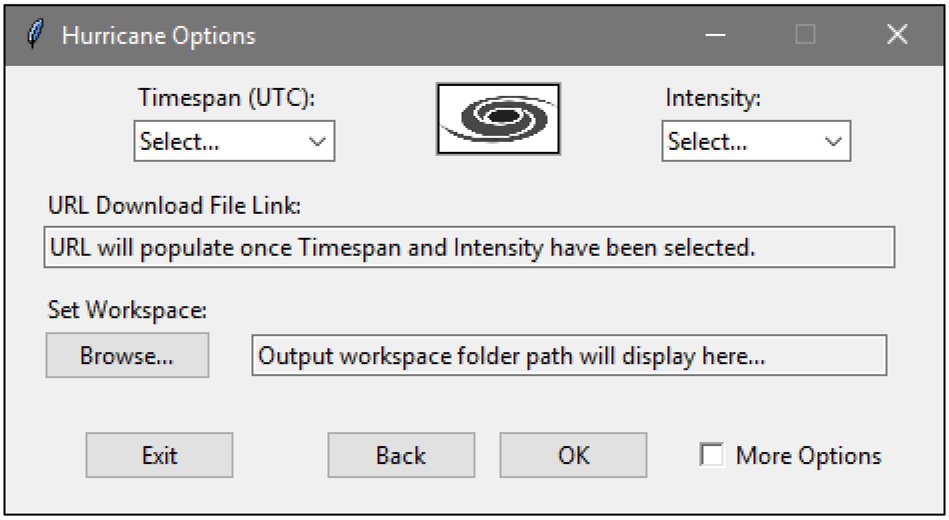

Figure 5

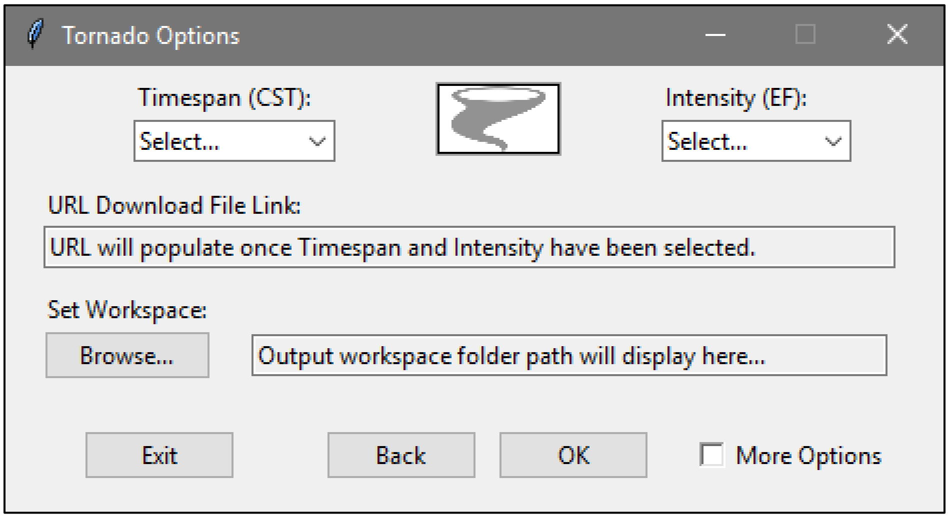

Figure 6

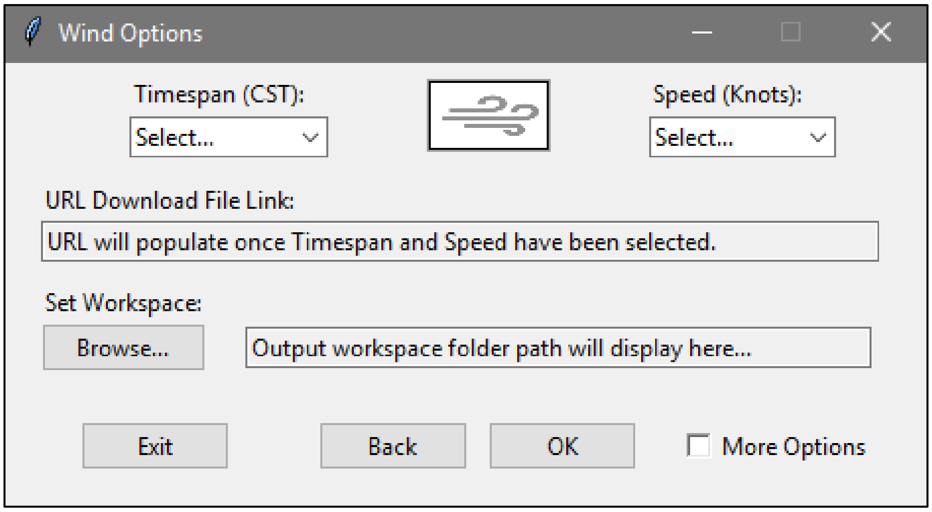

Figure 7

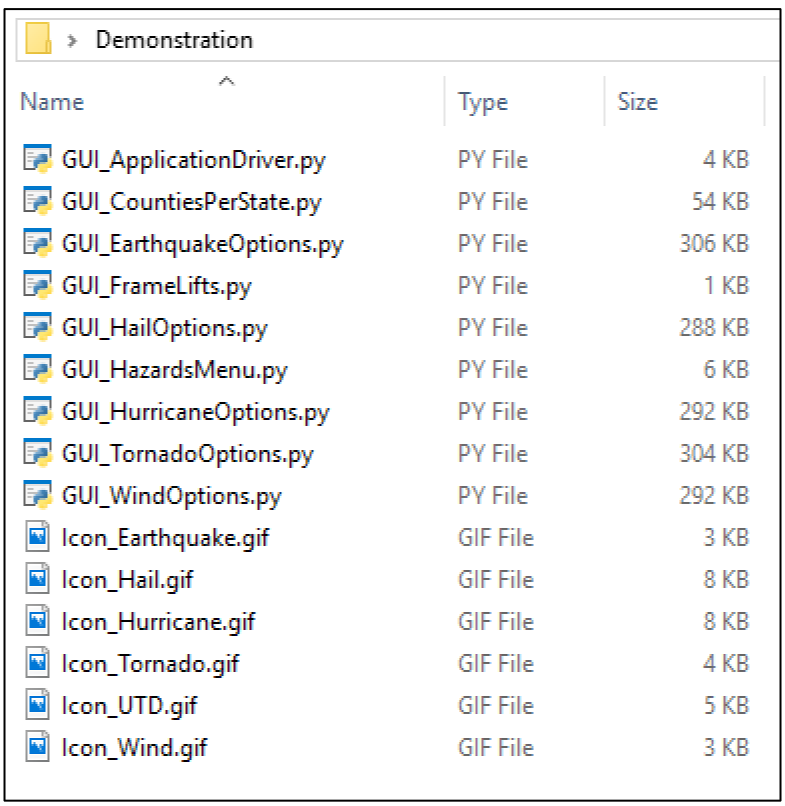

Figure 8

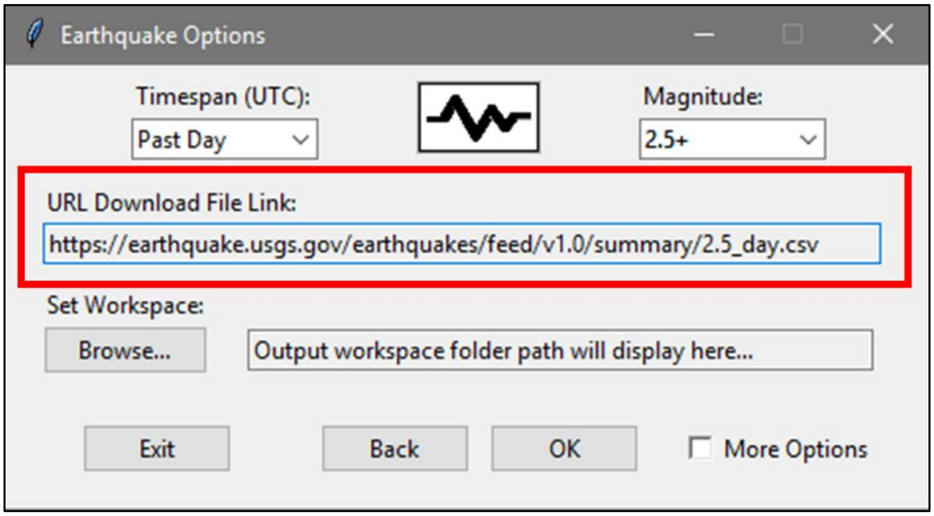

Figure 9

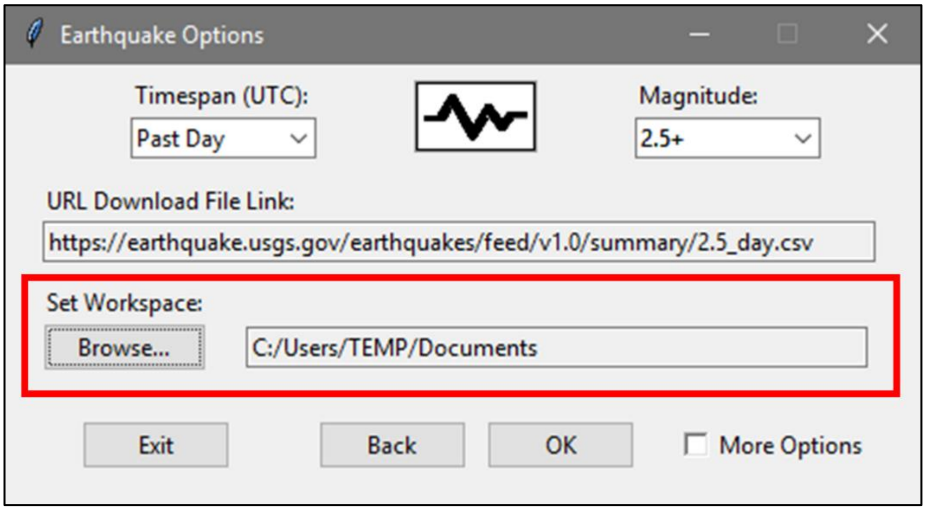

Figure 10

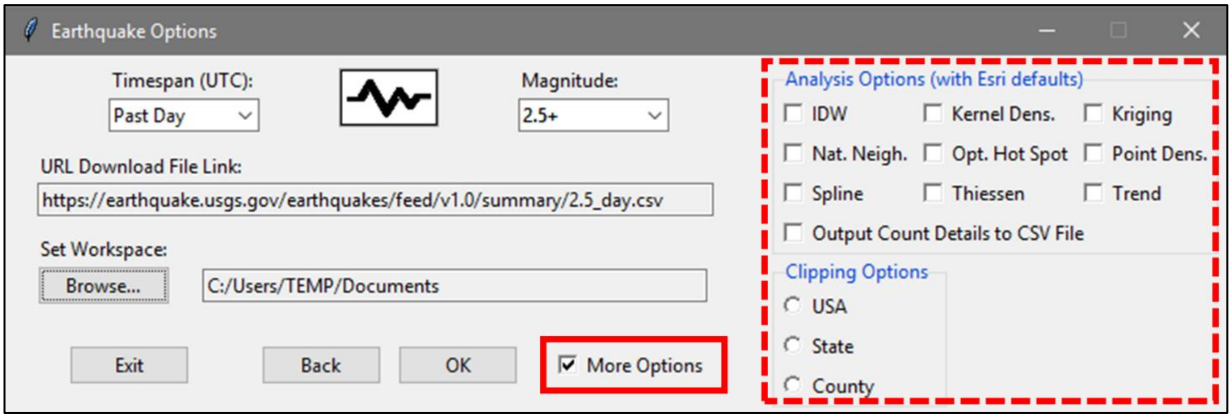

Figure 11

Figure 12

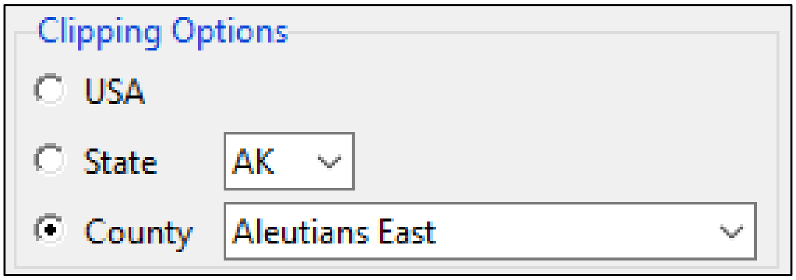

Figure 13

Timespan:	Past 30 Days					
Magnitude Range:	2.5+					
Worldwide Earthquakes:	1758					
USA (and DC) Earthquakes:	245					
State Earthquakes - CA (FIPS: 06):	56					
Statewide Earthquake Data						
V	V	V	V	V		
Minimum Magnitude	Maximum Magnitude	Mean Magnitude	Median Magnitude	Mode Magnitude		
2.45	3.87	2.86	2.79	No Unique Mode		
Earthquake Data per County						
V	V	V	V	V	V	V
Counties with Earthquakes	Earthquake Count	Minimum Magnitude	Maximum Magnitude	Mean Magnitude	Median Magnitude	Mode Magnitude
Alpine	1	2.82	2.82	2.82	2.82	2.82
Contra Costa	4	2.48	2.91	2.73	2.77	No Unique Mode
Fresno	3	2.57	2.95	2.81	2.92	No Unique Mode
Humboldt	2	2.55	2.58	2.56	2.56	No Unique Mode
Imperial	2	2.48	2.67	2.58	2.58	No Unique Mode
Inyo	1	2.73	2.73	2.73	2.73	2.73
Kern	2	3.69	3.78	3.73	3.73	No Unique Mode
Lake	4	2.46	3.49	2.85	2.73	No Unique Mode
Lassen	1	2.46	2.46	2.46	2.46	2.46
Los Angeles	1	2.5	2.5	2.5	2.5	2.5
Mendocino	1	2.84	2.84	2.84	2.84	2.84
Mono	2	2.57	3.6	3.08	3.08	No Unique Mode
Monterey	2	2.84	3.11	2.97	2.97	No Unique Mode
Riverside	5	2.68	3.87	3.15	2.91	No Unique Mode
San Benito	8	2.46	3.3	2.82	2.81	No Unique Mode
San Bernardino	4	2.84	3.29	3.09	3.12	No Unique Mode
San Diego	1	2.47	2.47	2.47	2.47	2.47
San Luis Obispo	2	2.61	2.93	2.77	2.77	No Unique Mode
Santa Barbara	1	2.53	2.53	2.53	2.53	2.53
Santa Clara	3	2.46	3.82	2.97	2.64	No Unique Mode
Shasta	2	2.52	2.8	2.66	2.66	No Unique Mode
Solano	1	2.61	2.61	2.61	2.61	2.61
Sonoma	3	2.45	2.71	2.62	2.7	No Unique Mode

Figure 14

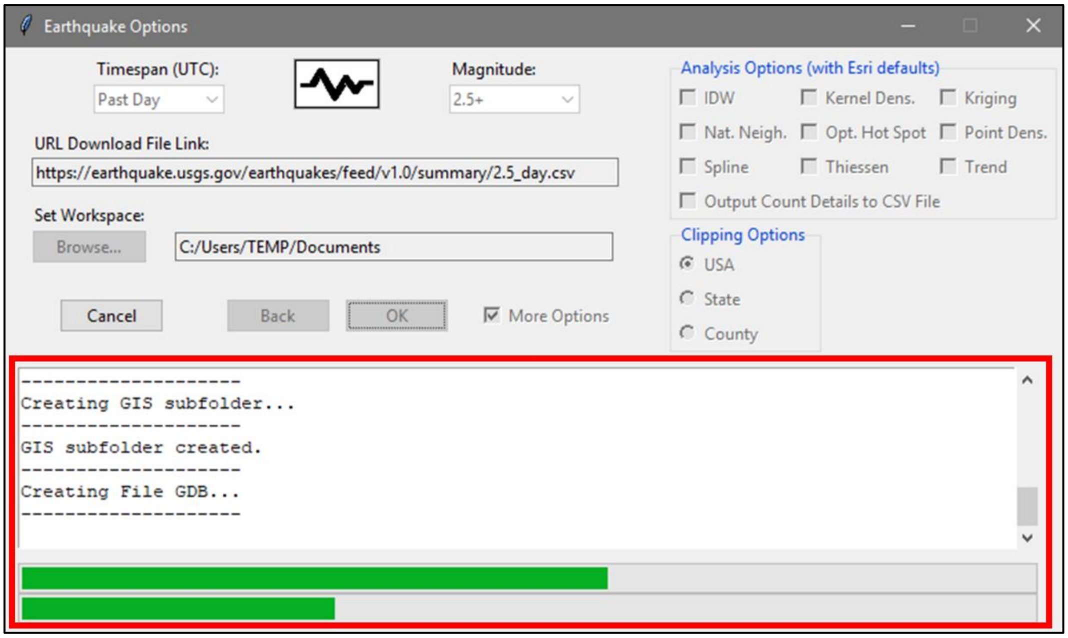

Figure 15

```
Error 400: Bad Request

33908 matching events exceeds search limit of 20000. Modify the search to match fewer events.

Usage details are available from https://earthquake.usgs.gov/fdsnws/event/1

Request:
/fdsnws/event/1/query.csv?starttime=2018-01-18%2000:00:00&endtime=2018-04-25%2023:59:59&minmagnitude=0&orderby=time

Request Submitted:
2018-04-25T12:26:16+00:00

Service version:
1.5.8
```

Figure 16

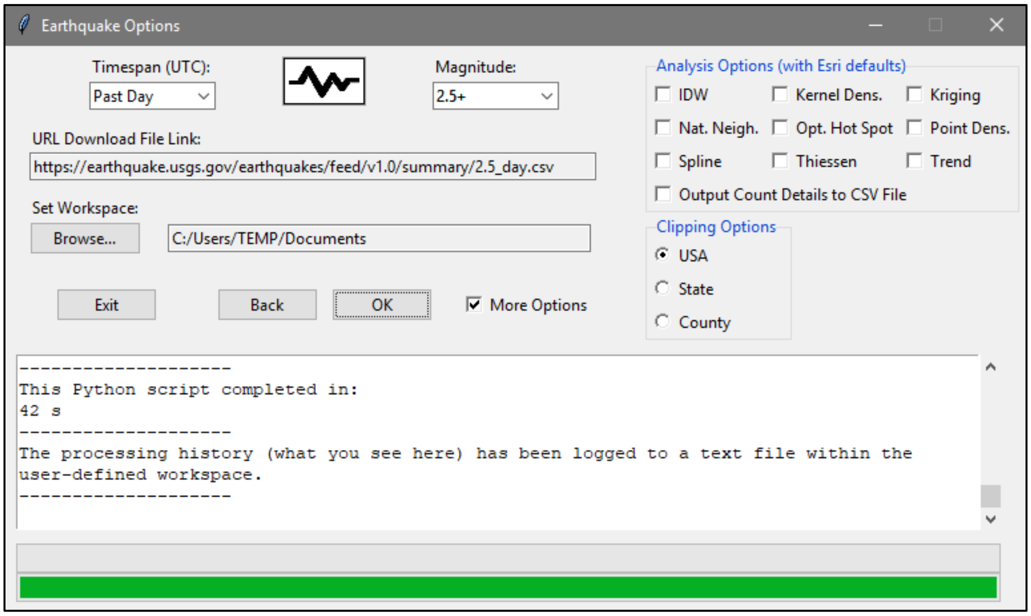

Figure 17

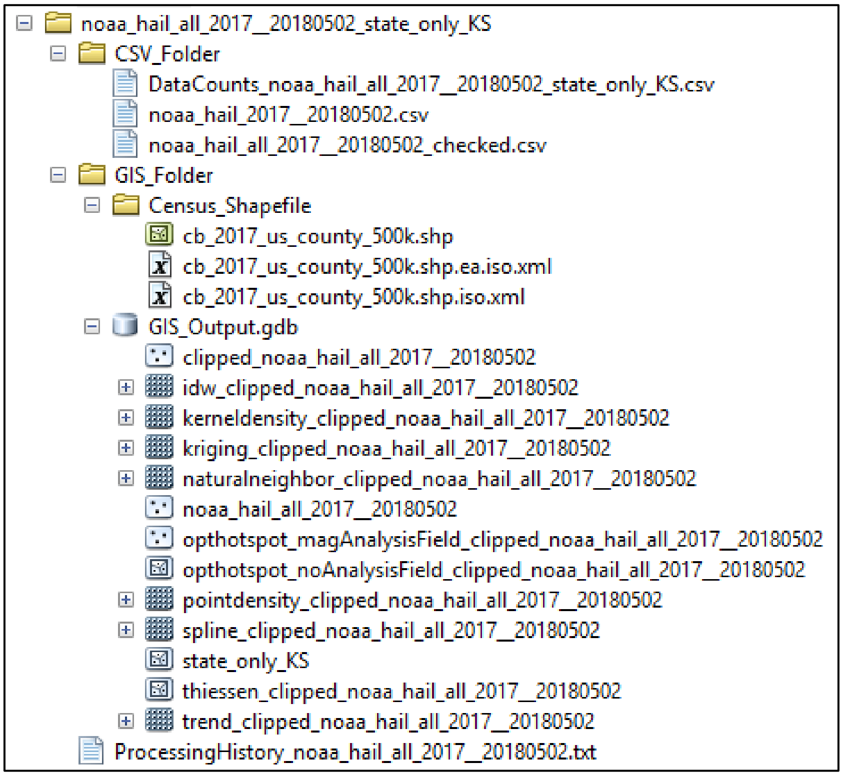

Figure 18

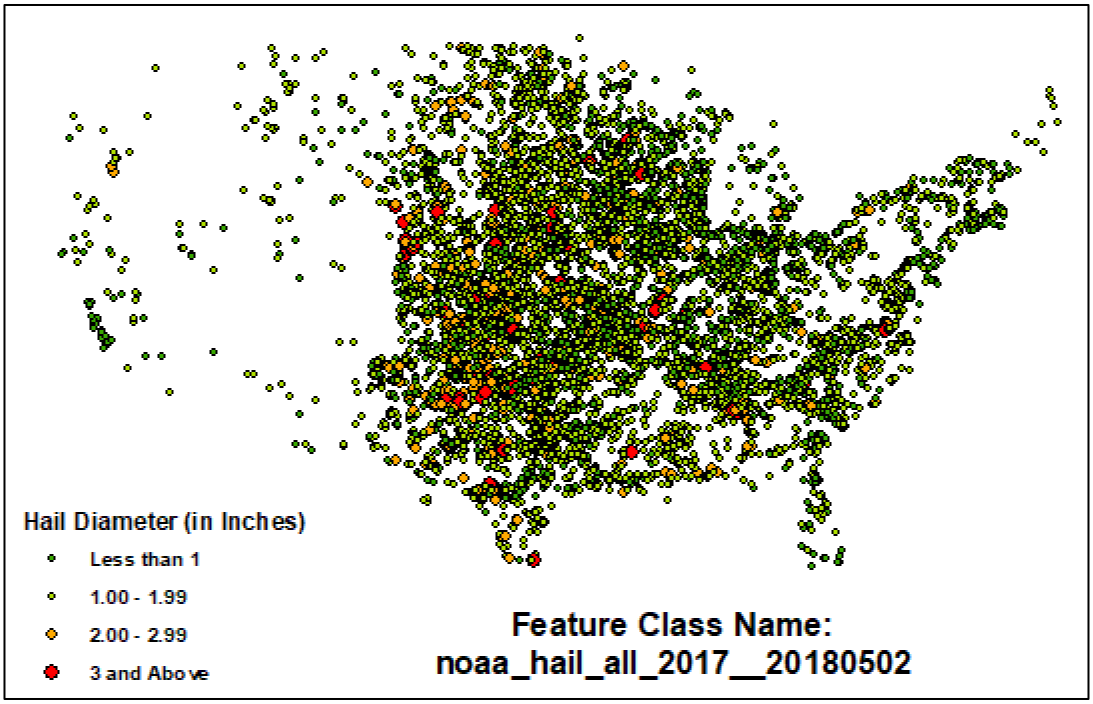

Figure 19

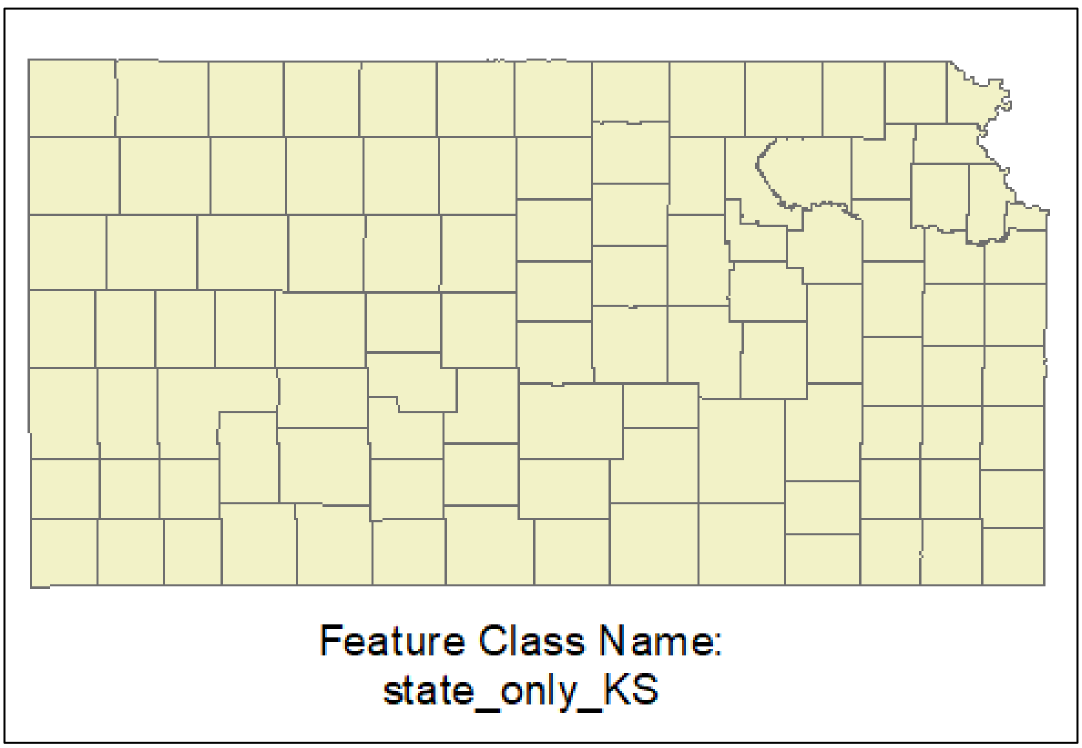

Figure 20

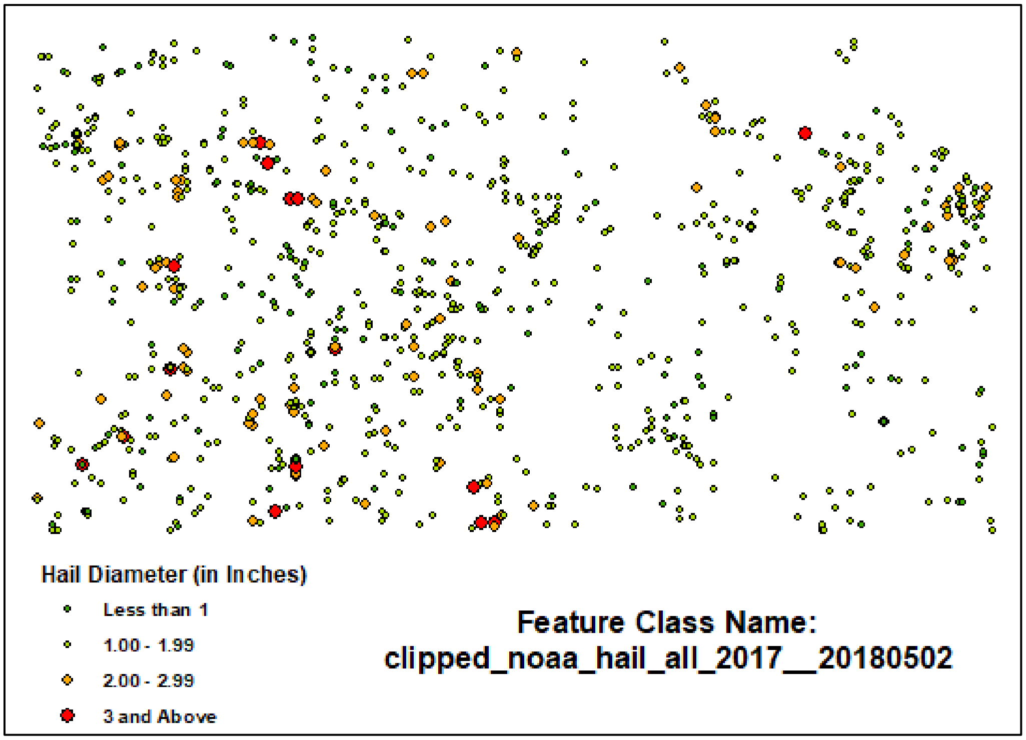

Figure 21

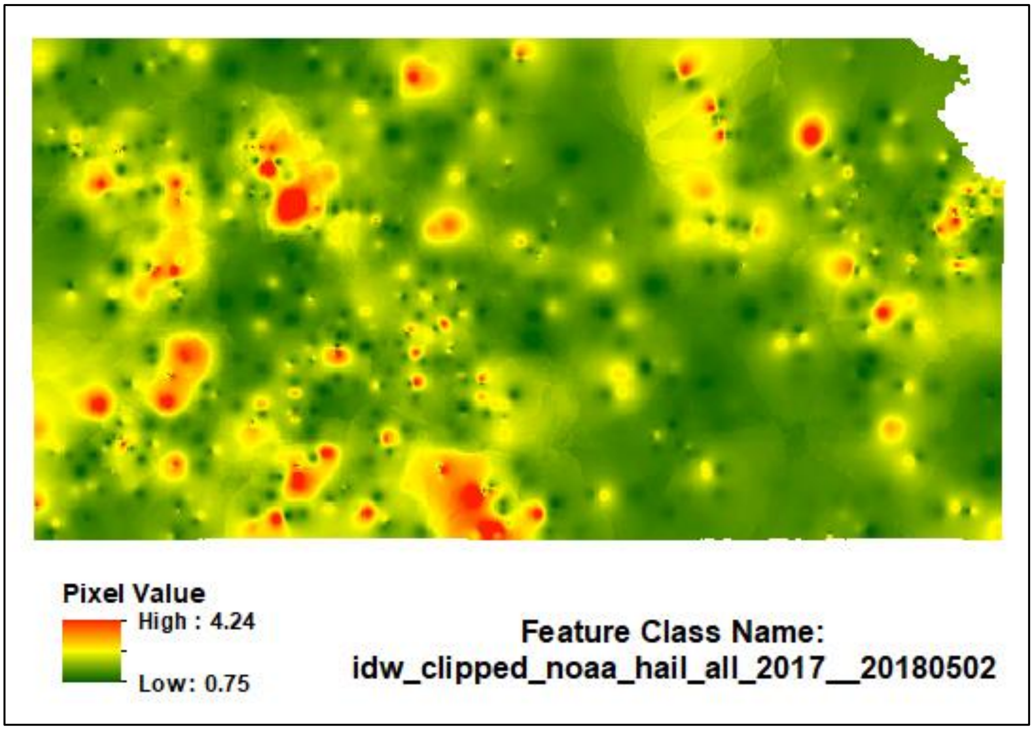

Figure 22

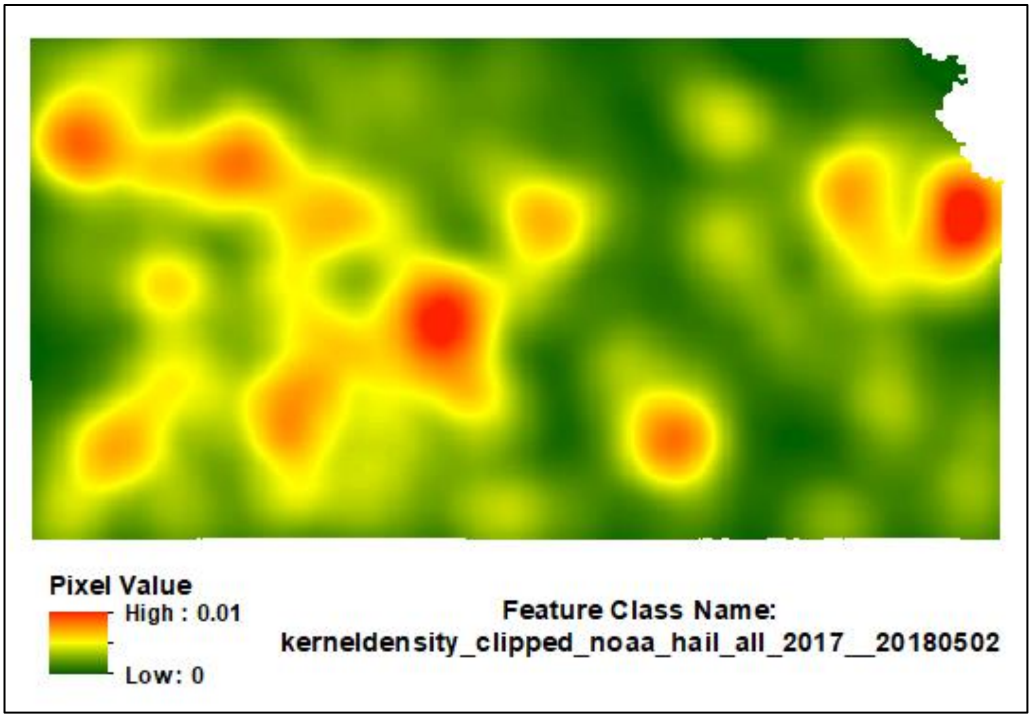

Figure 23

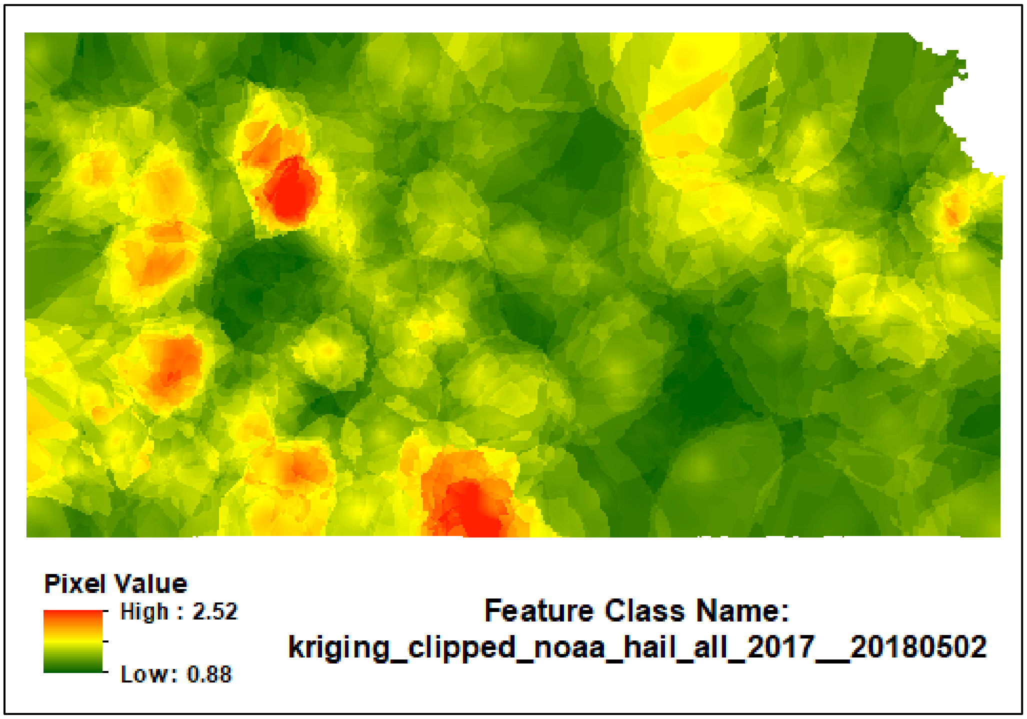

Figure 24

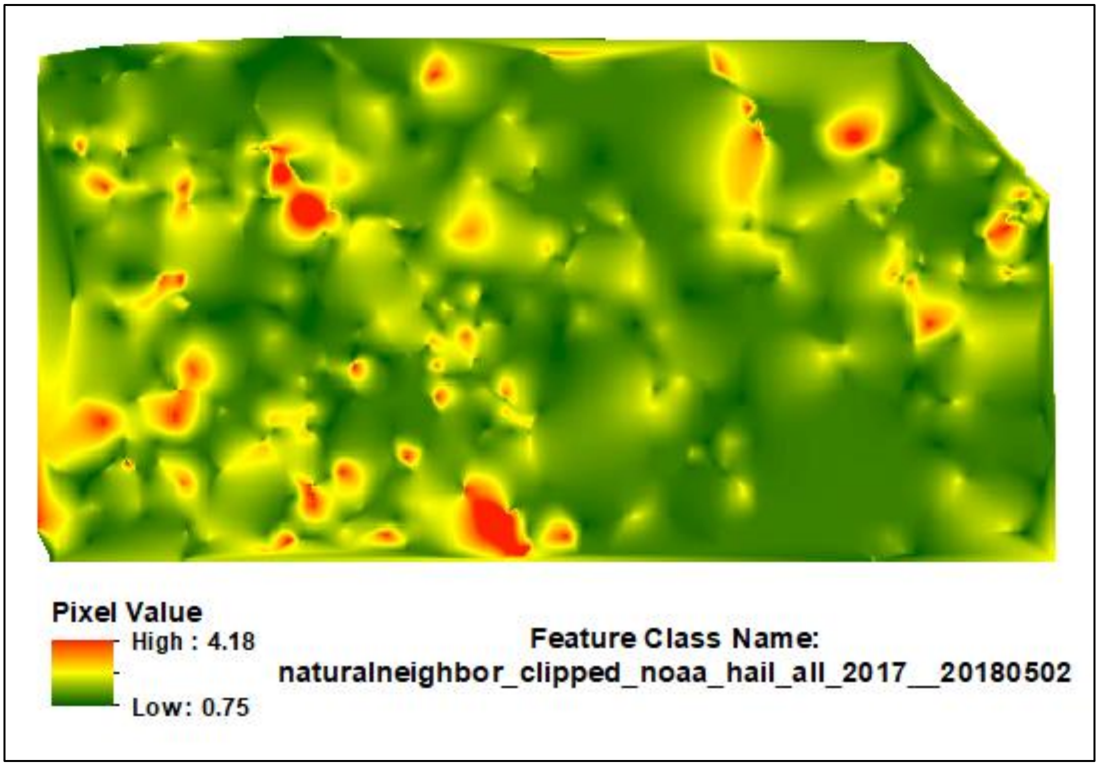

Figure 25

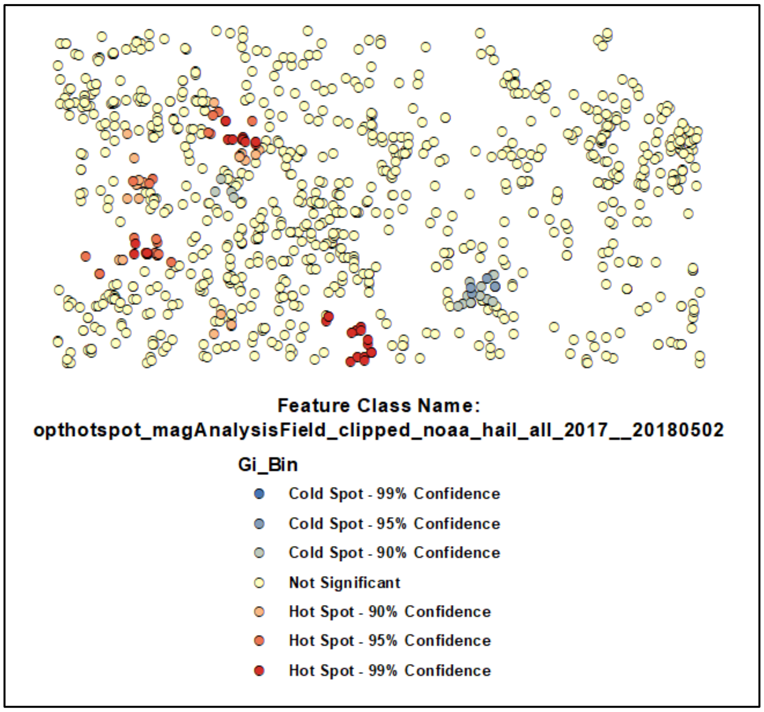

Figure 26

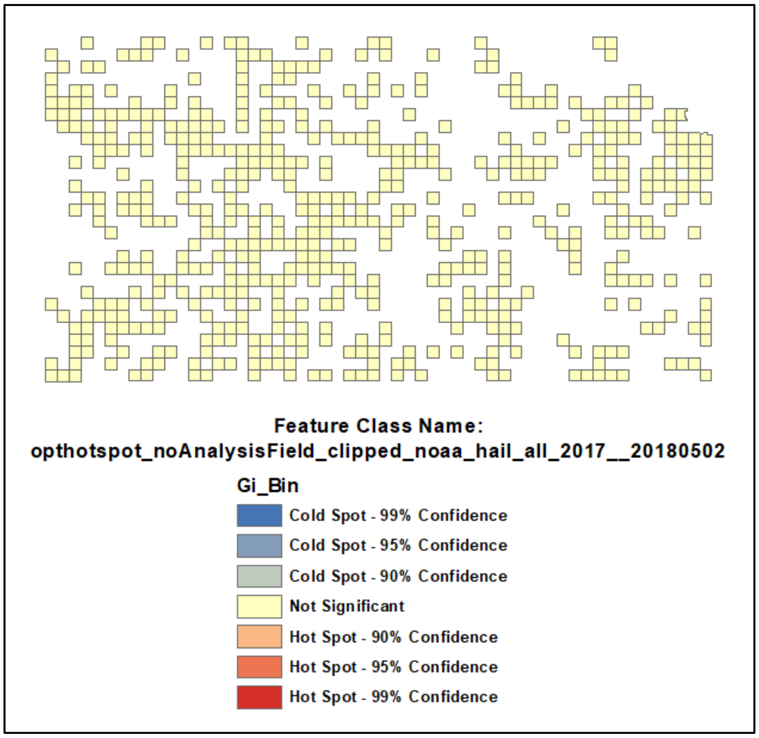

Figure 27

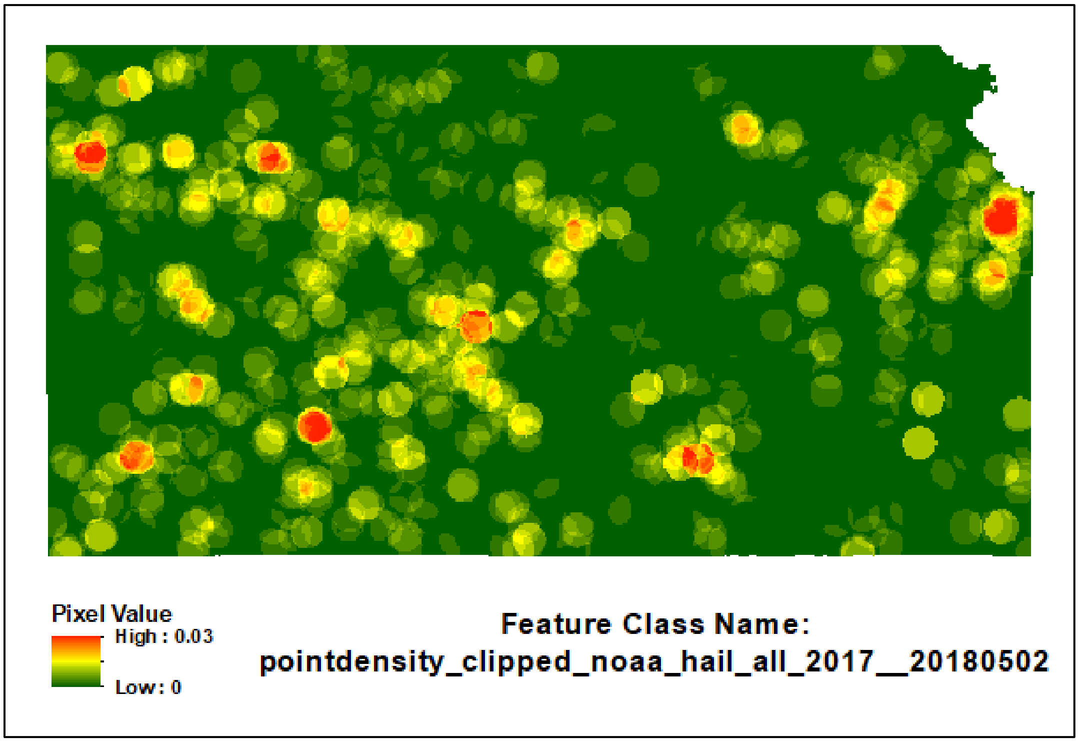

Figure 28

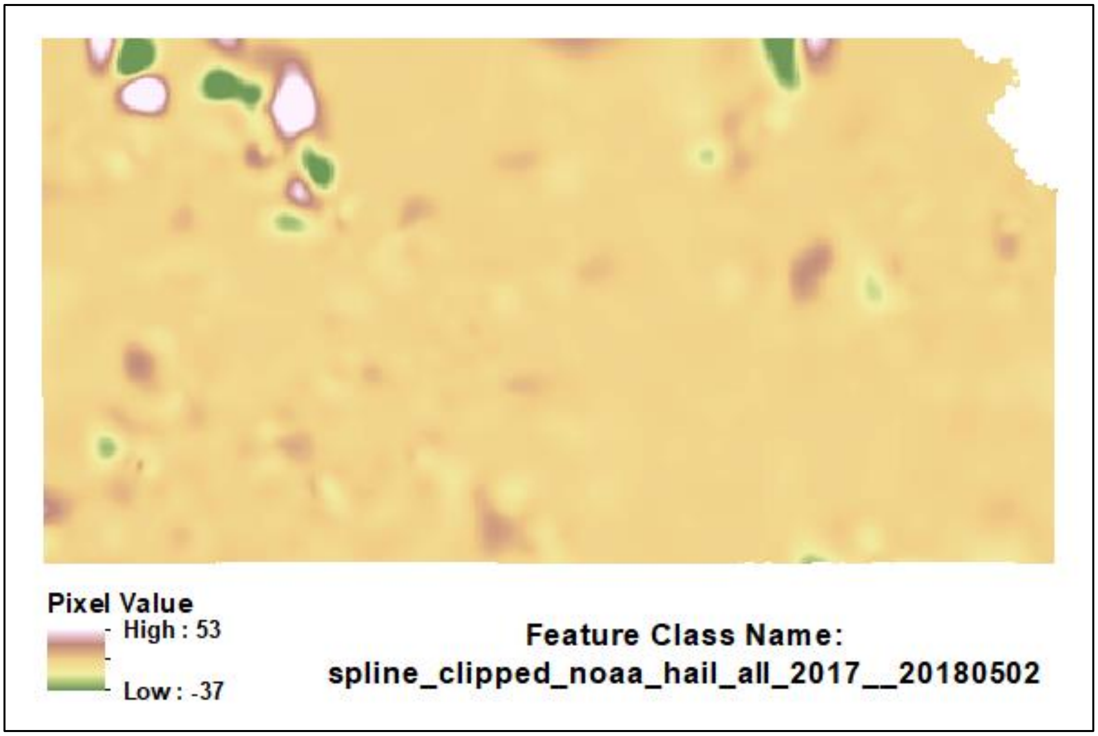

Figure 29

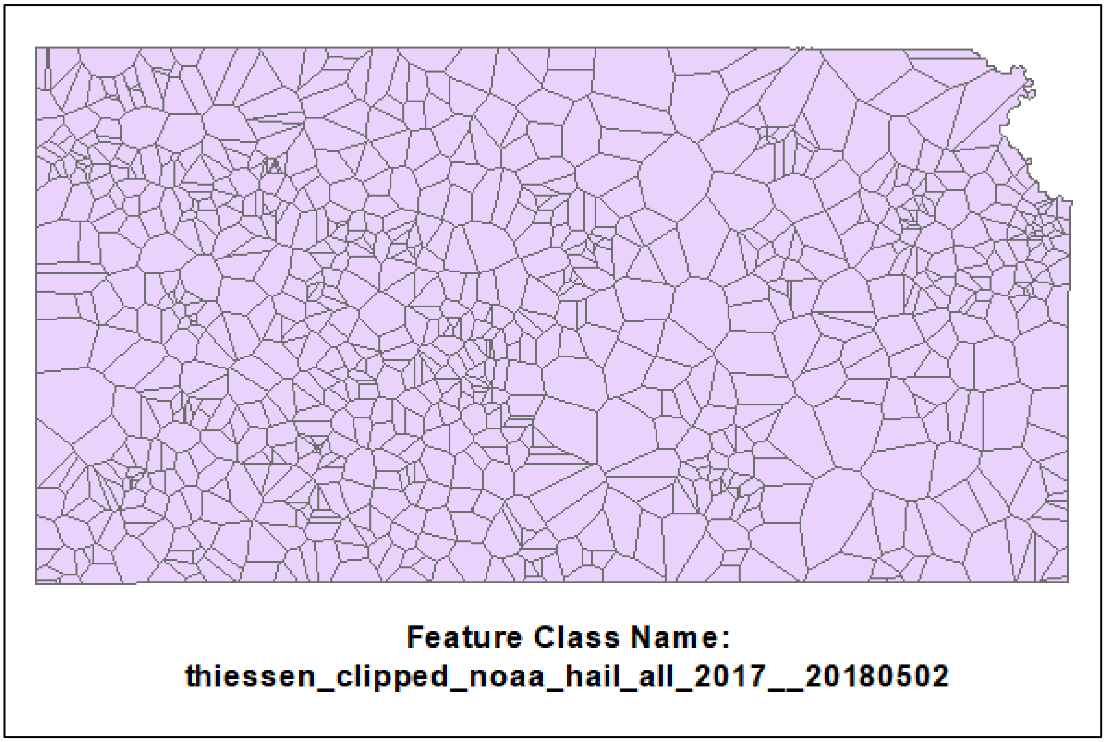

Figure 30

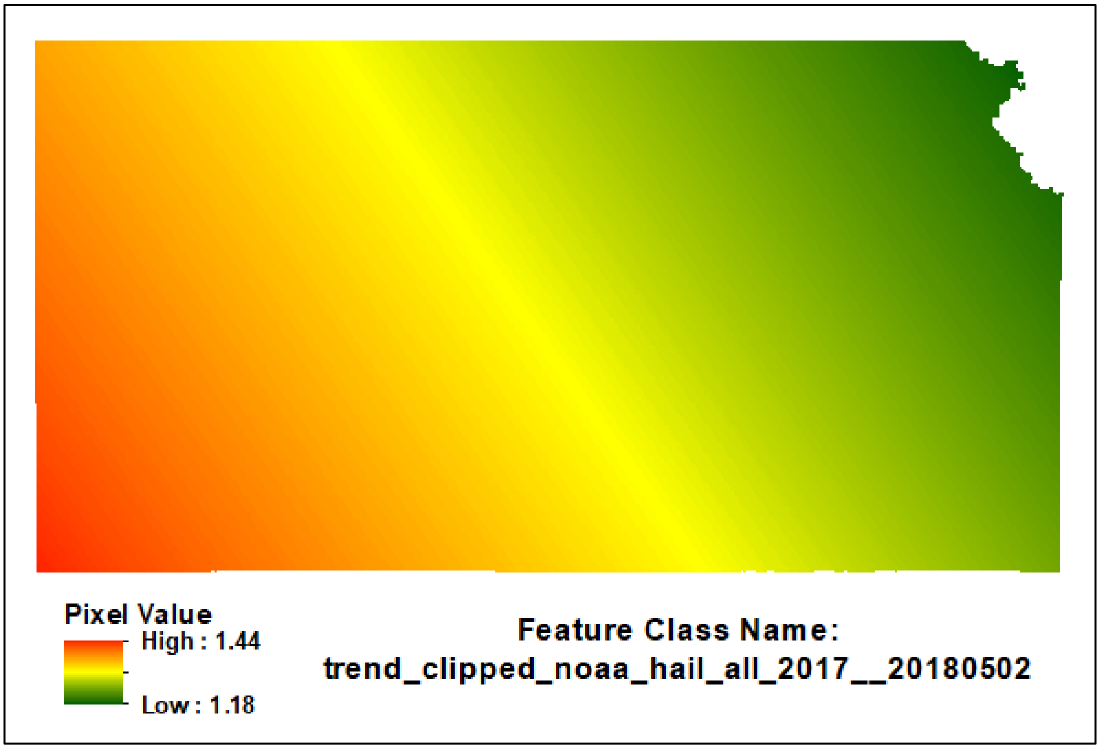

Figure 31

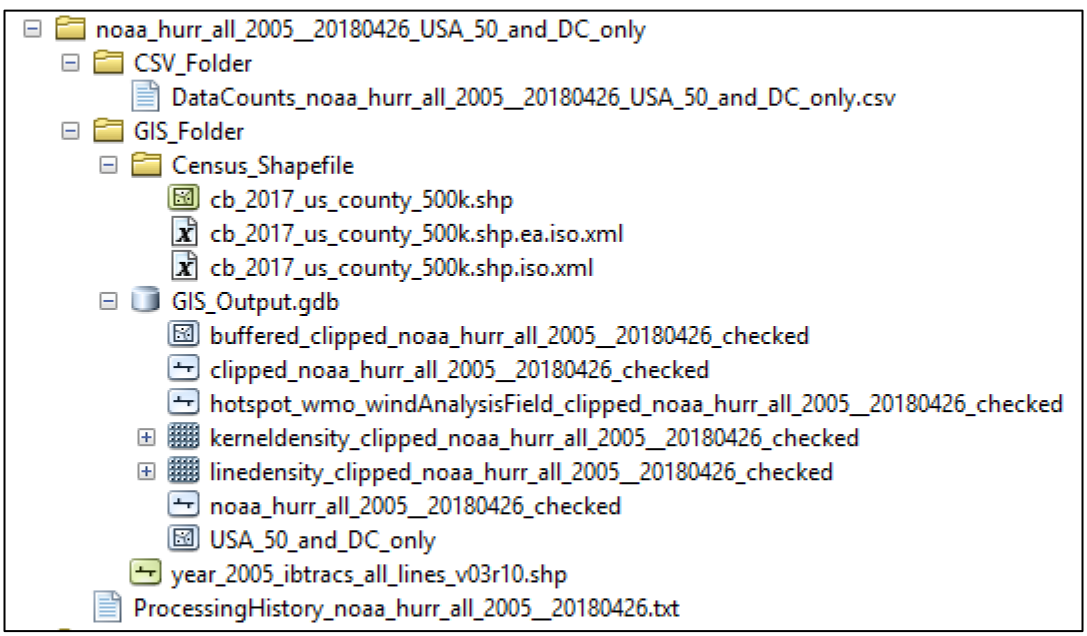

Figure 32

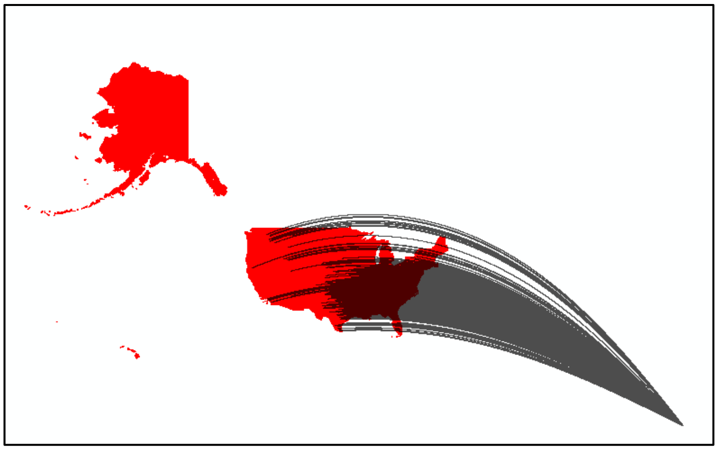

Figure 33

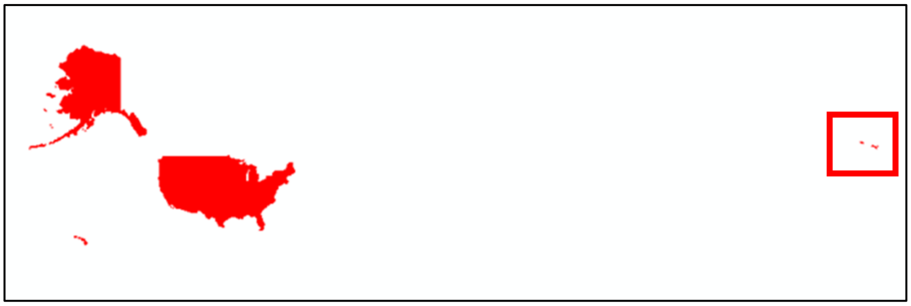

Figure 34

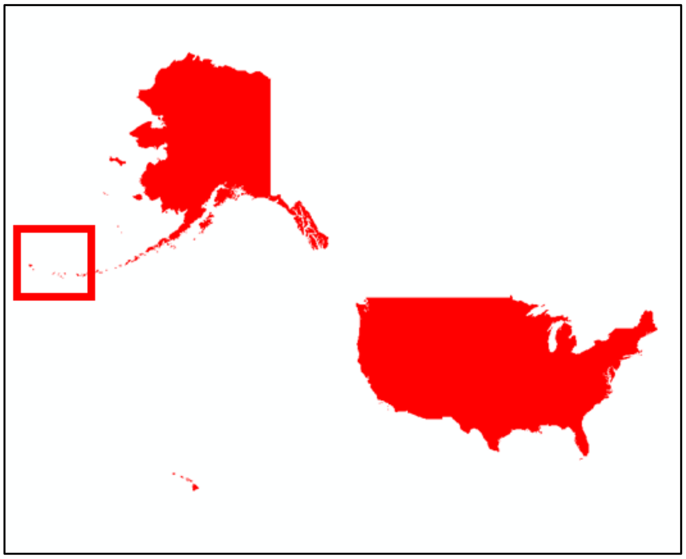

Figure 35