Flood Monitoring and Analysis using Google Earth Engine (Python)

Students:

- **Ayan Mandal** – Roll No: **2410030019**
- **Siddhartha Kaushik** – Roll No: **25SCS1003004626**
- **Aman Sharma** – Roll No: **2410030051**

Branch/Sem: CSE (AIML) / 4th Semester
College: IILM University, Greater Noida
Guide: Mr. Shobit Agarwal
Academic Year: 2025–26

1. Abstract

Floods are among the most frequent natural disasters, causing severe damage to life, property, agriculture, and infrastructure. Traditional flood monitoring methods are slow and limited by ground access. This project presents a cloud-based approach for flood monitoring and analysis using Google Earth Engine (GEE) with Python. Sentinel-1 SAR satellite imagery was used to generate pre-flood and post-flood composites for Assam, India. Change detection and thresholding techniques were applied to extract flooded areas. The results were visualized on interactive maps and the flooded area was estimated in square kilometers. The proposed approach enables fast, scalable, and reliable flood assessment to support disaster management and planning.

---

2. Introduction

Flooding is a recurring problem in many parts of India, particularly in Assam, due to heavy monsoon rainfall and river overflow. Remote sensing provides timely and large-area coverage for monitoring floods. Google Earth Engine (GEE) is a cloud-based geospatial platform that hosts petabytes of satellite data and enables large-scale processing without heavy local computation. In this project, Sentinel-1 SAR data is used because radar imagery works even in cloudy and rainy conditions, which are common during floods. The integration of GEE with Python in Google Colab provides an efficient workflow for flood detection and analysis.

---

3. Domain

Remote Sensing and Geospatial Analysis
(Also applicable to Environmental Monitoring and Disaster Management)

---

4. Objectives

- To monitor flood-affected areas using satellite imagery
- To generate pre-flood and post-flood composites
- To detect flood extent using change detection
- To visualize flooded regions using Google Earth Engine (Python)
- To estimate flooded area in square kilometers
- To analyze flood impact using Land Use Land Cover (LULC)

---

5. Tools and Technologies

- **Google Earth Engine (GEE)** – Cloud-based geospatial analysis
- **Python** – Programming language
- **Google Colab** – Execution environment
- **Geemap** – Interactive mapping library
- **Google Cloud Project** – Non-commercial GEE access

---

6. Dataset Description

6.1 Sentinel-1 SAR

Sentinel-1 provides radar imagery unaffected by cloud cover and illumination, making it suitable for flood mapping.

Dataset: COPERNICUS/S1_GRD
Type: Raster satellite imagery (numerical values)

6.2 Land Use Land Cover (LULC)

Provides land classification such as agriculture, forest, urban, and water.
Dataset: ESA WorldCover 2020

---

7. Study Area

The study area selected for this project is **Assam, India**, which is highly prone to seasonal floods. Administrative boundaries were used to clip satellite images to the region of interest.

---

8. Methodology

1. Create and configure a Google Cloud project and enable Earth Engine (non-commercial)
2. Authenticate Google Earth Engine in Google Colab using Python API
3. Select Assam as the study region
4. Collect Sentinel-1 SAR images for pre-flood (15 May–15 June 2022) and post-flood (20 June–20 July 2022) periods
5. Generate median composites to reduce noise
6. Perform change detection (After – Before)
7. Apply thresholding to derive flood mask
8. Visualize results on an interactive map
9. Overlay flood mask with LULC data
10. Calculate flooded area in square kilometers

---

9. Implementation

The implementation uses the Google Earth Engine Python API and Geemap for visualization. Sentinel-1 images are filtered by region, date, and instrument mode. Median composites are generated for pre- and post-flood periods. Flood extent is derived using backscatter change detection and thresholding. The total flooded area is calculated using pixel area aggregation.

10. LINKS

Colab Notebook Link

https://colab.research.google.com/drive/1OJH-RBHWnjGxKqiabWOl5oRcGL-_8RN8

Python Notebook - Untitled0.ipynb

Python code :

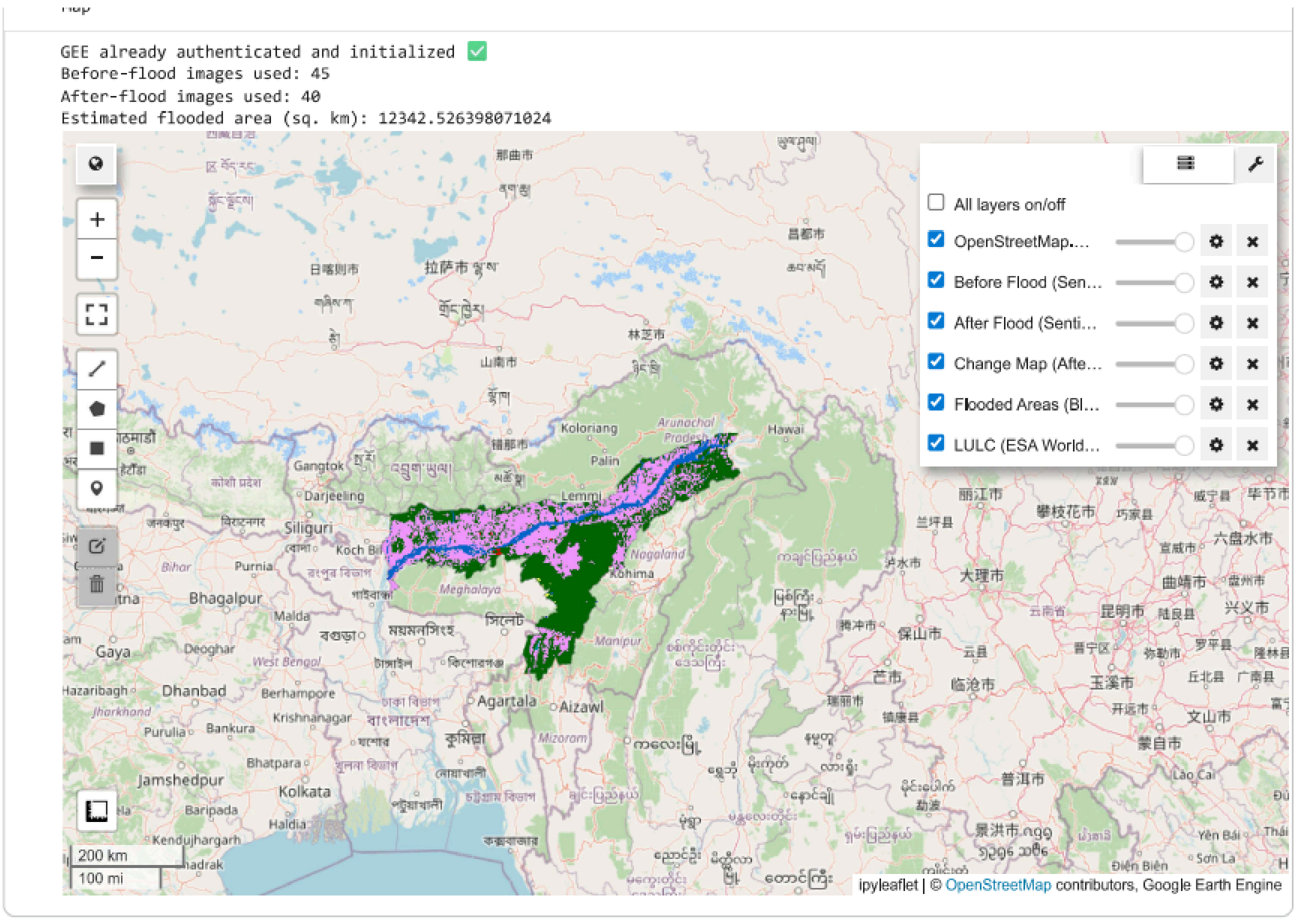
```
# Flood Monitoring and Analysis using Google Earth Engine (Python)
# Author(s): Ayan Mandal, Siddhartha Kaushik, Aman Sharma


# Install once per session (Colab)
!pip install -q earthengine-api geemap


import ee
import geemap


# Authenticate & Initialize GEE (Colab-safe)
try:
    ee.Initialize(project='college-gee-project')  # replace with your project ID if needed
    print("GEE already authenticated and initialized ")
except:
    ee.Authenticate(auth_mode='notebook')
    ee.Initialize(project='college-gee-project')
    print("GEE authenticated and initialized ")

# 1) Study area: Assam
region = ee.FeatureCollection("FAO/GAUL/2015/level1") \
    .filter(ee.Filter.eq('ADM1_NAME', 'Assam'))

# 2) Date ranges
before_start, before_end = '2022-05-15', '2022-06-15'
after_start, after_end   = '2022-06-20', '2022-07-20'

# 3) Sentinel-1 VV collections
before_ic = ee.ImageCollection('COPERNICUS/S1_GRD') \
    .filterBounds(region) \
    .filterDate(before_start, before_end) \
    .filter(ee.Filter.eq('instrumentMode', 'IW')) \
    .select('VV')

after_ic = ee.ImageCollection('COPERNICUS/S1_GRD') \
    .filterBounds(region) \
    .filterDate(after_start, after_end) \
    .filter(ee.Filter.eq('instrumentMode', 'IW')) \
    .select('VV')


# 4) Image counts

print("Before-flood images used:", before_ic.size().getInfo())

print("After-flood images used:", after_ic.size().getInfo())


# 5) Composites

before_flood = before_ic.median().clip(region)

after_flood  = after_ic.median().clip(region)


# 6) Change detection

change = after_flood.subtract(before_flood)


# 7) Flood mask (threshold)

THRESHOLD = -2

flood_mask = change.lt(THRESHOLD)


# 8) Flooded area (sq. km)

pixel_area = flood_mask.multiply(ee.Image.pixelArea())

area_stats = pixel_area.reduceRegion(

    reducer=ee.Reducer.sum(),

    geometry=region.geometry(),

    scale=30,

    maxPixels=1e13

)

flood_area_sqkm = ee.Number(area_stats.get('VV')).divide(1e6)
```

```
print("Estimated flooded area (sq. km):", flood_area_sqkm.getInfo())


# 9) LULC overlay

lulc = ee.Image('ESA/WorldCover/v100/2020').clip(region)


# 10) Visualization

Map = geemap.Map()

Map.centerObject(region, 7)

Map.addLayer(before_flood, {'min': -25, 'max': 0}, 'Before Flood')

Map.addLayer(after_flood,  {'min': -25, 'max': 0}, 'After Flood')

Map.addLayer(change, {'min': -5, 'max': 5}, 'Change Map')

Map.addLayer(flood_mask.updateMask(flood_mask), {'palette': ['0000FF']}, 'Flooded Areas')

Map.addLayer(lulc, {}, 'LULC')

Map
```

11. Results and Discussion

The results show clear flood-affected regions highlighted in blue over Assam. A total of **45 pre-flood images** and **40 post-flood images** were used to generate the composites. The estimated flooded area was **12,342.53 sq. km**, indicating significant inundation during the selected flood period. The LULC overlay revealed that agricultural lands and low-lying regions were the most affected. The results demonstrate the effectiveness of SAR-based flood mapping and cloud-based geospatial analysis.

Output :



```
GEE already authenticated and initialized ✅
Before-flood images used: 45
After-flood images used: 40
Estimated flooded area (sq. km): 12342.526398071024
```

12. Advantages

- **Operates under cloudy and rainy conditions**
- **Scalable cloud-based processing**
- **Rapid analysis for large regions**
- **Minimal local hardware requirements**
- **Useful for disaster management**

---

13. Limitations

- **Requires internet connectivity**
- **Threshold selection affects accuracy**
- **SAR noise can introduce uncertainty**
- **Limited ground truth validation**

---

14. Precautions

- **Select appropriate pre- and post-flood dates**
- **Use consistent polarization**
- **Apply noise reduction techniques**
- **Validate with field data when possible**

---

15. Applications

- **Flood risk assessment**
- **Disaster response planning**

- **Agricultural damage analysis**
- **Urban planning and resilience**
- **Climate change impact studies**

---

16. Conclusion

This project demonstrates the potential of Google Earth Engine and Sentinel-1 SAR data for rapid flood monitoring and analysis. The cloud-based workflow enables efficient processing and visualization of large geospatial datasets, making it suitable for disaster management applications. The methodology can be extended to other regions and disaster types.

---

17. Future Scope

- **Real-time flood monitoring dashboards**
- **Integration with IoT sensors**
- **Machine learning-based flood prediction**
- **Multi-year flood trend analysis**
- **Automated alert systems**

18. References

- **Google Earth Engine Documentation**
- **Sentinel-1 User Guide (ESA)**
- **ESA WorldCover Dataset**
- **Research papers on SAR-based flood mapping**