A Cloud-Based Flood Warning System For Forecasting Impacts to Transportation Infrastructure Systems

Jon Goodall
Associate Professor, Civil and Environmental Engineering
Associate Director, Link Lab

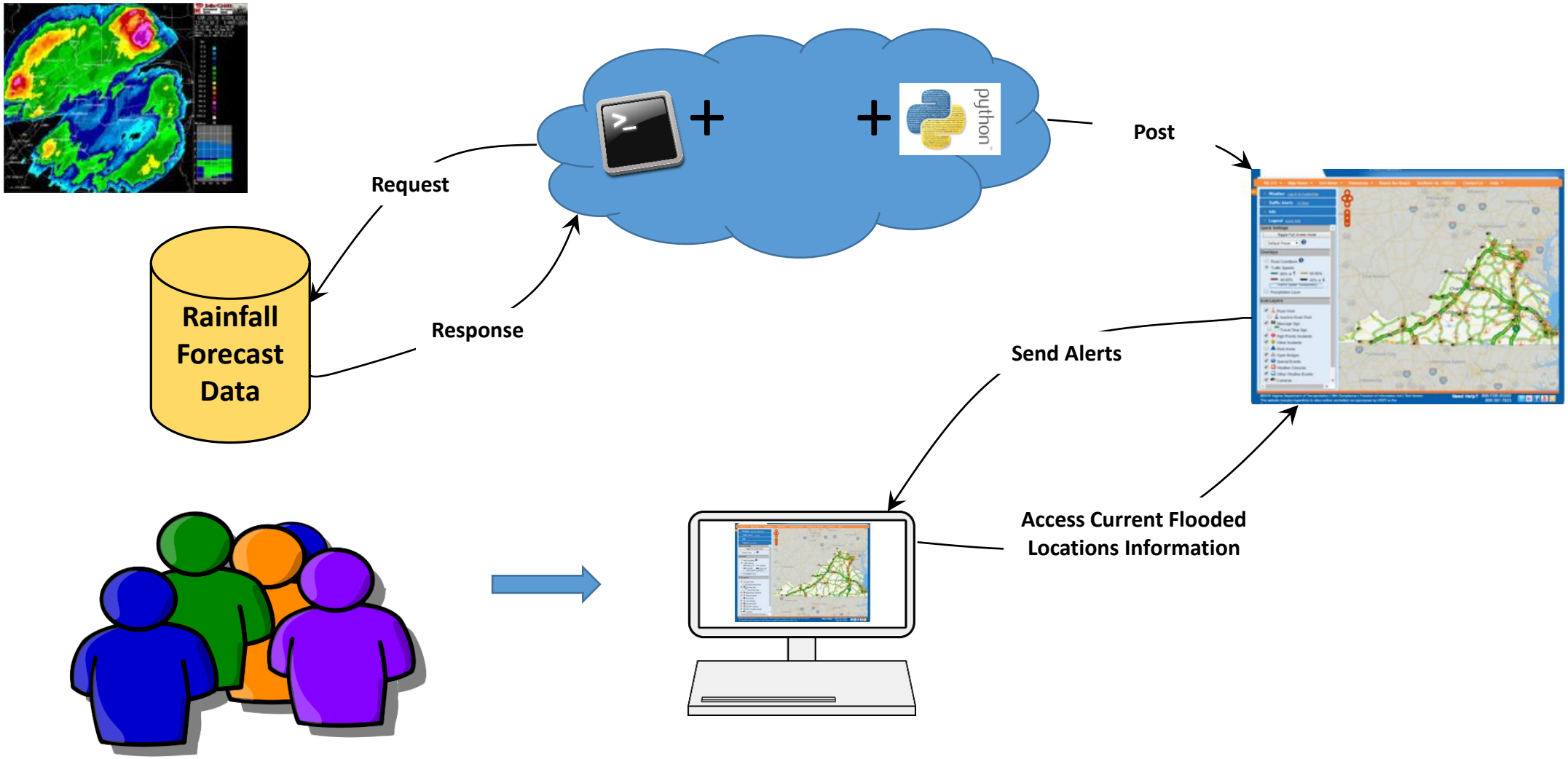
April 09, 2018

Research Motivation

- Predicting **street scale flooding**, especially for low relief terrains, is a challenge.
- For these regions, we need **high resolution datasets** and **2-D hydrodynamic models**.
- Flood predictions are needed **in near real-time** in a form that **aids decision makers**.

Chloe Riffle, 7, watches as she is surrounded by water on Sunday, Sept. 4, 2016 in the Ocean View section of Norfolk, Virginia.
https://weather.com/news/climate/news/coastal-flooding-global-warming-influence

Need for an Automated, Accurate Flood Prediction System



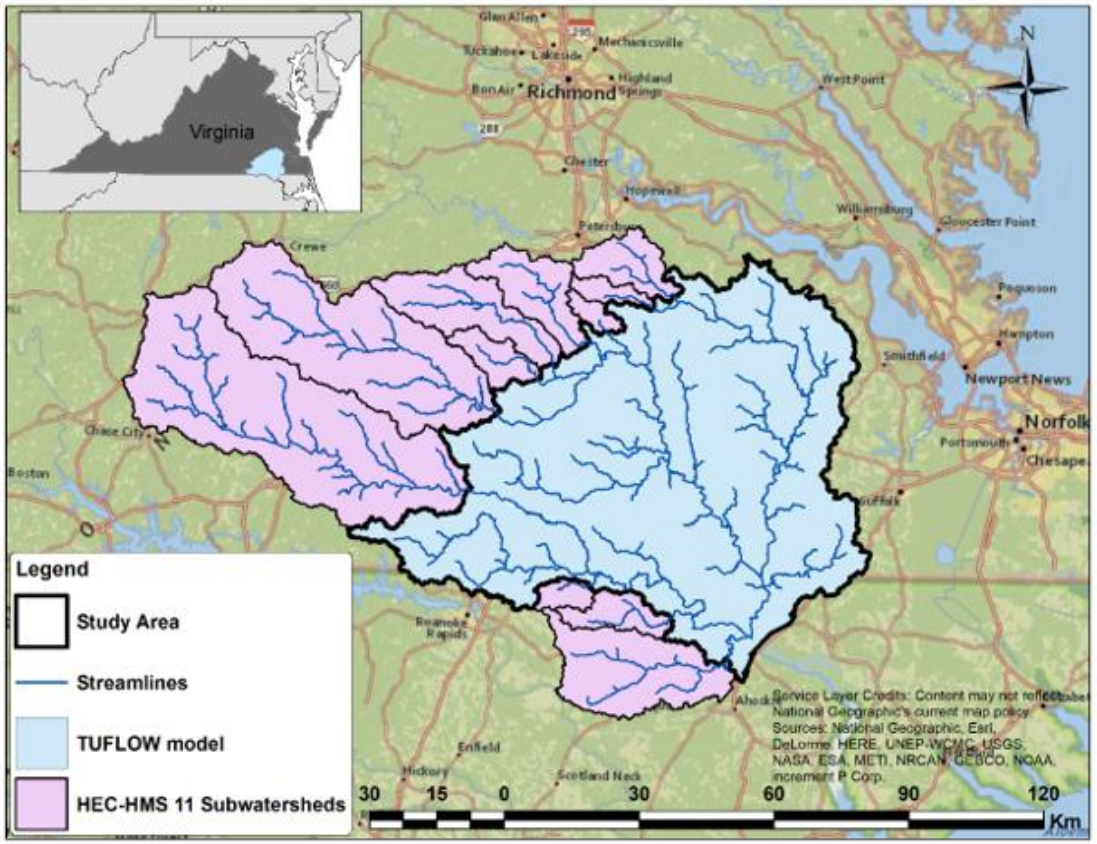
Objectives

- **Objective 1:** Build a **flood warning system** based on modern cyberinfrastructure using an **automated workflow** for obtaining the **real-time rainfall forecast data**.

- **Objective 2:** **Execute** the model using **cloud computing resources** to identify the flooded roadways and bridges location in **short-time duration** for warning and emergency management purposes.

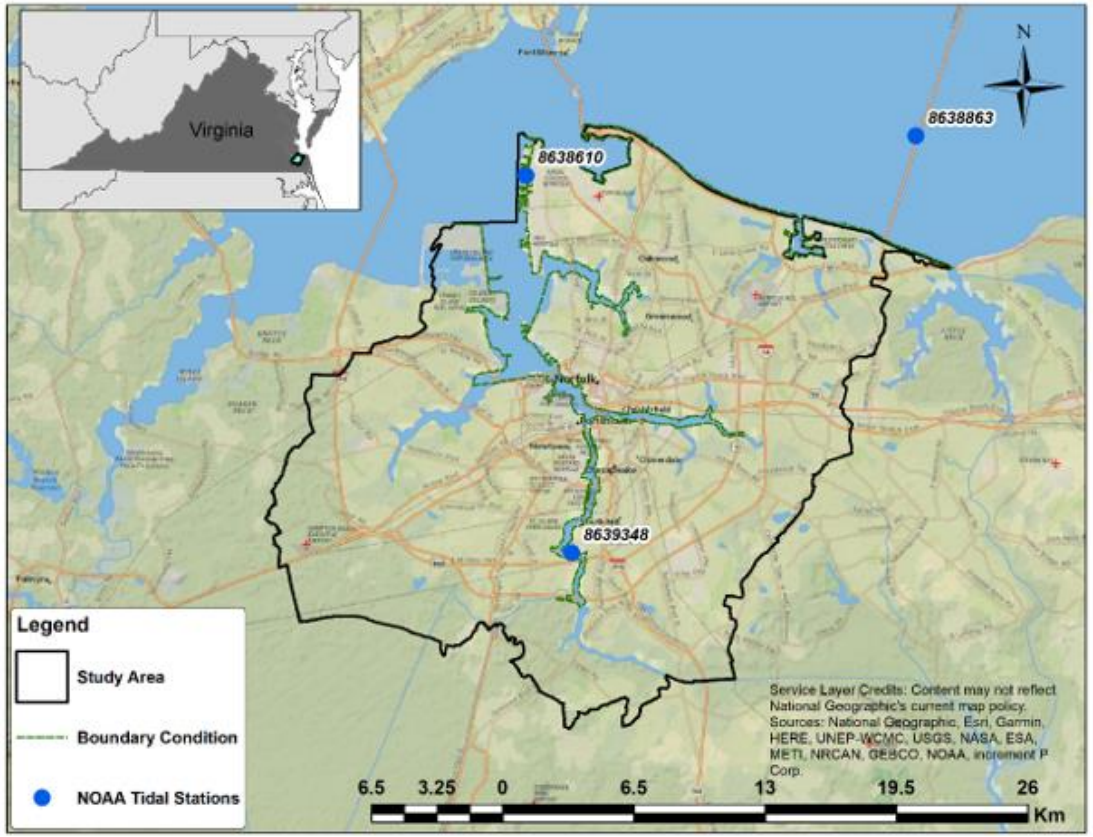
- **Objective 3:** Generate **online inundation map** with the location of the **flooded roadways** and **bridges with the ability to send automated warning email messages**.

Study Areas



VDOT's Hampton Roads District

(Rural Area)

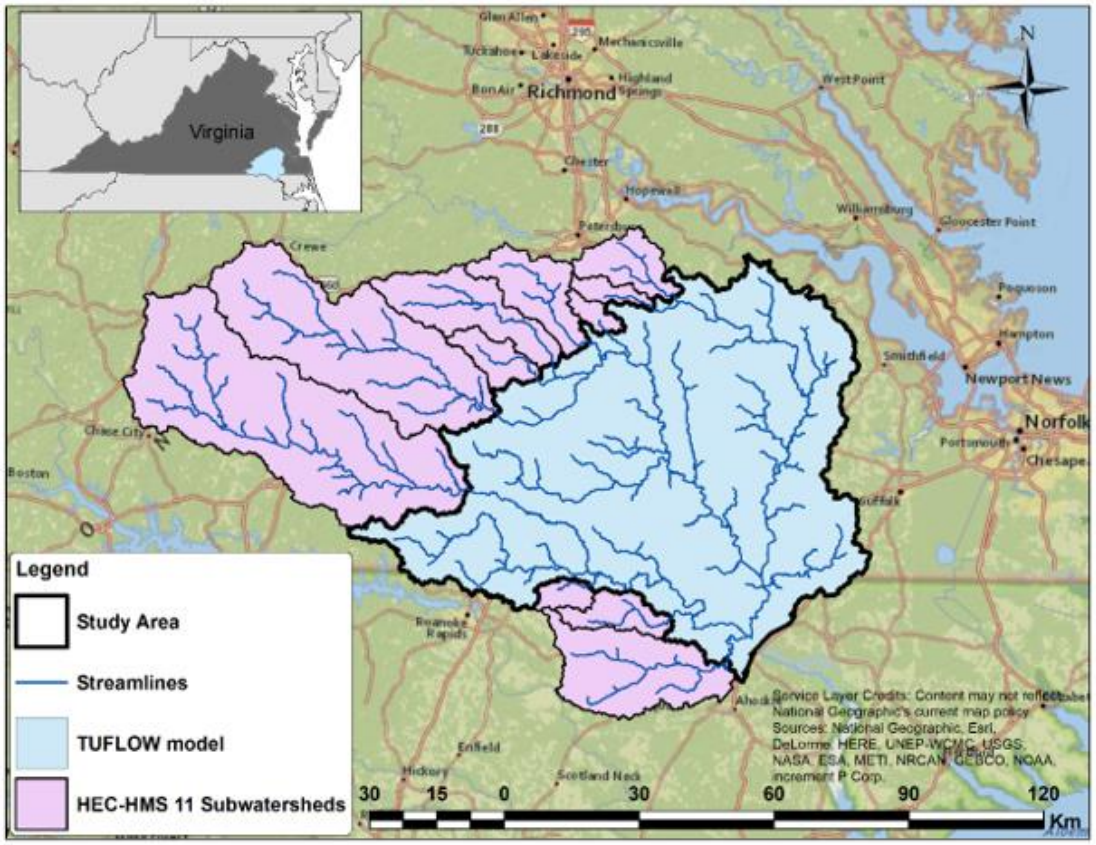


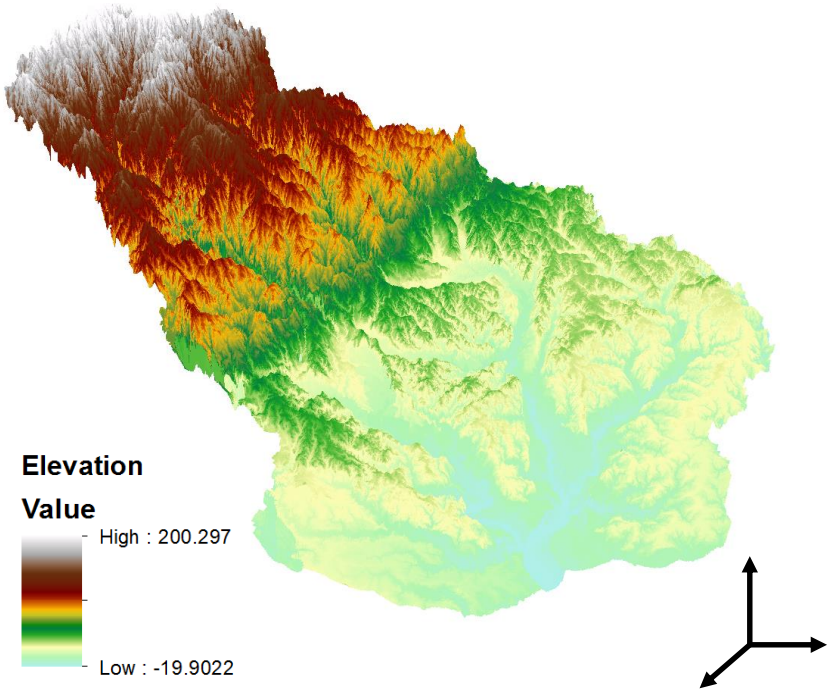
Norfolk Area

(Urban Area)

Modeling Work - VDOT's Hampton Roads District

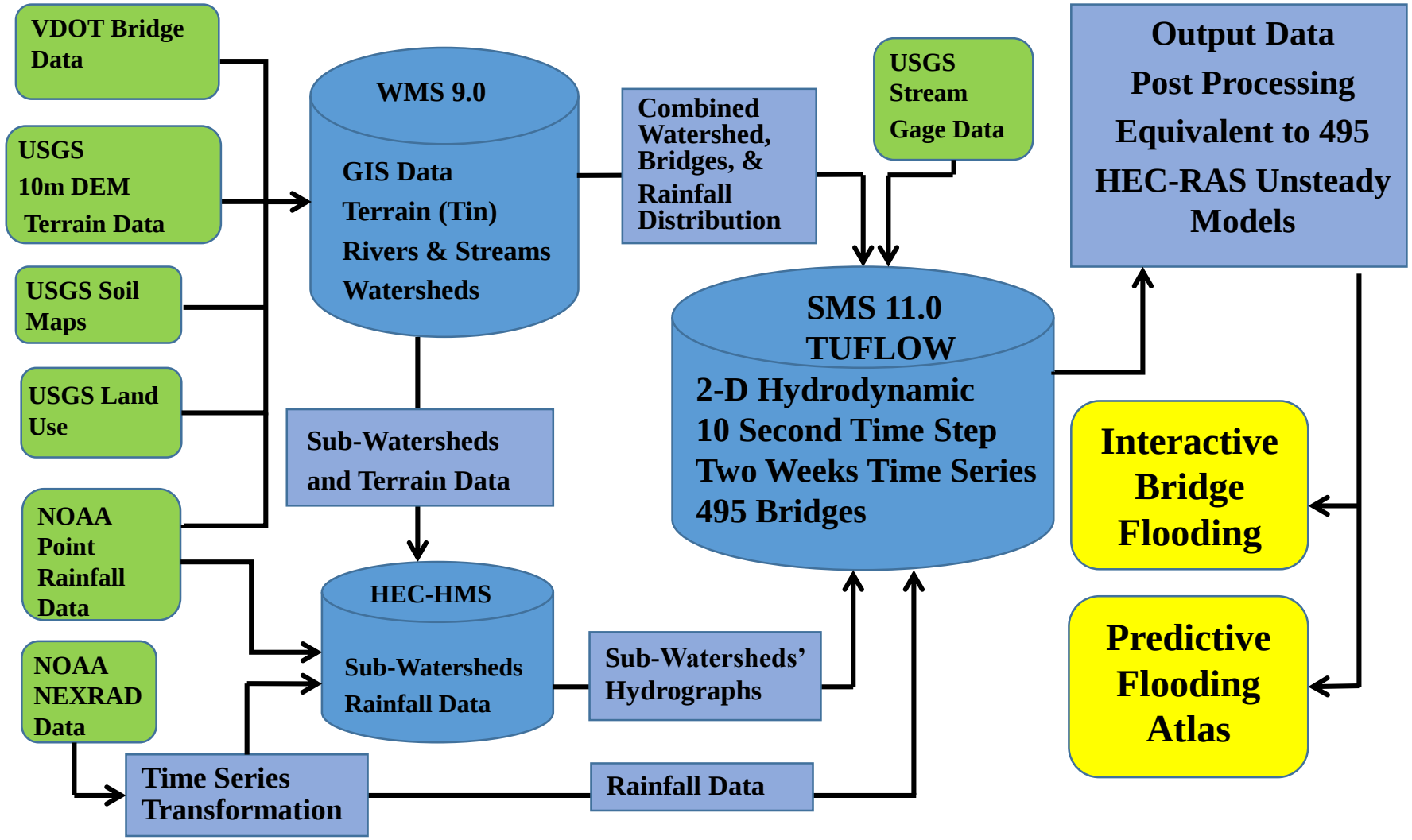
- Hampton Roads district, Virginia which is 11×10^3 km^2 of combined watersheds
- This Study area includes about 500 VDOT bridges and culverts
- 11 subwatersheds that surround the study area are incorporated as boundary conditions





Regional River Severe Storm Model (R^2S^2)

R^2S^2 is a diverse, multi-function digital platform that offers various applications to VDOT's Districts and Residencies, regional Emergency Services, and Environmental Agencies.



HWR Water Resources

Enhancing the Model

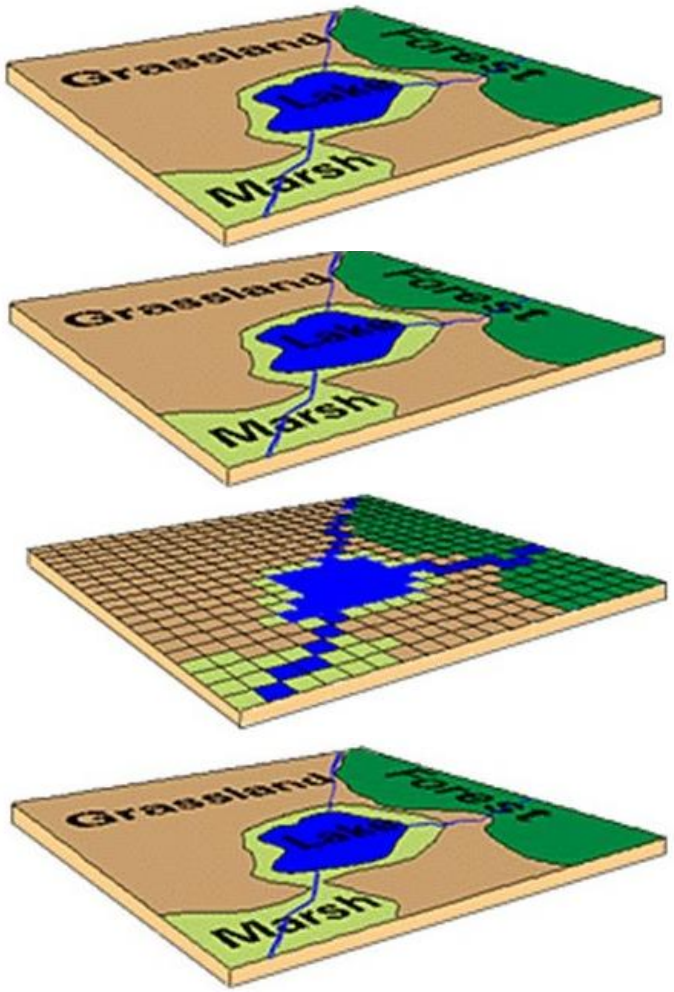
Model main Component

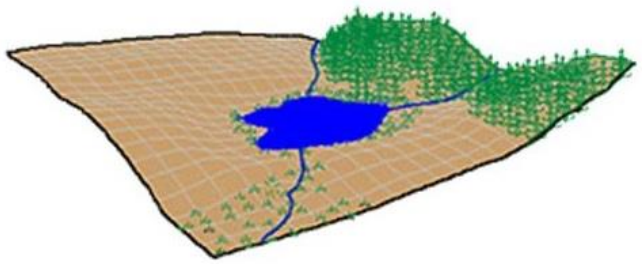
Streamflow line Shapfile Generated from the DEM

Road Shapfile files that include the Road Crossing and Elevation with streamflow lines

DEM Rater with 10m Resolution

Manning's Coefficient Shapefile Based on the Landuse

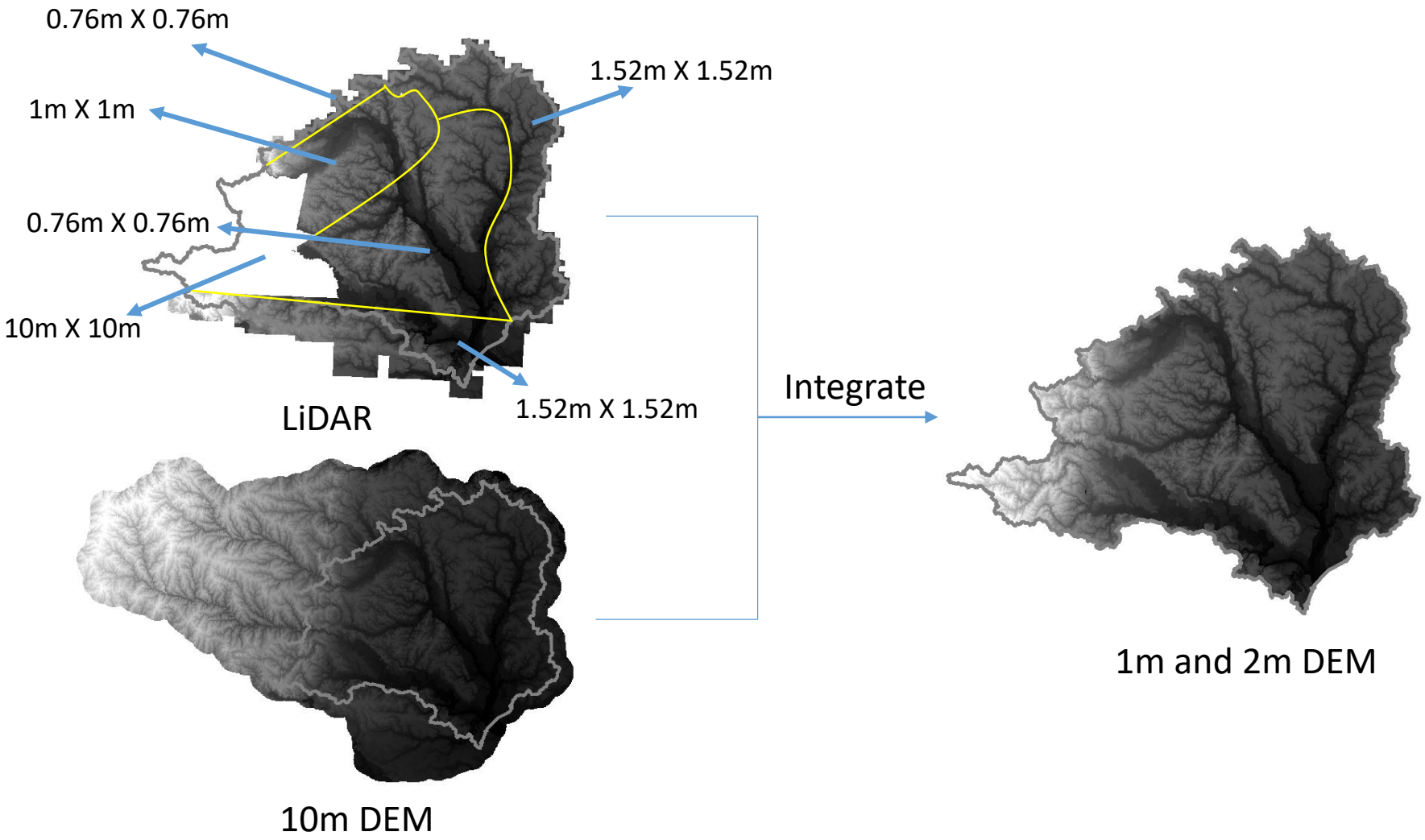




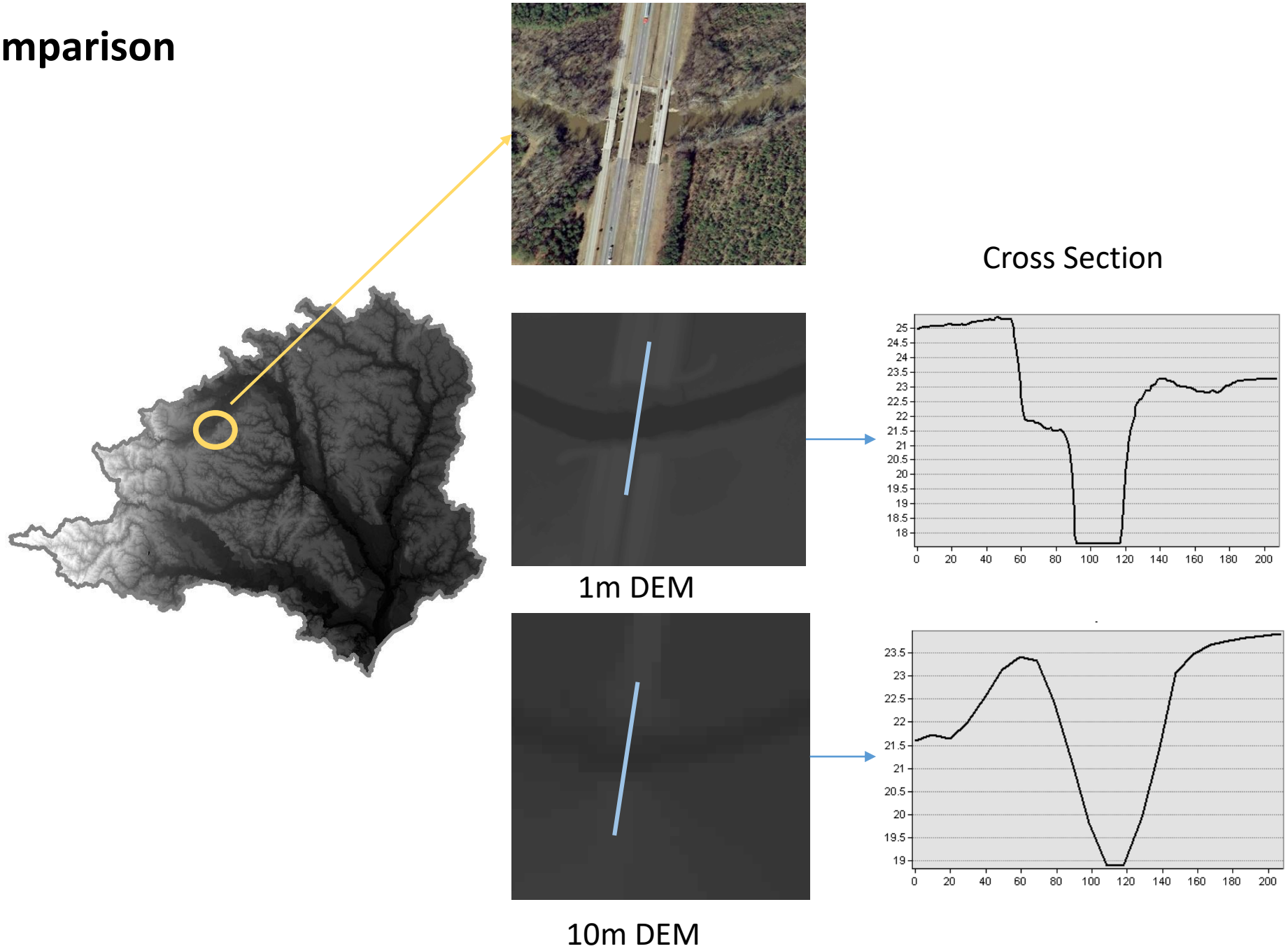
TUFLOW Model

DEM Raster Enhancement Progress

High resolution LiDAR data with resolution from 0.76m to 1.52m is available in most part of the study area.

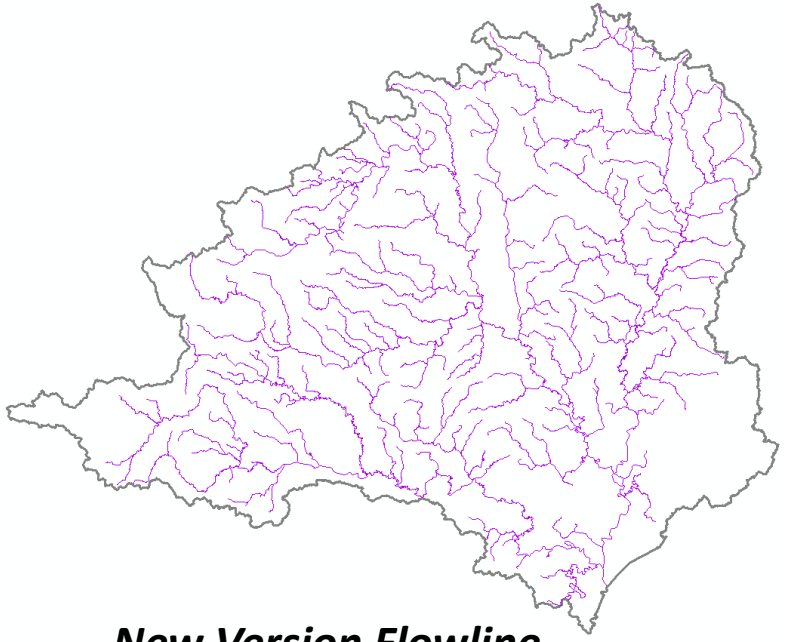


DEM Comparison



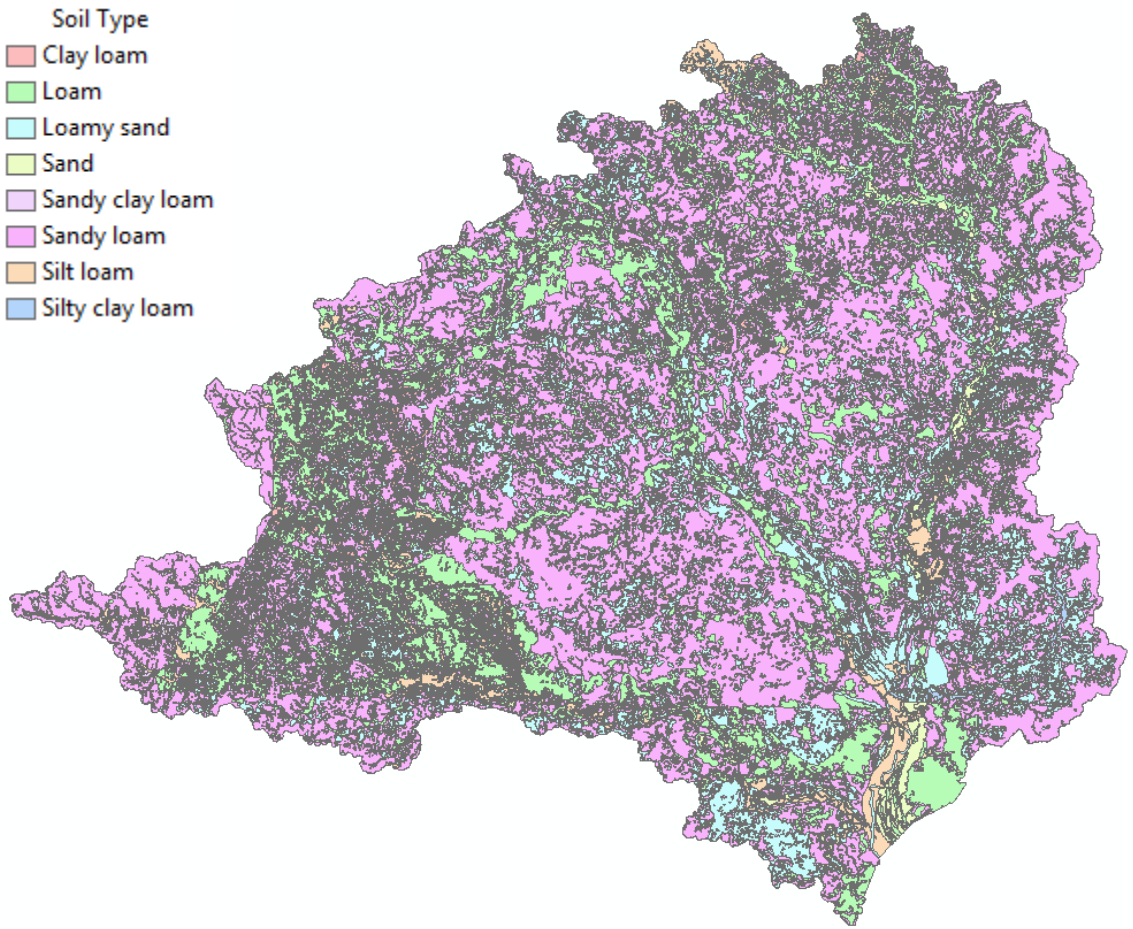
Streamline

Old Version Flowline

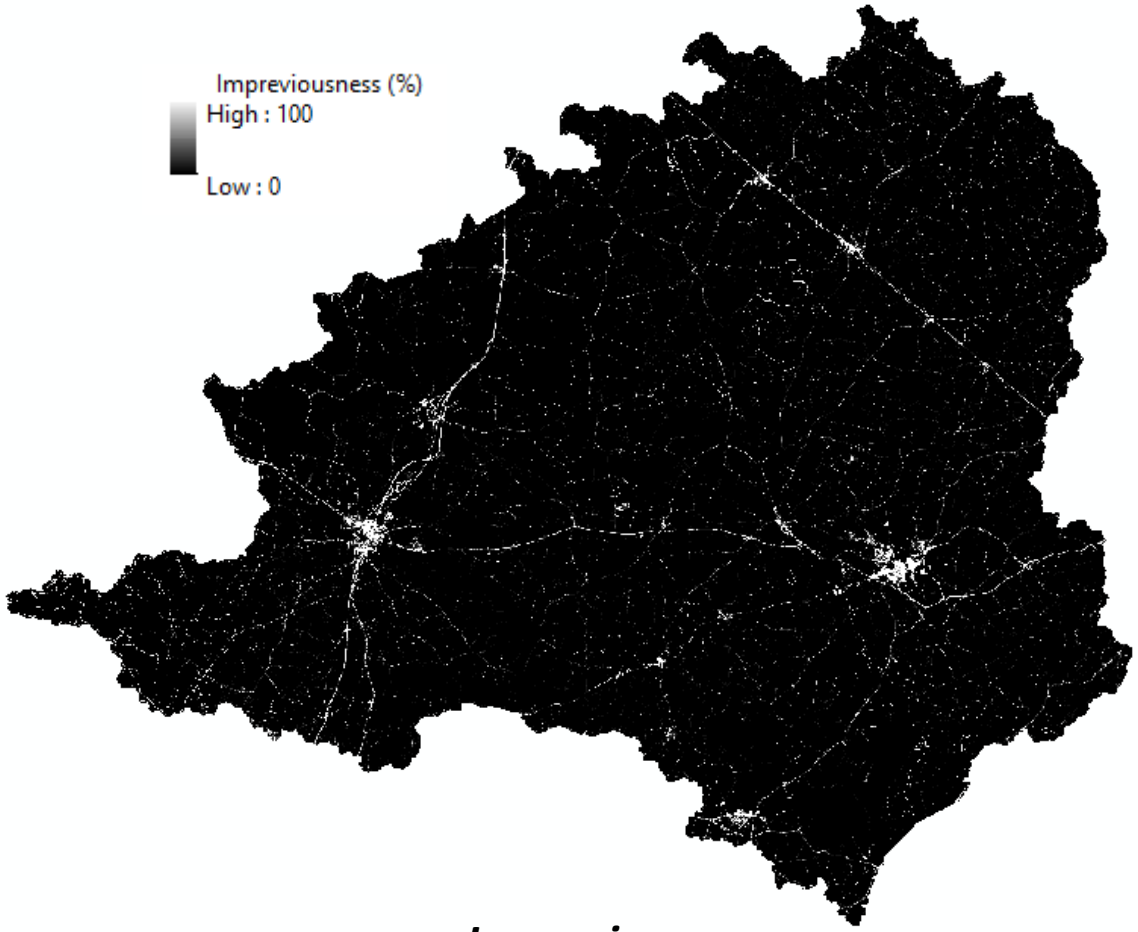


New Version Flowline

Infiltration Implementation



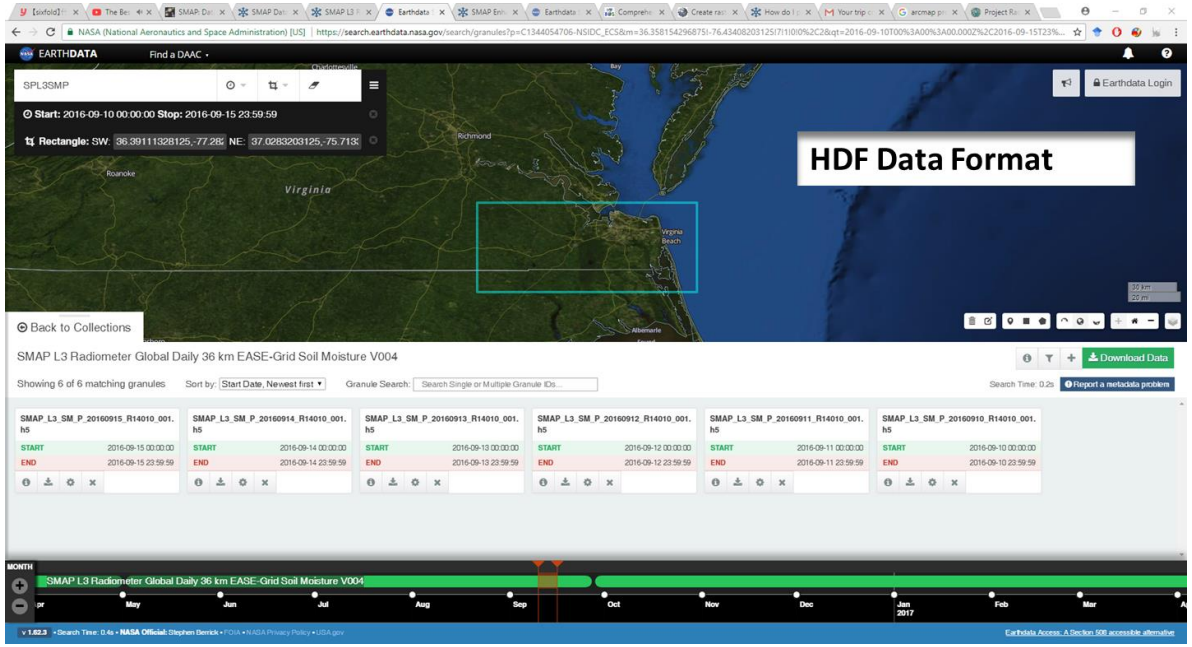
Soil Type

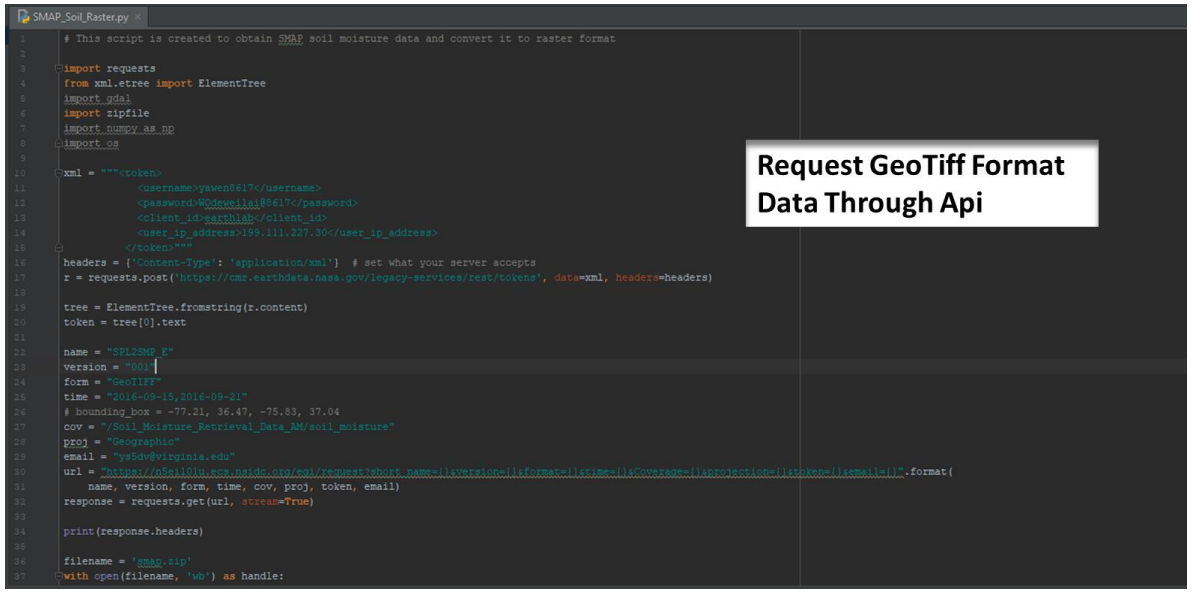


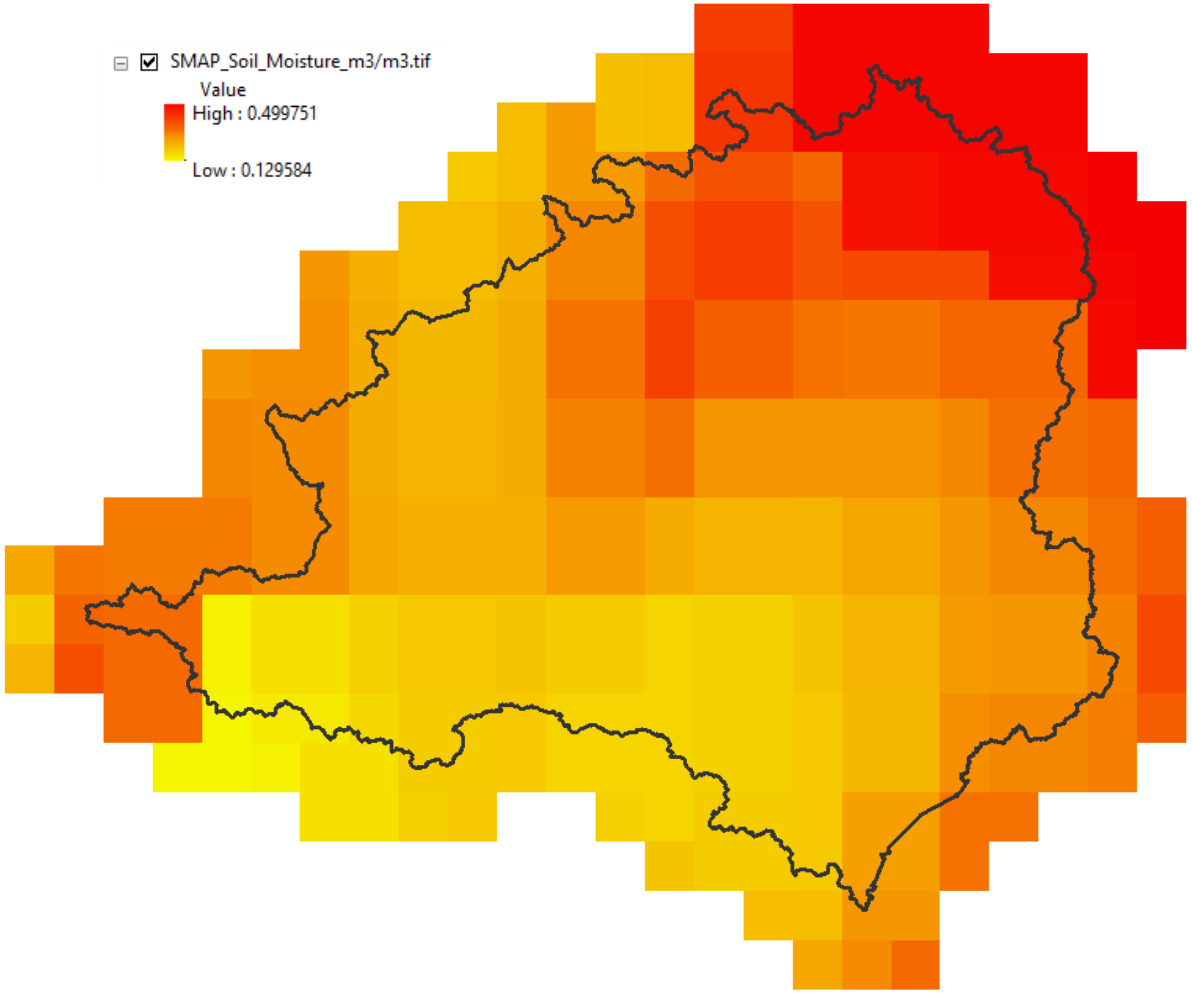
Imperviousness

Very small percentage of area has a imperviousness ratio greater then zero.

Prepare Soil Moisture from SMAP to TUFLOW



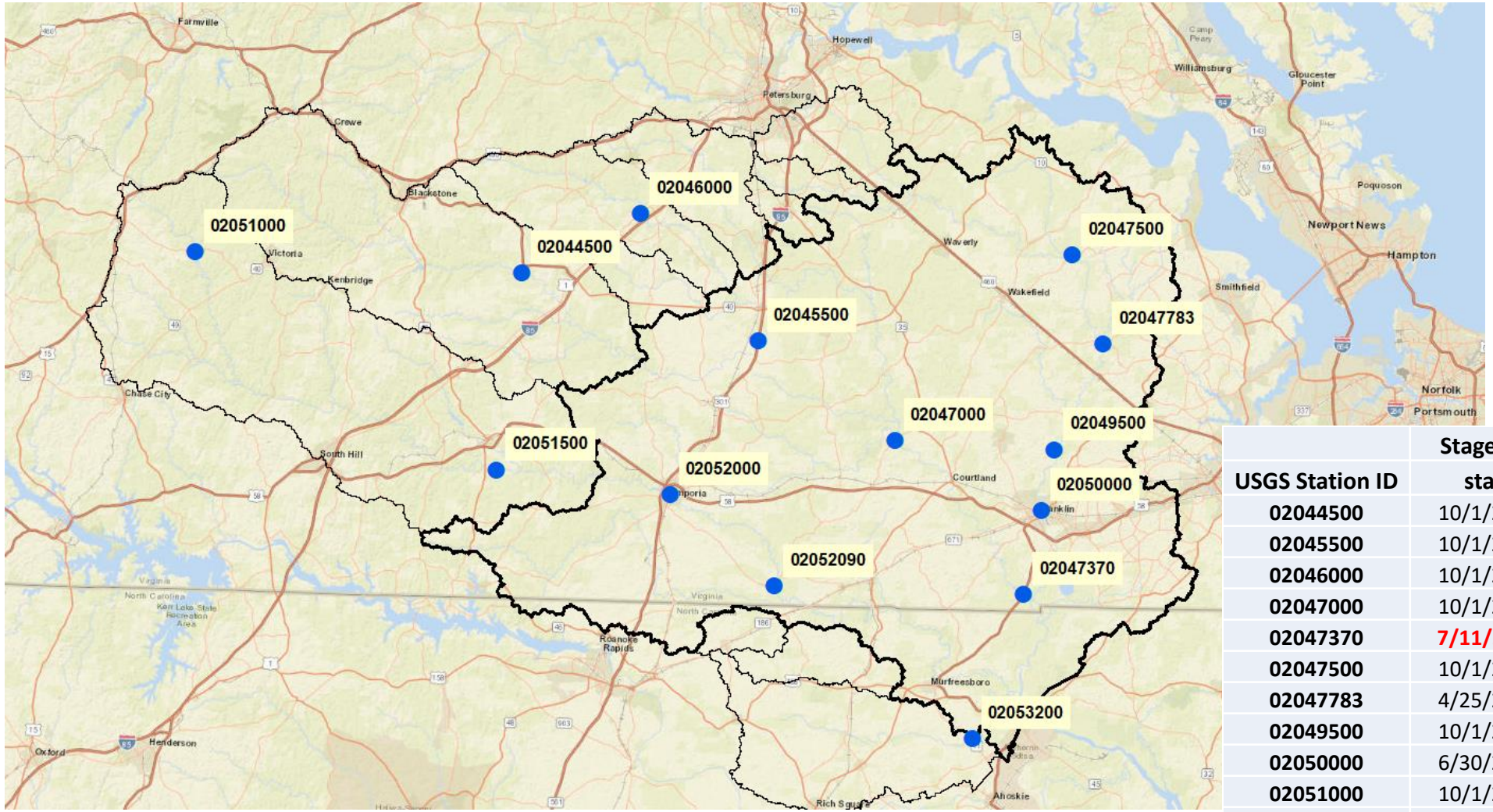




SAMP Soil Moisture (m3/m3)

Observation Data Preparation

USGS Stage Gage Data



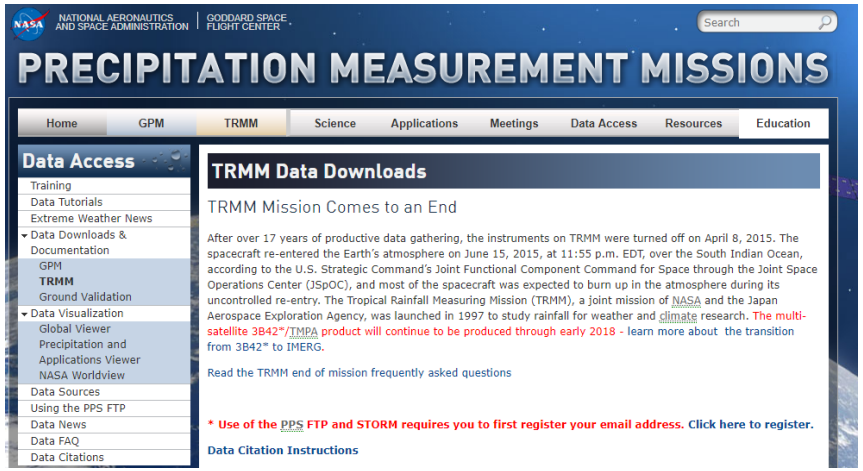
	Stage Depth Data Availability	
USGS Station ID	**start**	**End**
02044500	10/1/2003	11/13/2017
02045500	10/1/2003	11/13/2017
02046000	10/1/2003	11/13/2017
02047000	10/1/2002	11/13/2017
02047370	**7/11/2013**	11/14/2017
02047500	10/1/2003	11/13/2017
02047783	4/25/2013	11/14/2017
02049500	10/1/2005	11/13/2017
02050000	6/30/2010	11/14/2017
02051000	10/1/2003	11/14/2017
02051500	10/1/2002	11/13/2017
02052000	10/1/2003	11/13/2017
02052090	11/26/2012	11/14/2017
02053200	10/1/1985	11/14/2017

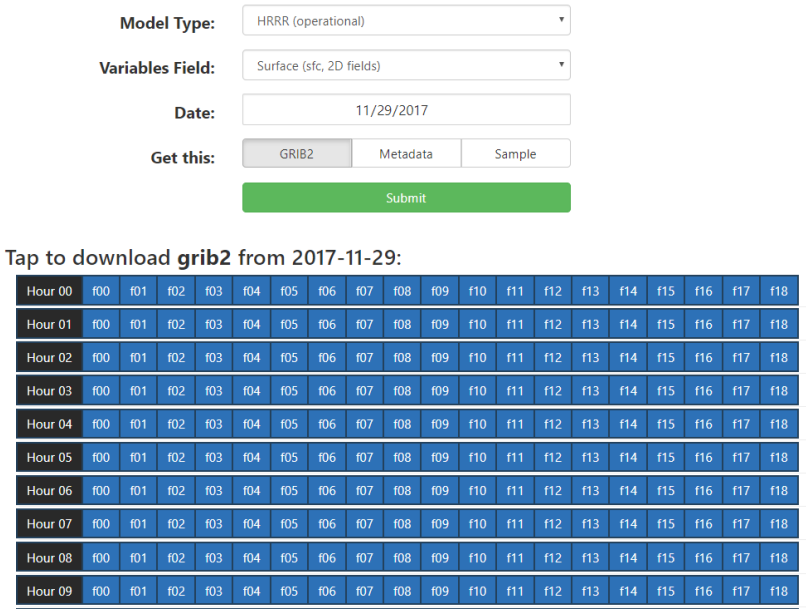
There are 10 USGS station located in the study. However, we believe more stations are necessary to better calibrating and verifying the model due to the complex and flat topographic.

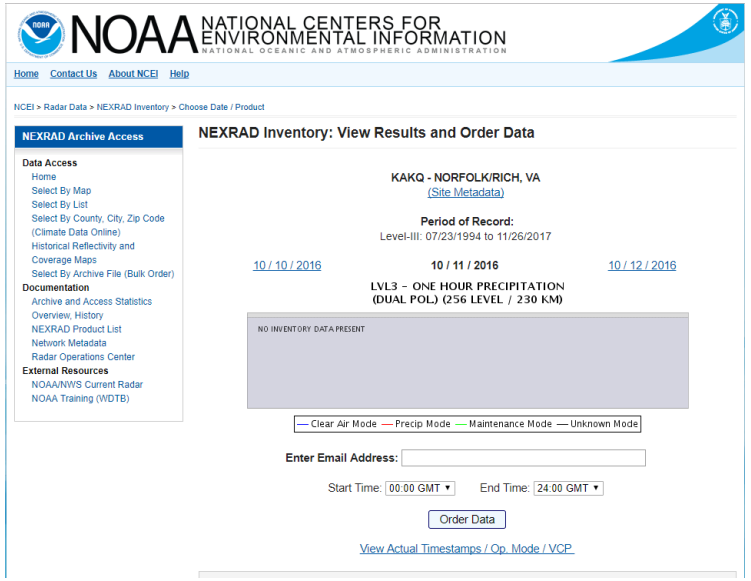
Observation Data Preparation

Observation Gridded Rainfall Data Preparation

- Three sources we explored to obtain the gridded rainfall data:
  - TRMM data
    - A code have been developed to pull and process the TRMM data for any given shapefile in the USA.
  - HRRR archived data
    - A code have been developed to pull and process the archived HRRR data from the University of Utah repository for any given shapefile in the USA.
  - NEXRAD data
    - However this is the best gridded rainfall data available but sometime the rainfall data was not recorded for critical storm events period. Also it is time consuming to order, download, and preprocess the data manual using NOAA Weather and Climate Tool (WCT).
  - All the codes are available through the project Phase II GitHub repository.





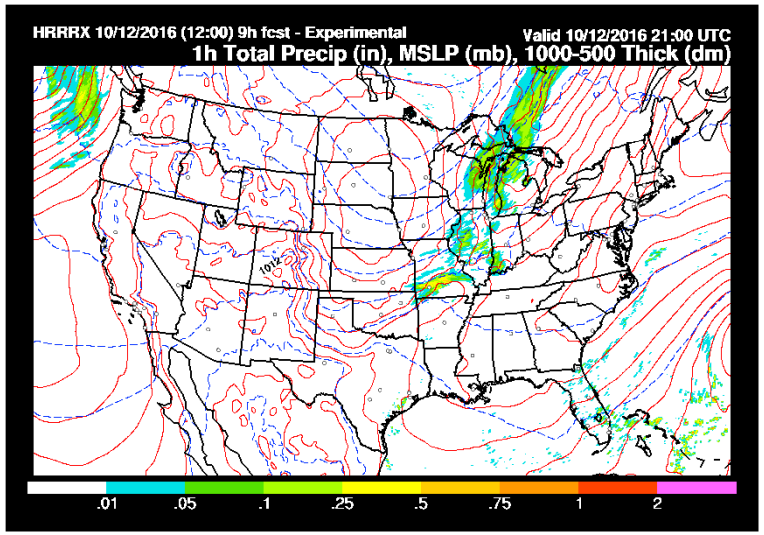


Rainfall Forecast Data Automation and Preparation for the Flood Warning System

Dataset	Data Provider	Relevant Data Product	Resolution		Forecast	Model Cycle
			Spatial (km)	Temporal (hrs)	(hrs)	
HRRR	NCEP	Surface total precipitation	3	1	18	24/day
RAP	NCEP	Surface total precipitation	13	1	18	24/day
NDFD	NWS	Quantitative precipitation forecast	5	6	72	8/day
NAM	NCEP	Surface total precipitation	12	1	36	4/day

High-Resolution Rapid Refresh (HRRR)

- NOAA/NCEP operational weather prediction system
- Forecasts hourly surface total precipitation at a **spatial resolution of 3-km**
- Comprised of a numerical forecast model and initialized by an analysis/assimilation system
- HRRR is run every hour of the day and forecasts out 18 hours on an hour time-step.
- We automated HRRR data access using Python and OPeNDAP

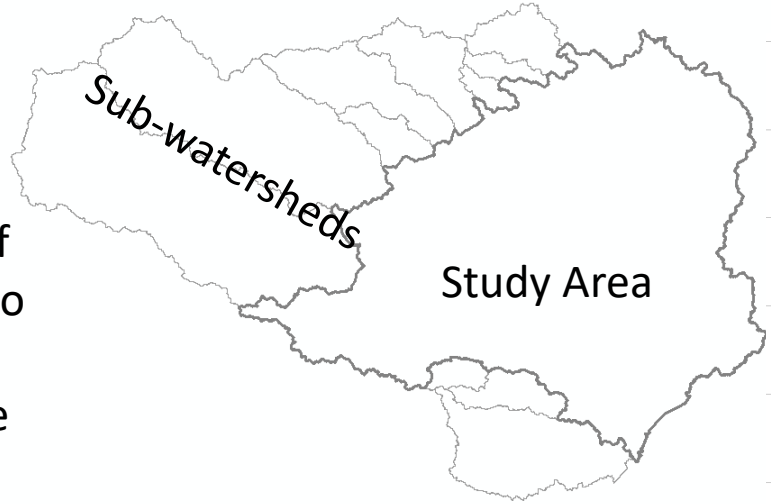


http://ruc.noaa.gov/hrrr/displayImage.cgi?image=hrrrcrefimage&width=859&model=hrrr&title=HRRR

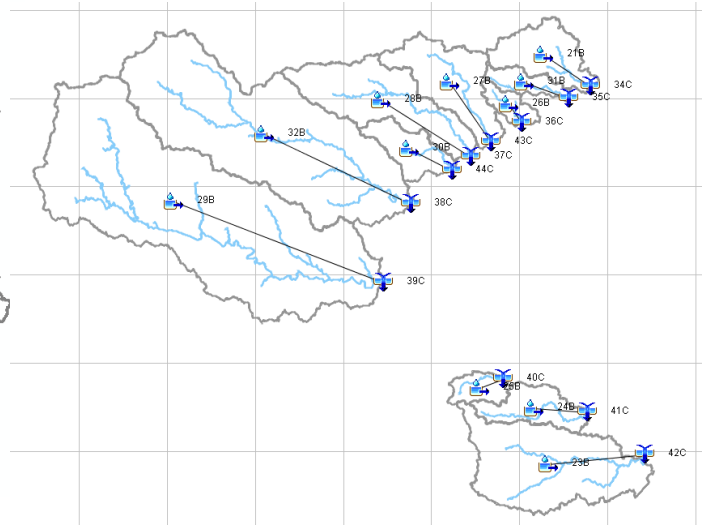
TUFLOW Model Boundary Input

Historical Runs

The HEC-HMS model was built to simulate the outlet flow rate of each sub-watershed to be applied as boundary input of the 2D model.



HEC-HMS model



Flood Forecasting Runs

The National Water Model simulation will be used as boundary input for flood forecasting system.

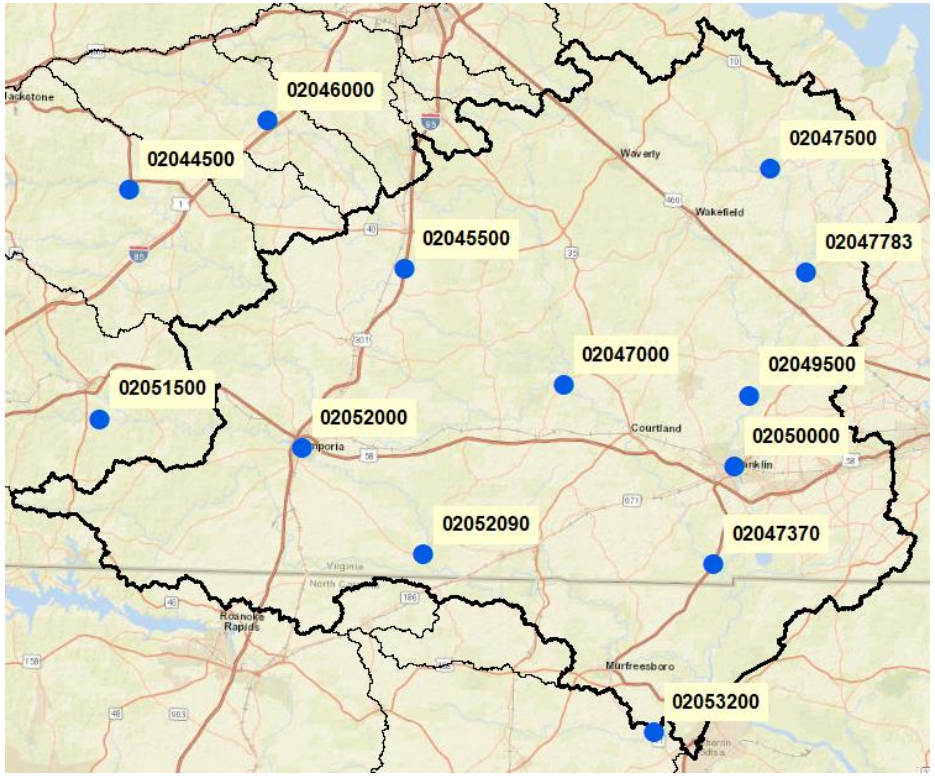
Current River Forecast Points (~3,600)

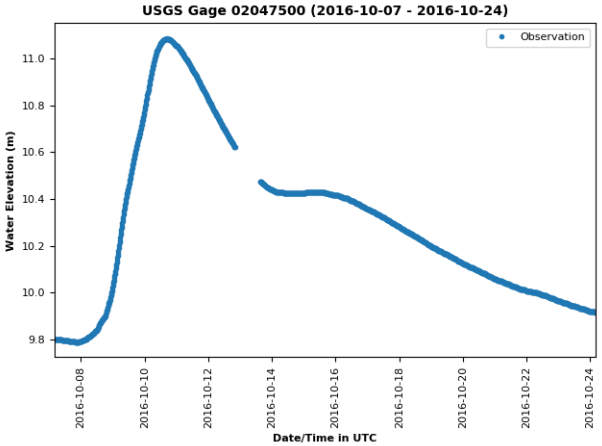
+

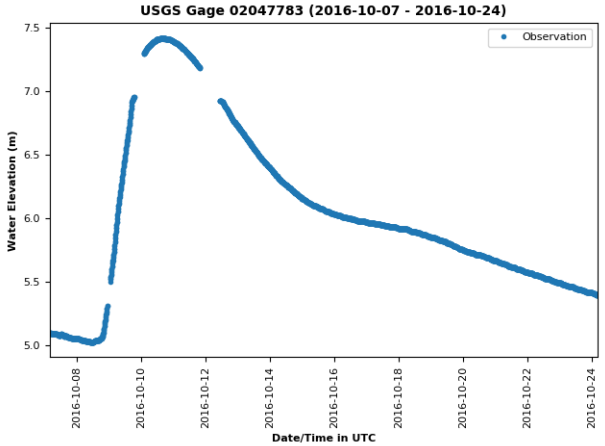
NWM Streamflow Output Points (~2.7 mil)

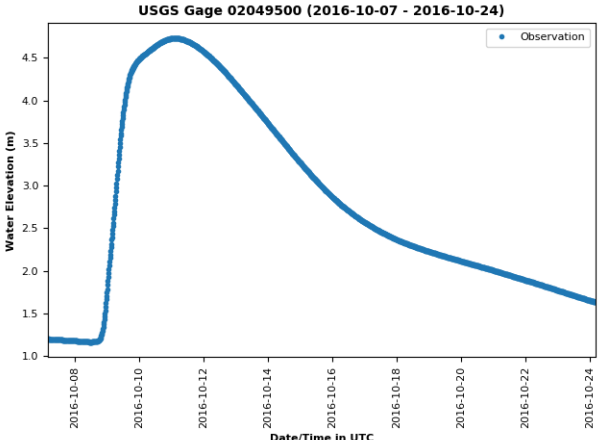
http://water.noaa.gov/about/nwm

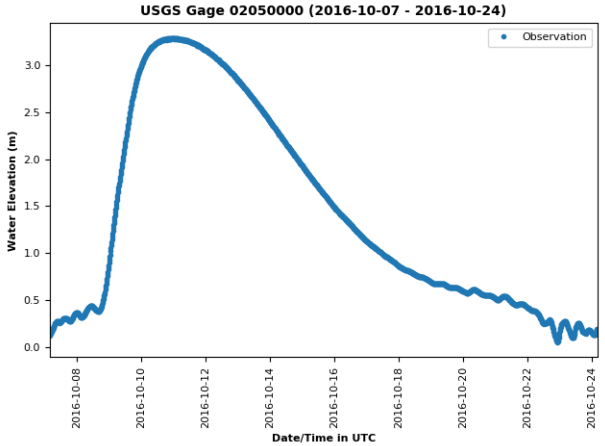
Hurricane Nicole, 2016

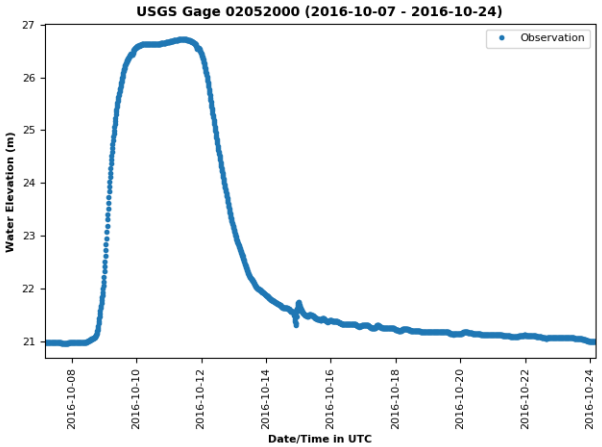


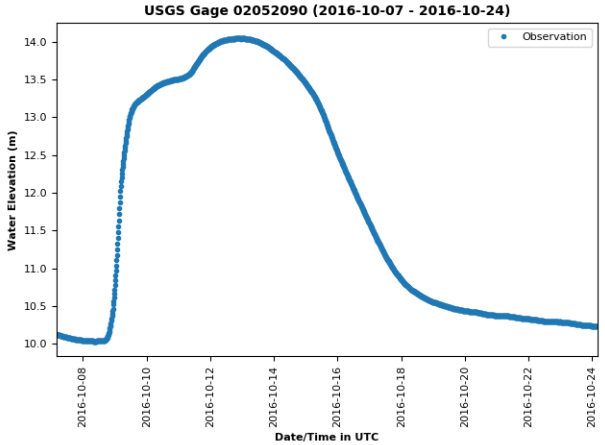




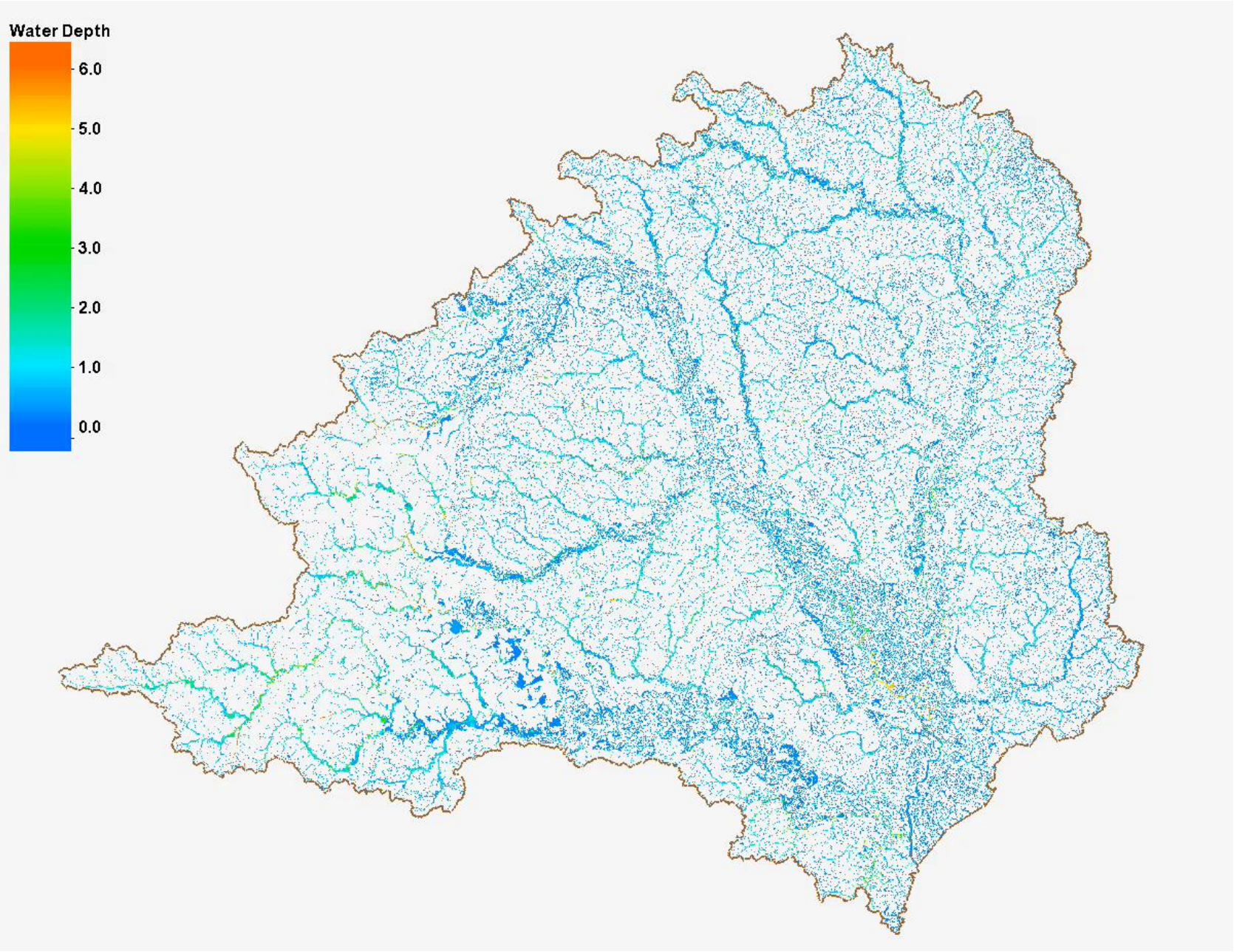




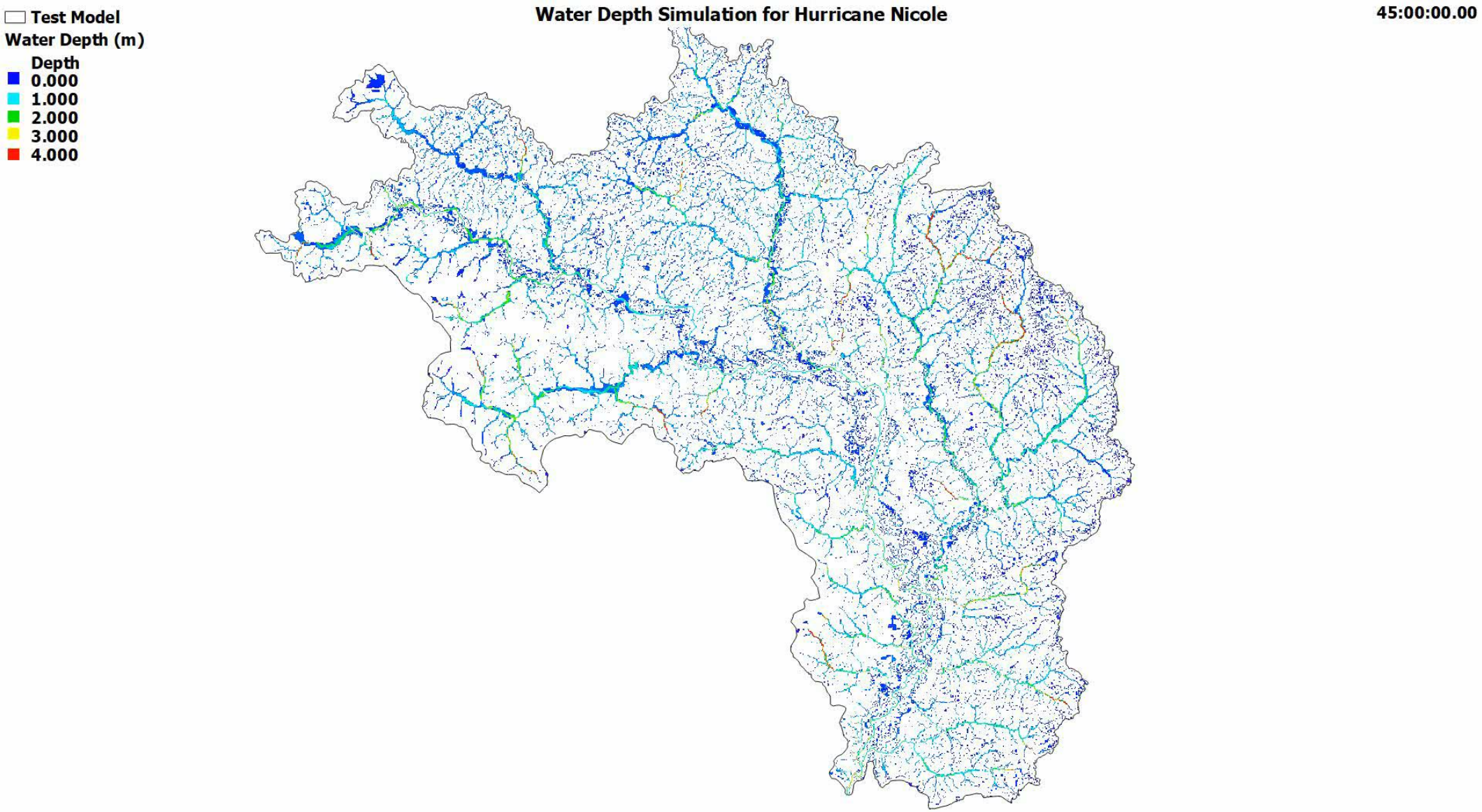




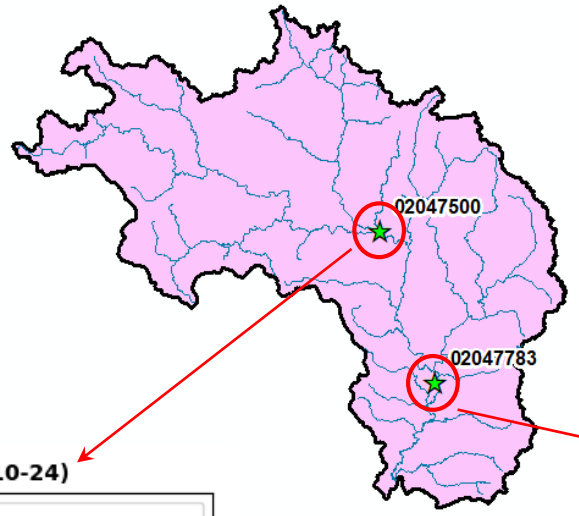
Hurricane Nicole, 2016

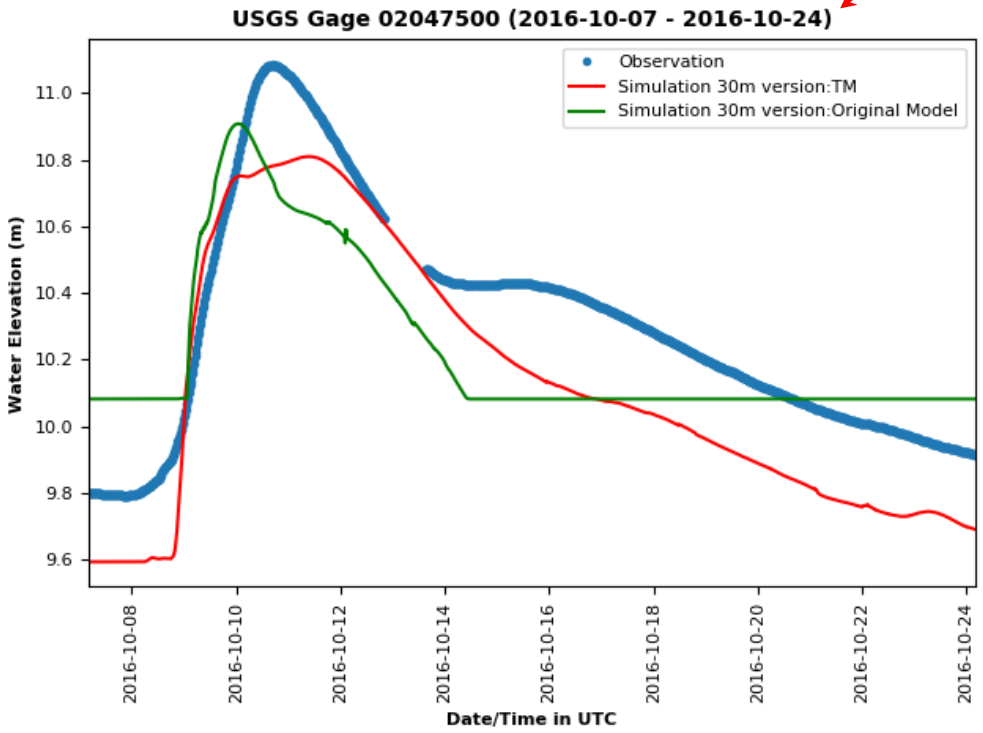


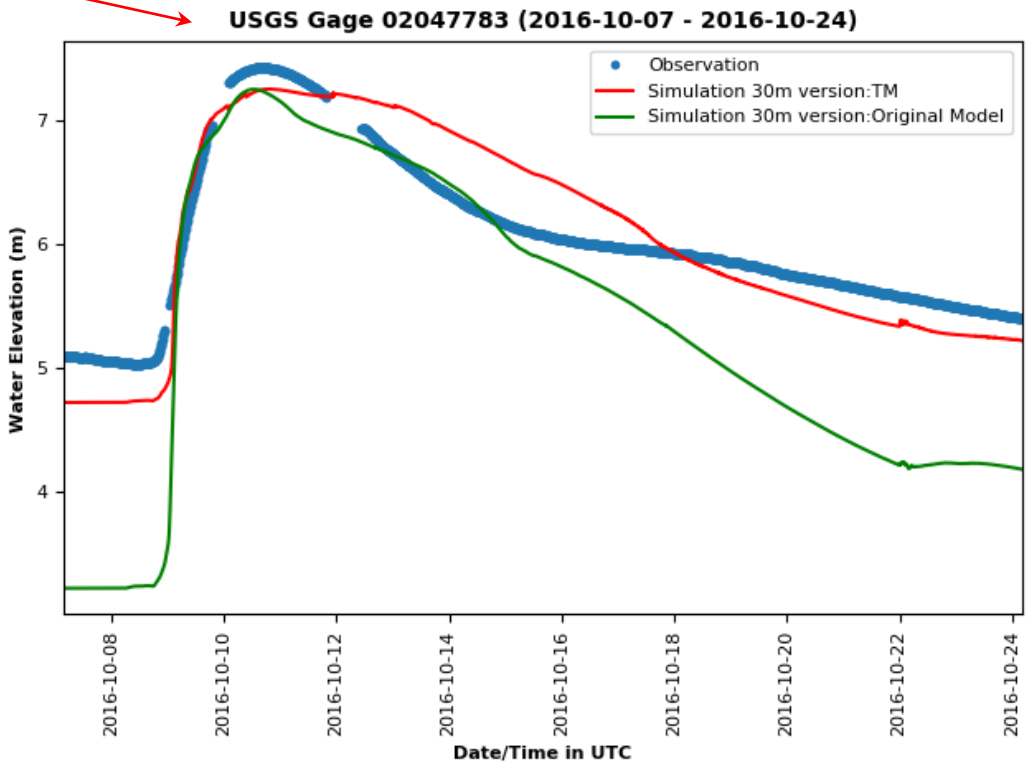
Preliminary Calibration for Hurricane Nicole, 2016



Preliminary Calibration for Hurricane Nicole, 2016

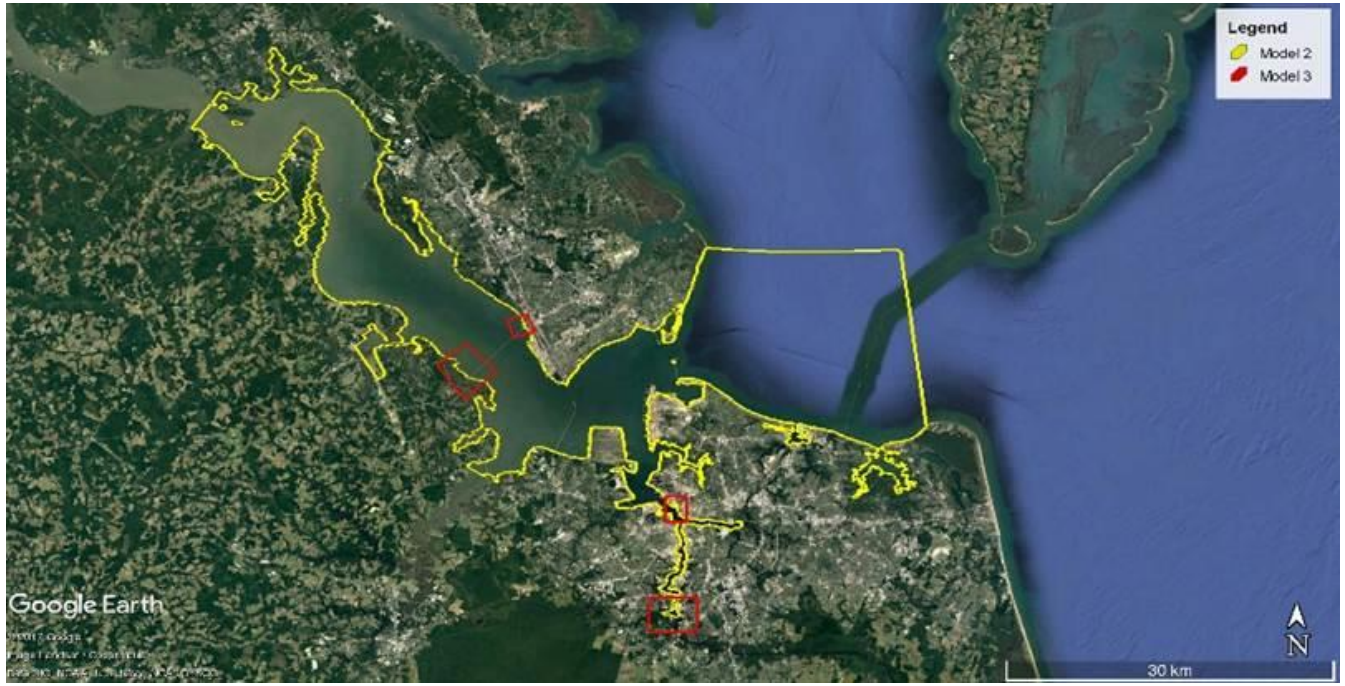




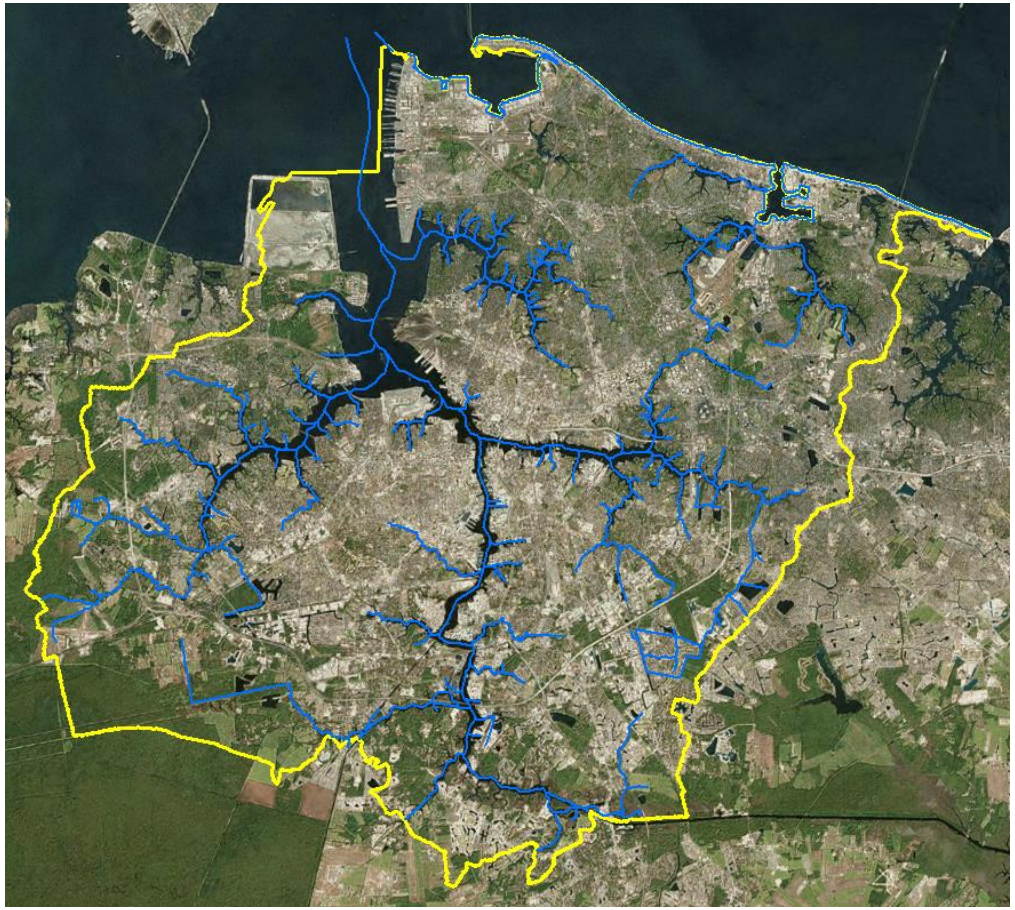


Modeling Work – Norfolk Area

A 2D hydrodynamic model needs to be developed to predict inundated roads in Norfolk Region.
The ODU storm surge model will provide the tide level boundary condition.

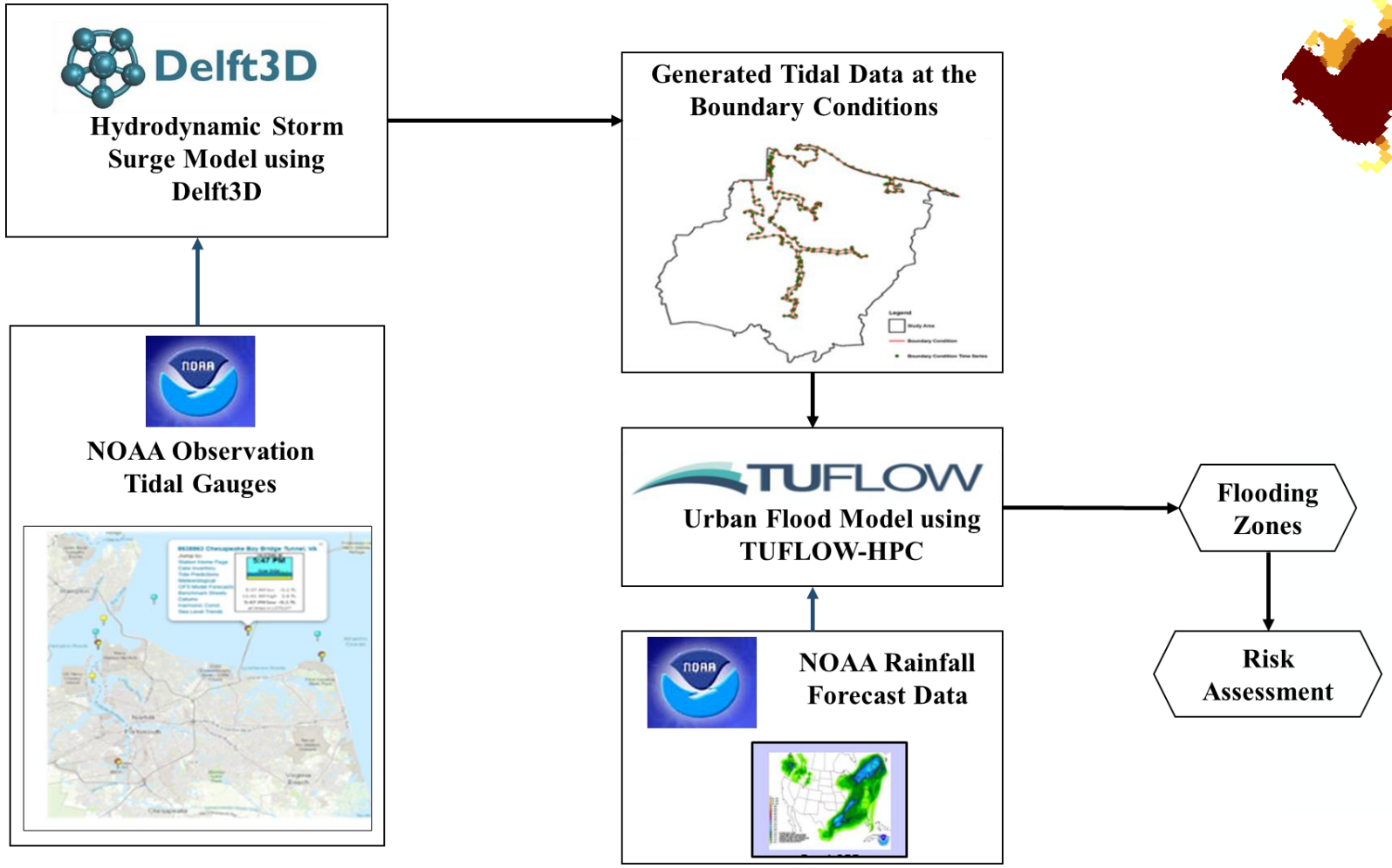


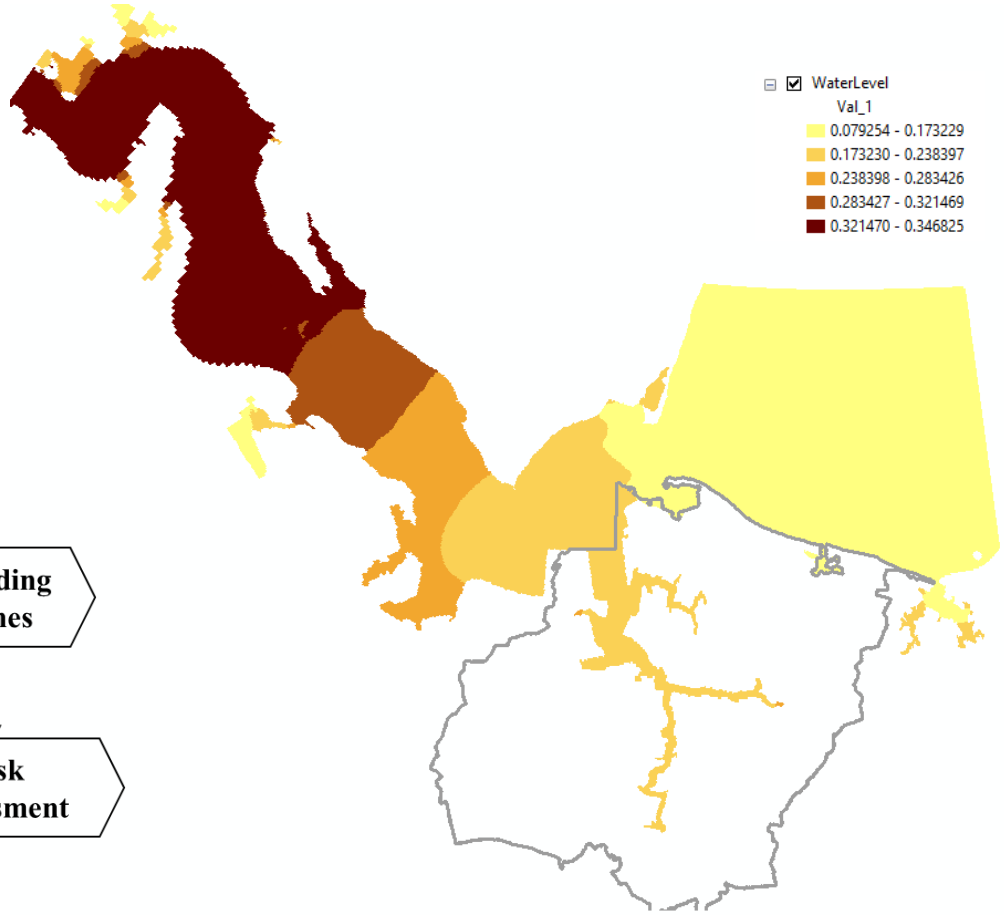
ODU Storm Surge Model



TUFLOW Overland Flood Model

Storm Surge and Urban Flood models Coupling Workflow





Storm surge model interaction with the urban flood model

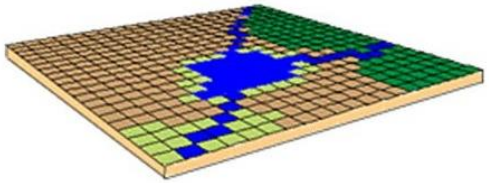
Model main Component

Storm Surge

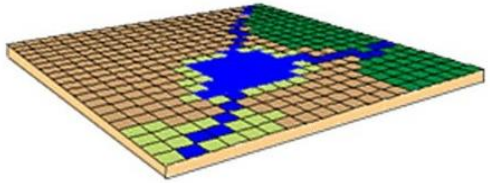
Precipitation

Forces

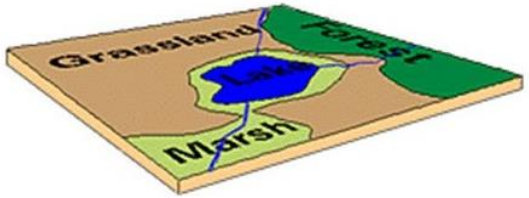
Impervious Ratio



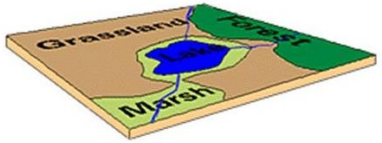
Soil Type



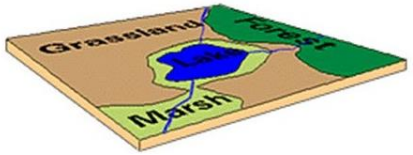
Manning's Coefficient Shapefile



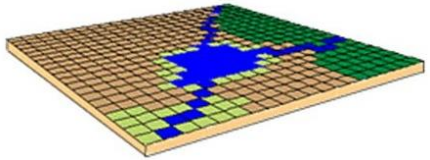
Flowline Shapfile



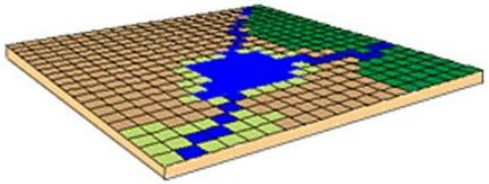
Road Network



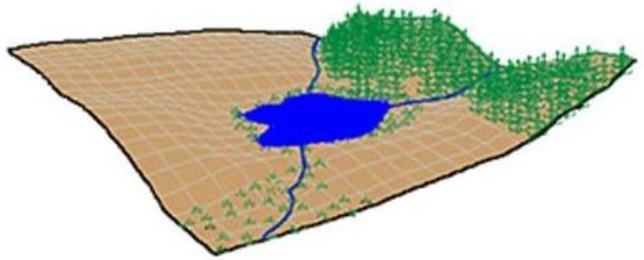
1m DEM and bathymetry



Topography Mesh



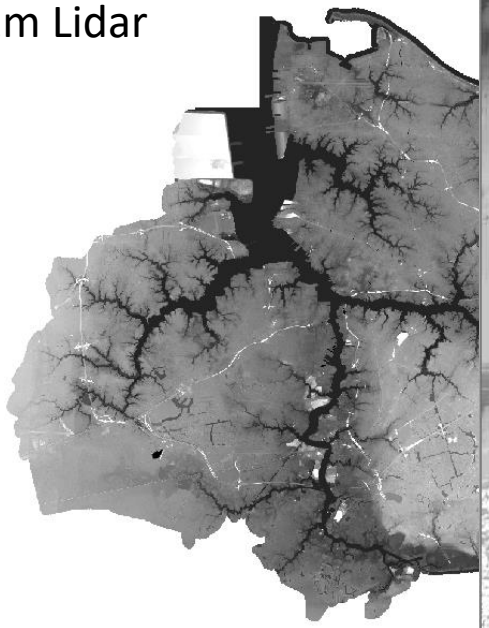
Buildings



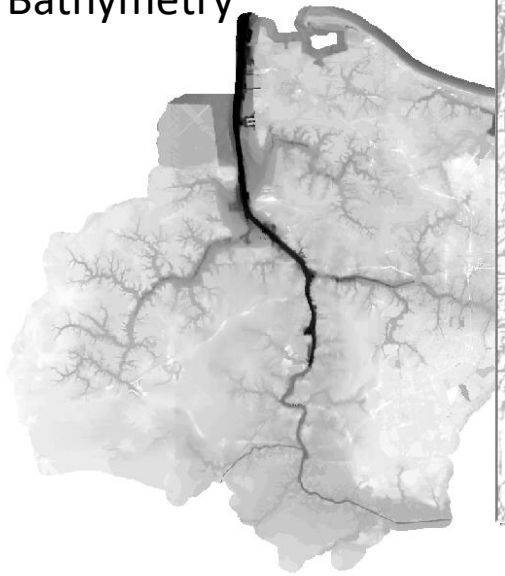
TUFLOW Model

Integrating High Resolution DEM

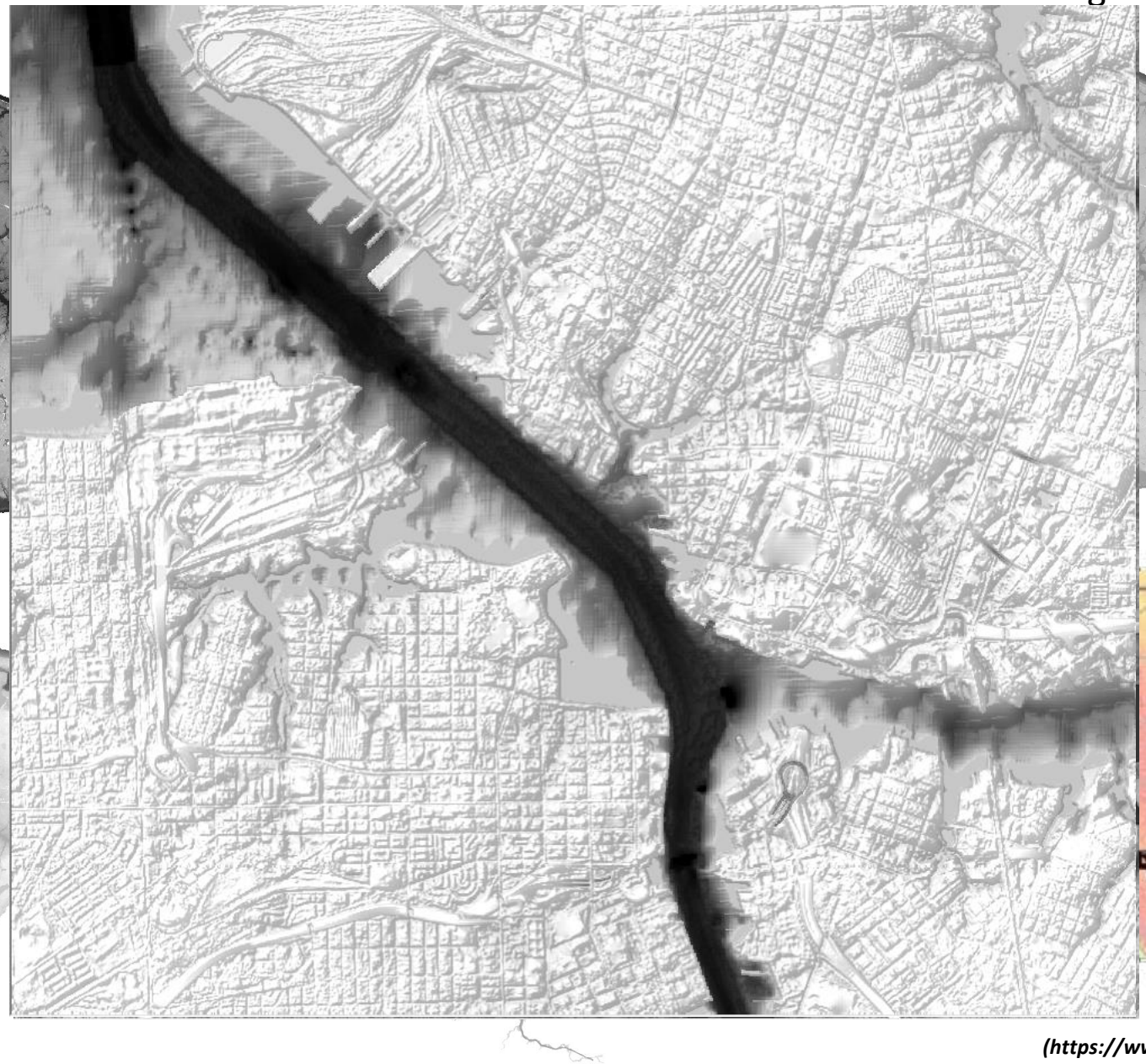
1m Lidar

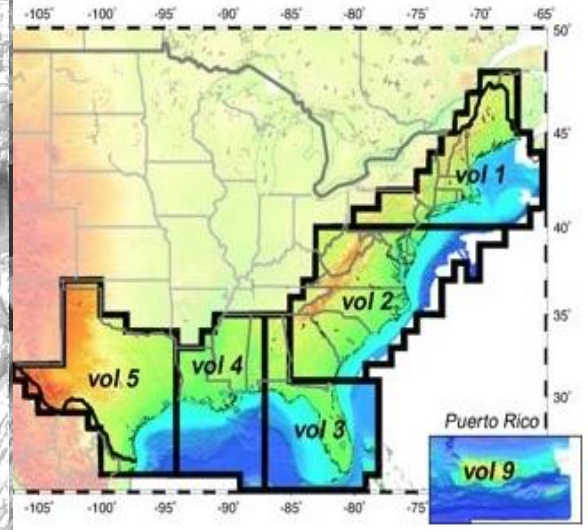


VB Bathymetry



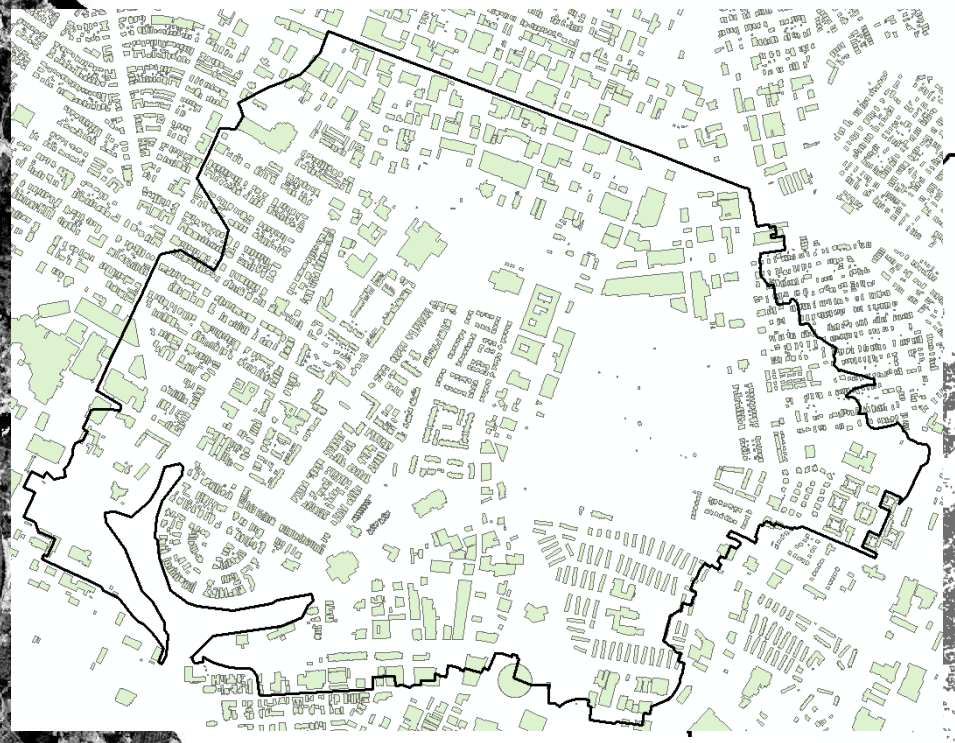
Integrated DEM



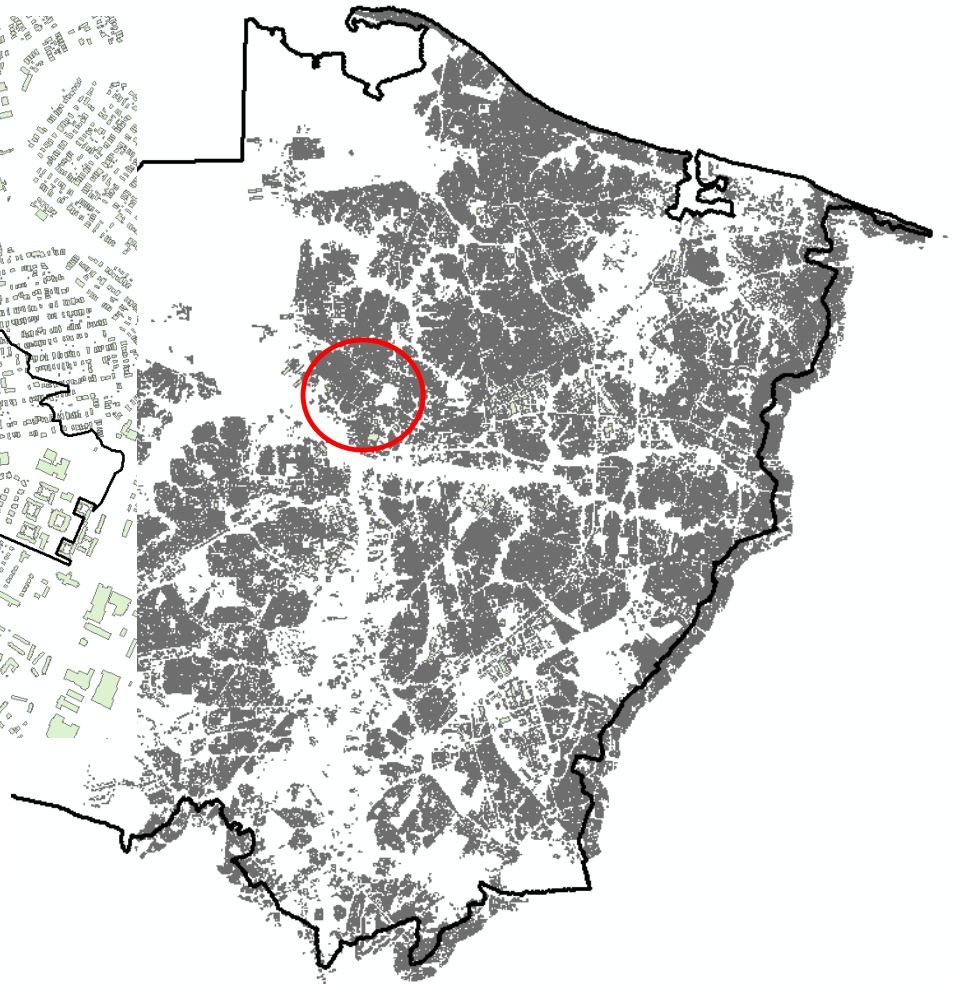


NOAA Bathymetry
(https://www.ngdc.noaa.gov/mgg/coastal/crm.html)

Imperviousness and Building Footprint



Building Footprint for Hague Community

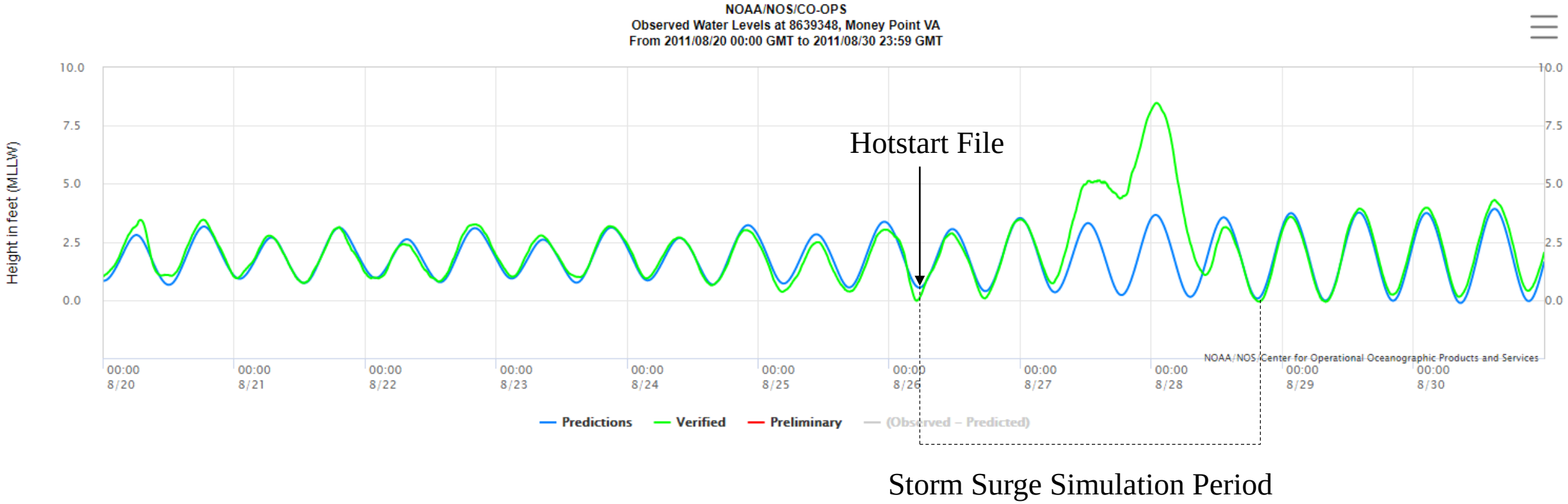


Surface Imperviousness

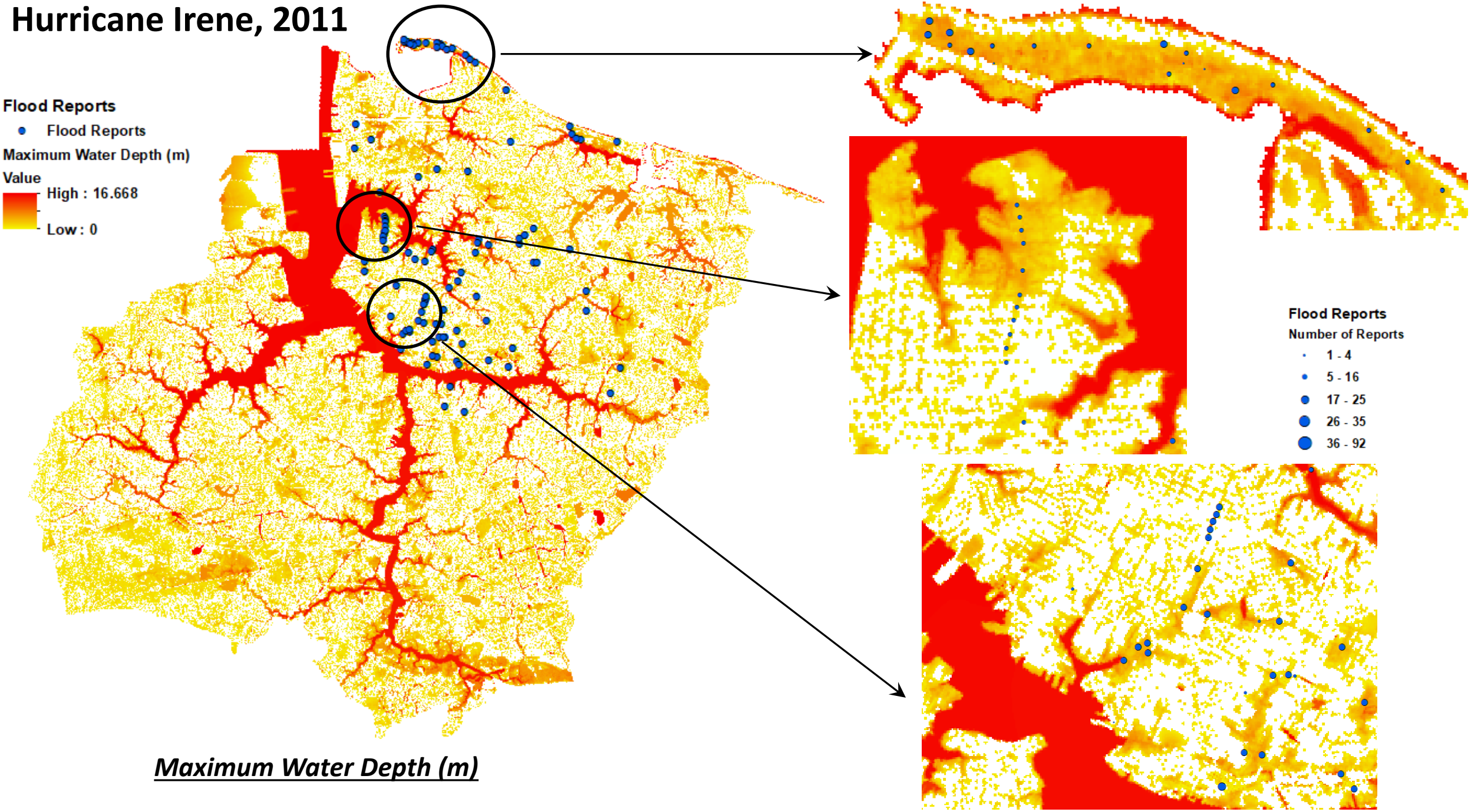
Building Footprint

Hurricane Irene, 2011

The storm surge simulation we have so far is from 26-Aug-2011 to 28-Aug-2011. We generated a hotstart file at the start time of the storm surge model by running the TUFLOW model with NOAA tidal level observation. Then, the hotstart file is feed into the model to let it run from wet condition.



Hurricane Irene, 2011



Maximum Water Depth (m)

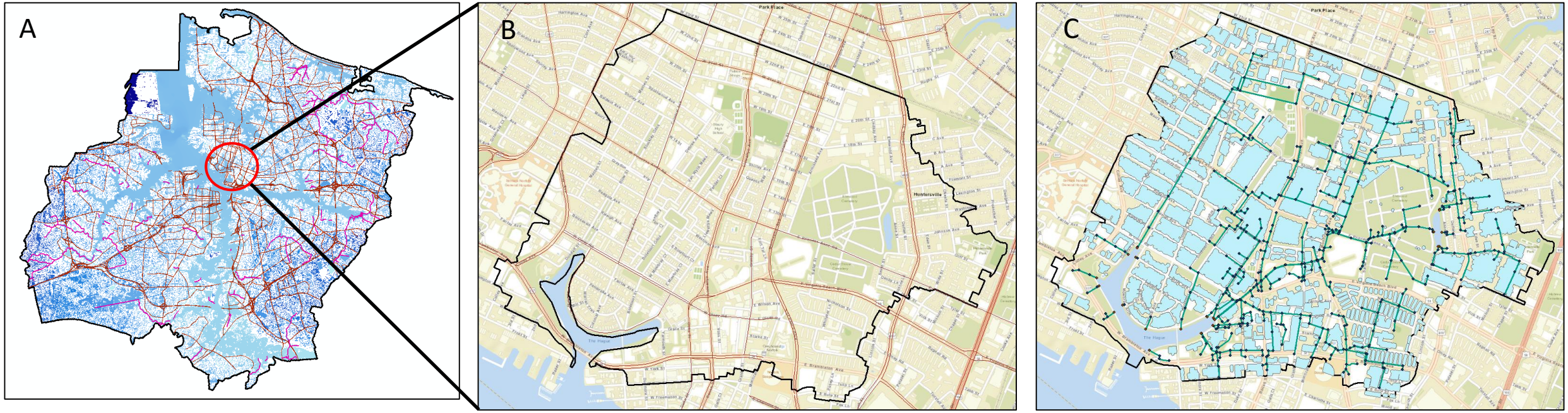
Hurricane Irene, 2011

Modeling the Hague Community

A. Using the current model domain.

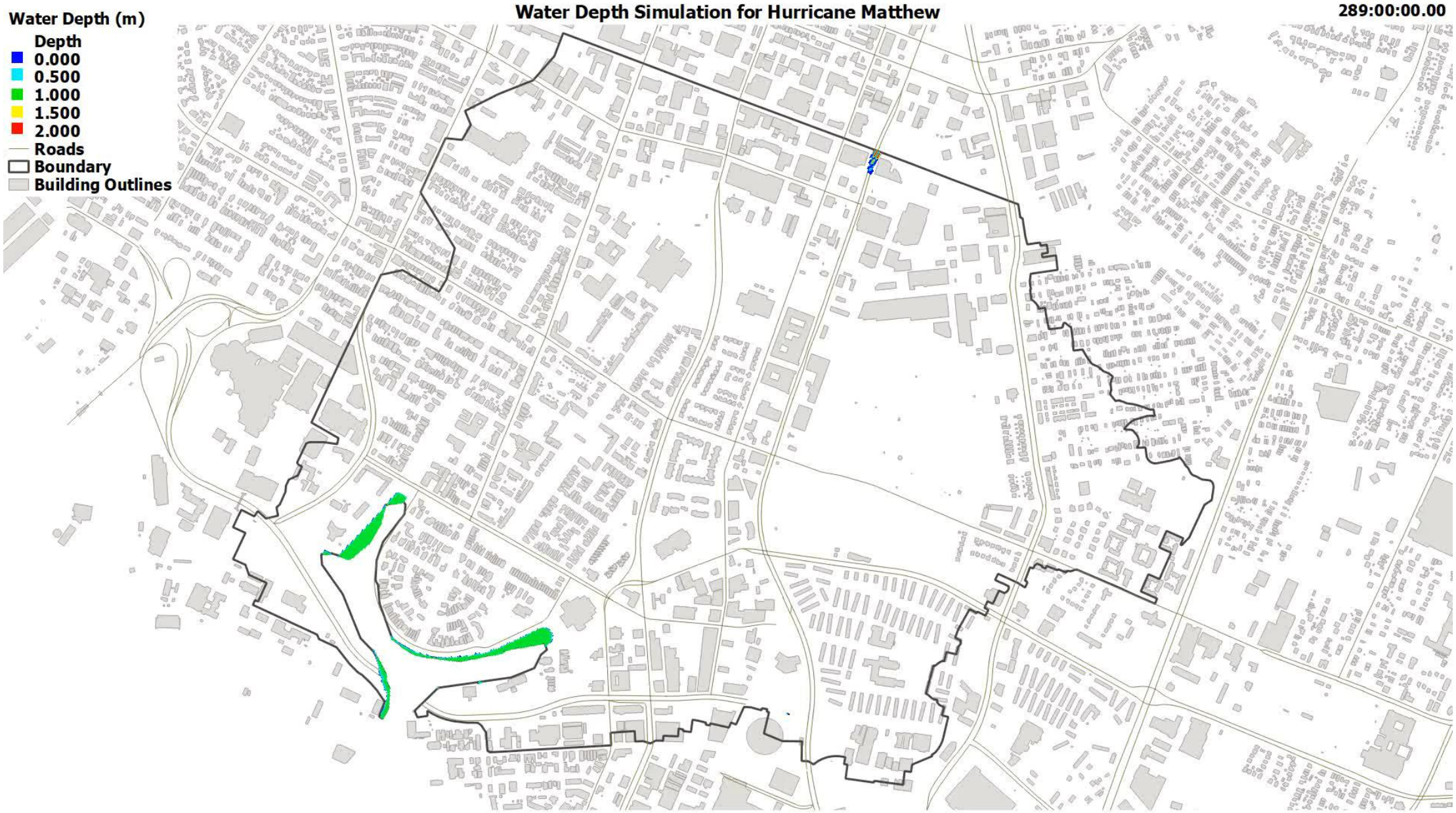
B. Create a small domain for the Hague community from the current model domain.

C. Using the small domain along with the storm drainage system for the Hague community.



Hurricane Irene, 2011

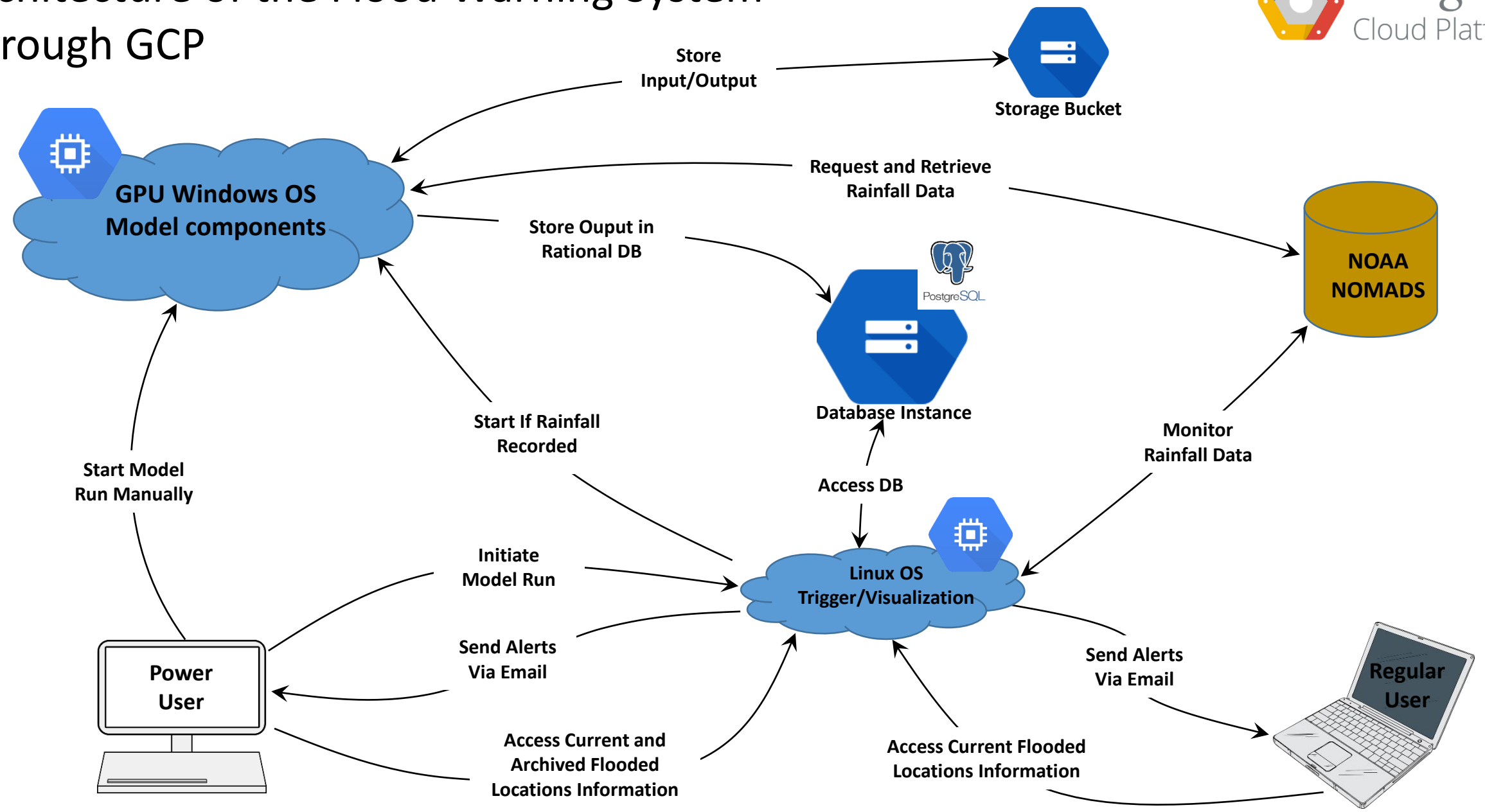
Modeling the Hague Community



Cloud-based Flood Warning System

https://vfis.uvahydroinformatics.org

Architecture of the Flood Warning System Through GCP



Model Running Speed on the GCP GPU Instance

- Google Cloud has completely customizable instances with up to 8 GPUs, either 8 GPUs of NVIDIA Tesla K80 (each $0.45/hr) or 4 GPUs of NVIDIA P100 (each $1.46/hr) (the maximum number of GPU on the current TUFLOW license is 8 GPU)

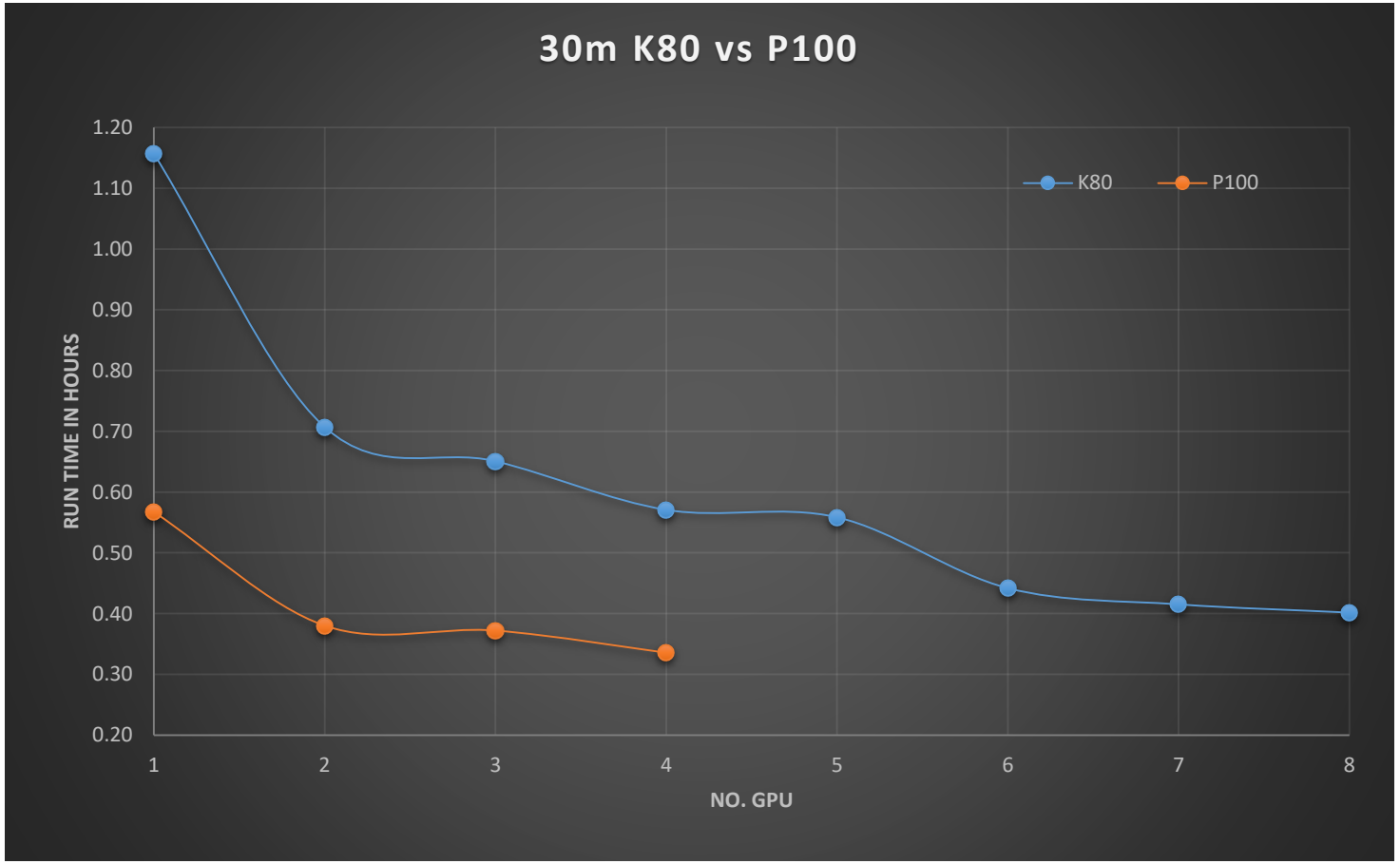
- Currently, we are using 4 GPUs of NVIDIA Tesla K80, 8 Gb of memory, and 50 Gb of storage as the basic configuration with cost of $2.078/hr ($0.278/hr without any GPUs).
  P100 is roughly 2x K80 in performance.

Processor	Cores	CUDA Cores	Frequency	GFLOPs (double)[1]	Memory	Memory B/W
NVIDIA Tesla K80 GPU (Kepler)	2 x 13 (SMX)	2 x 2,496	562 MHz	2 x 1,455	2 x 12 GB	2 x 240 GB/s
NVIDIA Tesla P100 GPU (Pascal)	56 (SM)	3,584	1,126 MHz	4,670	16 GB	720 GB/s

Grid Cell Resolution (m)	Running Time (hrs) for 19 hours modeling	CPUs	RAM
50	0.17	4	8 GB
40	0.27	4	8 GB
30	0.6	4	10 GB (+$0.009/hr)
20	1.70	4	20 GB (+$0.06/hr)
10	8.75	10 (+$0.2/hr)	65 GB (+$0.26/hr)

Model Running Speed on the GCP GPU Instance

- Focusing on 30m resolution grid cell size, the following chart shows the different in the running time using K80 vs P100 GPUs.



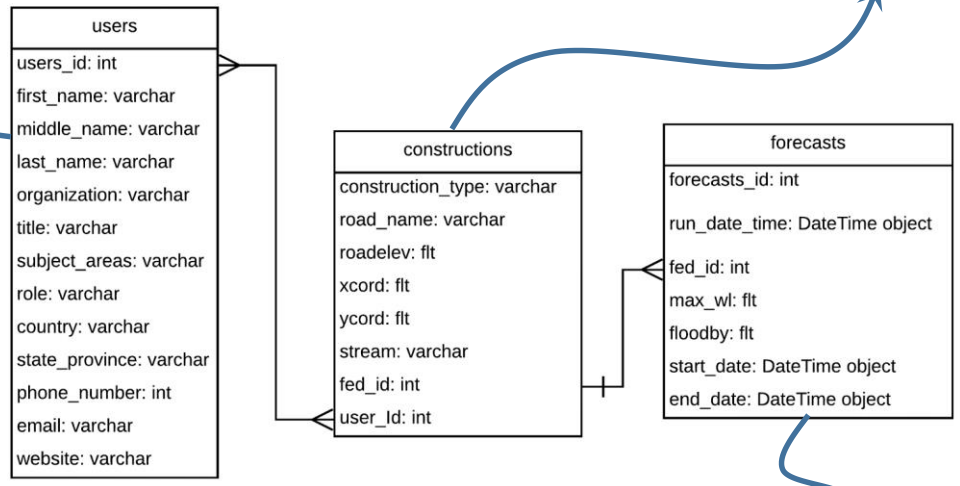
- using 4 P100 GPUs, the 20m resolution grid cell size runs in 0.91 hrs

Website SQL Database

- The database interaction is done by using SQLAlchemy

SQLAlchemy

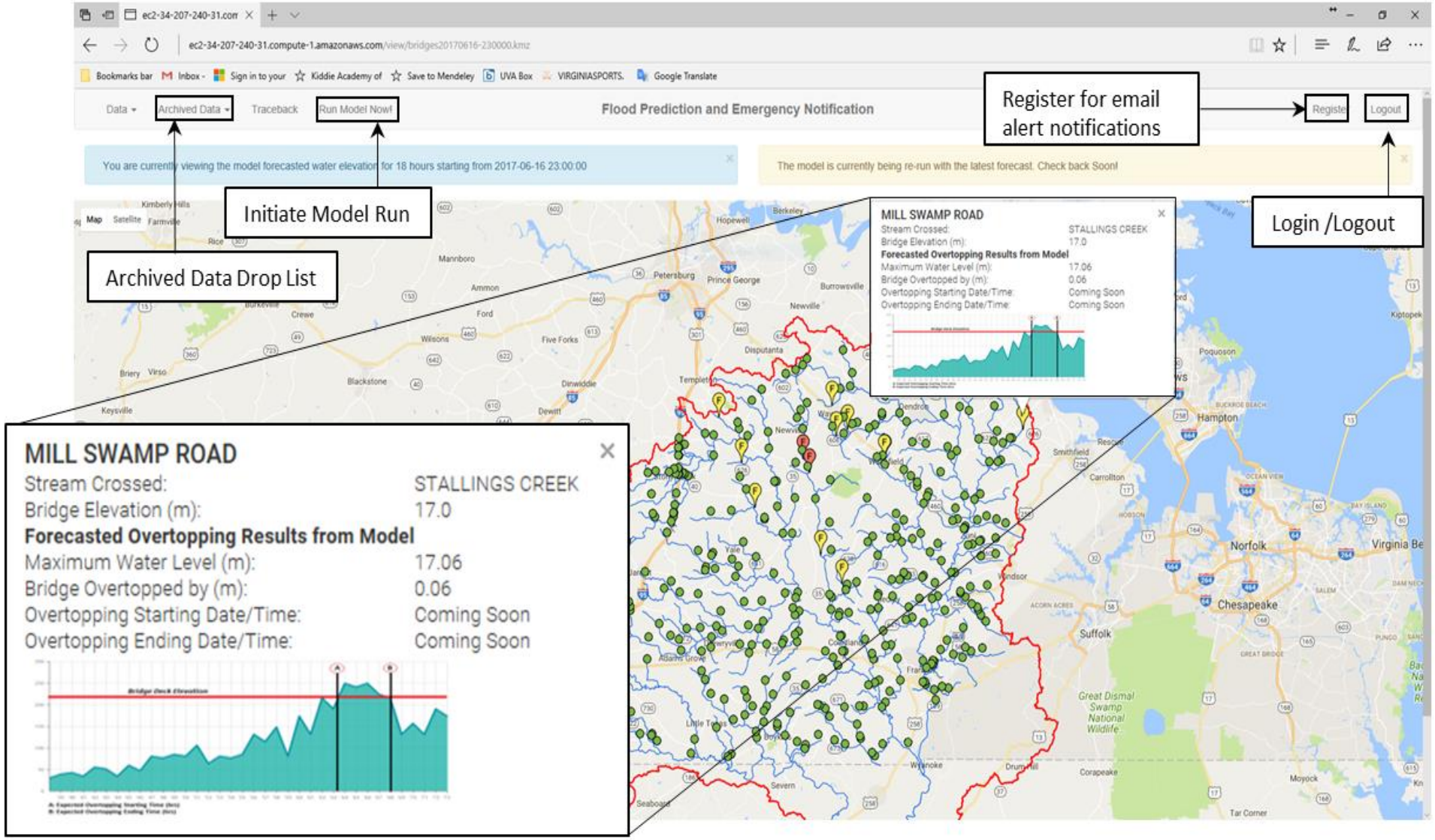
```python
class Constructions(db.Model):
    __tablename__ = 'constructions'
    fedid = db.Column(db.String, primary_key=True)
    roadname = db.Column(db.String)
    xcord = db.Column(db.Float)
    ycord = db.Column(db.Float)
    stream = db.Column(db.String)
    roadelev = db.Column(db.Float)
    forecasts = db.relationship("Forecast")
```




```python
class User(db.Model):
    __tablename__ = 'users'
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(32), index=True)
    password_hash = db.Column(db.String(64))
    first_name = db.Column(db.String())
    middle_name = db.Column(db.String())
    last_name = db.Column(db.String())
    organization = db.Column(db.String())
    title = db.Column(db.String())
    subject_areas = db.Column(db.String())
    role = db.Column(db.String())
    country = db.Column(db.String())
    state_province = db.Column(db.String())
    phone_number = db.Column(db.String())
    website = db.Column(db.String())

    def hash_password(self, password):
        self.password_hash = pwd_context.encrypt(password)

    def verify_password(self, password):
        return pwd_context.verify(password, self.password_hash)

    def as_dict(self):
       return {c.name: getattr(self, c.name) for c in self.__table__.columns}
```

```python
class Forecast(db.Model):
    __tablename__ = 'forecasts'
    id = db.Column(db.Integer, primary_key=True)
    start_date = db.Column(db.String)
    maxwl = db.Column(db.Float)
    floodedby = db.Column(db.Float)
    end_date = db.Column(db.String)
    construction_fed_id = db.Column(
        db.Integer, db.ForeignKey('constructions.fedid'))
    run_date_time = db.Column(db.String)
```

Visualization Website



https://vfis.uvahydroinformatics.org

Thank You

Jonathan L. Goodall
goodall@virginia.edu