HMSD Group-7 Assignment-1

Study Region: Coimbatore

—

Thakkallapally Rao - 2020101037
Gangam Reddy - 2020101021
Prateek Kumar Patel - 2019111022
Yash Agarwal - 2020114005
Konda Jayant Reddy - 2019102010
Yadavalli Lakshmi - 2020101103
Adwait Raste - 2019111027

Introduction

The Coimbatore region, situated in the southern part of India, has witnessed rapid urbanization and population growth in recent years, leading to increased pressure on its hydrological systems. Such changes in land use, coupled with the ongoing impacts of climate change, have necessitated a comprehensive study to assess the hydrological characteristics and potential flood scenarios in this area.

This report outlines the methodology and findings of a rigorous geospatial analysis and hydrological modeling project aimed at understanding the complex hydrological dynamics of the Coimbatore region.

In the following sections of this report, we will delve into the specifics of the methodology employed, the process of data acquisition and processing, the hydrological modeling approaches used, and the assessment of potential flood scenarios.

Data Downloaded

DEM Data

Name of the Data Set : C1_DEM_16b_2005-2014_V3R1_76E11N_C43E

Data prepared by : NRSC

Name of the Satellite : Cartosat-1

Sensor : PAN(2.5m) Stereo Data

Tile Name : C43E, C43F, C43K, C43L

File Format : GeoTIFF

Bits per Pixel : 16bit

Spatial Resolution : 1 arc sec

Spatial Resolution Unit : m

Shapefile Data

Source : https://github.com/datameet/Municipal_Spatial_Data/tree/master/Coimbatore

Coimbatore Municipal Data contains the ward map of Coimbatore reorganized.

Rainfall Data

Source : https://chrsdata.eng.uci.edu/

Dataset : Persiann-CCS

TimeStamp : October 2021

Resolution : 0.04degree x 0.04degree

Data Processing

1. Clipping the DEM Data

This code opens a list of TIF files, merges them into a single mosaic, and saves the resulting mosaic as "mosaic.tif." It utilizes the rasterio library to handle geospatial data and mosaic the input TIF files.

It then employs the mask function to clip the DEM data using a specified shapefile called 'wards.geometry'. The crop=True parameter ensures that only the portions of the DEM that intersect with the shapefile are retained.

The code prepares metadata for the clipped dataset, including the driver type, dimensions (height and width), and transformation information.

Finally, it creates a new GeoTIFF file named 'clipped.tif' and writes the clipped DEM data into it while preserving the metadata.

2. Preprocessing the DEM Data

An area of internal drainage with cells surrounded by higher elevations is called a depression in a digital elevation model (DEM).

It is necessary to remove these depressions from the DEM, usually by increasing the cell value of the depression to match the lowest overflow point.

Artificial pits or sinks, which are cells without any downstream cells surrounding them, can be produced when a DEM is created. To avoid isolating portions of the watershed, these pits must be removed.

We search the entire DEM dataset for pits, update the depth of each pit to that of its smallest neighbor, and then fill each pit.

3. Calculating Drainage Directions

The code proceeds to calculate the drainage directions for each cell in the DEM. It iterates through each cell (excluding the border cells) in a nested loop.

For each cell (i, j), it calculates the flow direction based on elevation differences between the current cell and its eight neighboring cells (N, NE, E, SE, S, SW, W, NW).

The D8 algorithm assigns a unique value to each of these directions, resulting in a combination of directions that represent the overall flow direction for the cell.

The code checks if the elevation in the current cell is higher than that in each of its neighbors. If so, it adds the corresponding value to the d8 array.

After processing all eight neighbors, the d8 value represents the combined flow direction for the cell.

4. Extracting a Sub-Region

The code defines the parameters of the sub-region to be extracted:

x_center and y_center: The coordinates (x, y) of the center of the sub-region. You can replace these coordinates with your specific values.

radius: The radius of the sub-region in terms of the number of cells.

Based on the center coordinates and radius, the code calculates the window (a rectangular area) to read from the full DEM data.

It determines the starting column (col_off) and row (row_off) offsets and the width and height of the window.

Using the calculated window, the code reads the sub-region from the full DEM data and stores it in the 'sub_dem' variable.

It then plots the sub-region using Matplotlib, with a color map ('jet') to visualize elevation differences.

The plot includes a title indicating that it shows a sub-region from the DEM.

5. Creating Flow Accumulation Map

The code proceeds to calculate the drainage directions for each cell in the sub-region DEM. It iterates through each cell (excluding the border cells) in a nested loop.

For each cell (i, j), it calculates the flow direction based on elevation differences between the current cell and its eight neighboring cells (N, NE, E, SE, S, SW, W, NW).

The D8 algorithm assigns a unique value to each of these directions, resulting in a combination of directions that represent the overall flow direction for the cell.

The code checks if the elevation in the current cell is higher than that in each of its neighbors. If so, it adds the corresponding value to the d8 array.

After processing all eight neighbors, the d8 value represents the combined flow direction for the cell.

This code is essential for hydrological analysis, as it determines how water flows across a sub-region of a terrain based on elevation differences. The resulting flow accumulation map can be used to identify areas where water accumulates and potentially forms streams or rivers. It's a valuable component of watershed delineation and flood modeling processes.

6. Creating Flow Inundation Map

The code defines two flood parameters:

flood_level: The elevation level above which flooding occurs. This value is set to 330 (you can adjust it as needed for your analysis).

flood_depth: The depth of the floodwaters when the elevation exceeds the flood level. It is set to 10 units (you can adjust it according to your units of measurement).

The code creates a flood inundation map called 'flood_map' using NumPy.

It uses the np.where function to set values in the 'flood_map' array based on a condition:

If the elevation value in the 'full_dem' array is greater than the 'flood_level', the corresponding value in the 'flood_map' is set to the 'flood_depth'.

Otherwise, if the elevation is below the 'flood_level', the 'flood_map' value remains 0, indicating no flooding.

The resulting 'flood_map' represents the areas of the sub-region that would be inundated or flooded when the elevation exceeds the specified flood level. This map can be used for flood risk assessment, emergency planning, and flood modeling in the selected sub-region.

7. Calculating Rainfall Intensity

This MATLAB code is designed to read and visualize GeoTIFF rainfall data within the spatial boundaries defined by a shapefile. It handles data preprocessing, such as replacing NoData values and calculating spatial parameters.

After loading the shapefile, the code creates a plot that displays the rainfall data within the intersection of the GeoTIFF extent and the shapefile's boundaries. This visualization allows for a clear representation of rainfall patterns within a specific geographic region.

8. Calculating Peak Discharge for Sub-Region

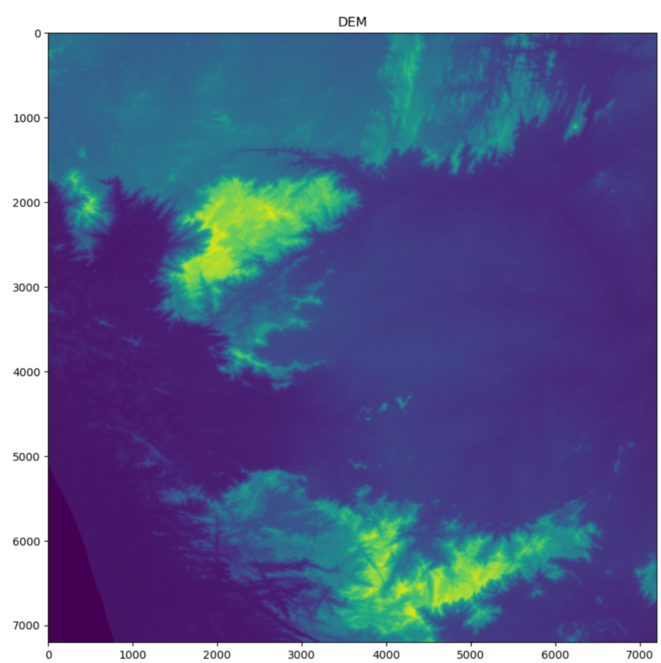
The code resulted in an incorrect output

9. Calculating Runoff Coefficient for the Sub-Region

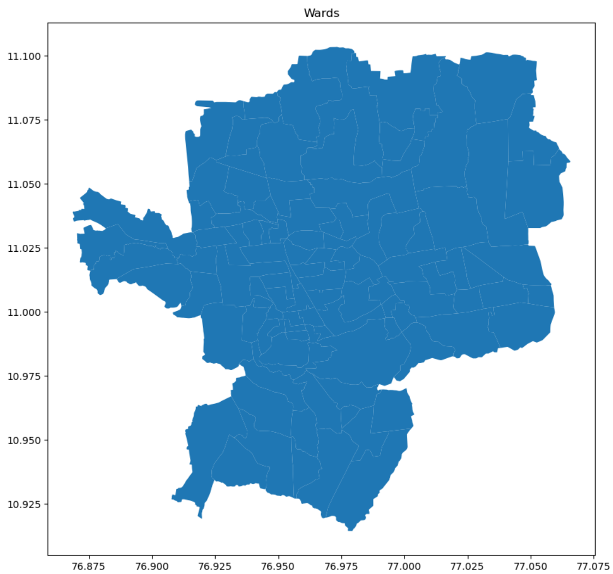
The code resulted in an incorrect output

Results

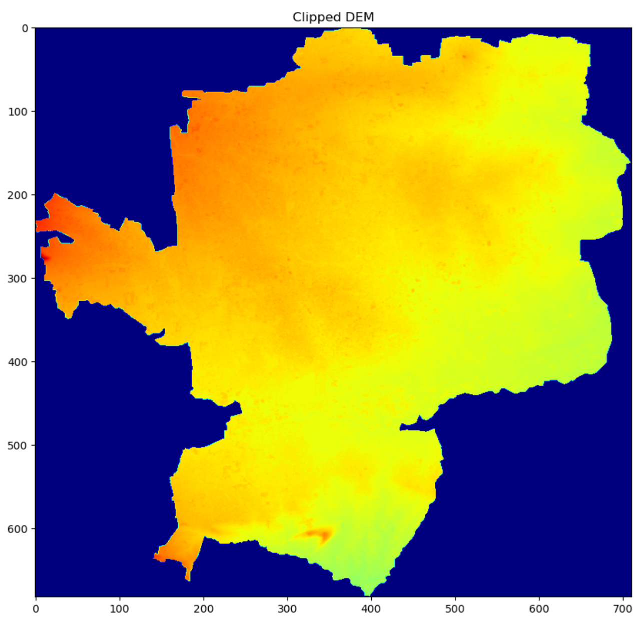
Reading DEM



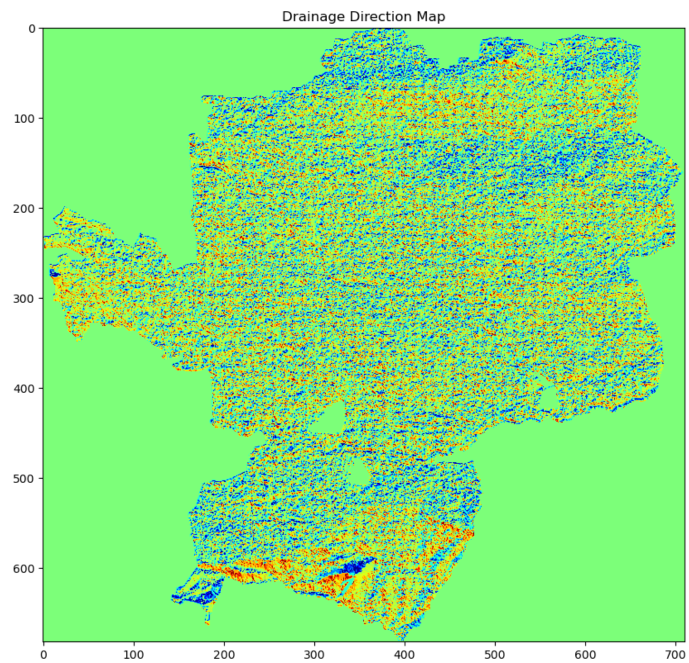
Reading Shapefile



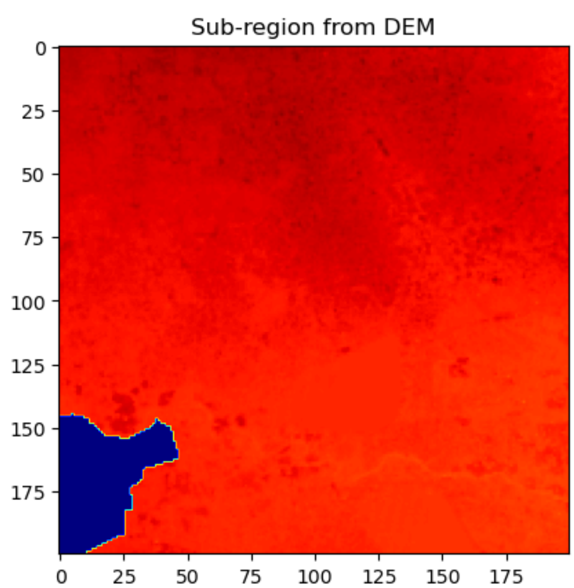
Clipped DEM



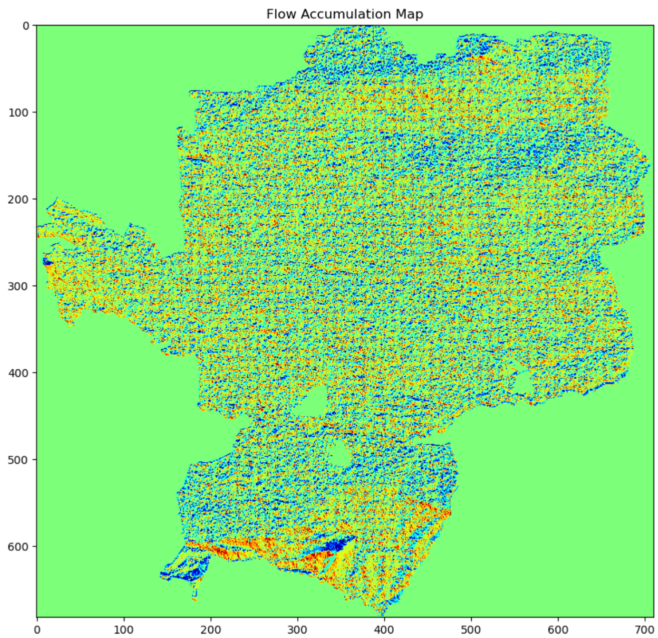
Drainage Direction Map



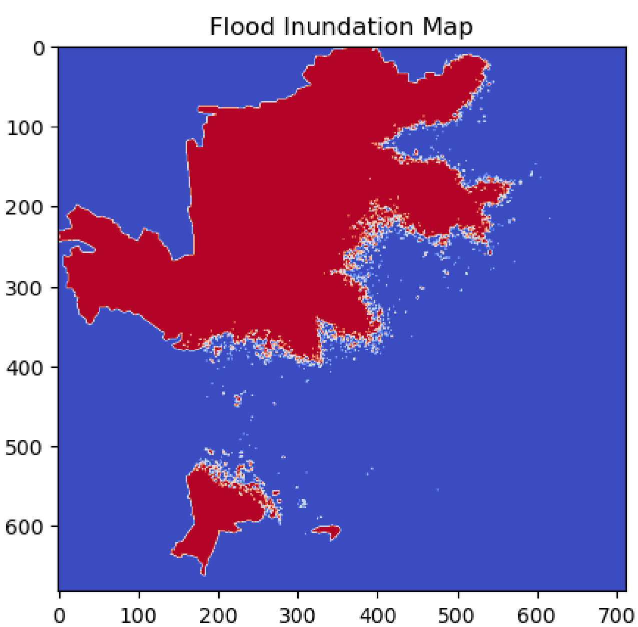
Sub-Region from DEM



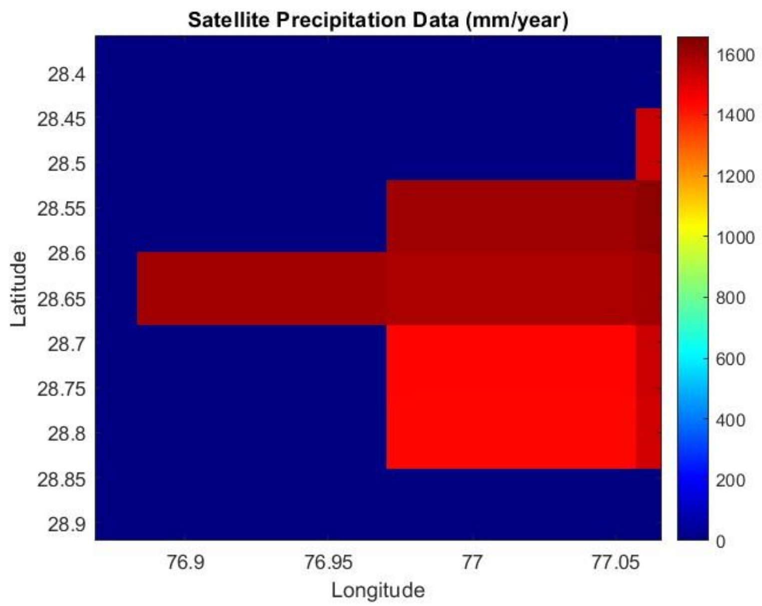
Flow Accumulation Map



Flow Inundation Map



Rainfall Intensity Map



Peak Discharge Map

Output not Correct

Runoff Coefficient Map

Output not Correct

Individual Contributions

Thakkallapally Rao : Clipping the DEM Data

Gangam Reddy : Preprocessing the DEM Data, helped in clipping.

Prateek Kumar Patel : Calculating Drainage Directions

Yash Agarwal : Extracting a Sub-Region

Konda Jayant Reddy : Creating Flow Accumulation Map

Yadavalli Lakshmi : Made Report and PPT and helped in debugging

Adwait Raste : Creating flood inundation Map, made Report and PPT