Project Name: Analyzing the relationship between roadways and flood zones for a selected study area

By: Noah Puchovsky

Share: You may share my project with future classes

Abstract:

The goal of this project was to take a study area and perform an analysis on the relationship between the study area's roadways and flood zones. Flooding is a significant natural hazard that is common across the globe. Parts of every single country in the world are subject to flooding. Severe flooding situations can lead to injury, death, and cause significant property damage to both buildings and infrastructure. Due to the changing climate across the globe, the severity and frequency of flooding scenarios has grown. I currently work in the insurance industry and our most common natural hazard losses stem from flooding. Scientists and engineers are constantly developing new flood models to get a better understanding of the flood risk for a certain area. Understanding the flood risk of an area allows for both people and businesses to better prepare for worst-case scenarios.

An item that can be overlooked when developing a flood emergency response plan is how the roadways will be affected by a flood. Concerns of flooded roadways include unusable roads, blocked evacuation routes, hindering emergency responders' path to a location, and creating an overall dangerous driving experience.

In this project, I have developed a python script that will allow the user to select a town or city and see the relationships between that town's roads and the official flood zones. Using data available from the state of Massachusetts I was able to complete a roadway flood analysis for the town of Mendon, Massachusetts. The script allows this analysis to be easily repeated in other towns and states.

Introduction:

The research question that this project looks at is "which roads will be affected by a flood in a given study area?" The motivation behind this project is my interest in natural hazard modeling and response. I currently work as a risk engineer at a large insurance carrier and complete natural hazard analysis during my everyday job. I work with insured clients everyday to help them understand their exposure to flooding and what they can do to try and mitigate any potential injuries or damage caused by flooding. Something that is commonly overlooked when developing a flood emergency response plan is how the flood scenario will impact nearby roadways. The goal of this project was to highlight which roadways may become unusable during a flood situation.

This tool would be beneficial to any emergency response planner either for a government organization or private organization. This tool will allow a user to select a certain city or town and create a map of the official flood zones and roadways for that area. Then using that information, geoprocessing tools are used to look at the relationships of the roadways and flood zones. This tool will allow for an emergency response planner to make a better plan for evacuation or emergency responders access to a critical location. For evacuation purposes, this will let the emergency response planners develop alternative routes by knowing which roads are most likely to be impacted by the flood. For emergency responders this tool can be used to develop alternative routes that will allow the quickest response time to a flooded area. In some situation flooding leads to people needing medical attention and response time by ambulances becomes extremely important in saving someone's life. Fire department response time is also essential in flood scenarios as fire can begin from short circuits caused by the flooding.

The tool that was developed for this project will be essential for emergency response planners and responders to create the best possible plans to mitigate the impacts of flooding. This is a topic that I am passionate about and is part of my everyday job.

Materials and Methods:

The first step in the project was choosing a study area to be the basis for future replications of the analysis. The study area that I chose to complete the initial analysis for was the town of Mendon, Massachusetts. This area was chosen as it is the town that I grew up in. Now that the study area was selected it was now time to gather appropriate data. The first piece of data that needed to be collected was a shapefile of all the cities and towns in the state of Massachusetts. I was able to gather this data from the mass.gov website. On the mass.gov site there is a separate section for MssGIS Data layers. The first file that I would be selecting from this site was the "2020 U.S. Census Towns" file. This file contains a shapefile of all the towns and cities in Massachusetts.

The next dataset that I needed was a dataset of all the roads in Massachusetts. This data set can also be found on the MassGIS Data Layers page on mass.gov. The title of this dataset was "2020 US Census TIGER Roads" file. This dataset contained a shapefile that has all roads in Massachusetts as line features.

The final data set that I would be gathering for this project was a data set that contains all FEMA Flood zones in Massachusetts. FEMA is the federal emergency management agency for the US and models flood zones for the entire country. This data was also obtained through the mass.gov website in the MassGIS Data Layers page. This shapefile contained polygons for all official flood zones as designated by FEMA.

Now that all the data was collected by reputable sources it was time to start developing the python script for analysis. The first step of the process would be to develop a list of all towns and cities in the state. To do this a function would need to be made that extracted all the town/city names in the dataset and put them in a separate table that lists the names in alphabetical order. A function was made that created a table of all available towns/cities that could be analyzed. To complete this a combination of arcpy tools would need to be used including CreateTable, AddField, SearchCursor, InsertCursor, and insertRow. Using the provided python functions a table listing only town/city names in alphabetical order could be made.

```
def list_towns():
    # set environment
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/mass_towns"

    # create list to store names of towns
    names = []

    # set variables for fc and field
    fc = "CENSUS2020TOWNS_POLY.shp"
    field = 'NAMELSAD20'

    # use search cursor with with and as to extract each town name in Massachusetts
    with arcpy.da.SearchCursor(fc, field) as cursor:
        # Loop through town shapefile and append to list of town names
        for row in cursor:
            name = row[0]
            names.append(name)

    # sort town names in list alphabetically
    sorted_names = sorted(names)

    # set variables to store list in new table
    out_path = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test"
    out_name = "town_list.dbf"

    # create table to store names in and add field for town names
    arcpy.management.CreateTable(out_path, out_name)
    arcpy.management.AddField("C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}".format(out_name), "Town_Names", "TEXT")

    # insert sorted town names into new table
    with arcpy.da.InsertCursor("C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}".format(out_name), "Town_Names") as cursor:
        for name in sorted_names:
            cursor.insertRow([str(name)])

    print("Available towns listed in table named Town_Names")
```

Figure 1 code block for list_towns function

The next step was to create a function that allowed the user to select the city or town that they would analyze. This function would ask the user if they were working with a city or town and then ask for the name of the city or town. Once the town/city was selected the arcpy analysis Select function was used to extract that town/city to its own file.

```
# create function to allow for user input to select a town
def select_town():
    # set environment to folder with shapefile of towns
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/mass_towns"
    arcpy.env.overwriteOutput = True

    # variable asking if user would like to work on a city or town
    city_or_town = input("Would you like to look at a city or town? ").lower()

    # conditional statement whether the user selected city or town
    # then ask the user for the name of the city or town and add city or town to then end of the user input
    if city_or_town == "city":
        town = input("What is the name of the city you would like to analyze? ") + " " + city_or_town

    elif city_or_town == "town":
        town = input("What is the name of the town you would like to analyze? ") + " " + city_or_town

    else:
        print("Did not correctly enter city or town")

    # create shapefile for selected city or town
    where_clause = '"NAMELSAD20" = \'{0}\''.format(town)

    # create variable that slices off city or town
    to = town[:-5]

    # input feature is the shapefile with all towns
    in_feature = "CENSUS2020TOWNS_POLY.shp"
    # output feature will be individual selected town
    out_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}.shp".format(to)

    # run Select function to select an individual town
    arcpy.analysis.Select(in_feature, out_feature, where_clause)
```

Figure 2 code block for select_town function

Once the town/city was separated into its own shapefile it was time to clip the roads and flood zones to the town/city. Two separate functions were developed that would allow the user to input the selected city/town and clip the Massachusetts roads and Massachusetts flood zone files to that town. The resulting datasets would include shapefiles that only showed the roads in the selected town and another shapefile that only showed the flood zones in the selected town. To complete this function the arcpy Clip function was used to create new shapefiles.

```
def clip_streets():
    # set environment
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test"
    arcpy.env.overwriteOutput = True

    # set local variables
    in_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/mass_roads/CENSUS2020TIGERROADS_ARC_GC.shp"
    town = input("Enter town you will analyze: ")
    clip_feature = "{0}.shp".format(town)
    out_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}_roads.shp".format(town)

    # run clip to clip roads in town
    arcpy.analysis.Clip(in_feature, clip_feature, out_feature)
```

Figure 3 code block for clip_streets function

```
def clip_fema():
    # set environment
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test"
    arcpy.env.overwriteOutput = True

    # set local variables
    in_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/mass_fema/FEMA_NFHL_POLY.shp"
    town = input("Enter town you will analyze: ")
    clip_feature = "{0}.shp".format(town)
    out_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}_fema.shp".format(town)

    # run clip to clip fema flood zones in town
    arcpy.analysis.Clip(in_feature, clip_feature, out_feature)
```

Figure 4 code block for clip_fema function

The next step was to develop a function that would return any roads that directly intersect with a flood zone. For this function, the arcpy intersect function was used with inputs of the clipped road file and clipped flood zone file. This function outputs a file that only includes the roads that directly intersect with a flood zone. An additional function was then developed using the intersect output file. The goal of this function was to print a list of all the road names that directly intersect with a flood zone. The arcpy SearchCursor and for loop were used to print each of the road names from the intersect file.

```
def intersect_fema():
    # set environment
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test"
    arcpy.env.overwriteOutput = True

    # set local variables
    town = input("Enter town you will analyze: ")
    road_feature = "{0}_roads.shp".format(town)
    fema_feature = "{0}_fema.shp".format(town)
    out_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}_intersect_road_fema.shp".format(town)

    # run intersect to find roads that intersect directly with flood zones
    arcpy.analysis.Intersect([road_feature, fema_feature], out_feature)
```

Figure 5 code block for intersect_fema function

```
def road_in_flood():
    # set environment
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test"
    arcpy.env.overwriteOutput = True

    # set local variables
    town = input("Enter town you will analyze: ")
    fc  = "{0}_intersect_road_fema.shp".format(town)
    field = 'FULLNAME'

    # use search cursor function with with and as
    with arcpy.da.SearchCursor(fc, field) as cursor:
        # loop through attribute table and print road names that intersect with flood zones
        for row in cursor:
            name = row[0]
            print("Road Name: " + name)
```

Figure 6 code block for road_in_flood function

The next step was to allow for a buffer analysis on the roads. The arcpy buffer function was used on the clipped road shapefile. This function would allow for the user to input the buffer distance and units for the roads. The function would then output a polygon shapefile that displayed a buffer around all the roads in the study area based on the user's inputs.

```
def buff():
    # set enivronment
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test"
    arcpy.env.overwriteOutput = True

    # set local variables
    town = input("Enter town you will analyze: ")
    in_feature = "{0}_roads.shp".format(town)

    # ask user for buffer distance and units
    dist = float(input("Enter distance of buffer around roads: "))
    unit = input("Enter units you would like to use (Feet, Meters): ")
    dist_field = "{0} {1}".format(dist, unit)

    out_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}_buffer.shp".format(town)

    # run buffer on roads
    arcpy.analysis.Buffer(in_feature, out_feature, dist_field)
```

Figure 7 code block for buff function

The next step was then to use the buffered road file that was created and perform an intersect function with the flood zone file. To do this the arcpy intersect function was used with inputs of the buffered road file and flood zone file. This tool would create an output file that shows all the roads that would be impacted by a flood zone based on the user's specified buffer distance.

```
def buff_intersect():
    # set environment
    arcpy.env.workspace = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test"
    arcpy.env.overwriteOutput = True

    # set local variables
    town = input("Enter town you will analyze: ")
    buff_feature = "{0}_buffer.shp".format(town)
    fema_feature = "{0}_fema.shp".format(town)
    out_feature = "C:/Documents/OU_Grad_School/GIS_5653/FinalProject/test/{0}_intersect_buff.shp".format(town)

    # run intersect to find roads that intersect directly with flood zones
    arcpy.analysis.Intersect([buff_feature, fema_feature], out_feature)
```

Figure 8 code block for buff_intersect function

Once all the functions were created, it was now time to create a main function that would allow for seamless integration of the developed functions. The main function was made to guide the user through the program to perform analyses that they selected. By stringing together, the above functions and with the use of conditional statements, the user could develop their own analysis based on what they need.

```
def main():
    # tell user to select town to analyze from the list
    print("Select town name from generated table")

    # run function that creates table of all selected
    list_towns()

    # run function that allows user to select town to analyze
    select_town()

    # run function to clip streets to town
    print("You will be asked again for town name to clip the streets to the selected town")
    clip_streets()

    # run function to clip flood zones to town
    print("You will be asked again for town name to clip the flood zones to the selected town")
    clip_fema()

    # ask user if they would like to see which roads intersect with flood zones
    intersect = input("Would you like to only see which roads directly intersect with flood zones? (Y or N) ")
    # if the user wants to see the intersections run the roads intersect with fema function
    if intersect == "Y":
        intersect_fema()

        #ask user if they would like to print the names of roads intersecting with flood zones
        print_names = input("Would you like to see the names of the roads that intersect with flood zones? (Y or N) ")
        # if user would like to see road names then run the print roads in flood zone function
        if print_names == "Y":
            road_in_flood()

    # ask user if they would like to run a buffer on the roads to see which how far roads have from flood zones
    buffer = input("Would you like to run a buffer on the roads to see proximity of flood zones and roads? (Y or N) ")
    # if user would like to perform buffer then run buffer function and buffer intersection function
    if buffer == "Y":
        print("You will be asked for town name to complete buffer")
        buff()
        print("You will be asked for town name to see intersections of buffered roads and flood zones")
        buff_intersect()
```

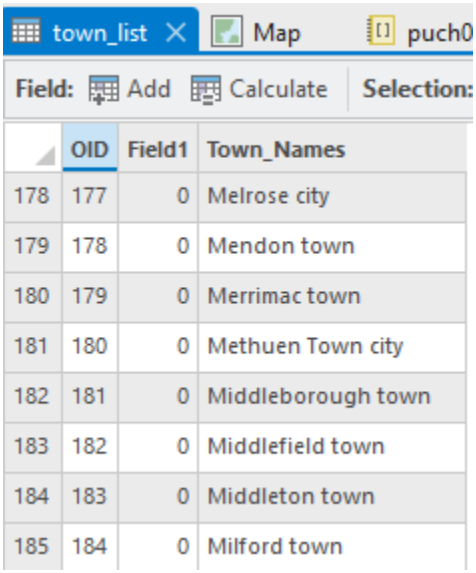
Figure 9 code block for main function

Results:

This section will show the results of the program script using the town of Mendon, Massachusetts. The first portion of the main function will ask the user to select a name from the generated table and will run the list_towns function. This results from the first step in the main function can be seen below.

```
Select town name from generated table
Available towns listed in table named Town_Names
```

Figure 10 program output for list_towns function



	OID	Field1	Town_Names
178	177	0	Melrose city
179	178	0	Mendon town
180	179	0	Merrimac town
181	180	0	Methuen Town city
182	181	0	Middleborough town
183	182	0	Middlefield town
184	183	0	Middleton town
185	184	0	Milford town

Figure 11 table created from list_towns function

After the user looks at the generated table of town names the select_town function will be run. This function will ask the user if they will be working with a town or a city. We will be entering 'town'. The program will then ask the user the name of the town. We will be entering 'Mendon'. The program will then create a new shapefile for Mendon. The results of the select_town function can be seen below.

```
Would you like to look at a city or town? town
What is the name of the town you would like to analyze? Mendon
```

Figure 12 program output for select_town function

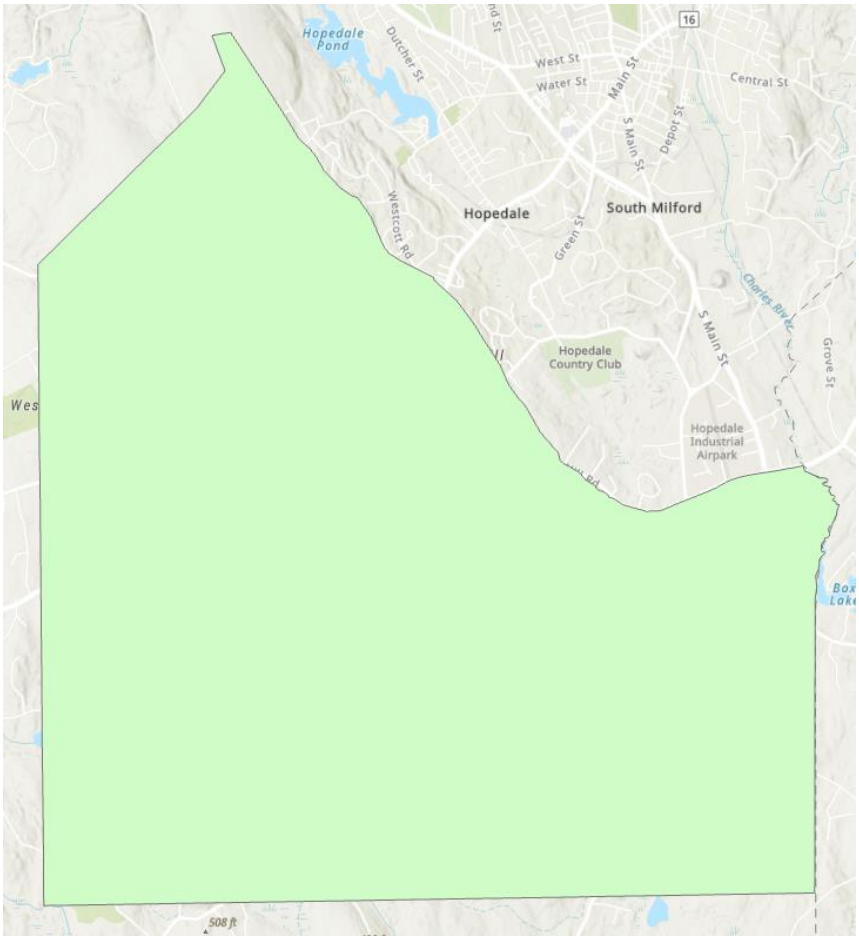


Figure 13 shapefile output from select_town function

Once the town is selected, the program will tell the user that you will be asked for the town name again so that the streets may be clipped to the town. The program will then run the clip_streets function. We will enter 'Mendon' as our town name to be analyzed. The results of the clip_streets function can be seen below.

```
You will be asked again for town name to clip the streets to the selected town
Enter town you will analyze: Mendon
```

Figure 14 program output for clip_streets function

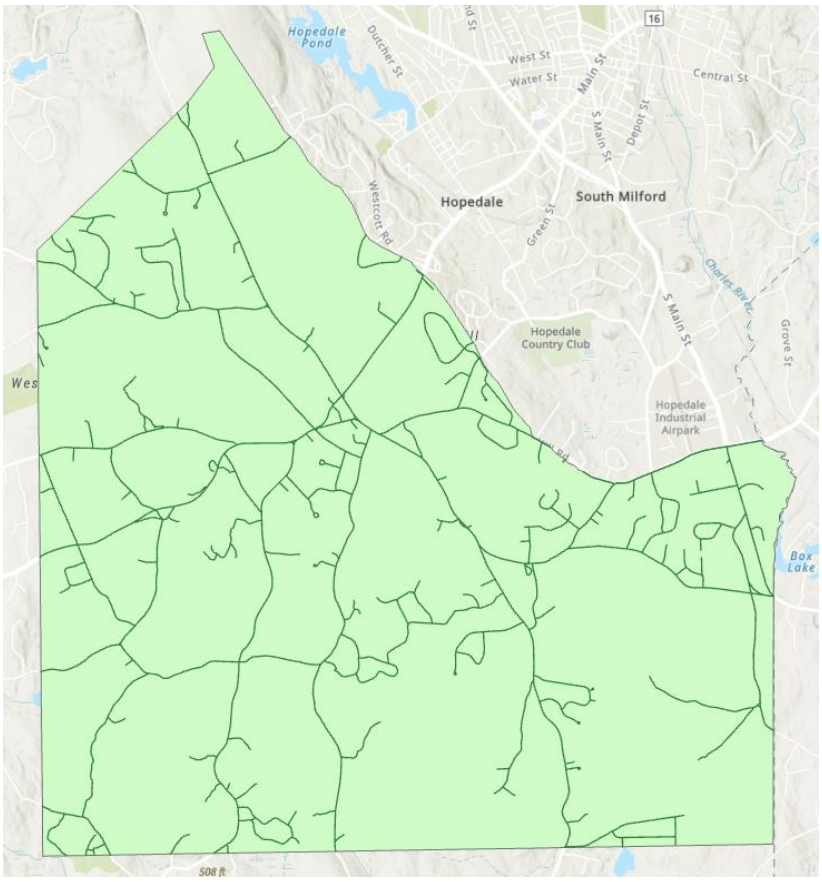


Figure 15 shapefile output of clip_streets function

The program will then tell the user to enter the town name again so that the flood zones may be clipped to the study layer. The program will run the clip_fema function. We will once again enter 'Mendon'. The results for the clip_fema function can be seen below.

```
You will be asked again for town name to clip the flood zones to the selected town
Enter town you will analyze: Mendon
```

Figure 16 program output for clip_fema function

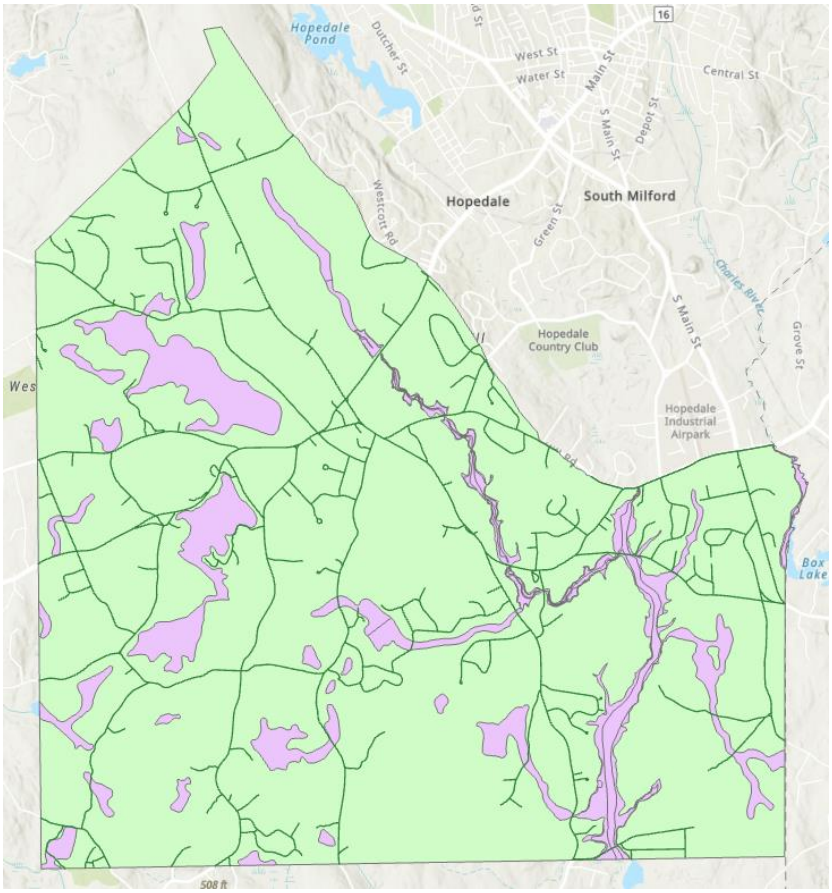


Figure 17 shapefile output for clip_fema function

The program will then ask the user if they would like to perform an analysis on the roads intersecting with the flood zones. We will enter 'Y'. The program will then run the intersect_fema function. The program will ask us for the town name, and we will enter Mendon. The results of the intersect_fema function can be seen below.

```
Would you like to only see which roads directly intersect with flood zones? (Y or N) Y
Enter town you will analyze: Mendon
```

Figure 18 program output for intersect_fema function

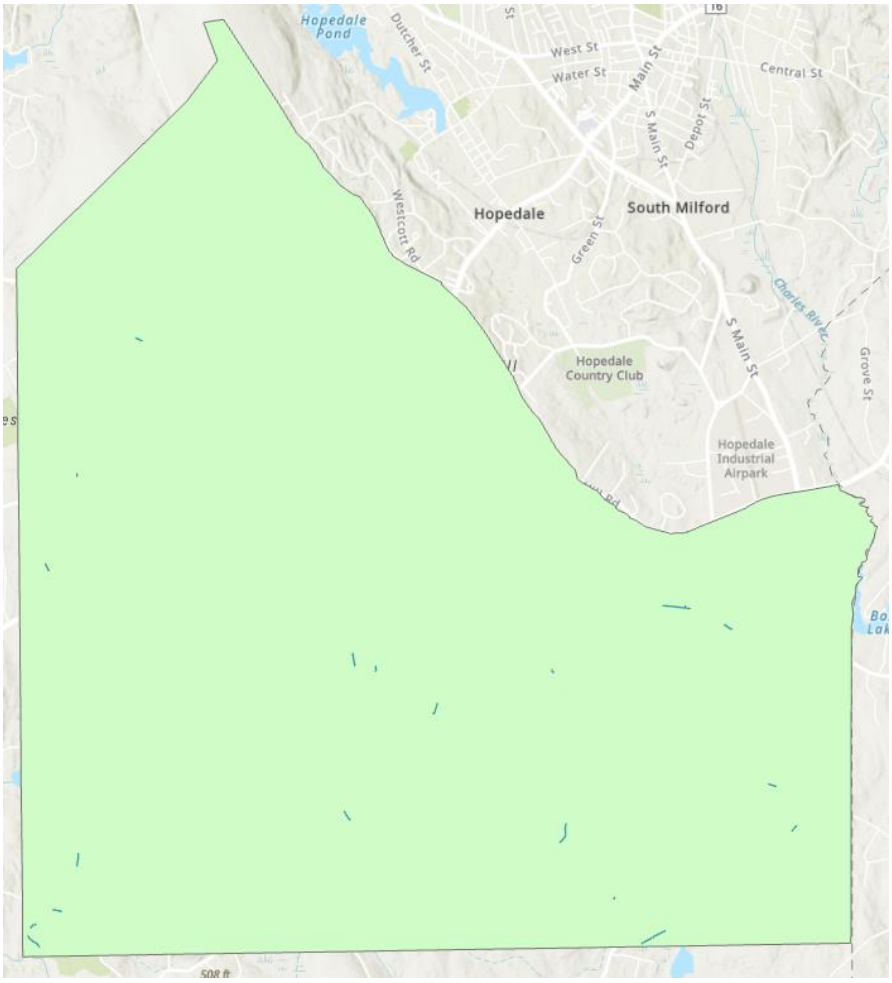


Figure 19 shapefile output for intersect_fema function

The program will then ask the user if you would to see a list of all the road names that intersect with flood zones. We will enter 'Y'. The program will then run the print_names function. We will enter 'Mendon' as the town we are analyzing. The results of the print_names function can be seen below.

```
Would you like to see the names of the roads that intersect with flood zones? (Y or N) Y
Enter town you will analyze: Mendon
```

Figure 20 program output for print_names function

```
Road Name: Hartford Ave E
Road Name: Thayer Rd
Road Name: Providence St
Road Name: Quissett Rd
Road Name: Bellingham St
Road Name: Bellingham St
Road Name: Millbrook Dr
```

Figure 21 printed list output from print_names function

The program will then ask the user if they would like to run a buffer analysis on the roads. We will enter 'Y'. The program will then run the buff and buff_intersect functions and ask the user for the town name. The buff function will also ask the user to enter the buffer distance and units. For this analysis we will be completing a 250 ft buffer analysis on the roads. The results of the buff and buff_intersect functions can be seen below.

```
Would you like to run a buffer on the roads to see proximity of flood zones and roads? (Y or N) Y
You will be asked for town name to complete buffer
Enter town you will analyze: Mendon
Enter distance of buffer around roads: 250
Enter units you would like to use (Feet, Meters): Feet
You will be asked for town name to see intersections of buffered roads and flood zones
Enter town you will analyze: Mendon
```

Figure 22 program output for buff and buff_intersect functions

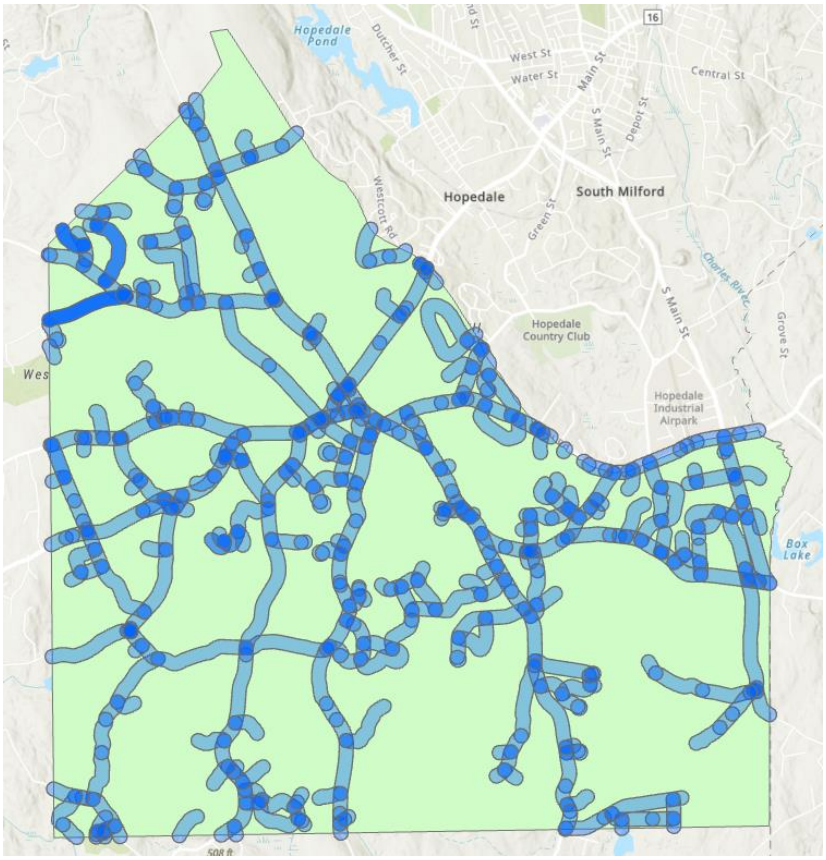


Figure 23 shapefile output for buff function

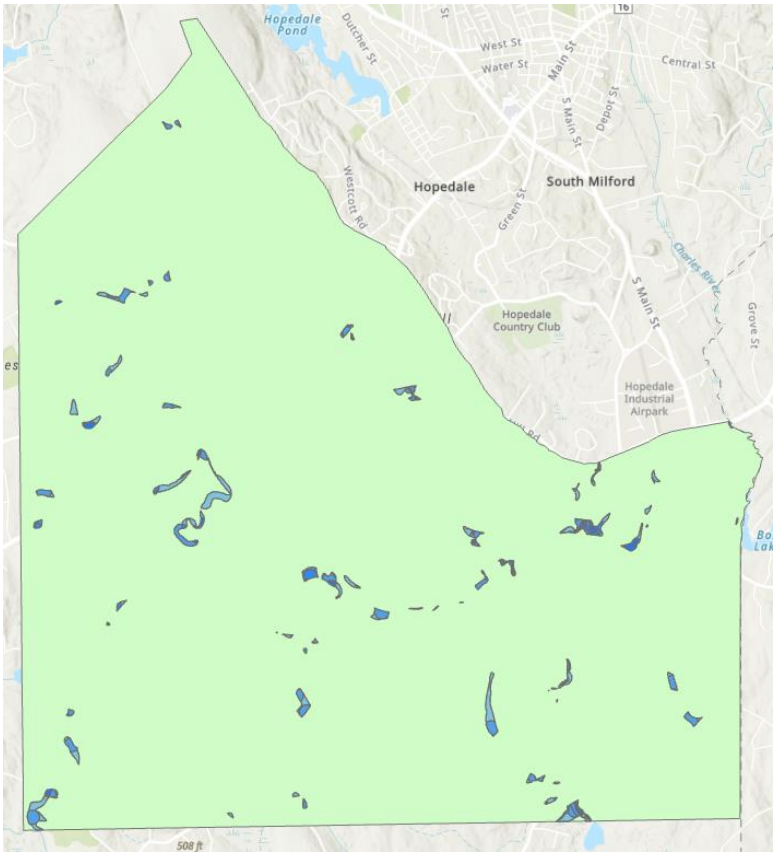


Figure 24 shapefile output for buff_intersect function

The program is now finished running and the user is left with several shapefiles that show the relationships between roads and flood zones in Mendon. As you can see in the figure showing the roads that directly intersect with flood zones, there are very few that have direct intersection relationships. This shapefile and map can be used however to highlight which roads to lookout for and avoid during a flood situation. Looking at the map with a 250-foot buffer around the roads it can be seen that there are more roads that may be impacted during severe floods. An emergency planner can use this map and information to highlight which areas may be impacted and develop alternative routes. The maps above show that there are areas across the entire town that will be impacted by flood scenarios. Key findings show that no area in Mendon will be completely safe from a flood. Even if the direct area someone is in is not impacted, there is a good chance that a road that they will need to use may not be available during a flood situation.

Discussion and Conclusions:

The results from the analysis of the relationship of flood zones and roads in Mendon, Massachusetts show that all parts of the town will be impacted by flooding. Using the maps that were

generated from the analysis an emergency response planner can develop alternative routes for evacuation and emergency response. The basic map of roads intersecting with flood zones shows that roads all over will be impacted by a flood. When using the map with the buffered roads it shows a larger number of roads that have the potential to be impacted in a severe flood event. During severe flood events the water typically flows well past the designated flood zone. These maps will allow for everyone to be better prepared in a situation to safely evacuate and get help. Different routes can be planned to use this information.

The way the program currently works, this type of analysis can be replicated for any town or city in the state of Massachusetts. All the users would need to do is enter a different town or city name such as 'Worcester' to program. The generated maps will then all be completed for Worcester, MA. This can also easily be completed for other states in the US. If town, road, and FEMA data is available for another state then the source data files can easily be changed to the state and the analysis can be completed for that state. What makes ArcGIS and arcpy so great is the ability to easily replicate analysis like this just by changing inputs. If this analysis was to be completed and replicated without the use of arcpy, then time would be wasted redoing the same operations repeatedly. This program allows for emergency planners to develop response plans for multiple areas very quickly. This could even be completed on a state level instead of town/city level by removing the select town functions.

The maps above are fairly basic in nature. One of the major limitations of this program is that the only flood modeling information is from FEMA. This information is freely available to all and because of that tends to be more basic than other flood models. It would be beneficial to include flood modeling data from private was organizations to get a more accurate understanding of flooding impact. Another limitation with this program is the lack of elevation data. Due to time constraints of the project, elevation data was not able to be included in this program. This is a limitation because there may be some roads that are within 50 feet from a flood zone but may have an elevation difference of 20 feet. If that is the case, then it is unlikely that the flood waters will ever reach the road. I would suggest that the next step for this program be to implement elevation data so that only roads that are within a set elevation from a flood zone are included.

References:

All data used for this program is courtesy of *"MassGIS (Bureau of Geographic Information), Commonwealth of Massachusetts EOTSS"*