FLASH FLOOD SCREENING TOOL

A Thesis

by

DEBAYAN MANDAL

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Francisco Olivera
Committee Members,	Andrew Klein
	Nasir Gharaibeh
Head of Department,	Robin Autenreith

December 2020

Major Subject: Civil Engineering

ABSTRACT

Flash floods have been a very unpredictive natural disaster. Previous attempts at zonation of flash floods based on predictive severity have been very data-intensive and exhaustive on focusing on a certain area. There are, however, simple models that utilize publicly available data to determine flash flood risks at specific locations. The Geographic Information System (GIS) –based tool, presented here, takes a Digital Elevation Model (DEM) and a Land-Use/Land-Cover map to compute basin characteristics. Flow velocity is modeled according to TR-55, which classifies it as sheet, shallow-concentrated and open-channel flow. With the help of various GIS tools, the drainage area of a user-given outlet point, as well as the flow lengths upstream to the headwaters and downstream to the outlet, can be calculated. These flow lengths are then used to classify the areas of sheet flow, while the areas of open-channel flow are user-defined. Velocities for sheet and shallow-concentrated flow are determined according to TR-55, while the velocities for open-channel flow are user-defined. Once the velocities are known, isochrones are determined and area vs. flow-time plots are developed. Intensity-duration-frequency (IDF) curves are used to define the precipitation for different concentration times and return periods. The rational method is then used to determine the expected peak flows. A sub-watershed of Little Walnut Creek, in Austin, will be valuated and shown as an example. This screening tool will help policymakers identify locations of potential high flash flood risk in which more-detailed hydrologic analysis would be needed.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was done under the supervision of Dr. Francisco Olivera and Dr. Nasir Gharaibeh of the Zachry Department of Civil and Environmental Engineering and Dr. Andrew Klein of the Department of Geography.

All work for the thesis was completed by the student, in collaboration with Dr. Francisco Olivera of the Zachry Department of Civil and Environmental Engineering.

Funding Sources

Graduate study was supported by a departmental fellowship from the Zachry Department of Civil and Environmental Engineering, Texas A&M University.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Flash floods are a critical natural disaster causing around five thousand casualties worldwide yearly and 16.8 annual average fatalities in Texas (Sharif et al., 2015). Particularly, the area from Dallas to San Antonio came to be known as "Flash Flood Alley" due to the frequent and intensive flash floods caused by high precipitation and topographic conditions. In this area, alone, each county has more than twenty flash floods in a given year (Sharif et al., 2015). Flash floods are the most prevalent cause of fatalities followed by floods in Texas (Sharif et al., 2015). Previous attempts at zonation of flash floods based on predictive severity have been very data-intensive and exhaustive on focusing on a certain area. There are, however, simple models that utilize publicly available data to determine flash flood risks at specific locations.

Despite averaging around five thousand flash flood fatalities per year worldwide (Jonkman, 2005; Grabs, 2010; Modrick and Georgakakos, 2015), acknowledgment of their severity and the need to take action to palliate them has been a recent phenomenon. This somewhat new concern for flash floods stems from the fast rate at which urbanization is occurring, which is limiting significant portions of the catchment to absorb the fraction of precipitation it did before development (Konrad 2003). Although there are numerous studies on flash floods, their number is still miniscule compared to the number of those conducted for longer-duration floods.

A Geographic Information System (GIS) -based tool to screen locations vulnerable to flash floods is presented here. While most of the existing tools for flash-flood analysis are very data intensive (Rozalis et al., 2010; Volkmann et al., 2010) – a majority of which are not readily available –, the tool presented here is based on publicly available data, at least for the United States.

2. LITERATURE REVIEW

According to the National Weather Service (2012), a flash flood takes less than six hours to reach the flood stage in a catchment of less than 250 km^2. This definition varies from country to country mostly to accommodate the regional differences.

Broadly classified, the separation of areas of high flash-flood risk to low flash-flood risk is done through rainfall-runoff and morphometric models.

The rainfall-runoff models used are data intensive requiring inputs of rainfall data from radars, streamflow data from hydrometric stations, soil distribution, land use and elevations. Many of the works use the SCS-CN method (Georgakakos, 1986; Brath et al., 1993) to compute runoff rate and kinematic wave to route the flow (Georgakakos, 1986; Brath et al., 1993; Yatheendradas et al., 2008; Rozalis et al., 2010; Volkmann et al., 2010). These hydrologic models can also be used in other applications, like estimating discharge at road and river intersections for very short time intervals (Naulin et al., 2013).

The latest works have focused on developing a Flash Flood Guidance (FFG). It uses a semi-distributed conceptual rainfall-runoff model to obtain rainfall depth in areas, which can cause imminent flooding, predominantly based on continuous soil moisture conditions (Moore, 1985; Norbiato et al., 2008). Additional work has also been done to account for ungauged basins. In those cases, a lumped hydrological model is used to derive flood probability at the outlet and threshold runoff is calculated from the model-based discharges. The threshold runoff is therefore mostly derived based on local statistics and model-based approach (Norbiato, Borg a, &

Dinale, 2009). Few models are capable of making reliable flash flood forecasts in urban watersheds (Hapuarachchi et al., 2011).

The tool proposed here has two advantages over existing ones – it takes publicly available data as input and thereby does not need extensive data collection and it outputs the flow versus time since the beginning of the storm for outlet locations of interest. This is a screening tool that narrows down potential areas of high risk of flash flood. Once the high-risk zones are narrowed down, more extensive models can generate more accurate results. This tool does not propose to replace existing models but rather acts as an ancillary to the process by identifying the locations where further analysis is needed. It can also work solo, depending on the level of accuracy desired.

Morphometric models recognize physical and social vulnerability parameters –. Morphometric parameters are calculated for watersheds, including parameters like stream order, number, length, bifurcation ratio, areal aspects, drainage density, stream frequency, circulatory ratio, length of overland flow, texture ratio, relief, ruggedness number, and wetness index. A relative weight is then assigned to these parameters using analytic hierarchy processes and the final weighted values are classified from high to low risk categories (Karagiorgos et al., 2016; Mohamed, 2019; Costache et al., 2019). These processes are followed by a vulnerability analysis with methods such as the Modified Frechet, Weibull, Exponential method (Bhaskar et al., 2000; Kim et al., 2013; Taha et al., 2017; Costache et al., 2019; Bisht et al., 2018; Lee et al., 2018). The drawback of this relatively simple approach is that the relative weight assignment to various factors can be very subjective. The computation however, is easier than hydrological models. The proposed tool does require less data and provides quick computation for an interested area, while being more technically viable. The hydrology adopted and its implemented is specified in the section below.

3. METHODOLOGY

This flash-flood screening tool takes the Digital Elevation Model, Land Use Land Cover data and the outlet location of interest, as inputs. It follows the flowchart given in Figure 1. This tool uses English units.

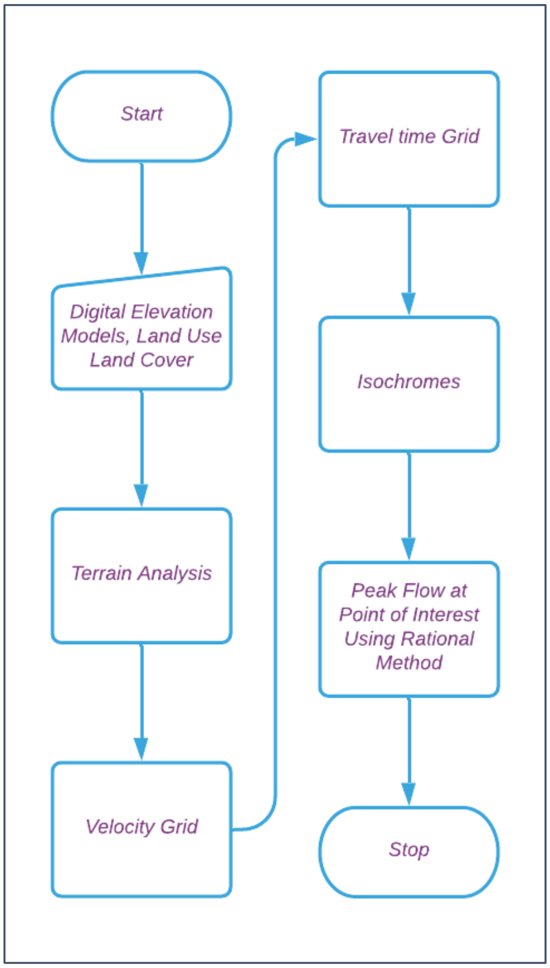

Start
Digital Elevation Models, Land Use Land Cover
Terrain Analysis
Velocity Grid
Travel time Grid
Isochromes
Peak Flow at Point of Interest Using Rational Method
Stop

Fig 1: Basic Methodology

The terrain analysis begins by employing the fill function to remove any depressions in the Digital Elevation Model. Followed by which, the flow direction function is invoked to produce a raster that signifies the direction of flow of water from one cell to an adjacent cell. Using this flow direction and outlet point, the watershed function lines the drainage area.

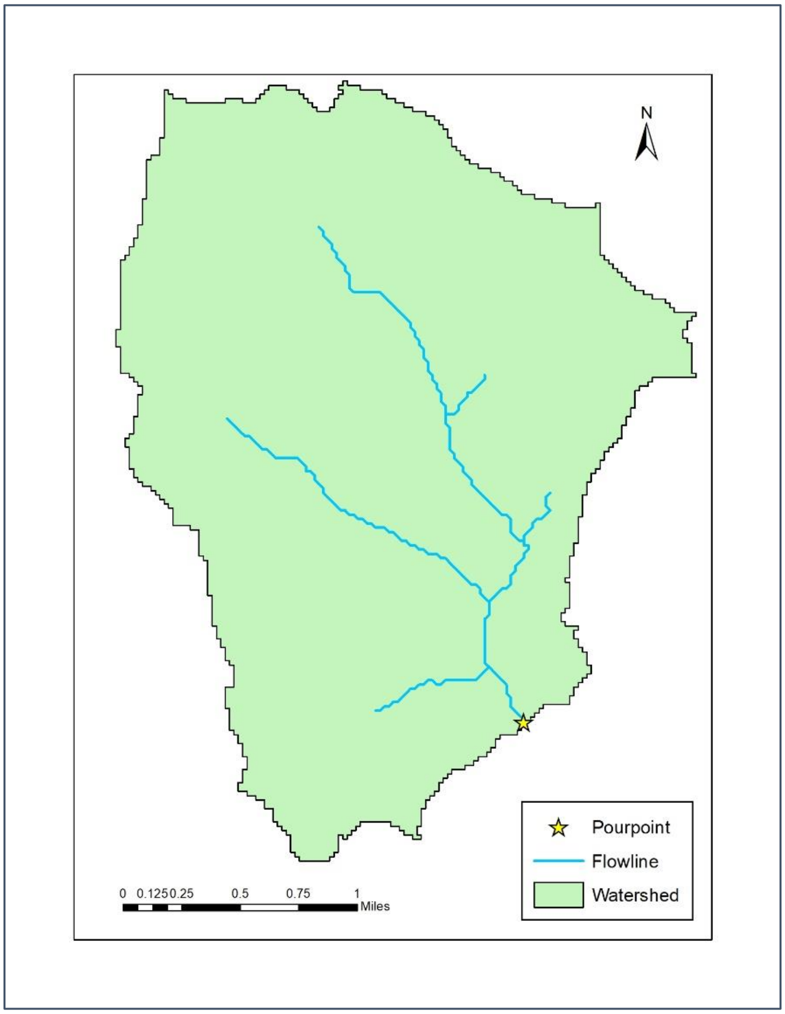


Fig 2: Watersheds for catchments on the rivers and creeks.

For optimizing the processing time, the previous results along with the inputs are clipped using the polygon obtained as watershed boundaries. The flow accumulation function uses the flow direction raster to populate each cell with the number of cells draining into them.

The flow is divided into three types: sheet flow, shallow concentrated flow, and open channel flow. Sheet flow occurs at the headwaters, usually over plane surfaces with a flow depth of 0.1 feet or less (National Engineering Handbook, 2008). Shallow Concentrated flow starts thereafter forming gullies and swales. Although it does not have a proper well-defined channel, the velocity equations are developed using wide, rectangular flow concepts (National Engineering Handbook, 2008). It is recommended by U.S.G.S that the open channels begin at locations where the blue lines indicating streams appear on United States Geological Survey (U.S.G.S) quadrangle sheets (1:24,000).

Usually the demarcation between sheet flow and shallow concentrated flow is taken as 300 feet (TR-55, 1986). This value has been debated and is reported to vary in the range of 100 to 300 feet for different types of drainage areas. Most of this variability is due to the varying length of flow in paved and unpaved areas. For unpaved areas, most of the sheet flow occurs within 100 feet, whereas, paved areas have longer lengths of sheet flow (Woodward and Welle, 1986). Therefore, the tool will give an option to the user to input this value in meters.

For demarcation of the start of open channel, an input of threshold drainage area is required. It is recommended to compare the USGS Quadrangle Maps and to the obtained raster to approximately match the streams nearby. The threshold value will be compared to the flow accumulation raster and the cells with values greater than the threshold will be designated as open channel flow cells raster.

The flowlength function then calculates the flow length upstream to the drainage divide. Zero values are associated with local headwaters or ridges. The output is in the units of feet and will usually be the sum of cell sides or diagonals. Therefore, the accuracy of this parameter is dependent on cell resolution.

The upstream flow length grid is then used to first demarcate sheet flow from shallow concentrated areas. The shallow concentrated areas outputted in this step will also contain open channel areas, which are removed later on during the mosaicking of velocity grids by assigning this shallow concentrated flow grid a lower priority value as compared to the open-channel flow-velocity raster. As the flow length is utilized in the velocity function for the sheet flow, it would be erroneous if the value yields zero feet per second. Therefore, the approximation is made to take an average by assigning the zero value cells a value of half the cell side size. The assumption is that flow starts from the middle of the cell and flows to a side. As cell size reduces, this assumption gets more accurate.

The primary formula utilized for sheet flow is obtained from National Engineering Handbook, Part 630 Hydrology, Chapter 15, USDA, NRCS. In page 15-4; we obtain the travel time equation at eq 15-8. This was developed by Welle and Woodward (1986) from Manning's kinematic solution. The equation is as follows in figure 3.

$$\mathrm{T} = \frac{0.007 * (\mathrm{nl})^{0.8}}{\mathrm{P}_2^{0.5} * \mathrm{S}^{0.4}}$$

where T = travel time, h; n = roughness coefficient; l = flow length, ft; P_2 = 2-year, 24-hour rainfall, in; S = Slope of hydraulic grade line, ft/ft

Fig 3: The Sheet Flow Travel Time Equation

To obtain a velocity equation:

Solving for L and changing the time units to seconds:

$$L = \frac{P_2^{0.625} * \ S^{0.5} * T_t^{1.25}}{56.46 * n}$$

Differentiating with respect to T; to get velocity term.

$$V = \frac{0.05 * \ P_2^{0.5} * \ S^{0.4} * L^{0.2}}{n^{0.8}}$$

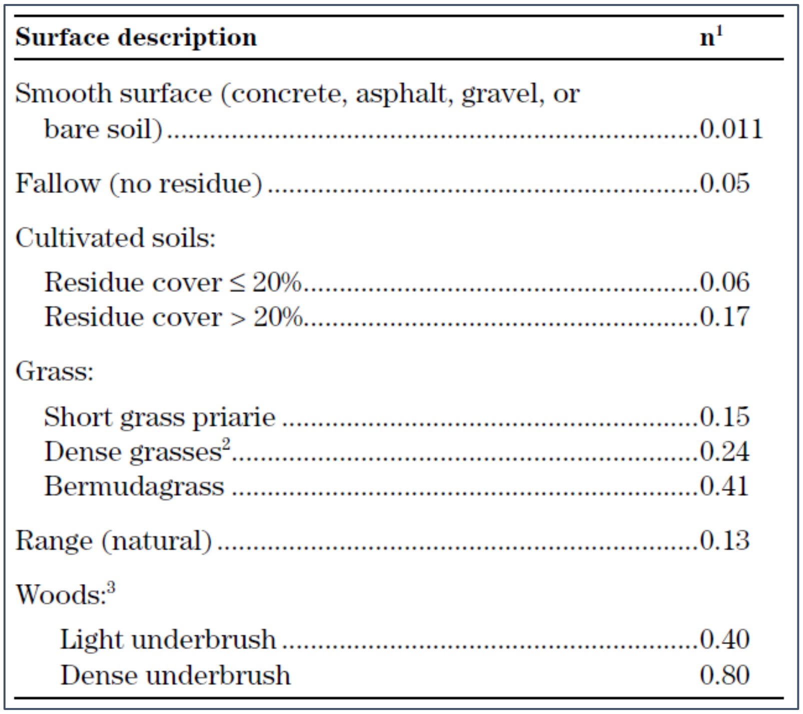
The units of the final equation components are as follows: -

Terms	Units
V	Feet per second
P_2	Inches
L	Feet
S, n	Dimensionless

Table 1: Units of Sheet Velocity Equation

To obtain the terrain slope, the slope function is used in “as percent rise” mode and is calculated from the Digital Elevation Model within the watershed. As our required equations need the slope in ratio form, the above raster is divided by one hundred to get rid of the percentage format.

An input for P2 value is taken in inches. P2 is the 2-year 24-hour precipitation depth, and be estimated using the corresponding intensity-duration-frequency curve. Next, the roughness coefficients for sheet flow will be taken from the table (National Engineering Handbook, 2008) in figure 4.

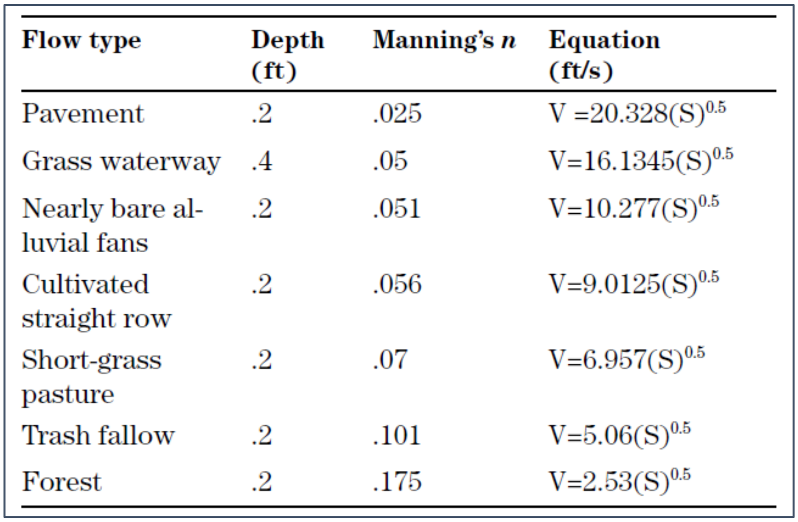


Surface description	n[1]
Smooth surface (concrete, asphalt, gravel, or bare soil)	0.011
Fallow (no residue)	0.05
Cultivated soils:	
Residue cover ≤ 20%	0.06
Residue cover > 20%	0.17
Grass:	
Short grass priarie	0.15
Dense grasses[2]	0.24
Bermudagrass	0.41
Range (natural)	0.13
Woods:[3]	
Light underbrush	0.40
Dense underbrush	0.80

Fig 4: Roughness Coefficients for sheet flow (National Engineering Handbook, 2008). Reprinted.

The tool will invoke a NLCD lookup table containing the corresponding n values for a particular land use. An option is given for another markup table if the user's land use is not sourced from NLCD. However, in that case, the user needs to provide a markup table in the editable excel file attached in the Documents folder.

For shallow concentrated flow, the velocity equation is obtained from National Engineering Handbook, Part 630 Hydrology, Chapter 15, USDA, NRCS. The velocity equations are obtained as a function of the slope and land use (see Figure 5). The coefficients of the equation are obtained in Table 15-2 in page 15-5 of the afore-mentioned document.



Flow type	Depth (ft)	Manning's n	Equation (ft/s)
Pavement	.2	.025	$V = 20.328(S)^{0.5}$
Grass waterway	.4	.05	$V=16.1345(S)^{0.5}$
Nearly bare alluvial fans	.2	.051	$V=10.277(S)^{0.5}$
Cultivated straight row	.2	.056	$V=9.0125(S)^{0.5}$
Short-grass pasture	.2	.07	$V=6.957(S)^{0.5}$
Trash fallow	.2	.101	$V=5.06(S)^{0.5}$
Forest	.2	.175	$V=2.53(S)^{0.5}$

Fig 5: The equations and coefficients for shallow concentrated flow (National Engineering Handbook, 2008). Reprinted

Similar to sheet flow classes, the tool will import a NLCD markup table. For other values to be taken or if the user's land use data is from a different source, the excel file must be edited to contain the new markup table in a new sheet.

The slope in the function still has the ratio format, thereby, no further calculations will be needed, and the function will be used as is for the shallow concentrated flow velocity equation.

Moving on to the open channel, there are many methods that can calculate open channel velocity. Thus, at this juncture, the tool allows for a direct input of the open channel velocity layer in ft/sec. If, however, the user is completely reliant on publicly available datasets, the tool gives an optional model that can be used.

In this optional model, the peak velocity in meters per second is given by Jobson (1996): -

$$\mathrm{V}(\frac{\mathrm{m}}{\mathrm{s}}) = 0.094 + 0.0143 * \text{Dimensionless D.A.}^{0.919} * \text{Dimensionless Q}^{-0.469} *$$

$$\text{streamslope}(\frac{\mathrm{m}}{\mathrm{m}})^{0.159} * \frac{\text{bankfull flow (cms)}}{\text{drainage area (m2)}}$$

D.A. here stands for drainage area, V for velocity and Q for flow. The flow value here is taken as bank full flow values to give maximum safety factor (Strager, 2012). The dimensionless D.A. is obtained from (Jobson, 1996): -

$$\text{Dimensionless D.A.} = \sqrt{g} * \frac{\text{drainage area}^{1.25}}{\text{average annual flow}}$$

The dimensionless Q is obtained from (Jobson, 1996): -

$$\text{Dimensionless Q} = \frac{\text{bankfull flow}}{\text{average annual flow}}$$

It is observed the factors required are drainage area, stream slope, average annual flow, and bank full flow.

For the drainage area grid, the tool multiplies the area of a cell size times the flow accumulation value added to one. The one is added as the cell itself is not being considered by the flow accumulation function, but it is also the part of drainage area. In other words, the equation is: -

(Flow Accumulation + 1) * Cell Area = D.A.

The stream slope is unique for each reach, where a stream reach is considered as the section of the stream between junctions, which depend on the area threshold for stream delineation. The stream link function was used to identify the reaches. Along the flowline elevations are extracted, and slope function is run again to get the required results.

As for the other two components, the daily mean stream flow values for gauges in and around the interested area need to be taken. The drainage area of the corresponding stream gauges is to be considered. The drainage area is calculated in square miles and flow in cubic feet per second. The stream gauges of very large rivers are recommended to be left out.

The annual average flow for each flow gauge needs to be calculated as the average of the daily mean flows. As bank full flow, it is defined as ninetieth percentile of the annual high flows (Strager, 2012). The greater the number of years of observations, the better it represents the actual value. The values of average annual flow and bank full flow are then regressed against the drainage area values to obtain the best-fit equation. The grid for bank full flow and average annual flow must be calculated based on the drainage area grid in square miles.

In the tool, the resultant velocity grid will be obtained by mosaicking all three velocity grids. The priority is set as first open, then sheet, followed by shallow-concentrated flow velocity. The velocity grid is then converted to feet per minute and then inverted. The flowlength function then calculates the travel time using the inverse velocity grid as weight in downstream mode. Contour lines in this travel time grid correspond to isochrones in minutes.

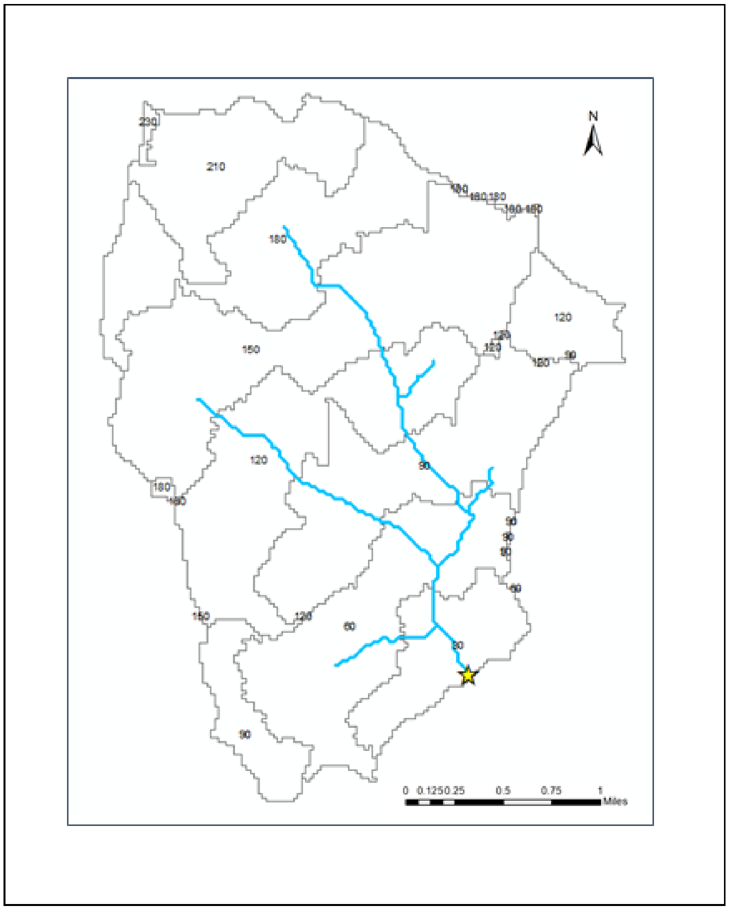


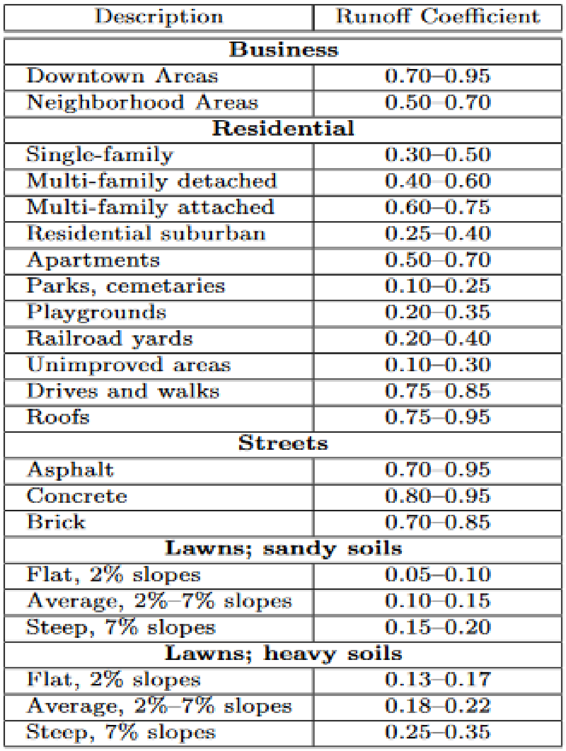
Fig 6: Isochrones of travel time

The flow in the outlet corresponding to rain fall is calculated by Rational Method (Poertner, 1974). The equation is used alongside the classic rational method extensively in TxDOT and other manuals circa 2002. The equation is as follows (Kuichling, 1889): -

$$Q_{\mathbf{p}} = C_{\mathrm{std}} * C_{\mathrm{f}} * i * A$$

Where C_{std} is the standard runoff coefficient, C_f is the frequency correction factor, i is the intensity in inches/hr, A is area in acres. This results in the flow in cfs. In the tool C_{std} is referred to as composite C.

As for the runoff coefficient, it is taken from the table (Thompson, 2006) in figure 7.

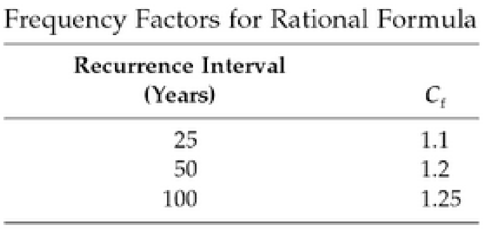


Description	Runoff Coefficient
Business	
Downtown Areas	0.70–0.95
Neighborhood Areas	0.50–0.70
Residential	
Single-family	0.30–0.50
Multi-family detached	0.40–0.60
Multi-family attached	0.60–0.75
Residential suburban	0.25–0.40
Apartments	0.50–0.70
Parks, cemetaries	0.10–0.25
Playgrounds	0.20–0.35
Railroad yards	0.20–0.40
Unimproved areas	0.10–0.30
Drives and walks	0.75–0.85
Roofs	0.75–0.95
Streets	
Asphalt	0.70–0.95
Concrete	0.80–0.95
Brick	0.70–0.85
Lawns; sandy soils	
Flat, 2% slopes	0.05–0.10
Average, 2%–7% slopes	0.10–0.15
Steep, 7% slopes	0.15–0.20
Lawns; heavy soils	
Flat, 2% slopes	0.13–0.17
Average, 2%–7% slopes	0.18–0.22
Steep, 7% slopes	0.25–0.35

Fig 7: Rational Coefficients in general (Thompson, 2006). Reprinted

The tool divides the slopes into flat, rolling, and hilly based on below 2, 2-7 and above 7 respectively. In between the ranges, the low value is assigned to the flat zone, median value is assigned to the rolling and the highest value is assigned to the hilly zone. In case of lawns the median value is taken for each slope classification.

The frequency correction factor is given by the following table (Debo, 2003): -



Frequency Factors for Rational Formula

Recurrence Interval (Years)	C_f
25	1.1
50	1.2
100	1.25

Table 2: Frequency Factors for Rational Formula (Debo, 2003). Reprinted

The formula for calculating composite coefficient value: -

$$C_v^{lit} = \frac{\sum_{i=1}^{n} C_i A_i}{\sum_{i=1}^{n} A_i} \qquad \text{(Kuichling, 1889)}$$

Where i is the ith sub-area with a particular land-use type, n is the number of land-use classes in the watershed, C_i is the literature-based runoff coefficient for the ith land-use class, and A_i is the area of the ith land-use class in the watershed.

The intensity values for every five minutes will be imported based on the County entered, for the return period rainfall of two-year, five-year, twenty five-year, fifty-year and hundred years. The tool has all the intensity values (TxDOT manual) for Texas state stored in it. After the calculations, the output of flow (cfs) vs time (minutes) will be displayed in graphical as well as tabular format.

A second method other than using the rational method is also made available in the alternate version of the tool. This method is derived from NRCS Unit hydrograph method.

$$Q = \frac{PRF * A}{T_p} * R$$

PRF – Peak Rate Factor taken as 484

A – Watershed area (sq.mi)

T_p – Peak Time (hrs)

R – Runoff depth (in)

The T_p is given by: -

$$T_p = 0.5 * t_d + t_{lag}$$

Where t_d is storm duration (hr) and t_{lag} is watershed lag time (hr)

$$t_d = \alpha * t_c$$

where α varies from 0.13 to 0.2

t_c – time of concentration

$$t_{lag} = 0.6 * t_c$$

Therefore, $T_p = (0.5 * \alpha + 0.6) * t_c$

The R is given by: -

$$R = i_R(t_c, T) * t_d$$

$$R = i_R(t_c, T) * \alpha * t_c$$

Where T is return Period

Reducing the flow equation: -

$$Q = \frac{PRF}{0.5 + \frac{0.6}{\alpha}} * i_R(t_c, \mathrm{T}) * A$$

To fit seven storms in the t_c, α is taken as 0.1428. Using SCS Runoff equation,

$$R_7 = \frac{(P_7 - 0.2 * S)\text{^}2}{(P_7 + 0.8 * S)}$$

Where R_7 is runoff of seven storms(in), S is potential maximum retention(in), P_7 is cumulative precipitation (in) and CN is curve number.

$$P_7 = i(t_c,T) * t_c$$

$$S = \frac{1000}{CN} - 10$$

Runoff depth for one storm: -

For one storm: - $R_1 = \alpha * R_7$

Runoff Intensity: - $i_R = \frac{R_1}{t_d} = \frac{R_7}{t_c}$

Convolution: -

$$T_p = (0.5 * \alpha + 0.6) * t_c = 0.67 * t_c$$

$$t_c = 1.49 * T_p$$

$$\frac{t}{T_p} = 1.49 * \frac{t}{t_c}$$

Taking values of Q_p at: -

t/tc	t/Tp	Q/Qp
0	0.00	0
0.14	0.21	0.109
0.29	0.43	0.358
0.43	0.64	0.724
0.57	0.85	0.96
0.71	1.06	0.994
0.86	1.28	0.874
1	1.49	0.69
	Sum =	**4.709**

Total response for all seven storms, substituting alpha as 0.1428: -

$$Q_p = 4.71 * 103 * \ i_R * A$$

$$Q_p(cfs) = 485.13 * \ i_R \left(\frac{in}{hr}\right) * A(sq.mi)$$

$$Q_p(cfs) = 0.76 * \ i_R \left(\frac{in}{hr}\right) * A(acre)$$

A CN grid input will be taken as an addition to the other inputs previously mentioned for employing this tool.

4. APPLICATION

This section showcases the application of the tool for Little Walnut Creek located in the Northern part of Austin. It has been named as "one of the steepest creeks" in the city.

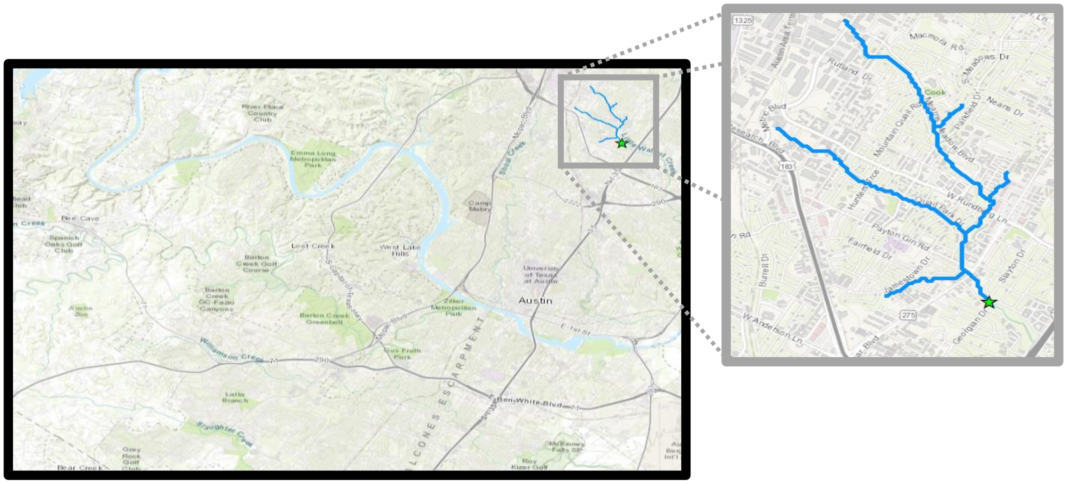
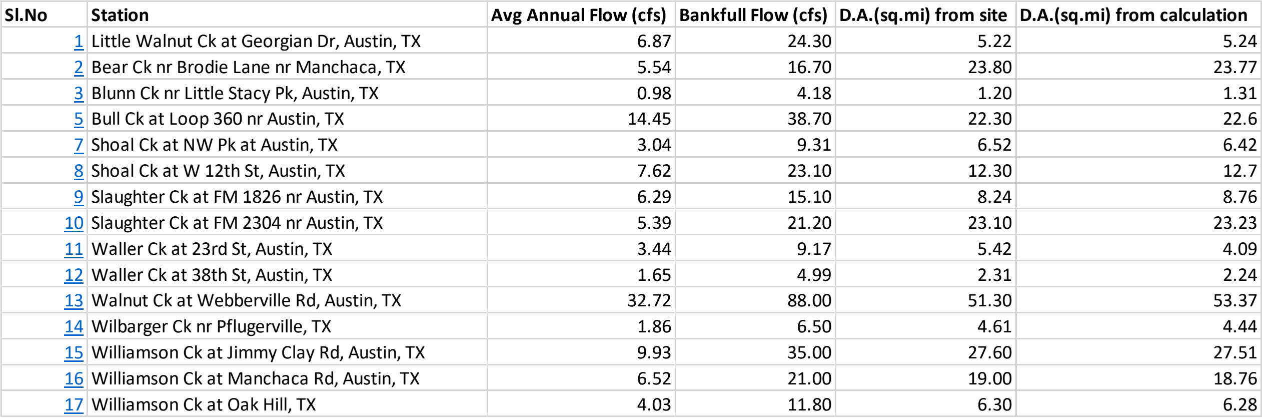


Fig 8: Little Walnut Creek

The daily mean flows for the stream gauges around Austin were taken for the regression relations. However, the datasets of the Colorado River were deleted, as it had a sizable chunk of its drainage area outside the considered map. The final table of the seventeen gauges were taken and regression analysis found the relations in between them.



Sl.No	Station	Avg Annual Flow (cfs)	Bankfull Flow (cfs)	D.A.(sq.mi) from site	D.A.(sq.mi) from calculation
1	Little Walnut Ck at Georgian Dr, Austin, TX	6.87	24.30	5.22	5.24
2	Bear Ck nr Brodie Lane nr Manchaca, TX	5.54	16.70	23.80	23.77
3	Blunn Ck nr Little Stacy Pk, Austin, TX	0.98	4.18	1.20	1.31
5	Bull Ck at Loop 360 nr Austin, TX	14.45	38.70	22.30	22.6
7	Shoal Ck at NW Pk at Austin, TX	3.04	9.31	6.52	6.42
8	Shoal Ck at W 12th St, Austin, TX	7.62	23.10	12.30	12.7
9	Slaughter Ck at FM 1826 nr Austin, TX	6.29	15.10	8.24	8.76
10	Slaughter Ck at FM 2304 nr Austin, TX	5.39	21.20	23.10	23.23
11	Waller Ck at 23rd St, Austin, TX	3.44	9.17	5.42	4.09
12	Waller Ck at 38th St, Austin, TX	1.65	4.99	2.31	2.24
13	Walnut Ck at Webberville Rd, Austin, TX	32.72	88.00	51.30	53.37
14	Wilbarger Ck nr Pflugerville, TX	1.86	6.50	4.61	4.44
15	Williamson Ck at Jimmy Clay Rd, Austin, TX	9.93	35.00	27.60	27.51
16	Williamson Ck at Manchaca Rd, Austin, TX	6.52	21.00	19.00	18.76
17	Williamson Ck at Oak Hill, TX	4.03	11.80	6.30	6.28

Table 3: Tabular format of data corresponding to the stream gauges

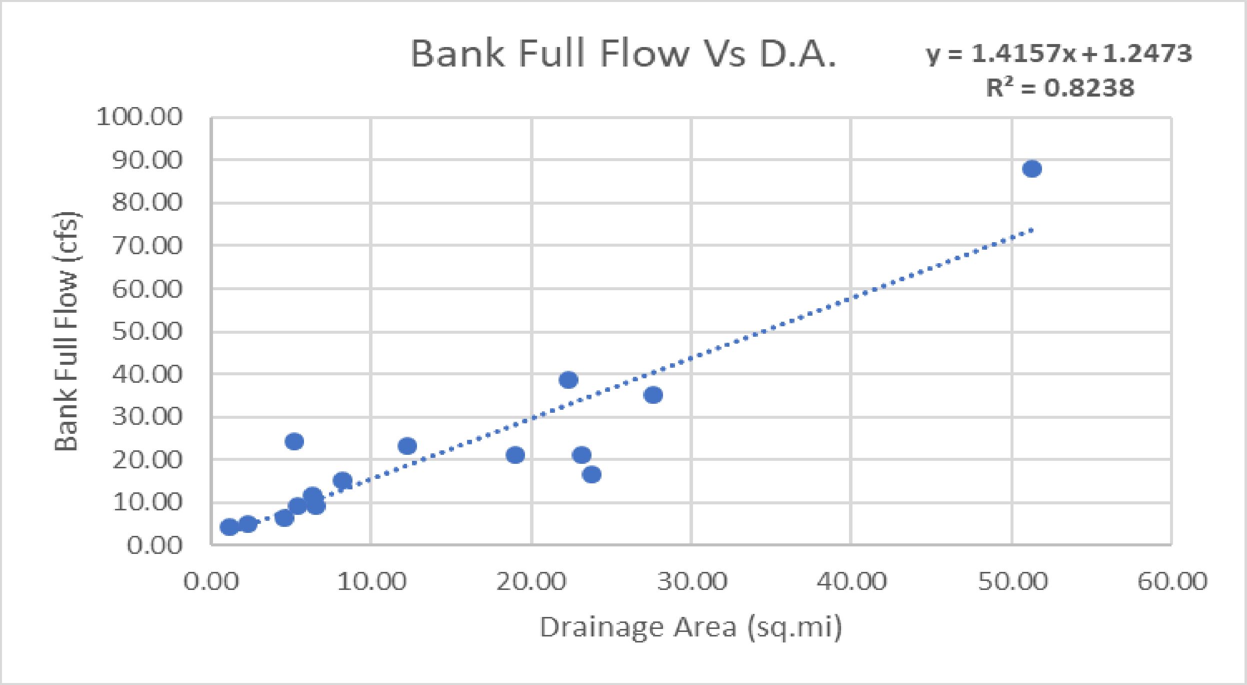


Fig 9: Relation between bank full flow vs drainage area

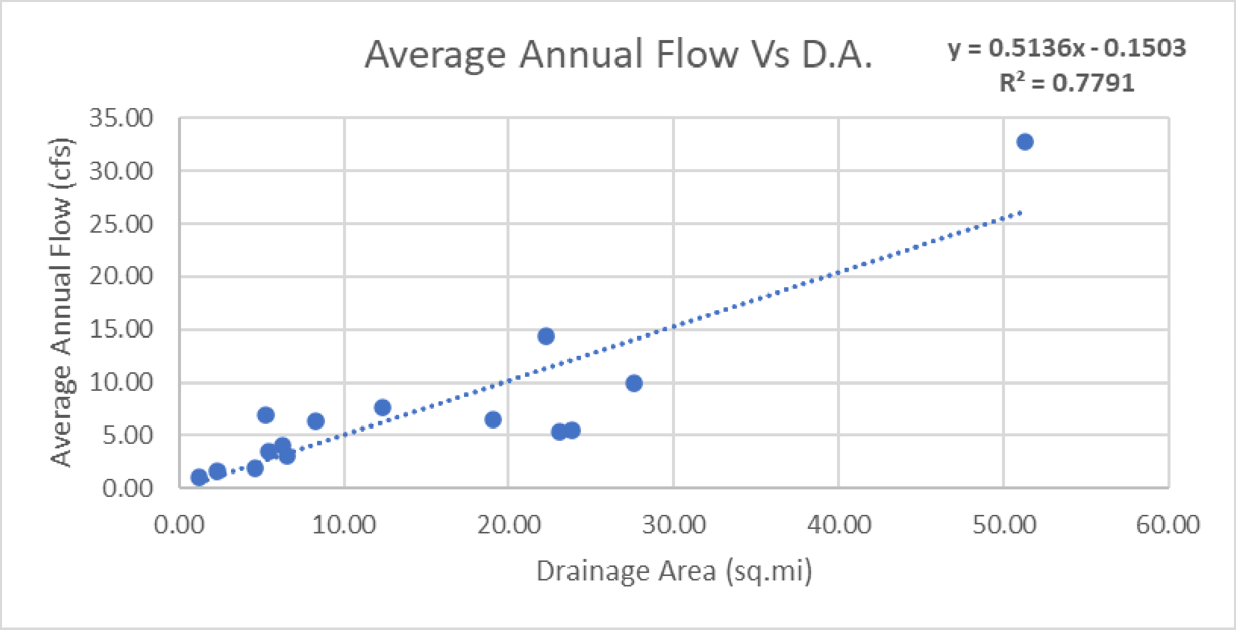


Fig 10: Relation between average annual flow vs drainage area

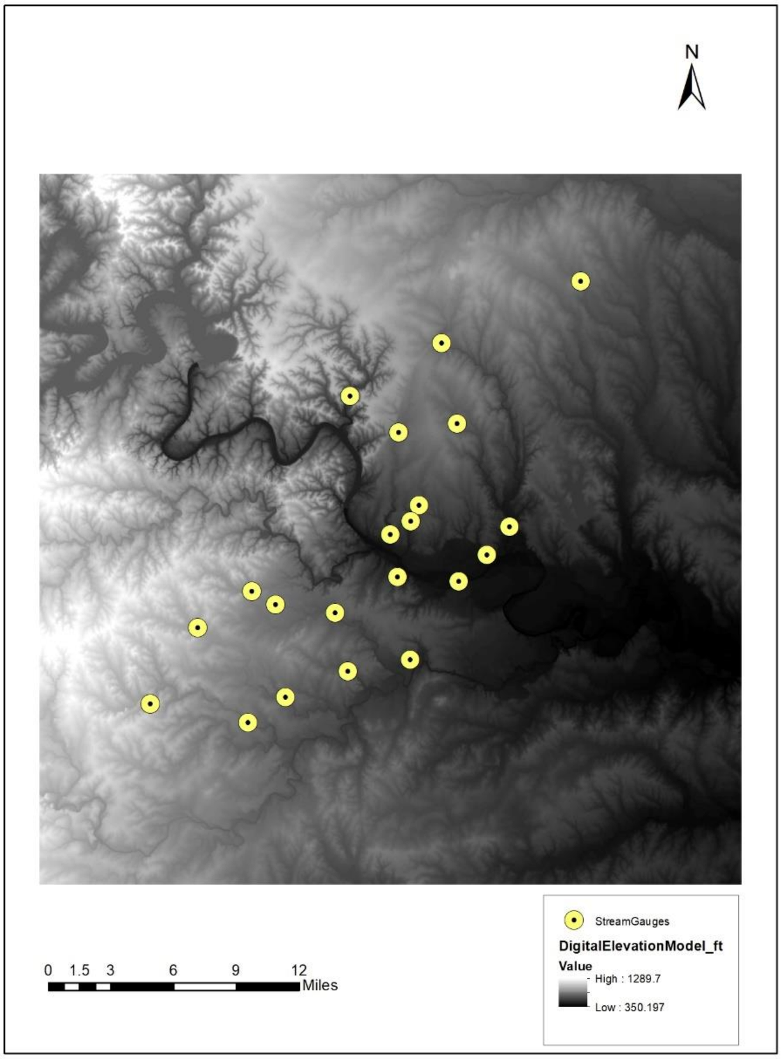


Fig 11: Stream Gauge Locations in Austin City

As can be seen, the R^2 values for both the graphs is 0.8238 and 0.7791, which quite reasonable. Therefore, these equations are used to form the grid of bank full flow and average annual flow.

After all the velocities are calculated, a random point was chosen and the values of flow velocity till three cells into the open channel along the flow direction is recorded to visualize the fluctuations in the velocity owing to slope amd landuse. As can be seen the figure 12, the sheet flow ended at the orange line of 91.44 meters from the start of flow. The shallow concentrated flow starts at the orange line and ends at the black one.

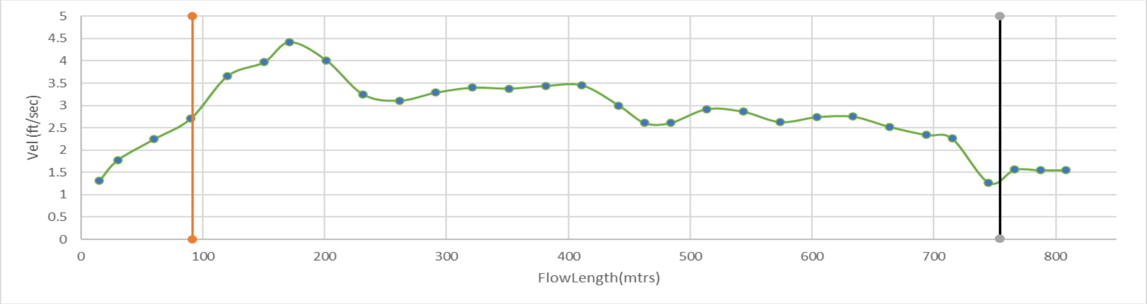


Fig 12: Velocity changing along a particular flowpath

Finally, after the formula for rational method was employed, the cumulative increase of area can also be observed in figure 13.

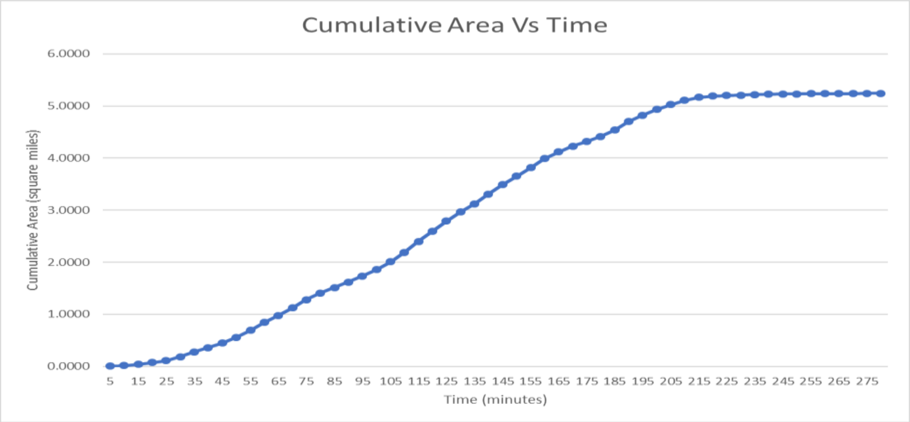


Fig 13: Graph of Cumulative Area vs time

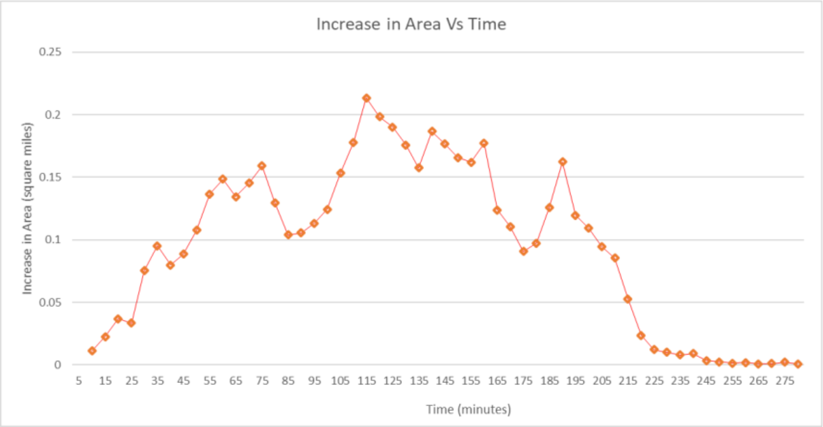
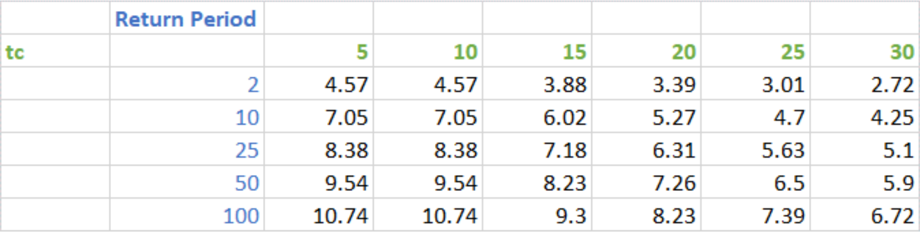


Fig 14: Graph of Increase in Area each five minutes

For brevity, a portion of the table of intensities with corresponding return period and time of concentration is being displayed (Tay, Neale, Herrmann, & Cleveland, 2015) in table 4.



	Return Period						
tc		5	10	15	20	25	30
	2	4.57	4.57	3.88	3.39	3.01	2.72
	10	7.05	7.05	6.02	5.27	4.7	4.25
	25	8.38	8.38	7.18	6.31	5.63	5.1
	50	9.54	9.54	8.23	7.26	6.5	5.9
	100	10.74	10.74	9.3	8.23	7.39	6.72

Table 4: Intensities from IDF for Texas (TxDOT Hydraulic Manual, 2019).

The final graph of flow in cubic feet per second vs time in minutes is obtained in figure 13.

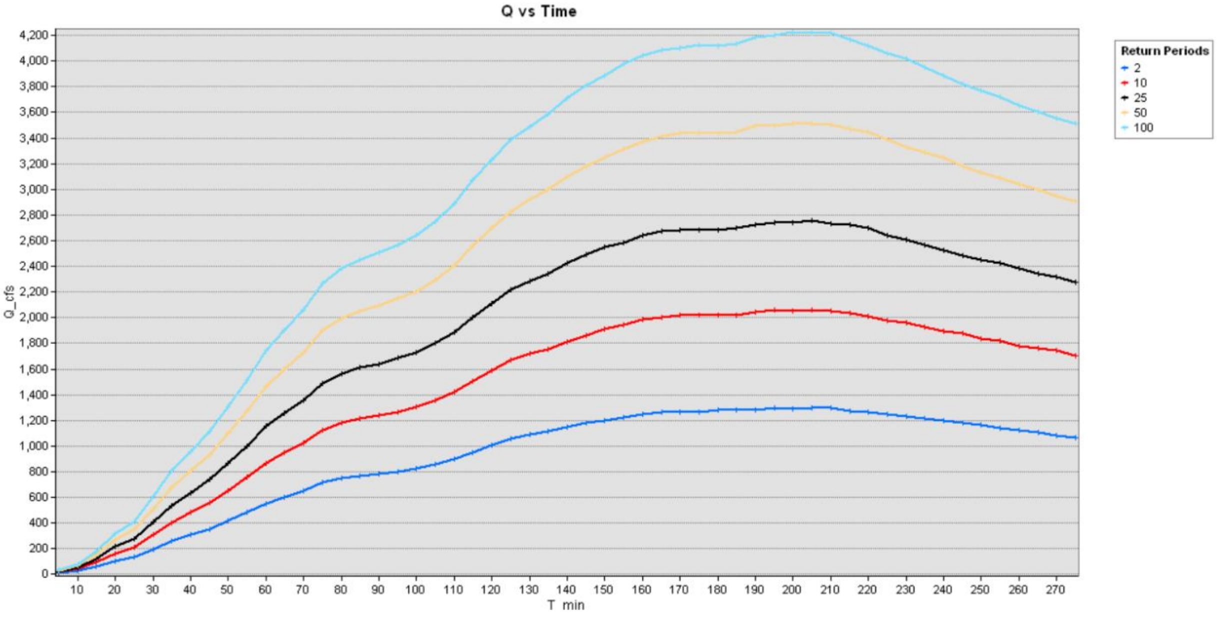


Fig 15: Output graph of Q vs Time for different return period rainfall

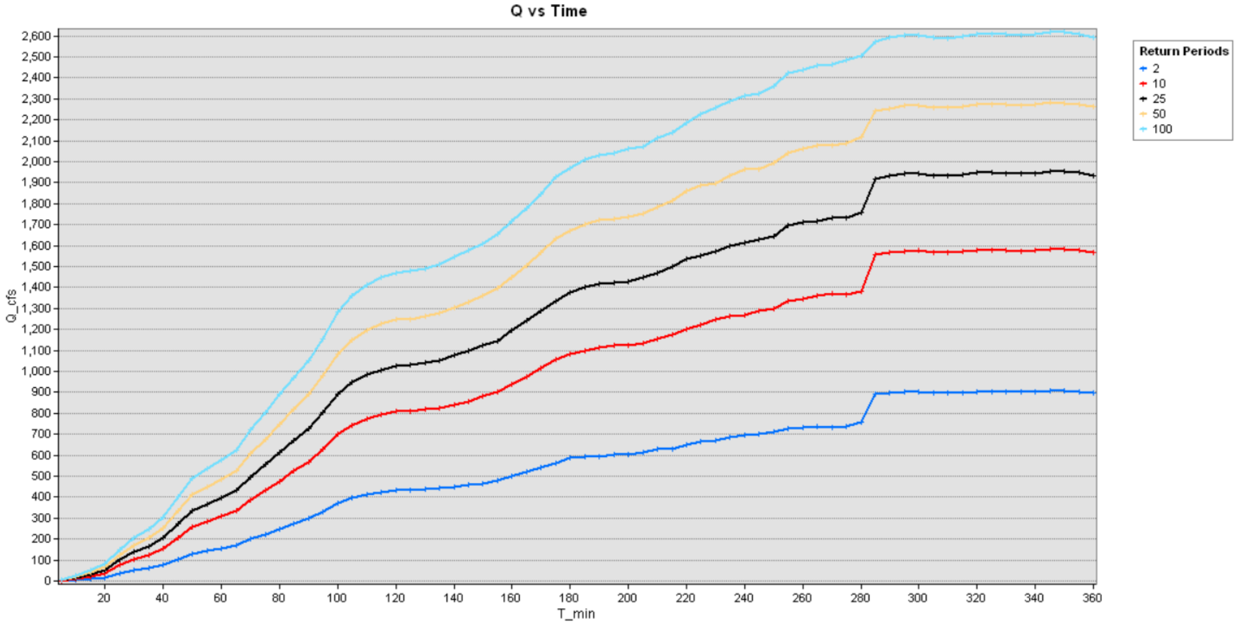


Fig 16: Output graph of Q vs Time for different return period rainfall (Version B)

Decisions concerning response time can be made based on these graphs. The second graph can be more relied upon as the study area is 3242 acres. The trend in the first graph dips due to the intensity decreasing with time of concentration, in the IDF curves. The trend in the second graph flatlines at the end due to decrease in the percentage increase in area. As the rate of increase in area slows down as the time increases, the city manager can decide based on available resources, at which time they choose to respond. At present, if it is not known where the high flows will occur, expensive machinery needs to be deployed across the study area. This leads to high budget requirement, which may not be available to many cities. However, this tool allows the city to narrow down the most probable spots. Having done so, the accuracy of results can be increased by utilizing tailored methods to the particular area. However, the cost shall be much less as many locations will be ruled out.

5. SUMMARY AND CONCLUSIONS

The flash flood screening tool, presented here, takes a Digital Elevation Model (DEM) and a Land-Use/Land-Cover map to compute basin characteristics. Flow is modeled according to TR-55, which classifies it as sheet, shallow-concentrated or open-channel flow. With the help of various Geographic Information Systems tools, the drainage area of a user-given outlet point, as well as the flow lengths upstream to the headwaters and downstream to the outlet, can be calculated. These flow lengths are then used to classify the areas of sheet flow, while the areas of open-channel flow are user-defined. Velocities for sheet and shallow-concentrated flow are determined according to TR-55, while the velocities for open-channel flow are user-defined. Once the velocities are known, isochrones are determined and area vs. flow-time plots are developed. Intensity-duration-frequency (IDF) curves are used to define the precipitation for different concentration times and return periods. The rational method is then used to determine the expected peak flows. This screening tool will help policymakers identify locations of potential high flash flood chance in which more-detailed hydrologic analysis would be needed.

As such the final graph can be used for various planning or zeroing down purposes. As the undulations of the curve can be seen – given the reduction in intensity and percentage of area increase, the resources of aid and time of aid in case of flash floods can be decided by the city managers. Alongside, as this is a screening tool, it aims to reduce the cost required for extensive analysis all around the city and allows for focusing on dangerous zones. If, however, a flow depth (stage) to discharge, i.e. Rating curve is available for those particular locations – then the obvious red flag zones can be screened out easily.

The two versions of the tool are given such that the first version would be applicable in analysis of smaller watersheds as the limitation of rational method also applies here. Therefore, watersheds below 200 acres can be evaluated with the first version of the tool. So, small creeks and streams will be the covered through this method. As for locations with larger catchment area, the version B of the tool can be employed for the same result. Therefore, larger streams can be evaluated through this.

In a nutshell, this tool aims at cutting down cost and work to screen out flash flood prone zones, and can be in the future, developed for socio-economic damage assessment.

REFERENCES

- Bhaskar, N. R., French, M. N., & Kyiamah, G. K. (2000). Characterization of Flash Floods in Eastern Kentucky. *Journal of Hydrologic Engineering, 5*(3), 327-331. doi:10.1061/(asce)1084-0699(2000)5:3(327)
- Brath, A., & Rosso, R. (1993). Adaptive calibration of a conceptual model for flash flood forecasting. *Water Resources Research, 29*(8), 2561-2572. doi:10.1029/93wr00665
- Costache, R., Hong, H., & Wang, Y. (2019). Identification of torrential valleys using GIS and a novel hybrid integration of artificial intelligence, machine learning and bivariate statistics. *Catena, 183*, 104179. doi:10.1016/j.catena.2019.104179
- Sharif, H. O., Jackson, T. L., Hossain, M. M., & Zane, D. (2015). Analysis of Flood Fatalities in Texas. *Natural Hazards Review*, *16*(1), 04014016. doi: 10.1061/(asce)nh.1527-6996.0000145
- Bisht, S., Chaudhry, S., Sharma, S., & Soni, S. (2018). Assessment of flash flood vulnerability zonation through Geospatial technique in high altitude Himalayan watershed, Himachal Pradesh India. *Remote Sensing Applications: Society and Environment, 12*, 35-47. doi:10.1016/j.rsase.2018.09.001
- Costache, R., & Bui, D. T. (2019). Identification of areas prone to flash-flood phenomena using multiple-criteria decision-making, bivariate statistics, machine learning and their ensembles. *Science of The Total Environment, 712,* 136492. doi:10.1016/j.scitotenv.2019.136492

- Cronshey, R., & Cronshey, R. (1986). *Urban hydrology for small watersheds*. (TR-55), Washington, D.C.: U.S. Dept. of Agriculture, Soil Conservation Service, Engineering Division.
- Debo, T. N., Reese, A. J., & Debo, T. N. (2003). *Municipal stormwater management*. Boca Raton, FL: Lewis Publishers.
- Hydraulic Design Manual, Texas Department of Transportation, September 2019.
- Georgakakos, K. P. (1986). A generalized stochastic hydrometeorological model for flood and flash-flood forecasting: 2. Case studies. *Water Resources Research, 22*(13), 2096-2106. doi:10.1029/wr022i013p02096
- Grabs, (2010). Regional Flash Flood Guidance and Early Warning System. Associated Programme on Flood Management, WMO, Switzerland (2010)
- Hapuarachchi, H. A., Wang, Q. J., & Pagano, T. C. (2011). A review of advances in flash flood forecasting. *Hydrological Processes, 25*(18), 2771-2784. doi:10.1002/hyp.8040
- Jobson, H.E., 1996, Prediction of traveltime and longitudinal dispersion in rivers and streams: U.S. Geological Survey Water-Resources Investigations Report 96–4013, 69 p.; https://pubs.usgs.gov/of/1996/4013.
- Jonkman, (2005). Global perspectives on loss of human life caused by floods. Nat. Hazards, 34 (2005), pp. 151-157
- Karagiorgos, K., Thaler, T., Heiser, M., Hübl, J., & Fuchs, S. (2016). Integrated flash flood vulnerability assessment: Insights from East Attica, Greece. *Journal of Hydrology, 541*, 553-562. doi:10.1016/j.jhydrol.2016.02.052

- Kim, B., & Kim, H. (2013). Evaluation of flash flood severity in Korea using the modified flash flood index (MFFI). *Journal of Flood Risk Management, 7*(4), 344-356. doi:10.1111/jfr3.12057
- Konrad, C. P. (2003). Effects of Urban Development on Floods. *Fact Sheet*. doi: 10.3133/fs07603
- Kuichling, E. (1889) The Relation between the Rainfall and the Discharge of Sewers in Populous Districts. Transactions of ASCE, 20, 1-60.
- Lee, E. H., & Kim, J. H. (2018). Development of a flood-damage-based flood forecasting technique. *Journal of Hydrology, 563*, 181-194. doi:10.1016/j.jhydrol.2018.06.003
- Modrick and Georgakakos, (2015) The character and causes of flash flood occurrence changes in mountainous small basins of Southern California under projected climatic change. J. Hydrol. Reg. Stud., 3 (2015), pp. 312-336
- Mohamed, S. A. (2019). Application of satellite image processing and GIS-Spatial modeling for mapping urban areas prone to flash floods in Qena governorate, Egypt. *Journal of African Earth Sciences, 158*, 103507. doi:10.1016/j.jafrearsci.2019.05.015
- Moore RJ. 1985. The probability-distributed principle and runoff production at point and basin scales. Hydrological Sciences Journal 30: 273–297.
- Naulin, J., Payrastre, O., & Gaume, E. (2013). Spatially distributed flood forecasting in flash flood prone areas: Application to road network supervision in Southern France. *Journal of Hydrology, 486*, 88-99. doi:10.1016/j.jhydrol.2013.01.044
- Noaa. (2004, November 1). Glossary. Retrieved from https://w1.weather.gov/glossary/
- Norbiato D, Borga M, Degli Esposti S, Gaume E, Anquetin S. 2008. Flash flood warning based on rainfall depth-duration thresholds and soil moisture conditions: An assessment

for gauged and ungauged basins. Journal of Hydrology 362(3–4): 274 –290, doi: 10.1016/j.jhydrol.2008.08.023

- Norbiato, D., Borga, M., & Dinale, R. (2009). Flash flood warning in ungauged basins by use of the flash flood guidance and model-based runoff thresholds. *Meteorological Applications, 16*(1), 65-75. doi:10.1002/met.126
- Poertner, H. (1974). Practices in Detention of Urban Stormwater Runoff. APWA Special Report No. 43. Washington, D.C.: American Public Works Association.
- Rozalis, S., Morin, E., Yair, Y., & Price, C. (2010). Flash flood prediction using an uncalibrated hydrological model and radar rainfall data in a Mediterranean watershed under changing hydrological conditions. *Journal of Hydrology, 394*(1-2), 245-255. doi:10.1016/j.jhydrol.2010.03.021
- Saharia, M., Kirstetter, P.-E., Vergara, H., Gourley, J. J., Hong, Y., & Giroud, M. (2017). Mapping Flash Flood Severity in the United States. *Journal of Hydrometeorology, 18*(2), 397–411. doi: 10.1175/jhm-d-16-0082.1
- Strager, M.P. (2012). Tools for Watershed Planning – Development of a Statewide Source Water Protection System (SWPS). In Nayak, P (Ed), Water Resources Management and Modeling (pp. 3-38). Croatia: InTech.
- Taha, M. M., Elbarbary, S. M., Naguib, D. M., & El-Shamy, I. (2017). Flash flood hazard zonation based on basin morphometry using remote sensing and GIS techniques: A case study of Wadi Qena basin, Eastern Desert, Egypt. *Remote Sensing Applications: Society and Environment, 8*, 157-167. doi:10.1016/j.rsase.2017.08.007
- Thompson, D.B. The Rational Method; Civil Engineering Department, Texas Tech University: Lubbock, TX, USA, 2006.

- U.S. Dept. of Agriculture, Soil Conservation Service. (1993). *National engineering handbook: Section 4: Part 630, Hydrologic engineering*. Washington, D.C.
- Volkmann, T. H., Lyon, S. W., Gupta, H. V., & Troch, P. A. (2010). Multicriteria design of rain gauge networks for flash flood prediction in semiarid catchments with complex terrain. *Water Resources Research, 46*(11). doi:10.1029/2010wr009145
- Wells, P. I., & Woodward, D. (1986). *Time of concentration*. Chester, PA: Northeast NTC.
- Yatheendradas, S., Wagener, T., Gupta, H., Unkrich, C., Goodrich, D., Schaffner, M., & Stewart, A. (2008). Understanding uncertainty in distributed flash flood forecasting for semiarid regions. *Water Resources Research, 44*(5). doi:10.1029/2007wr005940

APPENDIX A

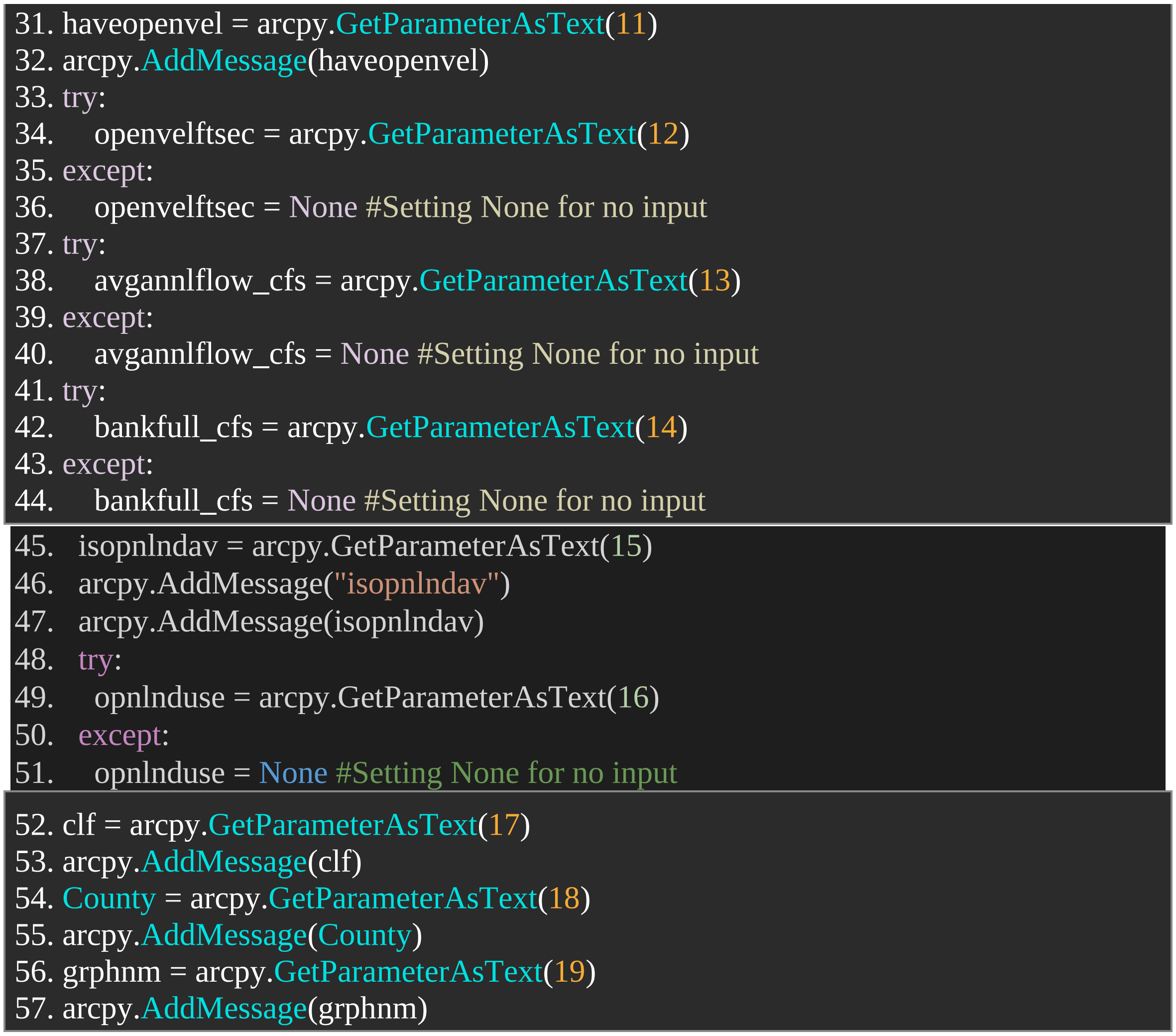
CODE MANUAL

```
import os
import random
import sys
import arcpy
from arcpy import env
from arcpy.sa import *
```

Fig 17: Importing libraries.

The arcpy, sys, random and os modules are imported in these lines. These modules will contain all the functions to be used in this tool.

```
path = arcpy.GetParameterAsText(0)
toolpath = arcpy.GetParameterAsText(1)
pourpoint = arcpy.GetParameterAsText(2)
dem = arcpy.GetParameterAsText(3)
dem = Raster(dem) #Forms the raster layer in the ArcMap interface
threshold = arcpy.GetParameterAsText(4)
arcpy.AddMessage(threshold)
p2 = arcpy.GetParameterAsText(5)
arcpy.AddMessage(p2)
landuse = arcpy.GetParameterAsText(6)
o = arcpy.GetParameterAsText(7)
arcpy.AddMessage(o)
try:
    pgnm = arcpy.GetParameterAsText(8)
except:
    pgnm = None #Setting None for no input
arcpy.AddMessage(pgnm)
try:
    pgname = arcpy.GetParameterAsText(9)
except:
    pgname = None #Setting None for no input
arcpy.AddMessage(pgname)
demarc = arcpy.GetParameterAsText(10)
arcpy.AddMessage(demarc)
```



```python
haveopenvel = arcpy.GetParameterAsText(11)
arcpy.AddMessage(haveopenvel)
try:
    openvelftsec = arcpy.GetParameterAsText(12)
except:
    openvelftsec = None #Setting None for no input
try:
    avgannlflow_cfs = arcpy.GetParameterAsText(13)
except:
    avgannlflow_cfs = None #Setting None for no input
try:
    bankfull_cfs = arcpy.GetParameterAsText(14)
except:
    bankfull_cfs = None #Setting None for no input
isopnlndav = arcpy.GetParameterAsText(15)
arcpy.AddMessage("isopnlndav")
arcpy.AddMessage(isopnlndav)
try:
    opnlnduse = arcpy.GetParameterAsText(16)
except:
    opnlnduse = None #Setting None for no input
clf = arcpy.GetParameterAsText(17)
arcpy.AddMessage(clf)
County = arcpy.GetParameterAsText(18)
arcpy.AddMessage(County)
grphnm = arcpy.GetParameterAsText(19)
arcpy.AddMessage(grphnm)
```

Fig 18: Parameterization.

The **path** variable stores the address of the geodatabase where all the files created will be stored in.

The **toolpath** variable stores the path of tool location folder is stored here, i.e. the ***address of the folder which has the ScrTool Folder*** stored in it.

The **pourpoint** variable stores the shapefile of the point outlet. The point outlet is the location the study is interested in.

The **dem** variable stores the input of Digital Elevation Model. The x, y, z units are taken in feet.

The **threshold** stores the threshold value for river delineation. This stores the lowest accumulating cell number for a cell to be considered as an open channel flow.

The **p2** variable stores the precipitation depth value of 2-year 24 hour in the units of inches.

The **landuse** variable stores the Land Use/ Land Cover layer.

The **o** variable stores the Boolean variable to check if land use is from NLCD.

The **pgnm** variable is an optional Parameter for new sheet name for land use markup table for sheet flow if it is not from NLCD.

The **pgname** variable is an optional Parameter for new sheet name for land use markup table for shallow concentrated flow if it is not from NLCD.

The **demarc** variable stores the upstream flowlength which differentiates the sheet flow from the shallow concentrated flow.

The **haveopenvel** variable stores the Boolean variable whether open flow velocity raster layer is available or otherwise if the optional model needs to be used.

The **openvelmsec** variable is the optional Parameter to store open channel flow velocity in ft/sec.

The **bankfull_cfs** variable is the optional Parameter to store bankfull flow in cfs if the optional model is used.

Similarly, the **avgannlflow_cfs** variable stores the average annual flow layer in cfs.

The **isopnlndav** variable stores the Boolean variable to check if there are some land use zones apart from the flow line which had open water or wetlands.

The **opnlnduse** variable stores the areas containing open water land use or wetlands.

The **clf** variable stores intervals for visual differentiation of time, to be produced in a layer so that the demarcations of area according to specified time is clearly visible to the user.

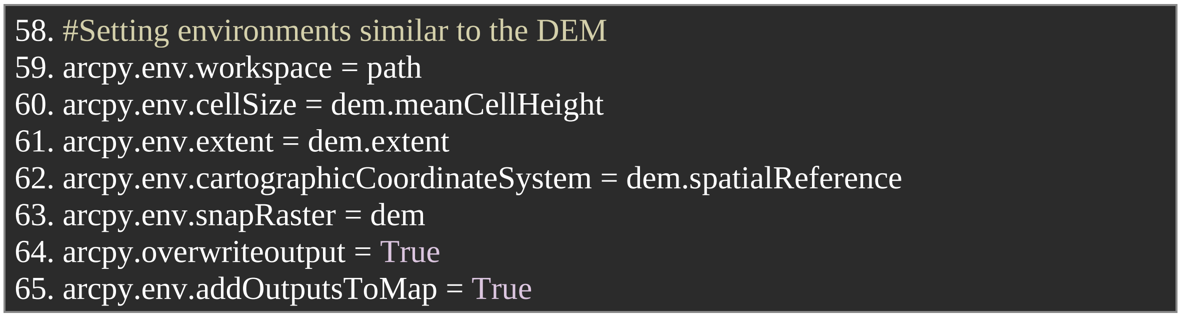
The **County** variable stores the county name to obtain intensity values stored in the excel file.

The **grphnm** variable stores the name desired for the final graph to be saved in.

The Raster functions are to form the layers on the Geographic Information System (GIS) document, so that it can be called on by the functions by layer name.

The AddMessage functions are visual confirmations of the value inputs in the tool.

The try except are used for the optional parameters so that it does not throw an error.



```
58. #Setting environments similar to the DEM
59. arcpy.env.workspace = path
60. arcpy.env.cellSize = dem.meanCellHeight
61. arcpy.env.extent = dem.extent
62. arcpy.env.cartographicCoordinateSystem = dem.spatialReference
63. arcpy.env.snapRaster = dem
64. arcpy.overwriteoutput = True
65. arcpy.env.addOutputsToMap = True
```

Fig 19: Setting Environments.

This section focuses on setting environments for the upcoming procedures. Some of them will be updated as we proceed through the code.

The 59th line sets the workspace to be similar to the provided file geodatabase.

The 60th line sets all cell size of the output layers to be similar to the Digital Elevation Model's (DEM) cell size.

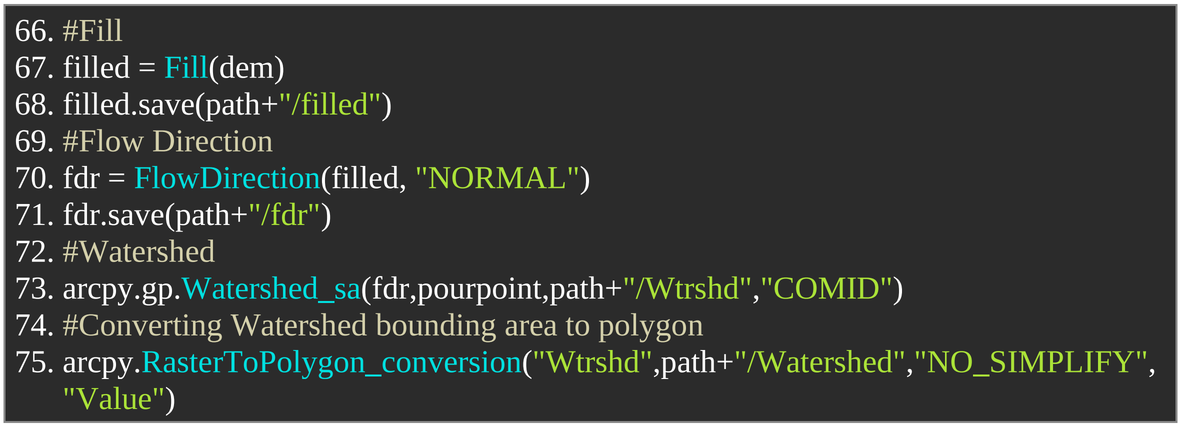
Then similarly in the 61st line the extent of computation is set to the current DEM.

The 62nd line sets the coordinate system to be similar to the DEMs'. This is why all the other inputted layers need to be in the similar properties as the DEM.

The 63rd line snaps all output raster to the DEM layer.

The 64th line overwrites outputs to make it easier for multiple runs of the tool.

The 65th line shows all outputs on map.



```
66. #Fill
67. filled = Fill(dem)
68. filled.save(path+"/filled")
69. #Flow Direction
70. fdr = FlowDirection(filled, "NORMAL")
71. fdr.save(path+"/fdr")
72. #Watershed
73. arcpy.gp.Watershed_sa(fdr,pourpoint,path+"/Wtrshd","COMID")
74. #Converting Watershed bounding area to polygon
75. arcpy.RasterToPolygon_conversion("Wtrshd",path+"/Watershed","NO_SIMPLIFY",
    "Value")
```

Fig 20: Watershed delineation.

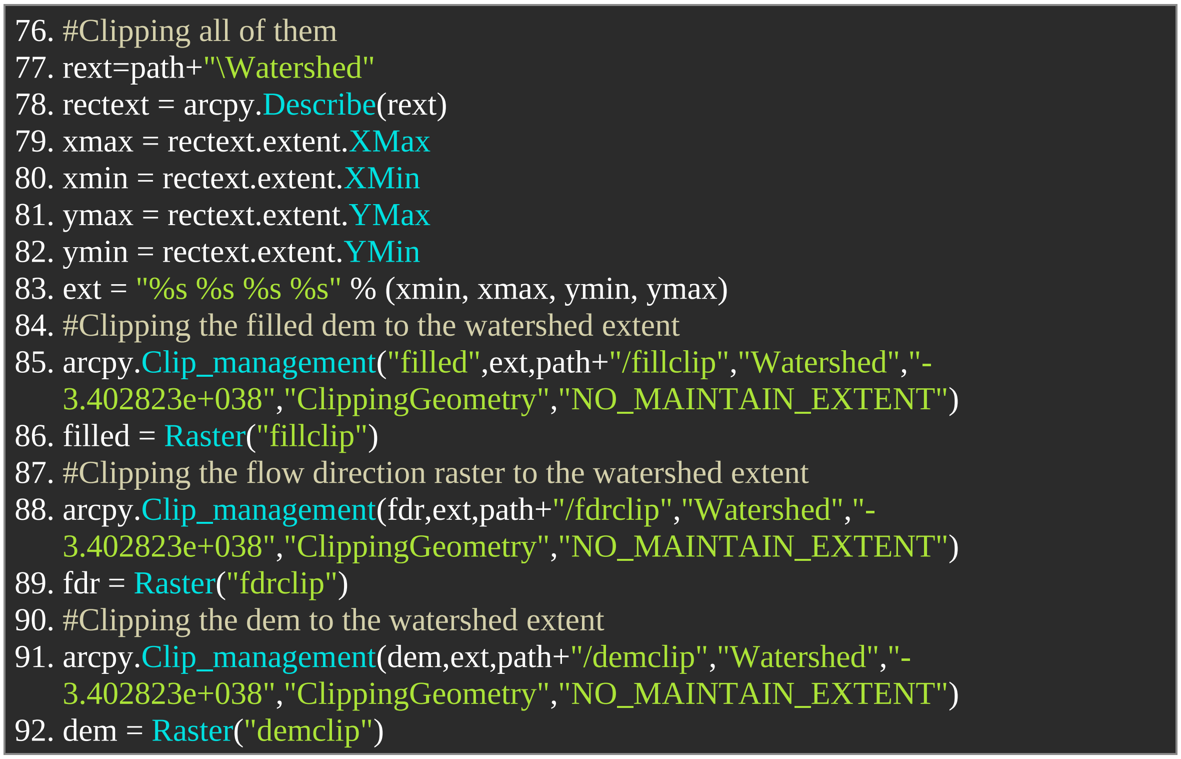
This set of lines delineates the watershed for the given pourpoint.

The 67th line uses the fill function on DEM to eradicate sinks and the 68th line saves it in the given geodatabase.

The 70th line calculates the flow direction on the filled DEM and the 71st line saves that in the geodatabase.

The 73rd line delineates the watershed from outlet and flow direction.

The 75th line vectorizes the obtained watershed zone for easier clipping.



```
76. #Clipping all of them
77. rext=path+"\Watershed"
78. rectext = arcpy.Describe(rext)
79. xmax = rectext.extent.XMax
80. xmin = rectext.extent.XMin
81. ymax = rectext.extent.YMax
82. ymin = rectext.extent.YMin
83. ext = "%s %s %s %s" % (xmin, xmax, ymin, ymax)
84. #Clipping the filled dem to the watershed extent
85. arcpy.Clip_management("filled",ext,path+"/fillclip","Watershed","-
    3.402823e+038","ClippingGeometry","NO_MAINTAIN_EXTENT")
86. filled = Raster("fillclip")
87. #Clipping the flow direction raster to the watershed extent
88. arcpy.Clip_management(fdr,ext,path+"/fdrclip","Watershed","-
    3.402823e+038","ClippingGeometry","NO_MAINTAIN_EXTENT")
89. fdr = Raster("fdrclip")
90. #Clipping the dem to the watershed extent
91. arcpy.Clip_management(dem,ext,path+"/demclip","Watershed","-
    3.402823e+038","ClippingGeometry","NO_MAINTAIN_EXTENT")
92. dem = Raster("demclip")
```

Fig 21: Clipping the results to match the desired watershed.

This section clips all the previous layers to the current watershed. For clipping the rectangular extent of the watershed also needs to be provided alongside the shapefile itself.

So, the **rext** variable saves the path of watershed polygon as a string. It is then called in the Describe function to access its properties.

The **rectext** variable stores the properties of the Watershed.

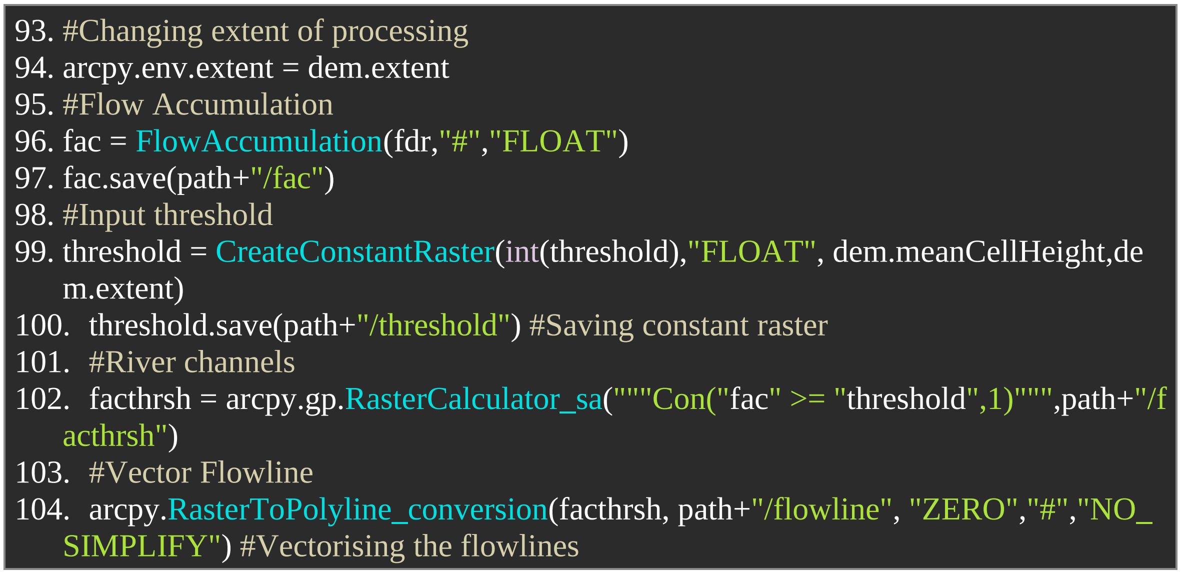
The **xmax, xmin, ymax, ymin** variable stores all the four extents of the Watershed separately and the **ext** variable brings them back together in a string in the format needed for input in the function.

The 85th line clips the filled raster according to the watershed polygon and the 86th line stores it in the **filled** variable. Maintain extent is turned off and clip in accordance to clipping geometry is turned on.

The 88th line clips the flow direction raster according to the watershed polygon and the 89th line stores it in the **fdr** variable. Maintain extent is turned off and clip in accordance to clipping geometry is turned on.

The 91st line clips the dem raster according to the watershed polygon and the 92nd line stores it in the **dem** variable. Maintain extent is turned off and clip in accordance to clipping geometry is turned on.

The **-3.402823e+038** value is the default no data value.



```
93. #Changing extent of processing
94. arcpy.env.extent = dem.extent
95. #Flow Accumulation
96. fac = FlowAccumulation(fdr,"#","FLOAT")
97. fac.save(path+"/fac")
98. #Input threshold
99. threshold = CreateConstantRaster(int(threshold),"FLOAT", dem.meanCellHeight,dem.extent)
100.  threshold.save(path+"/threshold") #Saving constant raster
101.  #River channels
102.  facthrsh = arcpy.gp.RasterCalculator_sa("""Con("fac" >= "threshold",1)""",path+"/facthrsh")
103.  #Vector Flowline
104.  arcpy.RasterToPolyline_conversion(facthrsh, path+"/flowline", "ZERO","#","NO_SIMPLIFY") #Vectorising the flowlines
```

Fig 22: Delineating river flow lines.

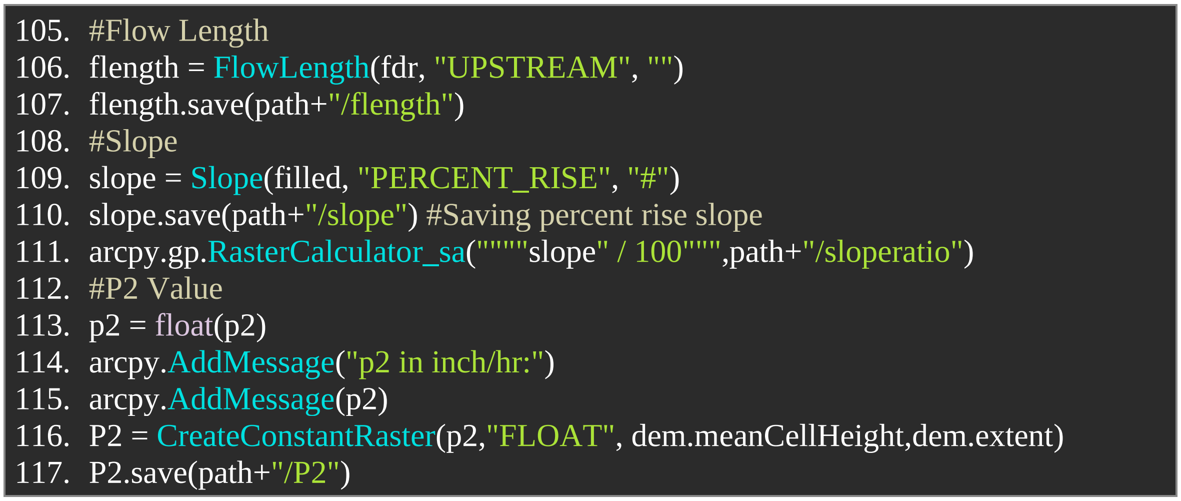
This section delineates the river lines/ open channel flow lines. The 94th line sets the extent from the previous DEM to the new clipped DEM.

The 96th line calls on the flow accumulation function using the flow direction layer in the **fac** variable and the resultant layer is stored in the geodatabase in the 97th line.

The 99th line creates a constant raster of the inputted threshold value for easier calculations and the resultant layer is stored in the geodatabase in the 100th line.

The 102nd layer delineates the river using the conditional to be above the threshold value to be assigned as 1 using raster calculator and stores this as **facthrsh** layer in the geodatabase.

The 104th line vectorizes **facthrsh** and stores it as **flowline.** The polygons are not simplified.



```
105.  #Flow Length
106.  flength = FlowLength(fdr, "UPSTREAM", "")
107.  flength.save(path+"/flength")
108.  #Slope
109.  slope = Slope(filled, "PERCENT_RISE", "#")
110.  slope.save(path+"/slope") #Saving percent rise slope
111.  arcpy.gp.RasterCalculator_sa(""""slope" / 100""",path+"/sloperatio")
112.  #P2 Value
113.  p2 = float(p2)
114.  arcpy.AddMessage("p2 in inch/hr:")
115.  arcpy.AddMessage(p2)
116.  P2 = CreateConstantRaster(p2,"FLOAT", dem.meanCellHeight,dem.extent)
117.  P2.save(path+"/P2")
```

Fig 23: Calculation of flow length, slope and p2 value.

This section calculates flow length, slope and p2 value, respectively.

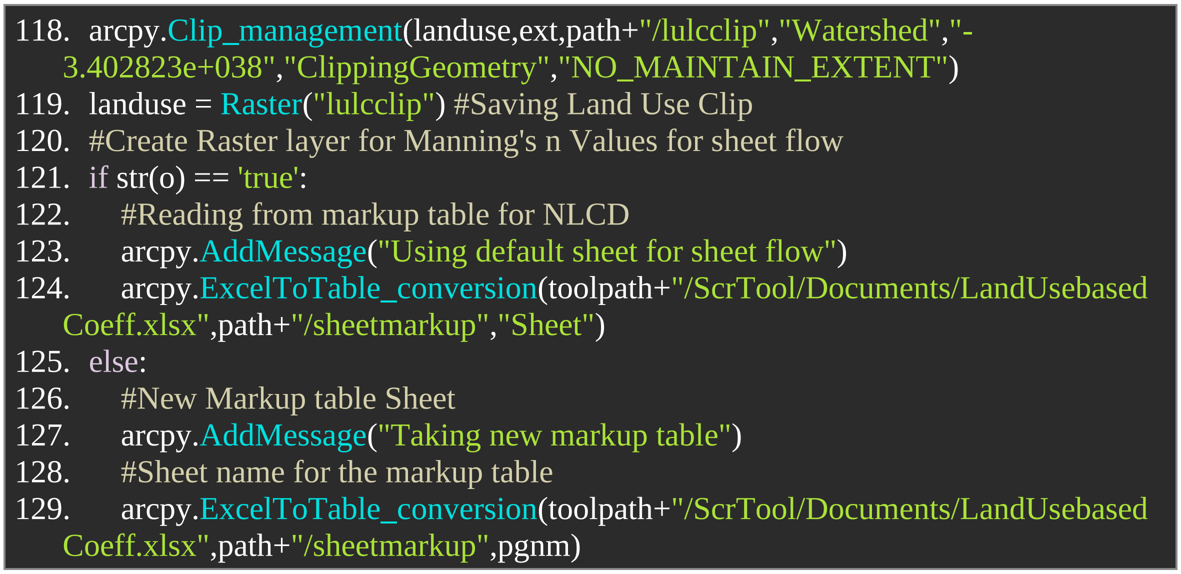
The 106th line calls on the flow length function to calculate upstream flow length, with given flow direction layer. It is then saved as **flength** layer in the geodatabase in line 107th.

The 109th line calculates slope from DEM in percentage rise and the 110th line stores it in geodatabase under the name **slope.**

The 111th line changes it into a ratio format as needed for our calculations by dividing the raster by 100 through raster calculator and saving it as **sloperatio.**

The 113th line puts **p2** variable in float type.

The 116th line creates a constant raster of the inputted the p2 value for easier calculations and the resultant layer is stored in the geodatabase as **P2** in the 117th line.



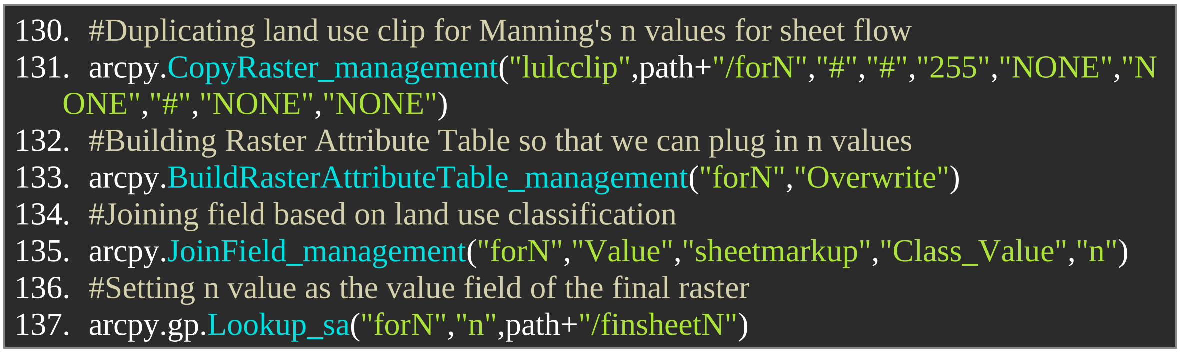
```
118.  arcpy.Clip_management(landuse,ext,path+"/lulcclip","Watershed","-
    3.402823e+038","ClippingGeometry","NO_MAINTAIN_EXTENT")
119.  landuse = Raster("lulcclip") #Saving Land Use Clip
120.  #Create Raster layer for Manning's n Values for sheet flow
121.  if str(o) == 'true':
122.      #Reading from markup table for NLCD
123.      arcpy.AddMessage("Using default sheet for sheet flow")
124.      arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/LandUsebased
    Coeff.xlsx",path+"/sheetmarkup","Sheet")
125.  else:
126.      #New Markup table Sheet
127.      arcpy.AddMessage("Taking new markup table")
128.      #Sheet name for the markup table
129.      arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/LandUsebased
    Coeff.xlsx",path+"/sheetmarkup",pgnm)
```

Fig 24: Calculation of flow length, slope and p2 value.

This section is used for preparing Land Use/ Land Cover layer for further calculations. In the Land Use layer, the open water and wetland cases need to be removed before inputting.

The 118th line clips the land use/ land cover to the watershed extent and stores it in as **lulcclip** in the geodatabase. The 119th line stores it in the **landuse** variable back again.

The if loop checks if the Land Use/ Land Cover layer is from NLCD and whichever the case, the respective table is imported from the excel file as the markup attribute table. The creation of this table is defined more broadly in the user's manual. This is for the sheet flow.



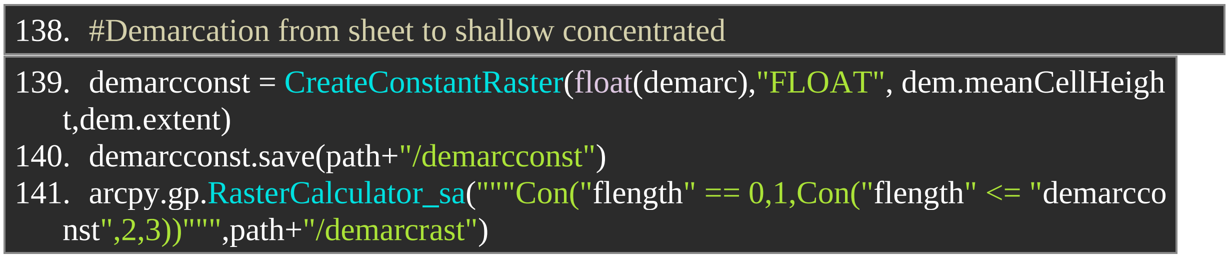
```
130.  #Duplicating land use clip for Manning's n values for sheet flow
131.  arcpy.CopyRaster_management("lulcclip",path+"/forN","#","#","255","NONE","NONE","#","NONE","NONE")
132.  #Building Raster Attribute Table so that we can plug in n values
133.  arcpy.BuildRasterAttributeTable_management("forN","Overwrite")
134.  #Joining field based on land use classification
135.  arcpy.JoinField_management("forN","Value","sheetmarkup","Class_Value","n")
136.  #Setting n value as the value field of the final raster
137.  arcpy.gp.Lookup_sa("forN","n",path+"/finsheetN")
```

Fig 25: Manning's n value allocation.

The 131st line duplicates **lulcclip** to **forN** to plug in the Manning's n values in the new raster. The 133rd line builds the attribute table consisting of unique classes as it gets lost in copying.

The 135th line joins the markup table and newly created attribute table on the basis of Value field in **forN** and Class_Value field in **sheetmarkup.** The field n is joined in.

The 137th line sets the n value as the value field of the raster layer which is saved as **finsheetN.**



```
138.  #Demarcation from sheet to shallow concentrated
139.  demarcconst = CreateConstantRaster(float(demarc),"FLOAT", dem.meanCellHeight,dem.extent)
140.  demarcconst.save(path+"/demarcconst")
141.  arcpy.gp.RasterCalculator_sa("""Con("flength" == 0,1,Con("flength" <= "demarcconst",2,3))""",path+"/demarcrast")
```

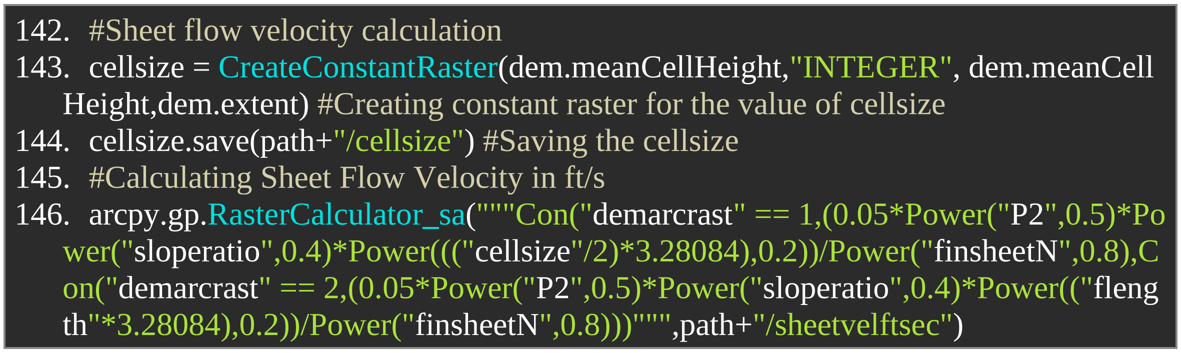
Fig 26: Demarcation of the sheet flow cells from the shallow concentrated ones.

This section deals with demarcating the sheet flow cells from the shallow concentrated ones.

The **demarcconst** in line 139, stores and creates a constant raster with the demarcation value for easier calculations.

The 140th line separately saves the layer in the geodatabase as **demarcconst.**

The 141st line separates the flowlength of zero, below threshold and above threshold as 1,2 and 3 respectively to identify the portions. As apparent, the 1 and 2 signify sheet flow and the 3 signify shallow concentrated flow.



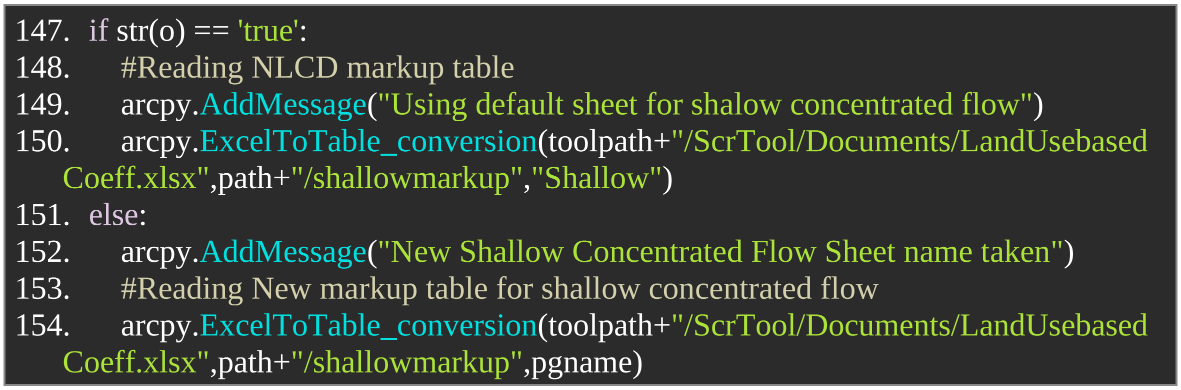
```
142. #Sheet flow velocity calculation
143. cellsize = CreateConstantRaster(dem.meanCellHeight,"INTEGER", dem.meanCellHeight,dem.extent) #Creating constant raster for the value of cellsize
144. cellsize.save(path+"/cellsize") #Saving the cellsize
145. #Calculating Sheet Flow Velocity in ft/s
146. arcpy.gp.RasterCalculator_sa("""Con("demarcrast" == 1,(0.05*Power("P2",0.5)*Power("sloperatio",0.4)*Power((("cellsize"/2)*3.28084),0.2))/Power("finsheetN",0.8),Con("demarcrast" == 2,(0.05*Power("P2",0.5)*Power("sloperatio",0.4)*Power(("flength"*3.28084),0.2))/Power("finsheetN",0.8)))""",path+"/sheetvelftsec")
```

Fig 27: Sheet Flow Velocity Calculation.

This section calculates sheet flow velocity in ft/sec.

The 143rd line creates a constant raster of the cellsize value taken from **dem** and saves it in **cellsize** variable. The 144th line saves it as the **cellsize** layer in the geodatabase.

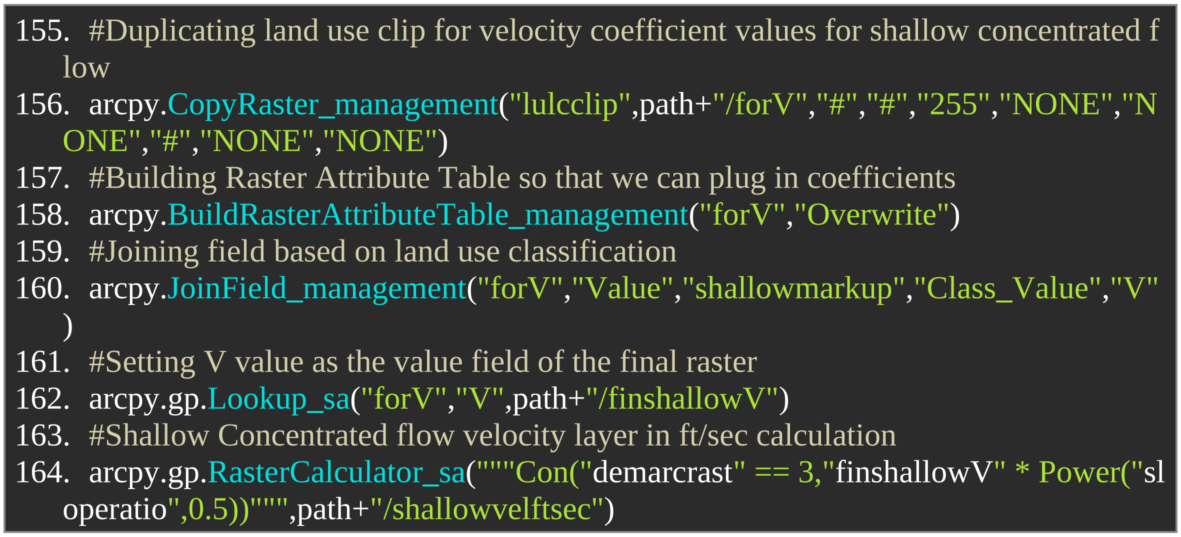
The 146th line calculates sheet flow velocity in ft/sec in the regions where **demarcconst** is 1 or 2, with the given formula using the previously obtained layers. The flowlength of zero is converted to half the cell size as minimum flow length. So, this gets better with better resolution.



```
147. if str(o) == 'true':
148.     #Reading NLCD markup table
149.     arcpy.AddMessage("Using default sheet for shalow concentrated flow")
150.     arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/LandUsebased Coeff.xlsx",path+"/shallowmarkup","Shallow")
151. else:
152.     arcpy.AddMessage("New Shallow Concentrated Flow Sheet name taken")
153.     #Reading New markup table for shallow concentrated flow
154.     arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/LandUsebased Coeff.xlsx",path+"/shallowmarkup",pgname)
```

Fig 28: Checking source of Land Use Land Cover.

The if loop checks if the Land Use/ Land Cover layer is from NLCD and whichever the case, the respective table is imported from the excel file as the markup attribute table. The creation of this table is defined more broadly in the user’s manual. This is for the shallow concentrated flow.



```
155. #Duplicating land use clip for velocity coefficient values for shallow concentrated flow
156. arcpy.CopyRaster_management("lulcclip",path+"/forV","#","#","255","NONE","NONE","#","NONE","NONE")
157. #Building Raster Attribute Table so that we can plug in coefficients
158. arcpy.BuildRasterAttributeTable_management("forV","Overwrite")
159. #Joining field based on land use classification
160. arcpy.JoinField_management("forV","Value","shallowmarkup","Class_Value","V")
161. #Setting V value as the value field of the final raster
162. arcpy.gp.Lookup_sa("forV","V",path+"/finshallowV")
163. #Shallow Concentrated flow velocity layer in ft/sec calculation
164. arcpy.gp.RasterCalculator_sa("""Con("demarcrast" == 3,"finshallowV" * Power("sloperatio",0.5))""",path+"/shallowvelftsec")
```

Fig 29: Shallow Concentrated Flow Velocity Calculation.

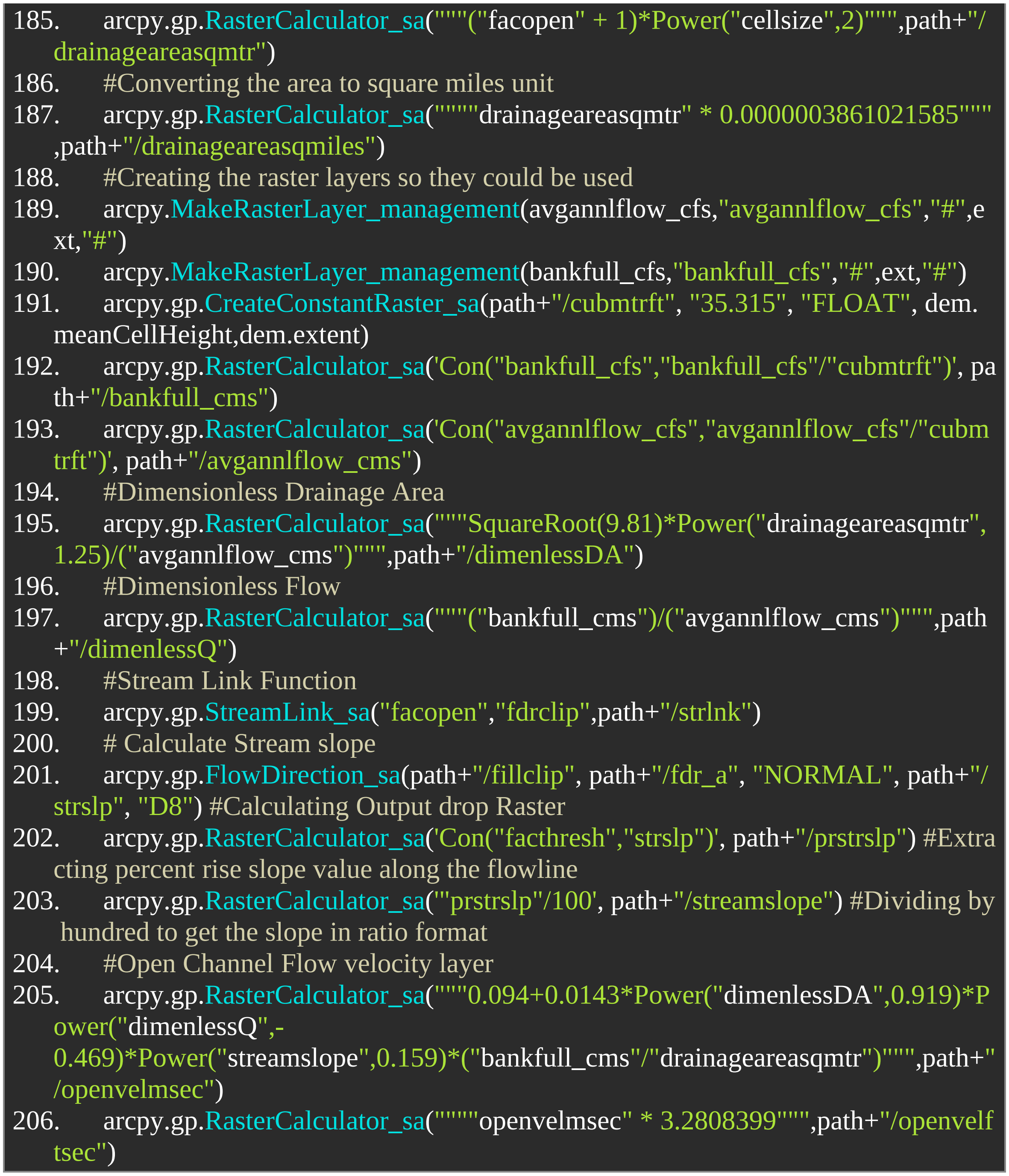
The 156th line duplicates **lulcclip** to **forV** to plug in the Velocity Coefficient values in the new raster. The 158th line builds the attribute table consisting of unique classes as it gets lost in copying.

The 160th line joins the markup table and newly created attribute table on the basis of Value field in **forV** and Class_Value field in **shallowmarkup.** The field V is joined in.

The 162nd line sets the V value as the value field of the raster layer which is saved as **finshallowV.**

The 164th line calculates shallow flow velocity in ft/sec with the `given formula in the region where **demarcconst** is 3, using the previously obtained layers. The formula is available in the Science Manual.

```
165.  #Open Channel calculation starts
166.  conv = CreateConstantRaster(3.28084,"FLOAT", dem.meanCellHeight,dem.extent) #Creating constant raster for conversion from m/sec to ft/sec
167.  conv.save(path+"/conv") #Saving the conversion factor
168.
169.  if str(haveopenvel) == 'true':
170.      #Creating inputted layer on Map document
171.      arcpy.MakeRasterLayer_management(openvelftsec,"openvelftsec","#",ext,"#")
172.  else:
173.      ##Optional Model that can be used for calculating open channel flow velocity
174.      if str(isopnlndav) == 'true':
175.          #To account for open water areas and wetlands in landuse
176.          arcpy.AddMessage("Open Water LandUse entered")
177.          arcpy.MakeRasterLayer_management(opnlnduse,"opnlnduse","#",ext,"#")
178.          arcpy.MosaicToNewRaster_management("facthrsh;opnlnduse",path,"facthresh","#","64_BIT","#","1","FIRST","FIRST")
179.      else:
180.          arcpy.AddMessage("No Open Water LandUse entered")
181.          arcpy.CopyRaster_management(in_raster="facthrsh", out_rasterdataset=path+"/facthresh", config_keyword="", background_value="", nodata_value="255", onebit_to_eightbit="NONE", colormap_to_RGB="NONE", pixel_type="64_BIT", scale_pixel_value="NONE", RGB_to_Colormap="NONE", format="Esri Grid", transform="NONE")  #Making duplicate raster to getfacthresh layer
182.      #Drainage Area per cell
183.      arcpy.gp.RasterCalculator_sa("""Con("facthresh","fac")""",path+"/facopen") #Raster Layer of flow accumulation for open channel
184.      #Adding one cell value to the flow accumulating celss and multiplying by the area of cells
```



```
arcpy.gp.RasterCalculator_sa("""("facopen" + 1)*Power("cellsize",2)""",path+"/drainageareasqmtr")
#Converting the area to square miles unit
arcpy.gp.RasterCalculator_sa(""""drainageareasqmtr" * 0.0000003861021585""",path+"/drainageareasqmiles")
#Creating the raster layers so they could be used
arcpy.MakeRasterLayer_management(avgannlflow_cfs,"avgannlflow_cfs","#",ext,"#")
arcpy.MakeRasterLayer_management(bankfull_cfs,"bankfull_cfs","#",ext,"#")
arcpy.gp.CreateConstantRaster_sa(path+"/cubmtrft", "35.315", "FLOAT", dem.meanCellHeight,dem.extent)
arcpy.gp.RasterCalculator_sa('Con("bankfull_cfs","bankfull_cfs"/"cubmtrft")', path+"/bankfull_cms")
arcpy.gp.RasterCalculator_sa('Con("avgannlflow_cfs","avgannlflow_cfs"/"cubmtrft")', path+"/avgannlflow_cms")
#Dimensionless Drainage Area
arcpy.gp.RasterCalculator_sa("""SquareRoot(9.81)*Power("drainageareasqmtr",1.25)/("avgannlflow_cms")""",path+"/dimenlessDA")
#Dimensionless Flow
arcpy.gp.RasterCalculator_sa("""("bankfull_cms")/("avgannlflow_cms")""",path+"/dimenlessQ")
#Stream Link Function
arcpy.gp.StreamLink_sa("facopen","fdrclip",path+"/strlnk")
# Calculate Stream slope
arcpy.gp.FlowDirection_sa(path+"/fillclip", path+"/fdr_a", "NORMAL", path+"/strslp", "D8") #Calculating Output drop Raster
arcpy.gp.RasterCalculator_sa('Con("facthresh","strslp")', path+"/prstrslp") #Extracting percent rise slope value along the flowline
arcpy.gp.RasterCalculator_sa('"prstrslp"/100', path+"/streamslope") #Dividing by hundred to get the slope in ratio format
#Open Channel Flow velocity layer
arcpy.gp.RasterCalculator_sa("""0.094+0.0143*Power("dimenlessDA",0.919)*Power("dimenlessQ",-0.469)*Power("streamslope",0.159)*("bankfull_cms"/"drainageareasqmtr")""",path+"/openvelmsec")
arcpy.gp.RasterCalculator_sa(""""openvelmsec" * 3.2808399""",path+"/openvelftsec")
```

Fig 30: Open Flow Velocity Calculation.

This section deals with the creation of open flow velocity in ft/sec.

The if else function checks through the variable **haveopenvel** if the user wants to input the open flow velocity directly or use the optional model provided which would require them to have layers for average annual flow and bank full flow. The process of obtaining these are explained in detail in the Science Manual.

If the user has the velocity grid, the tool makes the layer available in the GIS document in the 171st line.

If the velocity grid is not available, then the program moves on to the else part.

The if inside the else section checks through the variable **isopnlndav** if there are extra zones apart from the dileneated zone which has land use as water or wetlands.

The 177th line makes the **opnlnduse** layer available to the current GIS document. And the 17th line merges it with **facthrsh** to form the layer **facthresh** to form the raster where open flow velocity formula, will be applicable.

The corresponding else loop just copies the **facthrsh** into **facthresh,** so thatupcoming lines of code can find this layer.

The 183rd line extract flow accumulation values from wherever the **facthresh** layer is present, i.e., the open channel flow cells. This is stored as **facopen** layer.

The 185th line adds the **facopen** layer with 1 to account for the cell itself which is not counted during flow accumulation calculation and the multiplied with the square of the cell size to obtain the drainage area in square meters. This is stored as **drainageareasqmtr.**

The 187th line converts this layer's units to that of square miles and stores it as **drainageareasqmiles.** This is required for the formulas.

The 189th and 190th line makes the average annual flow and bank flow layers available in the GIS document to be used.

The 191st line stress a constant raster of conversion factor from cubic meter and cubic feet.

The 192nd line and 193rd line converts the average annual flow and bank full flow to cubic meter per second.

The 195th line uses the formula to calculate dimensionless drainage area, which needs to be plugged in the final formula.

The 197th line uses the formula to calculate dimensionless flow, which needs to be plugged in the final formula.

The 199th line uses Stream Link function to find unique streams and stores it as **strlnk**.

The 201st line uses the flow direction function to calculate output drop raster which calculates slope as a percentage along the flow direction.

The 202nd line extracts those values at the cells where open channels are present, i.e., **facthresh** zones.

The 203rd line divides those values by 100 to get the slope ratio and stores it as **streamslope.**

The 205th line calculates the open flow velocity in m/sec using the formula and stored as **openvelmsec**. It is then converted to ft/sec in the 206th line and stored as **openvelftsec.**

```
207.  #Mosaic the velocities into one layer
208.  arcpy.MosaicToNewRaster_management("openvelftsec;sheetvelftsec;shallowvelftsec",path,"vel_ftsec_all","#","64_BIT","#","1","FIRST","FIRST")
209.  #Convert velocity units to meters per minute
210.  arcpy.gp.RasterCalculator_sa(""""vel_ftsec_all" * 18.2888""",path+"/vel_mtrmin_all")
211.  #Inverse Velocity
212.  arcpy.gp.RasterCalculator_sa("""1/("vel_mtrmin_all")""",path+"/inverse_velocity")
```

```
213. #Travel time in minute
214. arcpy.gp.FlowLength_sa(fdr,path+"/time_min","DOWNSTREAM","inverse_velocity")
```

Fig 31: Time Calculation.

This section obtains the weightage layer and calculates the time.

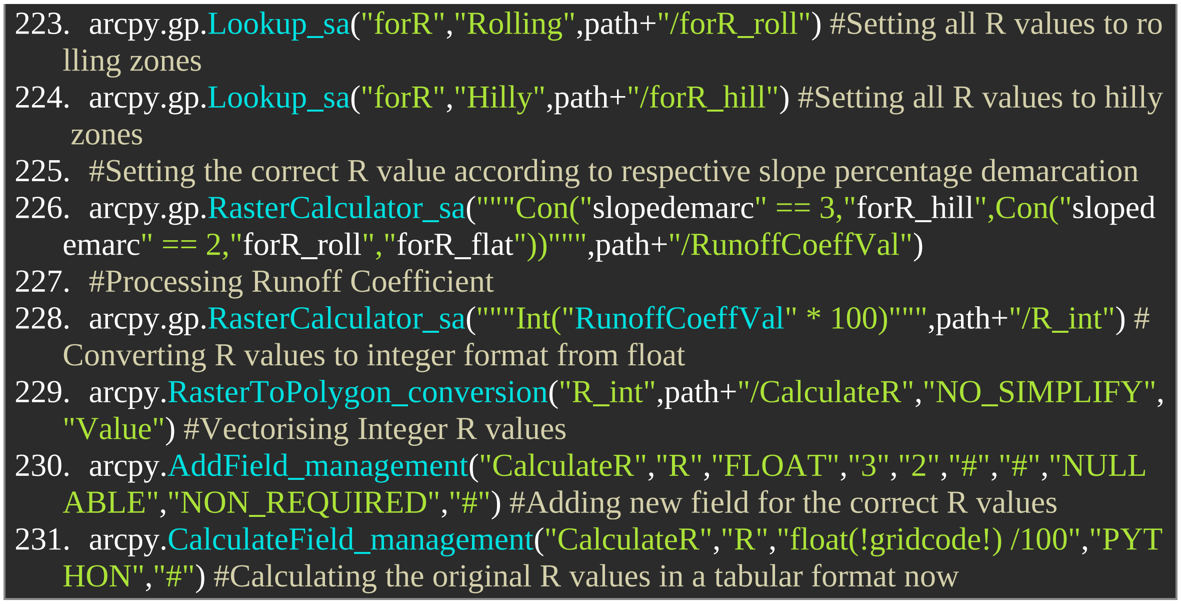
The 208th line mosaics the raster layers of velocities together with the preference of open channel velocity first followed by sheet flow and shallow concentrated flow, respectively. This is stored in the layer named **vel_ftsec_all.**

The 210th line converts the ft/sec unit to mtr/min for calculation convenience and stores it in layer **vel_mtrmin_all.**

The 212th line inverses the previous raster layer by calculating one divided by the layer and stores it as **inverse_velocity** layer.

The 214th line calculates downward flowlength using flow direction and **inverse_velocity** as weightage. This is stored as **time_min** layer. The unit is in minutes.

```
215. #Slope Percentage Demarcation - 1 = Flat; 2 = Rolling; 3 = Hilly
216. arcpy.gp.RasterCalculator_sa("""Con("slope"<=2,1,Con("slope"<=7,2,3))""",path+"/slopedemarc")
217. #Create Runoff coefficient layer
218. arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/LandUsebasedCoeff.xlsx",path+"/runcoefmarkup","RunoffCoeff")
219. arcpy.CopyRaster_management("lulcclip",path+"/forR","#","#","255","NONE","NONE","#","NONE","NONE")
220. arcpy.BuildRasterAttributeTable_management("forR","Overwrite") #Building Attribute table for easy plug in of numbers
221. arcpy.JoinField_management("forR","Value","runcoefmarkup","Class_Value","Flat;Rolling;Hilly") #Joining values with markup table
222. arcpy.gp.Lookup_sa("forR","Flat",path+"/forR_flat") #Setting all R values to flat zones
```



```
223.  arcpy.gp.Lookup_sa("forR","Rolling",path+"/forR_roll") #Setting all R values to rolling zones
224.  arcpy.gp.Lookup_sa("forR","Hilly",path+"/forR_hill") #Setting all R values to hilly zones
225.  #Setting the correct R value according to respective slope percentage demarcation
226.  arcpy.gp.RasterCalculator_sa("""Con("slopedemarc" == 3,"forR_hill",Con("slopedemarc" == 2,"forR_roll","forR_flat"))""",path+"/RunoffCoeffVal")
227.  #Processing Runoff Coefficient
228.  arcpy.gp.RasterCalculator_sa("""Int("RunoffCoeffVal" * 100)""",path+"/R_int") #Converting R values to integer format from float
229.  arcpy.RasterToPolygon_conversion("R_int",path+"/CalculateR","NO_SIMPLIFY","Value") #Vectorising Integer R values
230.  arcpy.AddField_management("CalculateR","R","FLOAT","3","2","#","#","NULLABLE","NON_REQUIRED","#") #Adding new field for the correct R values
231.  arcpy.CalculateField_management("CalculateR","R","float(!gridcode!) /100","PYTHON","#") #Calculating the original R values in a tabular format now
```

Fig 32: Runoff Coefficient Value Allocation.

This section deals with storing the runoff coefficient values for each cell.

The 216th line demarcates slopes into Flat, Rolling and Hilly from the **slope** layer and marks them as 1,2 and 3, respectively. The below 2% is considered as flat, below 7% is considered as rolling and above is considered as hilly. This is stored as **slopedemarc.**

The 218th line imports the markup table for runoff coefficients based on land use and slope divisions. It is stored as **runcoefmarkup.**

The 219th line makes a duplicate raster of **lulcclip** to plug in R values. It is stored as **forR.** The 220th line builds the attribute table consisting of unique classes as it gets lost in copying.

The 221st line joins the markup table and newly created attribute table on the basis of Value field in **forR** and Class_Value field in **runcoefmarkup.** The field Flat, Hilly and Rolling is joined in.

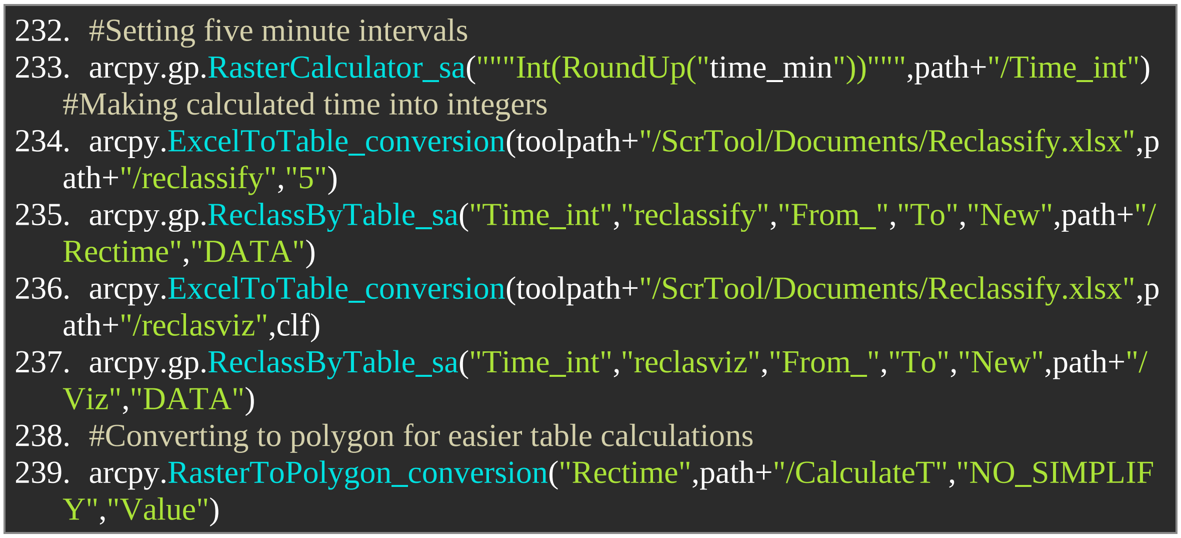
The 222nd, 223rd, 224th line sets the Flat, Rolling and Hilly column value as the value field of the raster layers which are saved as **forR_flat, forR_roll, forR_hill** respectively.

The 226th line sets the R values properly with the conditional checking the slope demarcation (**slopedemarc**). This is stored as **RunoffCoeffVal.**

The 228th line converts the R values to integer after multiplying by 100 so that it can be converted to polygon. It is stored as **R_int.**

The 229th line vectorizes the **R_int** and stores as **CalculateR.**

The 230th line adds a new column, "**R**" in the **CalculateR,** and the 231st line divides the gridcode values by 100 to obtain back the original R values. Now these could be used for upcoming calculations.



```
232.  #Setting five minute intervals
233.  arcpy.gp.RasterCalculator_sa("""Int(RoundUp("time_min"))""",path+"/Time_int") #Making calculated time into integers
234.  arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/Reclassify.xlsx",path+"/reclassify","5")
235.  arcpy.gp.ReclassByTable_sa("Time_int","reclassify","From_","To","New",path+"/Rectime","DATA")
236.  arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/Reclassify.xlsx",path+"/reclasviz",clf)
237.  arcpy.gp.ReclassByTable_sa("Time_int","reclasviz","From_","To","New",path+"/Viz","DATA")
238.  #Converting to polygon for easier table calculations
239.  arcpy.RasterToPolygon_conversion("Rectime",path+"/CalculateT","NO_SIMPLIFY","Value")
```

Fig 33: Isochrones.

The 233rd line converts the time values to integer by rounding off the fractions and stores it as **Time_int.**

The 234th line imports the classifications of time into the upper bound of five-minute intervals as separating zones of time. This is stored as **reclassify.** Then it is used through the reclassify function and resultant layer is stored as **Rectime** in line 235th.

The 236th line imports the classifications of time into the upper bound of user-specified-minute intervals as separating zones of time. This is used for visual purpose. It is stored as **reclasviz.** Then it is used through the reclassify function and resultant layer is stored as **Viz** in line 237th.

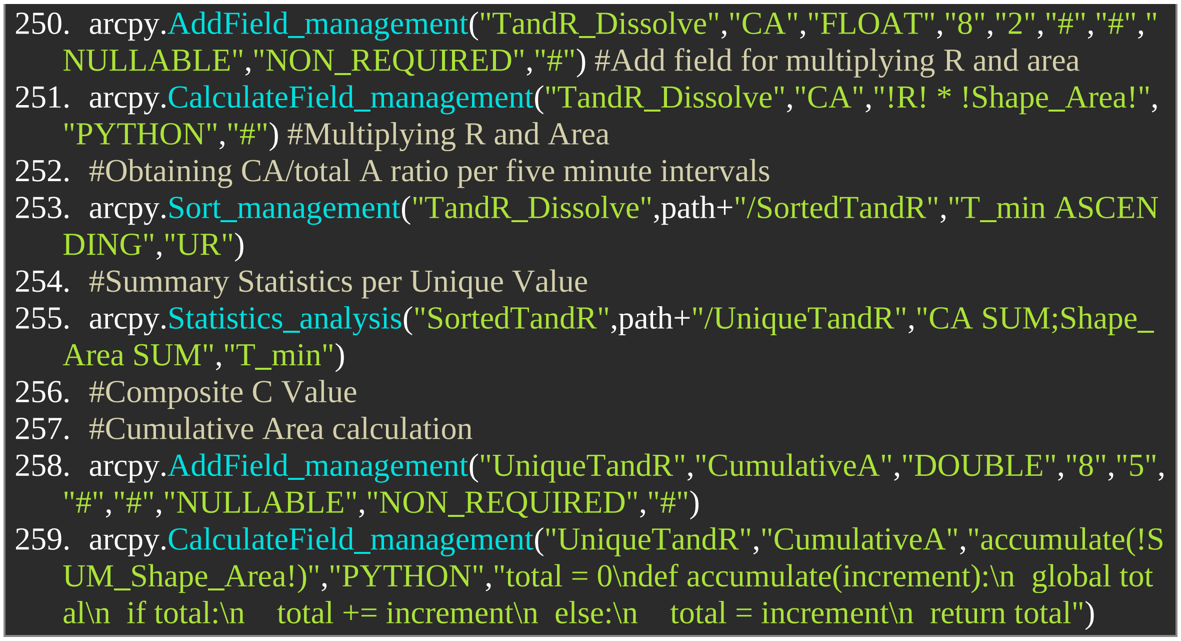
The 239th line vectorizes **Rectime** to **CalculateT** for easier calculations.

```
240.  #Storing the maximum time in minutes in a global variable
241.  time_min = Raster("time_min") #Storing rthe time raster in a variable
242.  global m #Globalising the variable
243.  m = time_min.maximum #Invoking maximum function to get the max value
```

Fig 34: Global Variable time limit.

The 243rd line here stores the maximum value of time in minutes in the global variable **m,** to be used in the future.

```
244.  #Intersecting and dissolving time and and runoff coefficients to obtain respective area.
245.  arcpy.AddField_management("CalculateT","T_min","FLOAT","5","2","#","#","NULLABLE","NON_REQUIRED","#")
246.  arcpy.CalculateField_management("CalculateT","T_min","float(!gridcode!)","PYTHON","#")
247.  arcpy.Intersect_analysis("CalculateT #;CalculateR #",path+"/TandR","ALL","#","INPUT")
248.  arcpy.Dissolve_management("TandR",path+"/TandR_Dissolve","T_min;R","#","MULTI_PART","DISSOLVE_LINES")
249.  #Multiplying C and A
```



```
250.  arcpy.AddField_management("TandR_Dissolve","CA","FLOAT","8","2","#","#","NULLABLE","NON_REQUIRED","#") #Add field for multiplying R and area
251.  arcpy.CalculateField_management("TandR_Dissolve","CA","!R! * !Shape_Area!","PYTHON","#") #Multiplying R and Area
252.  #Obtaining CA/total A ratio per five minute intervals
253.  arcpy.Sort_management("TandR_Dissolve",path+"/SortedTandR","T_min ASCENDING","UR")
254.  #Summary Statistics per Unique Value
255.  arcpy.Statistics_analysis("SortedTandR",path+"/UniqueTandR","CA SUM;Shape_Area SUM","T_min")
256.  #Composite C Value
257.  #Cumulative Area calculation
258.  arcpy.AddField_management("UniqueTandR","CumulativeA","DOUBLE","8","5","#","#","NULLABLE","NON_REQUIRED","#")
259.  arcpy.CalculateField_management("UniqueTandR","CumulativeA","accumulate(!SUM_Shape_Area!)","PYTHON","total = 0\ndef accumulate(increment):\n  global total\n  if total:\n    total += increment\n  else:\n    total = increment\n  return total")
```

Fig 35: Area Calculation.

The 245th line adds a new column, "**T_min**" in the **CalculateT,** and the 246th line converts the gridcode values into float format from integer and stores in the new column. Now these could be used for upcoming calculations.

In the 247th line the **CalculateT** and **CalculateR** are intersected and result is stored as **TandR.** In the 248th line they were dissolved to unique values for T and R, stored as **TandR_Dissolve.**

The 250th line adds a new column, "**CA**" in the **TandR_Dissolve,** and the 251st line multiplies R value with shape area in that column.

The 252nd line sorts the time values in ascending order and stores as **SortedTandR.** This is then used in Statistics Analysis function to obtain values for unique time values. The **CA** product and the **Shape_area** are obtained as sum totals in the final table. This is stored as **UniqueTandR.**

The 258th line adds a new column, "**CumulativeA**" in the **UniqueTandR,** and the 259th line it calculates cumulative area for each time value increment.

```
260. #Cumulative product per time increment
261. arcpy.AddField_management("UniqueTandR","CumulativeCA","DOUBLE","8","5","#","#","NULLABLE","NON_REQUIRED","#")
262. arcpy.CalculateField_management("UniqueTandR","CumulativeCA","accumulate(!SUM_CA!)","PYTHON","total = 0\ndef accumulate(increment):\n  global total\n  if total:\n    total += increment\n  else:\n    total = increment\n  return total")
263. #Composite C calculation
264. arcpy.AddField_management("UniqueTandR","CompositeC","DOUBLE","8","5","#","#","NULLABLE","NON_REQUIRED","#")
265. arcpy.CalculateField_management("UniqueTandR","CompositeC","!CumulativeCA! / !CumulativeA!","PYTHON","#")
266. #Cf Value
267. arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/Cf.xlsx",path+"/Cf","Sheet1")
268. #i Value
269. arcpy.ExcelToTable_conversion(toolpath+"/ScrTool/Documents/iValues.xlsx",path+"/iRequired",County)
270. #Area in Acres
271. arcpy.AddField_management("UniqueTandR","A_acres","DOUBLE","8","5","#","#","NULLABLE","NON_REQUIRED","#")
272. arcpy.CalculateField_management("UniqueTandR","A_acres","!CumulativeA! /4047","PYTHON","#")
```

Fig 36: Rational Method.

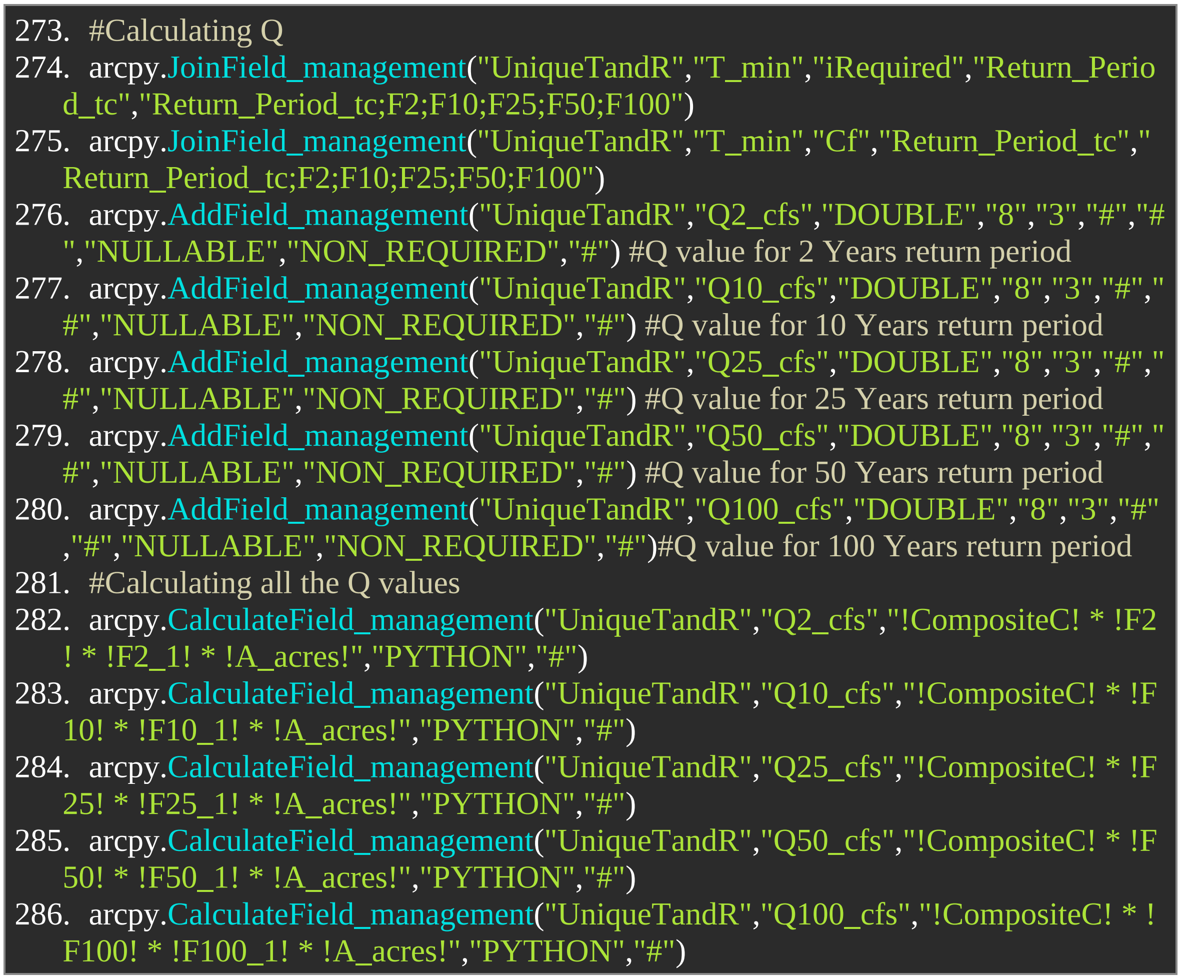
The 261st line adds a new column, “**CumulativeCA”** in the **UniqueTandR,** and the 262nd line it calculates cumulative CA value for each time value increment.

The 264th line adds a new column, “**CompositeC”** in the **UniqueTandR,** and the 265th line it calculates composite C value using the formula. The formula is mentioned broadly in the Science Manual.

The 267th line imports the Cf value from the Excel file for each return periods under the name **Cf.**

The 269th line imports the intensity values from the Excel file for each return periods in accordance to the county name provided and stores it under the name **iRequired.**

The 271st line adds a new column, “**A_acres”** in the **UniqueTandR,** and the 272nd line it converts the cumulative area values from square meters to acres.



```
273.  #Calculating Q
274.  arcpy.JoinField_management("UniqueTandR","T_min","iRequired","Return_Period_tc","Return_Period_tc;F2;F10;F25;F50;F100")
275.  arcpy.JoinField_management("UniqueTandR","T_min","Cf","Return_Period_tc","Return_Period_tc;F2;F10;F25;F50;F100")
276.  arcpy.AddField_management("UniqueTandR","Q2_cfs","DOUBLE","8","3","#","#","NULLABLE","NON_REQUIRED","#") #Q value for 2 Years return period
277.  arcpy.AddField_management("UniqueTandR","Q10_cfs","DOUBLE","8","3","#","#","NULLABLE","NON_REQUIRED","#") #Q value for 10 Years return period
278.  arcpy.AddField_management("UniqueTandR","Q25_cfs","DOUBLE","8","3","#","#","NULLABLE","NON_REQUIRED","#") #Q value for 25 Years return period
279.  arcpy.AddField_management("UniqueTandR","Q50_cfs","DOUBLE","8","3","#","#","NULLABLE","NON_REQUIRED","#") #Q value for 50 Years return period
280.  arcpy.AddField_management("UniqueTandR","Q100_cfs","DOUBLE","8","3","#","#","NULLABLE","NON_REQUIRED","#")#Q value for 100 Years return period
281.  #Calculating all the Q values
282.  arcpy.CalculateField_management("UniqueTandR","Q2_cfs","!CompositeC! * !F2! * !F2_1! * !A_acres!","PYTHON","#")
283.  arcpy.CalculateField_management("UniqueTandR","Q10_cfs","!CompositeC! * !F10! * !F10_1! * !A_acres!","PYTHON","#")
284.  arcpy.CalculateField_management("UniqueTandR","Q25_cfs","!CompositeC! * !F25! * !F25_1! * !A_acres!","PYTHON","#")
285.  arcpy.CalculateField_management("UniqueTandR","Q50_cfs","!CompositeC! * !F50! * !F50_1! * !A_acres!","PYTHON","#")
286.  arcpy.CalculateField_management("UniqueTandR","Q100_cfs","!CompositeC! * !F100! * !F100_1! * !A_acres!","PYTHON","#")
```

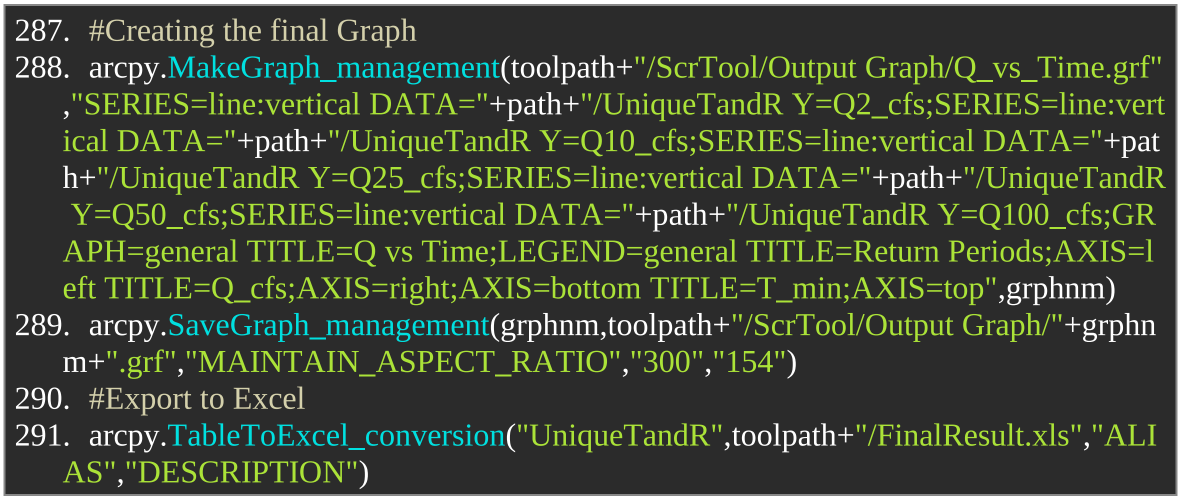
Fig 37: Flow value for each return period calculation.

This section calculates the Q values for different return periods for each time value.

The 274th joins the intensity values into the **UniqueTandR** table based on time value increments.

The 275th joins the Cf values into the **UniqueTandR** table based on time value increments.

The 276th, 277th, 278th, 279th and 280th line creates new fields to calculate the flows for different return periods of 2, 10, 25, 50 and 100 years. The formula for it is given in the science manual. The 282th, 283rd, 284th, 285th and 286th lines calculate the same using the columns.



```
287.  #Creating the final Graph
288.  arcpy.MakeGraph_management(toolpath+"/ScrTool/Output Graph/Q_vs_Time.grf","SERIES=line:vertical DATA="+path+"/UniqueTandR Y=Q2_cfs;SERIES=line:vertical DATA="+path+"/UniqueTandR Y=Q10_cfs;SERIES=line:vertical DATA="+path+"/UniqueTandR Y=Q25_cfs;SERIES=line:vertical DATA="+path+"/UniqueTandR Y=Q50_cfs;SERIES=line:vertical DATA="+path+"/UniqueTandR Y=Q100_cfs;GRAPH=general TITLE=Q vs Time;LEGEND=general TITLE=Return Periods;AXIS=left TITLE=Q_cfs;AXIS=right;AXIS=bottom TITLE=T_min;AXIS=top",grphnm)
289.  arcpy.SaveGraph_management(grphnm,toolpath+"/ScrTool/Output Graph/"+grphnm+".grf","MAINTAIN_ASPECT_RATIO","300","154")
290.  #Export to Excel
291.  arcpy.TableToExcel_conversion("UniqueTandR",toolpath+"/FinalResult.xls","ALIAS","DESCRIPTION")
```

Fig 38: Output graph.

The 288th line makes the final graph using the details on **UniqueTandR** of flow per time in different return periods using the graph template of a pre-saved graph. It is then saved in the Output Graph folder with the user given name in the line 289th.

The **UniqueTandR** table is then exported as an Excel File for later use if deemed required in the main folder in line 291st.

The second method exchanges the runoff coefficient calculation with curve number. The curve number is taken from TR-55 and an weighted average on the basis of area is used.

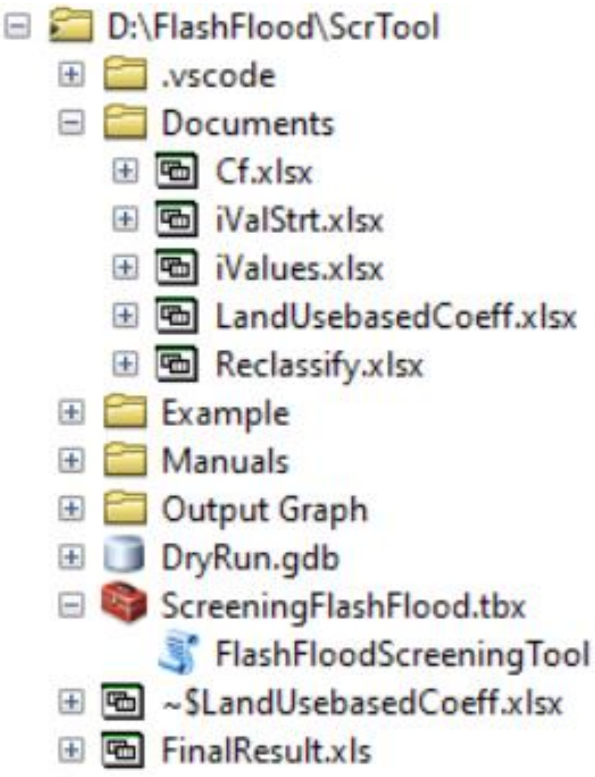
APPENDIX B

USER’S MANUAL

The link where it can be downloaded:

https://drive.google.com/drive/folders/14NYTWVGtY9UWPNTUQBCW60KEibh0rbkC?usp=sharing

The tool comes in a Folder called **ScrTool.** It can be kept under any folder. In this example, the parent folder is D:/FlashFlood. The different components of the tool can be seen in the next image: -

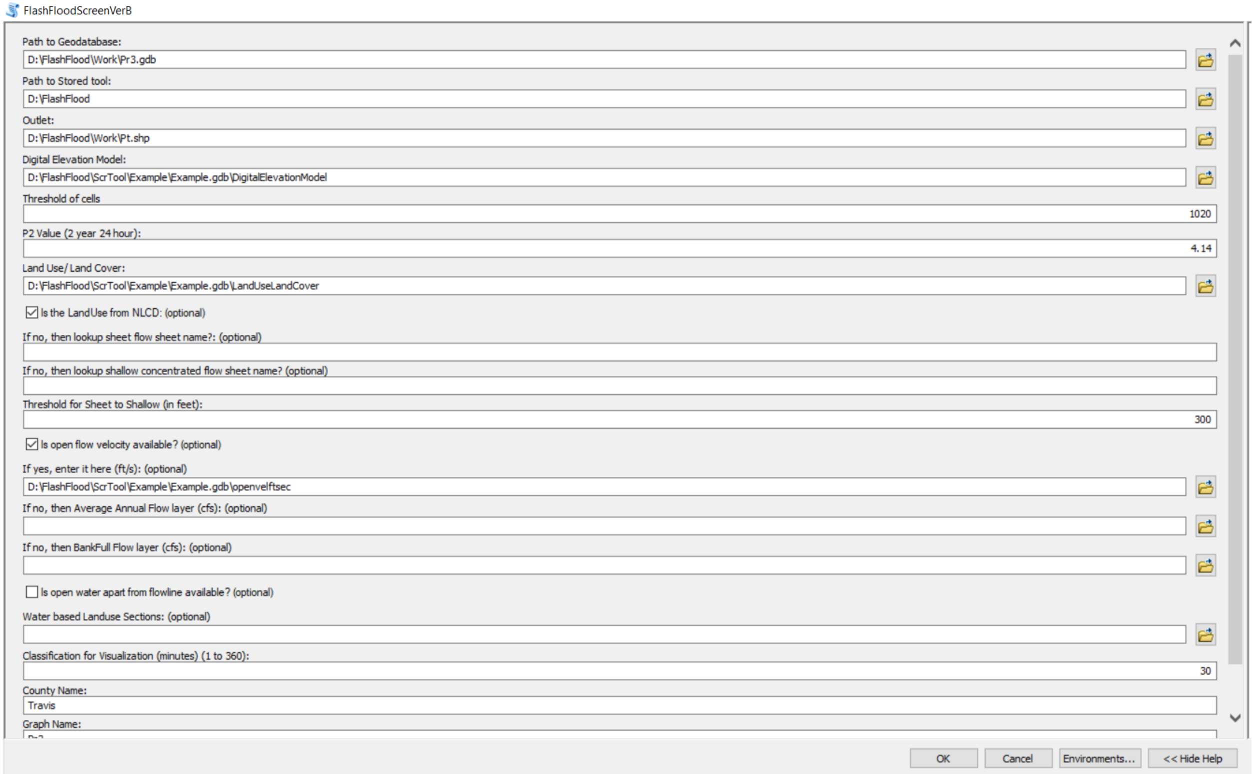


```
D:\FlashFlood\ScrTool
    .vscode
    Documents
        Cf.xlsx
        iValStrt.xlsx
        iValues.xlsx
        LandUsebasedCoeff.xlsx
        Reclassify.xlsx
    Example
    Manuals
    Output Graph
    DryRun.gdb
    ScreeningFlashFlood.tbx
        FlashFloodScreeningTool
    ~$LandUsebasedCoeff.xlsx
    FinalResult.xls
```

Fig a: Entire Tool with its components

In the Screening Flash Flood toolbox, the script is call Flash Flood Screening Tool. This is the executable. Double-click it to start running the tool.

In the documents folder, the various excel files have the coefficients and values stored in. It is a database that is called by the tool and can also be modified as per the user's needs. Each of them is explained below the tool parameters description.



FlashFloodScreenVerB

Path to Geodatabase:
D:\FlashFlood\Work\Pr3.gdb

Path to Stored tool:
D:\FlashFlood

Outlet:
D:\FlashFlood\Work\Pt.shp

Digital Elevation Model:
D:\FlashFlood\ScrTool\Example\Example.gdb\DigitalElevationModel

Threshold of cells
1020

P2 Value (2 year 24 hour):
4.14

Land Use/ Land Cover:
D:\FlashFlood\ScrTool\Example\Example.gdb\LandUseLandCover

☑ Is the LandUse from NLCD: (optional)

If no, then lookup sheet flow sheet name?: (optional)

If no, then lookup shallow concentrated flow sheet name? (optional)

Threshold for Sheet to Shallow (in feet):
300

☑ Is open flow velocity available? (optional)

If yes, enter it here (ft/s): (optional)
D:\FlashFlood\ScrTool\Example\Example.gdb\openvelftsec

If no, then Average Annual Flow layer (cfs): (optional)

If no, then BankFull Flow layer (cfs): (optional)

☐ Is open water apart from flowline available? (optional)

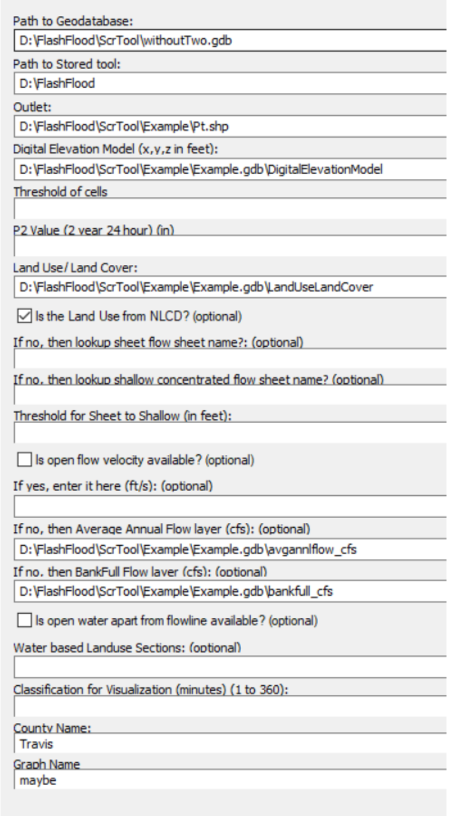
Water based Landuse Sections: (optional)

Classification for Visualization (minutes) (1 to 360):
30

County Name:
Travis

Graph Name:

Fig b: Tool's dialog



Path to Geodatabase:
D:\FlashFlood\ScrTool\withoutTwo.gdb

Path to Stored tool:
D:\FlashFlood

Outlet:
D:\FlashFlood\ScrTool\Example\Pt.shp

Digital Elevation Model (x,y,z in feet):
D:\FlashFlood\ScrTool\Example\Example.gdb\DigitalElevationModel

Threshold of cells

P2 Value (2 year 24 hour) (in)

Land Use/ Land Cover:
D:\FlashFlood\ScrTool\Example\Example.gdb\LandUseLandCover

☑ Is the Land Use from NLCD? (optional)

If no, then lookup sheet flow sheet name?: (optional)

If no, then lookup shallow concentrated flow sheet name? (optional)

Threshold for Sheet to Shallow (in feet):

□ Is open flow velocity available? (optional)

If yes, enter it here (ft/s): (optional)

If no, then Average Annual Flow layer (cfs): (optional)
D:\FlashFlood\ScrTool\Example\Example.gdb\avgannlflow_cfs

If no, then BankFull Flow layer (cfs): (optional)
D:\FlashFlood\ScrTool\Example\Example.gdb\bankfull_cfs

□ Is open water apart from flowline available? (optional)

Water based Landuse Sections: (optional)

Classification for Visualization (minutes) (1 to 360):

County Name:
Travis

Graph Name
maybe

Fig c: Expanded view of the parameters.

Inputs need to be as follows: -

Path to Geodatabase: The address of the geodatabase where all the files created will be stored in.

Path to Stored Tool: Address of the parent folder of the ScrTool Folder.

Outlet: The point outlet is the location the study is interested in.

Digital Elevation Model (x, y, z in feet): Input of Digital Elevation Model. The x, y, z unit are all taken in feet.

Threshold: The lowest accumulating cell number for a cell to be considered as an open channel flow.

P2 Value (2 year 24 hour) (mm): The p2 variable stores the precipitation depth value of 2-year 24 hour in the units of inches.

Land Use/ Land Cover: The Land Use/ Land Cover layer.

Is the Land Use from NLCD: Checked only if land use is from NLCD.

If no, then lookup sheet flow sheet name: New sheet name for land use markup table for sheet flow if it is not from NLCD.

If no, then lookup shallow concentrated flow sheet name: New sheet name for land use markup table for shallow concentrated flow if it is not from NLCD.

Threshold from sheet to shallow (in feet): The upstream flowlength which differentiates the sheet flow from the shallow concentrated flow.

Is open flow velocity available? Checked if open flow velocity raster layer is available. Unchecked if the optional model needs to be used.

If yes, enter it here (ft/s): Open channel flow velocity raster layer in ft/sec.

If no, then average annual flow layer (cfs): The average annual flow layer in cubic feet per second.

If no, then bankfull flow layer (cfs): The bankfull flow layer in cubic feet per second.

Water Based Landuse Sections: Optional parameter for raster layer of the areas containing open water land use or wetlands.

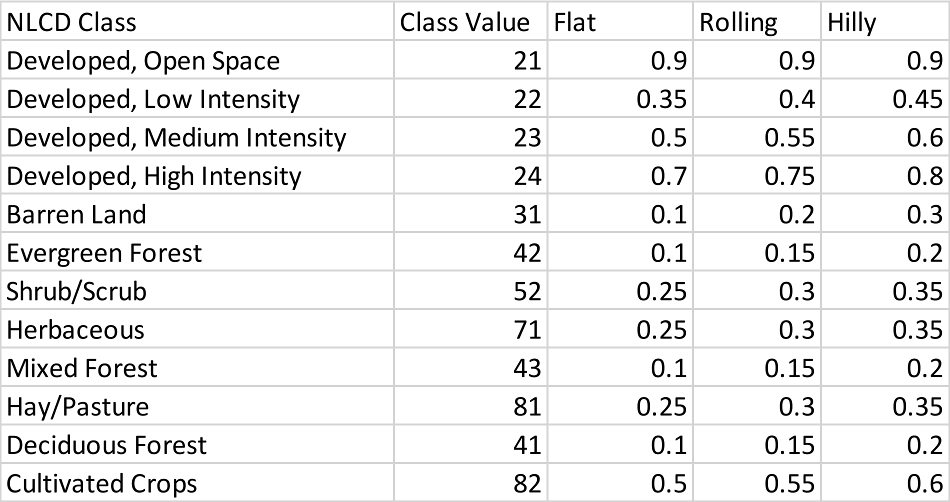
Classification for Visualization (minutes) (1 to 360): Intervals for visual differentiation of time, to be produced in a layer so that the demarcations of area according to specified time is clearly visible to the user. The classification tables go up to 360 minutes. Any extension wanted can be added by the user in the excel file.

County Name: The county name where the study area is located in.

Graph Name: The name desired for the final graph to be saved in.

Documents Folder

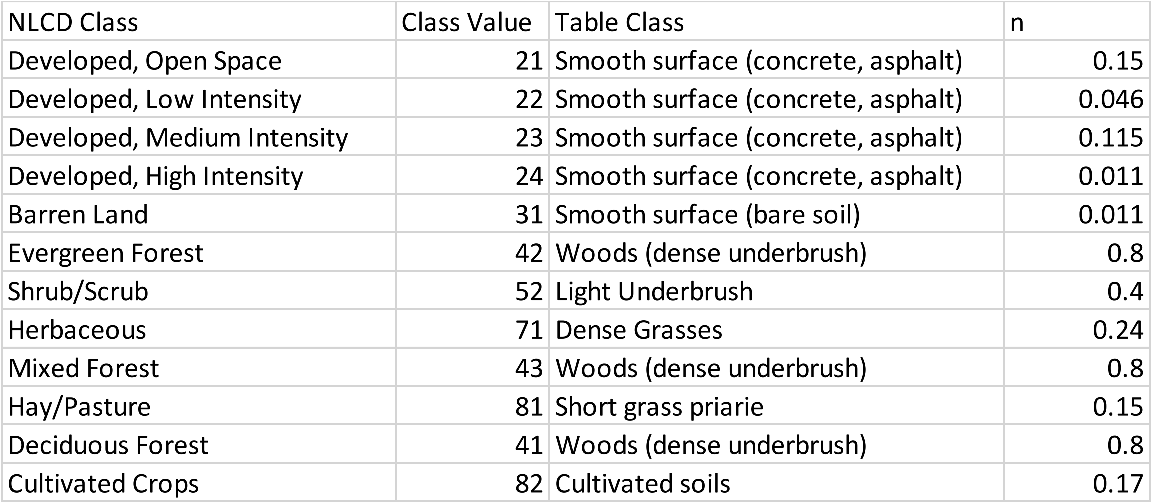
Excel Filename: LandUsebasedCoeff.xlsx, Sheet name: RunoffCoeff Sheet



NLCD Class	Class Value	Flat	Rolling	Hilly
Developed, Open Space	21	0.9	0.9	0.9
Developed, Low Intensity	22	0.35	0.4	0.45
Developed, Medium Intensity	23	0.5	0.55	0.6
Developed, High Intensity	24	0.7	0.75	0.8
Barren Land	31	0.1	0.2	0.3
Evergreen Forest	42	0.1	0.15	0.2
Shrub/Scrub	52	0.25	0.3	0.35
Herbaceous	71	0.25	0.3	0.35
Mixed Forest	43	0.1	0.15	0.2
Hay/Pasture	81	0.25	0.3	0.35
Deciduous Forest	41	0.1	0.15	0.2
Cultivated Crops	82	0.5	0.55	0.6

This is the format of runoff coefficient markup sheet. Not all values are shown for brevity. The first column shows NLCD Class name. The second column is the class values. **If the land use is not from NLCD, these values need to be edited.** The coefficients are segregated in flat, rolling, and hilly in the next three columns. **If the user has local variables available, which are more accurate for that study area, then these values need to be edited.**

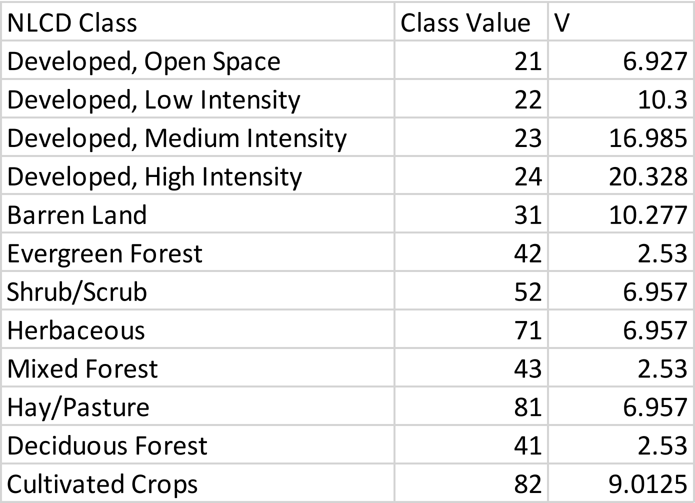
<u>Excel Filename: LandUsebasedCoeff.xlsx, Sheet name: Sheet Sheet</u>



NLCD Class	Class Value	Table Class	n
Developed, Open Space	21	Smooth surface (concrete, asphalt)	0.15
Developed, Low Intensity	22	Smooth surface (concrete, asphalt)	0.046
Developed, Medium Intensity	23	Smooth surface (concrete, asphalt)	0.115
Developed, High Intensity	24	Smooth surface (concrete, asphalt)	0.011
Barren Land	31	Smooth surface (bare soil)	0.011
Evergreen Forest	42	Woods (dense underbrush)	0.8
Shrub/Scrub	52	Light Underbrush	0.4
Herbaceous	71	Dense Grasses	0.24
Mixed Forest	43	Woods (dense underbrush)	0.8
Hay/Pasture	81	Short grass priarie	0.15
Deciduous Forest	41	Woods (dense underbrush)	0.8
Cultivated Crops	82	Cultivated soils	0.17

This is the format of sheet flow markup sheet for Manning's n values. Not all values are shown for brevity. The first column shows NLCD Class name. The second column is the class values. The third column shows the class name of the source table and the fourth column shows corresponding n values. The source of these values is explained in the Science Manual. **If land use is not from NLCD, entirely a new sheet must be made. The new sheet's name needs to be given at the tool. The tool joins on the basis of class values. If more accurate n values are available, then the fourth column must be changed.**

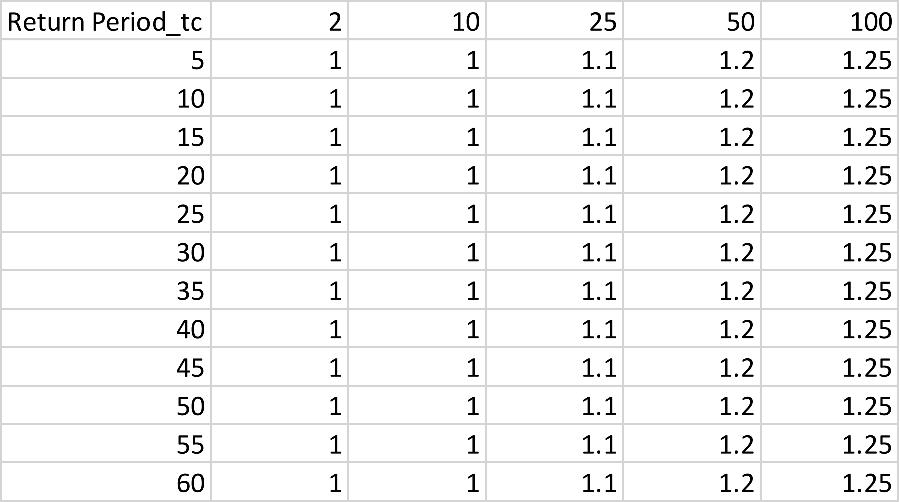
Excel Filename: LandUsebasedCoeff.xlsx, Sheet name: Shallow Sheet



NLCD Class	Class Value	V
Developed, Open Space	21	6.927
Developed, Low Intensity	22	10.3
Developed, Medium Intensity	23	16.985
Developed, High Intensity	24	20.328
Barren Land	31	10.277
Evergreen Forest	42	2.53
Shrub/Scrub	52	6.957
Herbaceous	71	6.957
Mixed Forest	43	2.53
Hay/Pasture	81	6.957
Deciduous Forest	41	2.53
Cultivated Crops	82	9.0125

This is the format of shallow concentrated flow markup sheet for Velocity Coefficient values. Not all values are shown for brevity. The first column shows NLCD Class name. The second column is the class values. The third column shows the Velocity Coefficient values corresponding to the particular Land use. The source of these values is explained in the Science Manual. **If land use is not from NLCD, entirely a new sheet must be made. The new sheet's name needs to be given at the tool. The tool joins on the basis of class values. If more accurate n values are available, then the fourth column must be changed.**

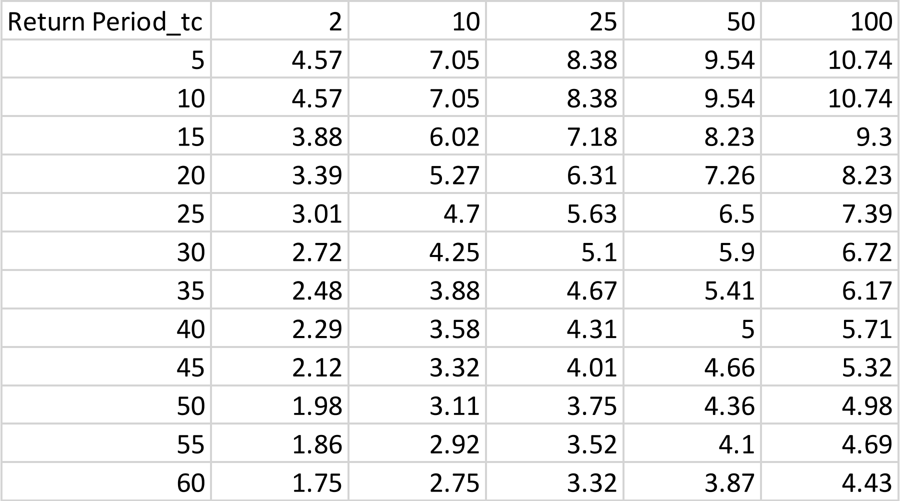
Excel Filename: Cf.xlsx, Sheet name: Sheet 1



Return Period_tc	2	10	25	50	100
5	1	1	1.1	1.2	1.25
10	1	1	1.1	1.2	1.25
15	1	1	1.1	1.2	1.25
20	1	1	1.1	1.2	1.25
25	1	1	1.1	1.2	1.25
30	1	1	1.1	1.2	1.25
35	1	1	1.1	1.2	1.25
40	1	1	1.1	1.2	1.25
45	1	1	1.1	1.2	1.25
50	1	1	1.1	1.2	1.25
55	1	1	1.1	1.2	1.25
60	1	1	1.1	1.2	1.25

This is the format of frequency correction factor markup sheet for different return periods. Not all values are shown for brevity. The first column gives time value increments. The first row gives return periods in years. The rest are all corresponding Cf values. **These can be changed if new values are wished to be used.**

<u>Excel Filename: iValues.xlsx, Sheet name: "County"</u>

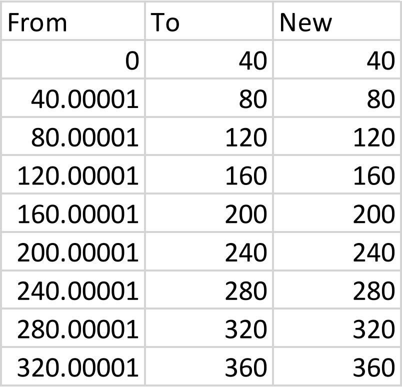


Return Period_tc	2	10	25	50	100
5	4.57	7.05	8.38	9.54	10.74
10	4.57	7.05	8.38	9.54	10.74
15	3.88	6.02	7.18	8.23	9.3
20	3.39	5.27	6.31	7.26	8.23
25	3.01	4.7	5.63	6.5	7.39
30	2.72	4.25	5.1	5.9	6.72
35	2.48	3.88	4.67	5.41	6.17
40	2.29	3.58	4.31	5	5.71
45	2.12	3.32	4.01	4.66	5.32
50	1.98	3.11	3.75	4.36	4.98
55	1.86	2.92	3.52	4.1	4.69
60	1.75	2.75	3.32	3.87	4.43

This is the format of intensity values for different return periods. Not all values are shown for brevity. The first column gives time value increments. The first row gives return periods in years. The rest are all corresponding intensity values for particular counties. Each county has a sheet name to it. **These can be changed if new values are wished to be used. For counties outside Texas, new sheets need to be added.**

This database itself is generated from **iValStrt.xlsx** using **IValueGenerate.py** code. This is not incorporated in the original code as it uses python 3. This excel file contains e, b, and d values for all counties in Texas. **This code can thereby be separately run to generate iValues.xlsx after changing the e, b, and d values if needed**.

Excel Filename: Reclassify.xlsx, Sheet name: “clf”

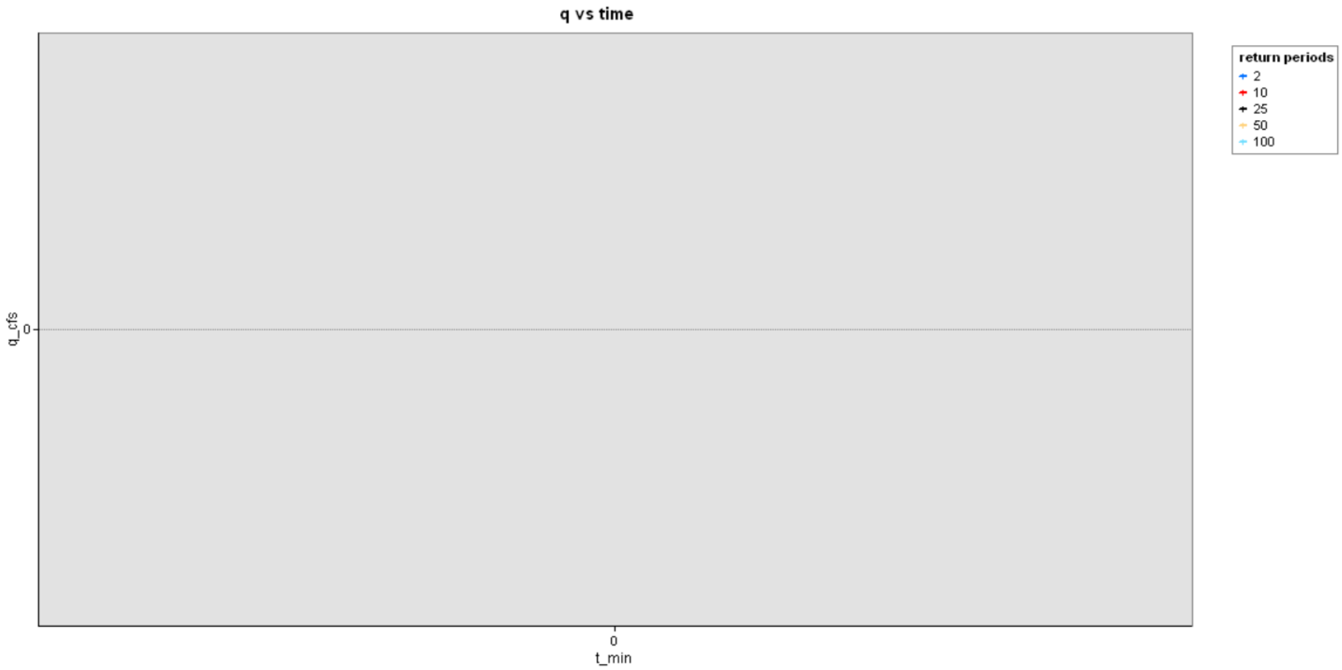


From	To	New
0	40	40
40.00001	80	80
80.00001	120	120
120.00001	160	160
160.00001	200	200
200.00001	240	240
240.00001	280	280
280.00001	320	320
320.00001	360	360

This is the reclassification table to set all the time values within specified intervals to the upper bound limit. This need not be changed at all. It contains all classifications for intervals from 1 to 360 minutes.

Output Graph Folder

Q_vs_Time.grf

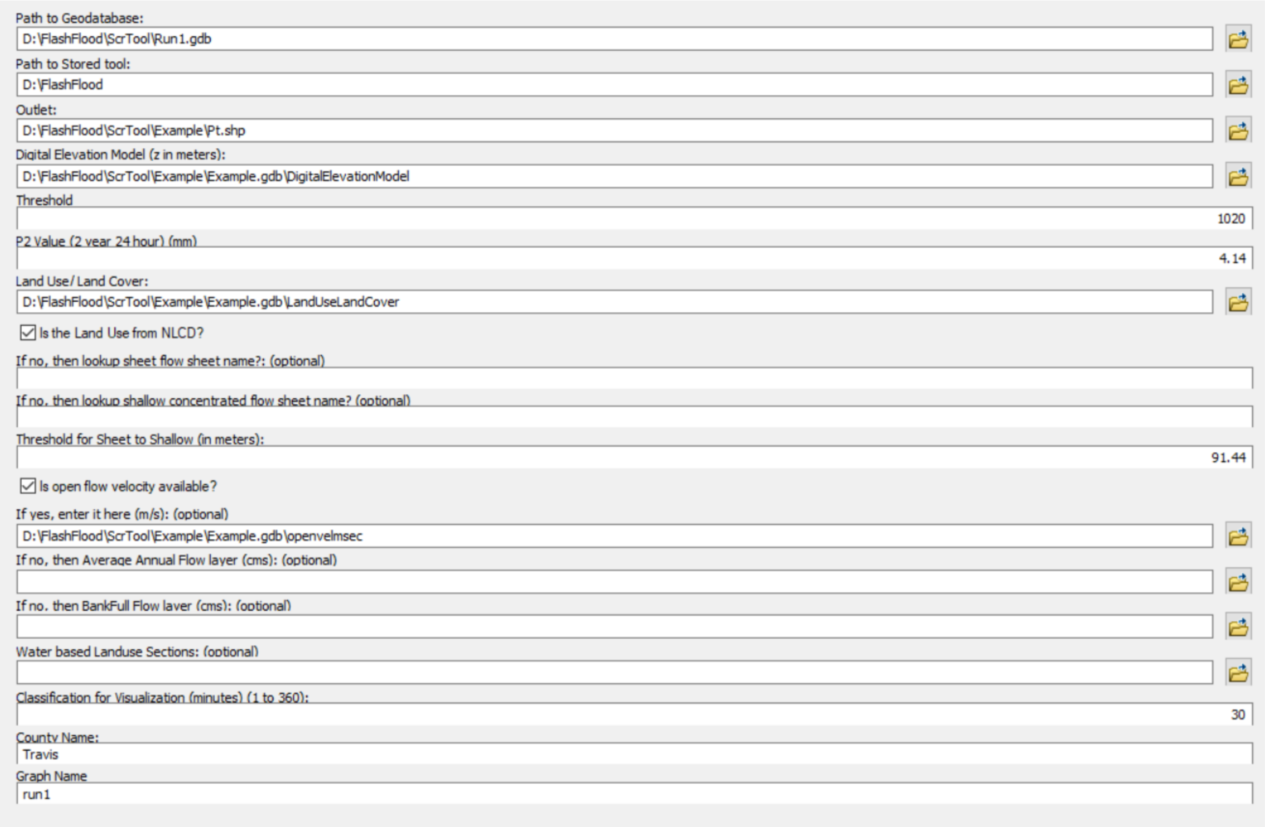


This is the graph template that is used. It can be changed if wished so.

Example Folder

Here, all the inputs for an example run are available. The figure below shows the required input:

-



Path to Geodatabase:
D:\FlashFlood\ScrTool\Run1.gdb

Path to Stored tool:
D:\FlashFlood

Outlet:
D:\FlashFlood\ScrTool\Example\Pt.shp

Digital Elevation Model (z in meters):
D:\FlashFlood\ScrTool\Example\Example.gdb\DigitalElevationModel

Threshold
1020

P2 Value (2 year 24 hour) (mm)
4.14

Land Use/ Land Cover:
D:\FlashFlood\ScrTool\Example\Example.gdb\LandUseLandCover

☑ Is the Land Use from NLCD?

If no, then lookup sheet flow sheet name?: (optional)

If no, then lookup shallow concentrated flow sheet name? (optional)

Threshold for Sheet to Shallow (in meters):
91.44

☑ Is open flow velocity available?

If yes, enter it here (m/s): (optional)
D:\FlashFlood\ScrTool\Example\Example.gdb\openvelmsec

If no, then Average Annual Flow layer (cms): (optional)

If no, then BankFull Flow layer (cms): (optional)

Water based Landuse Sections: (optional)

Classification for Visualization (minutes) (1 to 360):
30

County Name:
Travis

Graph Name
run1

As for the version B, an extra field is added to input the CN layer. For the example, CN is decided based on hydrologic soil groups and land use consulting TR-55. However, there are a lot of tools to calculate curve number per cell, viz. CatchmentSim, etc. As an example, the CN grid is attached.

APPENDIX C

Video on how to operate with no open channel velocity data: -

UserGuide1.mp4

Video on how to operate with open channel velocity data: -

UserGuide2.mp4