ABSTRACT

CUEVAS LÓPEZ, TOMÁS. Prediction of Peak Water Levels during Tropical Cyclones with Deep Learning. (Under the direction of Casey Dietrich).

Storm-driven flooding is a severe hazard for coastal communities and regions. Computational models can predict the combined effects of tides, winds, and flooding due to tropical cyclones, including in real-time, but requirements for the models' runtime make it challenging to consider simulations of the full range of storm uncertainty. To address this problem, researchers have developed neural networks, trained on libraries of storm surge simulations, to predict ensembles of coastal flooding in seconds. However, existing neural networks do not consider the interaction between storm surge and astronomical tides nor storms of any duration. Moreover, they are trained on datasets tailored to represent only extreme conditions. We aim to develop a neural network to predict the peak total water levels for any storm at specific locations along the North Carolina coast.

In this research, we implemented a neural network to predict peak values for total water level (tides and storm surge) at multiple stations, considering astronomical tides and storm tracks of any duration as inputs. To create the training library, we simulated 1,813 synthetic tropical cyclones based on historical data in the North Atlantic Ocean, with a specific focus on storms that affect North Carolina. These simulations used a full-physics hydrodynamic model with variable spatial resolution of about 50 m near the coast. The outputs were downscaled to grayscale images with a higher and constant resolution of 15 m, enhancing the flood predictions by considering small-scale topographic features, and then used as training data for the neural network. The many-to-one deep learning model predicts a single peak total water level in time at multiple locations in space using time series of the offshore astronomical tide and track parameters as inputs. We used the model to make probabilistic predictions of peak total water levels for observed and perturbed tracks of several historical storms that affected North Carolina.

We showed that the neural network performed well (with errors ranging from 8 to 43 cm) in predicting peak total water levels at nine locations in North Carolina. We applied the neural network to make probabilistic predictions of peak total water levels for observed and perturbed tracks of historical storms. For each storm, the neural network predicted at nine stations for 101 storm scenarios (the true/historical storm and 100 perturbations) in less than 10 seconds. The performance for the observed historical storms was similar to those obtained in process-based simulations, but with a significant gain in computational runtime.

Prediction of Peak Water Levels during Tropical Cyclones with Deep Learning

by
Tomás Cuevas López

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Civil Engineering

Raleigh, North Carolina
2024

APPROVED BY:

DEDICATION

To my wonderful family: my wife María Ignacia and our children Trini and Gaspi.
For their love, support, and being the best motivation.

BIOGRAPHY

Tomás was born in Valdivia, Chile, in 1991. He graduated from secondary school in Concepcion, Chile, in 2009. A few months later, Chile was struck by the 2010 8.8 earthquake and tsunami. This event pushed Tomás to pursue his bachelor's in civil engineering specializing in Hydraulic engineering at the University of Chile. After graduating in 2016, he joined PRDW Consulting Port and Coastal Engineers, where he worked for almost six years doing numerical modeling of coastal processes and data science. In 2022, he joined the Coastal & Computational Hydraulics Team at North Carolina State University to work with Dr. Casey Dietrich in storm surge modeling and machine learning. Besides work and academics, Tomás enjoys hanging out with friends and family, especially spending time outside with his wife María Ignacia and children Trini and Gaspi.

ACKNOWLEDGEMENTS

There are so many people I would like to acknowledge that helped me along the way. First, I thank my wife and my children. María Ignacia, you have been the most amazing life partner I could ever have; your support through the last 18 years has been more than fundamental for me. Thank you for your love and especially for your optimism; it makes everything better. Trini and Gaspi, thank you for those smiles. You offered me a break from work daily during our playtime or bike rides, which has always been the best part.

I thank my parents, Conrado and Gabriela, for their unconditional support and encouragement throughout my life. Your dedication and hard work have been a great example to me. You taught me to persevere regardless of the challenges I face. I am forever grateful for your guidance and love. To my siblings, thank you for your support and advice and for always being available for me. I thank my family-in-law for always being there for us; your support has been very important.

To my advisor Dr. Casey Dietrich, thank you for supporting and encouraging me in our machine learning journey. Your calm and patience when the results didn't look as expected were very important to me, and I'm sure it will help me be a better scientist and engineer. I am also thankful to my committee for their expertise and advice along the way. Specifically to Dr. Lobaton for his help with the neural network development, to Dr. Anarde for always making me think about the broader impacts of my research, and to Dr. Anderson for his continued advice and for helping me to shape this work from the beginning. I thank the coastal team, undergrads, lab mates, old members, and faculties; you made my time at NCSU great.

I'm very thankful to my friends and former colleagues at PRDW; I really enjoyed the time I worked there. Especially to Javier Vasquez for the countless feedback on my reports and Benjamin Carrion for mentoring and introducing me to numerical modeling and data science. Lastly, I thank all my friends from school, undergrad, and life for their support, especially in the last part of 2023, which was very challenging for me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation

Water pushed inland by tropical cyclones (TC) is a severe hazard for coastal communities. As one example, in September 2022, the massive category-four TC Ian made landfall near Cayo Costa in southwestern Florida with wind speeds higher than 70 m/s, generating more than 4.5 m of *storm surge* – the rise of water levels due to the combined effects of wind and atmospheric pressure. These high waters pushed inland and devastated the area. The storm caused 66 deaths (Bucci et al. 2023) and damages of more than $112 billion (National Centers for Environmental Information 2023). Over the last decade, TCs have caused more than 750 deaths (Lau et al. 2022) and more than $480 billion in economic loss in the United States (Smith et al. 2020). TCs are intensifying because of global warming (Knutson et al. 2020), increasing the risks of damage due to winds, precipitation, storm surges, and waves (Emanuel 2021; Martinez 2020). Because of these severe hazards, providing accurate and timely forecasts of TC-driven coastal flooding is critical.

Starting with the formation of TCs, which later can turn into hurricanes, the National Hurricane Center (NHC) forecasts the storm's track, intensity, and size, among other parameters, at least every six hours (Cangialosi 2023). To predict the storm's effects on coastal water levels and flooding, scientists and engineers use the forecasts issued by the NHC as inputs to coastal hydrodynamic models. One of the most common tools for modeling storm surge is the ADvanced CIRCulation model (ADCIRC; Luettich, Westerink, and Scheffner 1992; Westerink et al. 2008). ADCIRC solves the Generalized Wave-Continuity Equation for the water levels and the vertically-integrated momentum equations for the current velocities (Luettich and Westerink 2004). The equations are solved on an unstructured triangular mesh, thus allowing a finer resolution where the coastal environment is more complex. ADCIRC outputs water levels and currents at every mesh vertex, allowing the generation of maps. It has been validated for storms in the U.S. Atlantic and Gulf coasts (Cialone et al. 2017; Vijayan et al. 2023). An example of ADCIRC for real-time forecasting is the automated ADCIRC Surge Guidance System (ASGS; Fleming et al. 2008). The ASGS produces a local storm surge prediction for each new NHC advisory (every 3 to 6 hours) to inform emergency management agencies and decision-makers.

On a high-performance computing system (HPC) with multiple CPUs, an ADCIRC TC simulation typically takes between 1 and 2 hours. The runtime is determined largely by the spatial resolution applied to represent the coastal environment. Higher resolution gives more accurate results, but it also makes the model more time-consuming. Given this, a trade-off exists between spatial resolution or accuracy and the model runtime (S. C. Hagen, Westerink, and Kolar 2000). Numerical modelers must balance the model producing accurate enough predictions without compromising the runtime. As coastal areas become more urbanized (Saginor and Ge 2017), modelers must represent a more complex coastal environment. In addition, as the population is increasing (ibid.), more people would need to be evacuated if needed. Thus, local storm surge predictions are expected at finer resolutions and in shorter time frames.

This tradeoff can make it prohibitive to study the inherent uncertainties of the forecast tropical cyclone parameters. Using a probabilistic framework — simulation of several perturbations of the same storm — to account for the forecast uncertainties may help in having more informative predictions and reducing the chances of under- or over-estimation of the storm surge (Di Liberto et al. 2011; Flowerdew, Horsburgh, and Mylne 2009; Mel and Lionello 2014; William J Pringle et al. 2023). For example, the ASGS simulates the most probable or consensus forecast from the NHC and a few perturbations. Nevertheless, probabilistic frameworks require many simulations to characterize the mean surge statistics and account for uncertainties (Donald T Resio et al. 2007). In storm surge real-time forecasting, addressing the uncertainty of the hurricane's evolution with physics-based models such as ADCIRC is challenging. The NHC uses another tool called Sea, Lake and Overland Surges from Hurricanes (SLOSH; Jelesnianski, J. Chen, and Shafer 1992) in their Probabilistic Tropical Storm Surge system (P-surge; Taylor and Glahn 2008). SLOSH makes several physical simplifications (Joyce et al. 2019) and utilizes numerical domains of limited extension to increase time efficiency.

An alternative to include the uncertainty of the track parameters is using surrogate models, which have become extremely popular for storm surge prediction in the last decade (Jia and Li 2012; Kyprioti, Adeli, et al. 2021; Kyprioti, Taflanidis, et al. 2021; Lee et al. 2021; Plumlee et al. 2021; William J Pringle et al. 2023; Taflanidis et al. 2013; Tiggeloven et al.

2021). Storm surge surrogate models are trained on large datasets and then, using data-driven approximations, estimate water levels for any new storm. Surrogate models can be soft-divided into two categories: statistical learning and deep learning models. In statistical methods, each instance of the dependent variables, e.g. storm surge, is described by a set of features or attributes (independent variables), e.g. wind speed, pressure, or eye position. In contrast, in deep learning, the model can find the essential features within the storm parameters that better explain each instance of storm surge.

Statistical learning models have been used widely (e.g. Jia and Li 2012; Kyprioti, Taflanidis, et al. 2021; Plumlee et al. 2021; William J Pringle et al. 2023; Taflanidis et al. 2013), showing promising results in the prediction of storm surge. Kyprioti, Irwin, et al. (2023) implemented a surrogate model using Gaussian Processes (GP; Rasmussen and C. K. Williams 2006) to predict time series of storm surge over a few thousand computation points, using the Coastal Hazards System-North Atlantic (CHS-NA) database (Nadal-Caraballo and Melby 2014) as training data. The North Atlantic Comprehensive Coastal Study (NACCS) database contains storm surge simulations for 1,050 synthetic tropical cyclones affecting the U.S. coast from Virginia to Maine, computed with a process-based hydrodynamic model. However, the simulations do not consider astronomical tides, and the synthetic storms are generated by perturbing parameters within realistic ranges, which has the effect that subsets of storms are similar, e.g. on parallel tracks. Even so, statistical learning methods can predict storm surges with high accuracy and have benefits such as high interpretability, relatively easy implementation, low risk of overfitting, and computationally cheap to train.

Deep learning (DL) models have become popular in coastal engineering applications in recent years. Neural networks (NN) are algorithms inspired by how the human brain recognizes, classifies, and predicts features within data. Compared to statistical learning models, NN are more computationally expensive to train and harder to implement and interpret. Nevertheless, they are better for learning complex non-linear relationships, handling high-dimensional data, and generalizing to unseen data (Goodfellow, Bengio, and Courville 2016). Some studies have predicted storm surge from time series of atmospheric variables at fixed stations a certain number of hours ahead of time (Bai and Xu 2022; Chao and Young 2022; K. Chen et al. 2022; S. Kim, Matsumi, et al. 2016; S. Kim, Pan, and Mase 2019; Tiggeloven et al. 2021; Wang et al. 2021) with good results, but lacking geospatial information about the storm track. Tiggeloven et al. (2021) implemented different NN architectures to predict storm surge, without astronomical tide, globally. This work used storm surge observations from 736 tide stations worldwide as the dependent variable and data from the ERA5 reanalysis (Hersbach et al. 2020), with a spatial resolution of 0.25°, as the independent variable. They extracted atmospheric variables at a centered box of 1.25°around the station. The box size prevents a prediction of the full storm, and atmospheric reanalysis are often unable to resolve the TC inner core of the storm (Hodges, Cobb, and Vidale 2017; Schenkel and Hart 2012).

Other authors followed a hybrid approach, using the hurricane's parameters as the independent variable and physics-based models' output as dependent variables (e.g. Ehsan Adeli et al. 2022; Ayyad, Hajj, and Marsooli 2022; Hashemi et al. 2016; S.-W. Kim et al. 2015; Lee et al. 2021; Pachev et al. 2023). The advantage of the hybrid approach is that the independent variable considers the full geospatial information of the storm, whereas the disadvantage is that many physics-based model simulations are needed for training. There are a few large libraries of storm surge ADCIRC simulations. Some examples are FEMA (2021), Owensby et al. (2020), Nadal-Caraballo and Melby (2014), and Dawson et al. (2021). They all focus on high-return period (extreme) storms for risk analysis, so they may not represent the average conditions. Lee et al. (2021) implemented a 1D Convolutional Neural Network (1D CNN; LeCun, Bengio, et al. 1995) combined with K-means clustering (Lloyd 1982; MacQueen et al. 1967) and Principal Component Analysis (Wold, Esbensen, and Geladi 1987) to predict peak storm surge (without astronomical tides) in the Chesapeake Bay using NACCS as training data. The authors used a portion of the track to have sequences of the same length as the only input. Pachev et al. (2023) implemented a two-step NN to classify the output point as wet or dry and then to predict the peak total water level for the wet points. This study considers the topography and bathymetry as inputs, but they removed the storm track's temporal component by considering the max, mean, and min values of the wind vectors, wind magnitude, and pressure. Both studies showed good agreement between the DL-predicted and ADCIRC values and highlighted that their models can predict in order of seconds. This time frame allows to study the track uncertainty by predicting the storm surge from several perturbations of the storm.

However, all the cited studies that combine deep learning techniques and physic-based storm surge models were trained with datasets developed to characterize the flooding for extreme events, because there is a limited number of large libraries of storm surge simulations, and because these libraries have been developed to study the risk associated with high-return period (extreme) inundation scenarios. The training data, especially when it does not consider the full temporal information of the storms, limits the usage of this model for emergency management because the NN must predict for any storm (extreme or not) of any duration. As far as we know, there has not been an attempt to implement a NN to predict peak total water level for storms of any duration, considering the astronomical tide as input, and trained with a dataset tailored to represent the extreme and average conditions of tropical cyclones. Considering the non-linear interaction between the astronomical tide and the storm surge is a key component because the effects of the peak storm surge happening at high tide can differ drastically from those at low tide. Additionally, as far as we know, there has not been an attempt to predict 2D maps of peak total water levels by using deep learning. For this purpose, a library would need to be developed with flood maps as raster images, and the NN would need to be designed to be extensible to handle images.

In this research, we propose a neural network for forecasting peak total water levels from TC-driven coastal flooding for North Carolina, considering astronomical tides and storms of any duration. The architecture is based on 1D CNN layers to handle the time evolution of the storm. We used a synthetic database of tropical cyclones based on historical data to generate the training library, with 10,000 years of information in the North Atlantic Ocean and more than 100,000 storm tracks (Bloemendaal et al. 2020). We reduced the number of storms by selecting a subset that may affect the North Carolina coast, and then we simulated the 1,813 storms using ADCIRC with random astronomical tides. Using an in-house and open-source Python software called Kalpana (Kalpana 2018), we downscaled the peak total water level to a higher and constant resolution (Rucker et al. 2021). Then, the library of high-resolution maps of TC-driven coastal flooding was used to train the proposed neural network to predict peak total water levels at any location. The NN needs only a time series of the following parameters: distance from the storm's eye to the prediction point, maximum wind speed, minimum pressure, radius to maximum wind speeds, and offshore astronomical tides.

First, we show the results of a neural network capable of simultaneously making predictions at multiple points, hereafter referred to as *multitask model*. The root mean squared error (RMSE) for unseen storms ranged from 8 to 43 cm for all tested locations. We used the trained model to make probabilistic predictions of the peak total water level for multiple perturbations of historic storm tracks. Finally, using simpler versions of the neural network, we assess its sensitivity to hyperparameters, data augmentation, and inputs. Although this model predicts the peak water levels at specific coastal locations, it has been designed to be extensible to flood maps in future work.

1.2 Objectives

The general objective of this work is to develop a neural network capable of predicting peak total water levels (tides and storm surge) from hurricane-driven coastal flooding with minimal computational time. The specific objectives are listed below:

- Define a subset of storms within a dataset created with a probabilistic model that represents the extreme and average conditions of tropical cyclones in North Carolina.
- Generate a training library of high- and constant-resolution peak surge maps using ADCIRC and a downscaling technique.
- Train a deep learning algorithm with data extracted from the high and constant-resolution peak surge maps library to make predictions for unseen storms.
- Use the neural network to predict historic storms' peak total water levels and compare the results with observations.
- Implement a probabilistic prediction framework by predicting peak total water level for several perturbations of recent historic storms.
- Assess the neural network sensitivity to hyperparameters, data augmentation, and inputs.

1.3 Funding Acknowledgment

This material is based upon work supported by the U.S. Department of Homeland Security under Grant Award Number 2015-ST-061-ND0001-01. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security. This research was also supported by the Chilean Agency of Research and Development (ANID).

Chapter 2

Background

In this research, we implemented a neural network to predict the peak total water levels (tides and storm surge) of storm-driven coastal flooding. The neural network relies on datasets and software from other sources, which are described in this chapter. We also describe our efforts to modernize a Python software, which we will use later to increase the resolution of the ADCIRC outputs in our NN development.

2.1 Study area

Coastal North Carolina (NC) is characterized by large estuaries with multiple river inputs, wide sounds with shallow bathymetry, and limited connections to the open ocean (Figure 2.1). The system of barrier islands of North Carolina, known as the Outer Banks, is the first line of protection from extreme sea conditions for coastal communities and ecosystems. Beaches and dunes on barrier islands dissipate wave energy before the water reaches the mainland. The barrier islands are low-lying land formations with elevations below 6 m (NAVD88). The ground surface elevation does not increase quickly at inland regions, so the entire state coast is prone to significant hurricane-driven flooding. In general, NC coastal configuration is very complex so that flooding drivers may vary in different areas.

Coastal NC is described well by geospatial datasets, and this research will utilize a digital elevation model (DEM) with topography and bathymetry with a 15 m resolution of the North Carolina coast (Figure 2.1). The data was obtained from the NOAA Digital Coast (Oceanic and Administration 2020), and it is part of the Coastal National Elevation Database (CoNED USGS 2020). The raster is projected on the NAD83(2011) / UTM zone 17N coordinate system. The unit of the elevation data is meters. The horizontal resolution of the DEM is 1 m, but in this research it was resampled to 15 m to reduce the file size and speed up the computations.

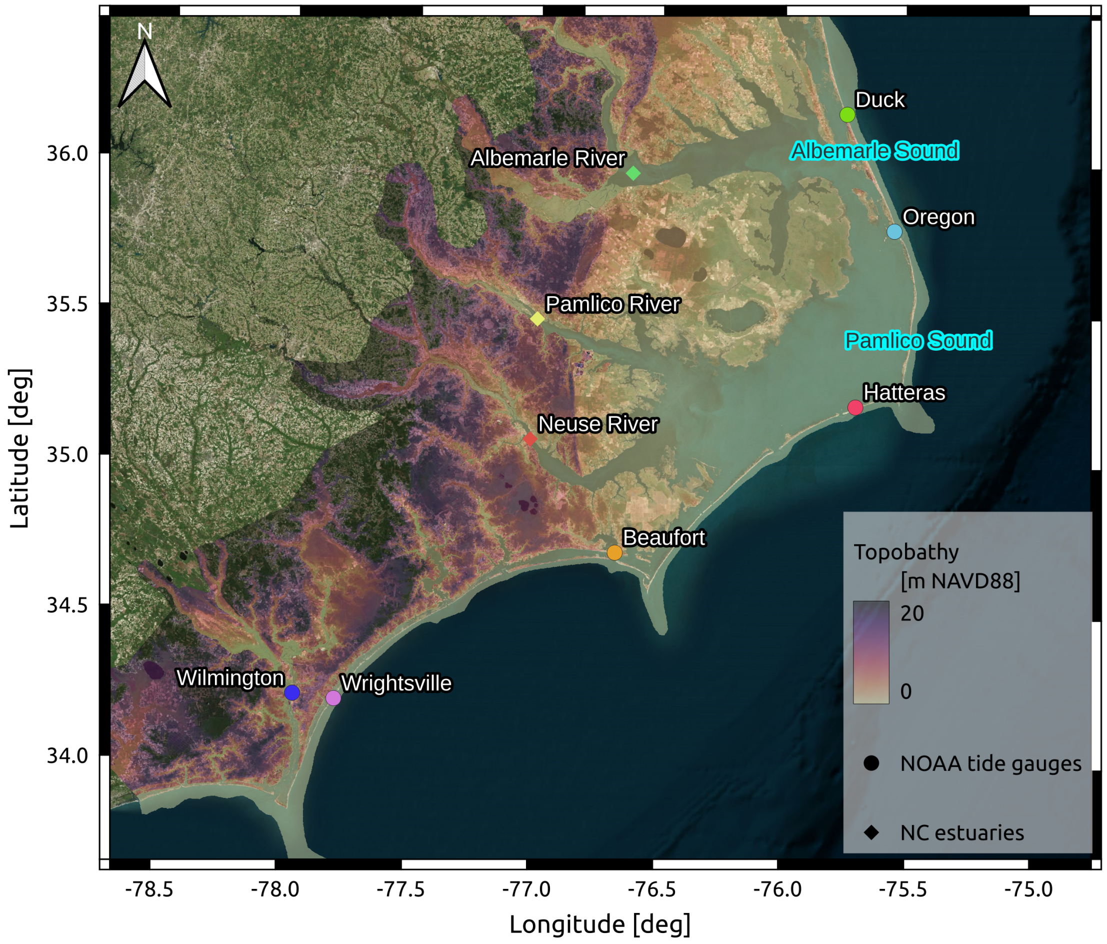


Figure 2.1: Digital elevation model (DEM) of coastal North Carolina. The background digital elevation model shows the coastal topography and bathymetry. Main NC rivers are marked with diamonds, and NOAA tide gauges are shown with dots. These nine locations will be used for making predictions with the neural network.

The population and urbanization of coastal areas are increasing (Saginor and Ge 2017), and most of the NC coast is low-lying and prone to flooding, even outside TCs (Kopp et al. 2015). Sea level rise (North Carolina Emergency Management 2014) and land subsidence (Johnston et al. 2021) increase the risk of flooding, highlighting the importance of accurate and real-time flood forecasting systems.

2.2 Tropical cyclones

2.2.1 Recent storms in North Carolina

Several tropical cyclones have affected the NC coast in the last 15 years. Irene (2011) formed as a tropical wave on August 15 to become a tropical depression on August 20 as it approached the Lesser Antilles. Just after its initial landfall in Puerto Rico, Irene reached a category-1 status. The storm intensified to category 3 as it moved toward the northwest, but then started to weaken to a strong category 1 storm as it approached the North Carolina coast. The NC coast started to be affected on August 26, and made landfall on August 27 near Cape Lookout. Along the North Carolina coast, the storm surge ranged between 0.8 to 2.1 m, with a maximum water level of 2.2 m observed at Oregon Inlet tide gauge (National Oceanic and Atmospheric Administration 2011).

Arthur (2014) formed as a low-pressure system front of South Carolina on 28 June, then it moved south to become more organized over the warm waters of east Florida. It reached hurricane status on July 3 near Savannah, Georgia, and continued to strengthen, reaching its peak intensity of 85 kt (category 1) on July 4 just off the NC coast. Arthur produced significant storm surge flooding along the NC coast. At Oregon Inlet, a storm surge of 1.4 m was observed, whilst surges between 0.6 and 0.8 m were recorded at Wrightsville Beach, Beaufort, Cape Hatteras, and Duck tide gauges (Berg 2015).

Matthew (2016) started as a strong tropical wave below 10°N latitude off the western coast of Africa on September 23. It became a tropical storm on September 28 northwards of Barbados in the Lesser Antilles of the West Indies (Stewart 2017). Matthew rapidly intensified to a category-5 storm on the Saffir-Simpson (Saffir 1974) between September 30 and October 1. It first made landfall near Haiti on October 4 with a category-4 status and then near the West End on Grand Bahama Island on October 7 as category 3. It covered the distance between Florida and North Carolina, running shore-parallel. The flooding in the NC coast varied significantly, with the highest levels recorded on the Outer Banks' sound side. The highest recorded water level at Cape Hatteras reached 1.8 m (Stewart 2017). In some Barrier Islands, the storm caused geomorphological impacts such as severe dune erosion, formation of temporary inlets, and lowering and shoreward extension (Backstrom, Loureiro, and Eulie 2022).

Florence (2018) developed off the west coast of Africa on 31 August 2018, and reached a category-4 status on September 4. The shore-perpendicular track made landfall as a category-1 storm near Wrightsville, NC, and moved slowly remaining near the coast for more than a day. Florence produced more localized impacts, specifically in southeast North Carolina. It was a major rainfall event, producing nearly 80 cm of rainfall in some areas. At landfall, Florence caused a devastating storm surge on the open coast and within the sounds and estuaries of NC. On the open coast, the estimated surge was between 1.5 and 2.5 m (Stewart and Berg 2019), and between 2.5 to 3.5 m along the Neuse River inside the sounds.

Dorian (2019) formed as a tropical depression on August 24 near Barbados in the Winward Islands and became a tropical storm later that same day. Dorian made landfall for the first time over Barbados on August 27, and then, after passing over the U.S. Virgin Islands, it started strengthening on its way to the Atlantic Ocean. It reached a category-3 status on August 30 and continued its rapid intensification to make landfall as a category-5 storm in the Bahamas. Dorian is the strongest hurricane in modern records to make landfall in the Bahamas. After landfall, it weakened to re-strengthen back to category 3 over the Gulfstream. On September 5th, a NOAA buoy offshore of South Carolina recorded a pressure of 959.2 mb. It made landfall on U.S. continental territory over Cape Haterras on September 6 with a category-2 status. Catastrophic storm surge flooding occurred in northwestern Bahamas; observations suggest water levels reached 2 m above ground level on the western end of Grand Bahama Island. North of North Carolina, inundation levels of 0.6 m above ground occurred (L A Avila and A. B. Hagen 2020).

Isaias (2020) became a tropical storm on July 30, when it was located southwards of Puerto Rico. It made landfall for the first time in the Dominican Republic and then strengthened into a hurricane when its center was located offshore of the northern coast of Hispaniola. Isaias continued to move northwestward to make landfall for the second time on 31 July as the center crossed Great Inagua Island in the southeastern Bahamas and then passed just west of the central Bahamas. It reached a peak category-1 intensity located south-southeast of Nassau. On August 2, Isaias made its closest approach to Florida, but then it turned toward the north-northeast to make the fourth and final landfall near Ocean Isle Beach, NC, on August 4 with a category-1 status. Isaias produced a storm surge of 1 to 2 m above the ground level along the southernmost NC coast. The highest inundation was observed in Brunswick County, NC. Two sensors recorded water levels around 2.5 m NAVD88 at Ocean Isle Beach and Oak Island, NC (A Latto and Berg 2021).

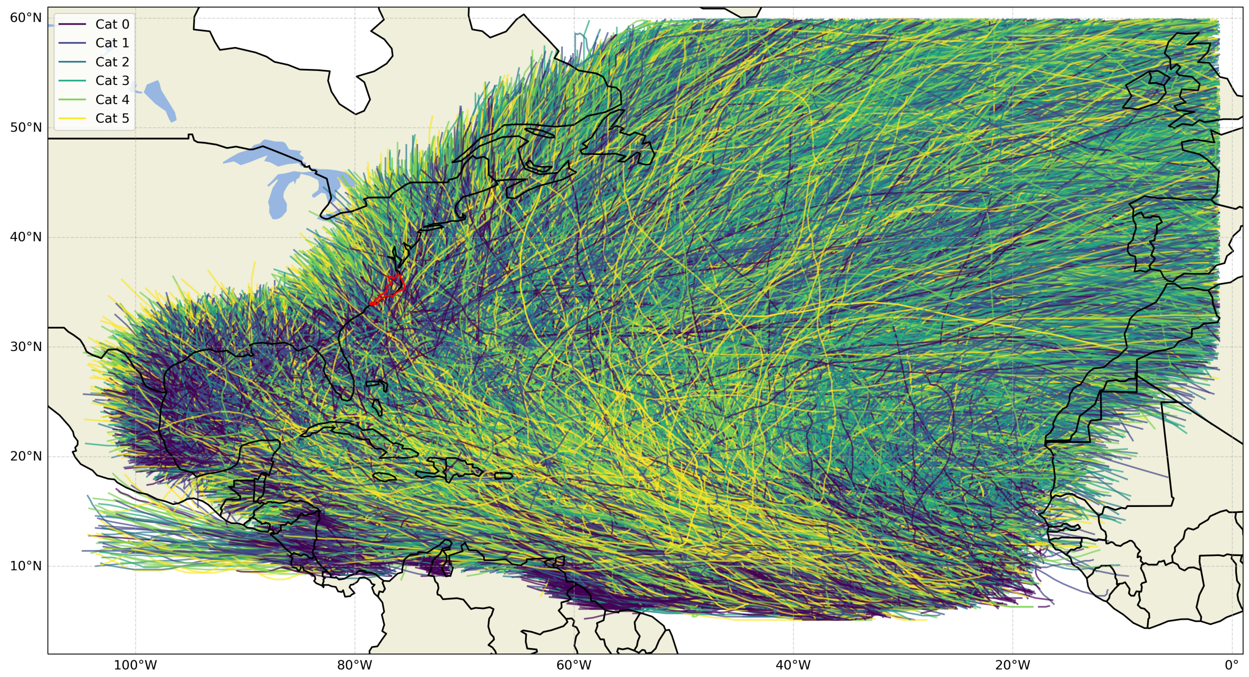


Figure 2.2: Synthetic storms from the STORM database and based on the IBTrACS for the North Atlantic Basin. The color of each track indicates the maximum category in the Saffir-Simpson scale. The plot contains more than 100,000 synthetic tracks. The red polygon on the U.S. Atlantic coast shows coastal NC.

2.2.2 Dataset of synthetic tropical cyclones

Although these historical storms were devastating to NC, the record of historic storms is less than 40 years long, so it does not have enough storms to develop a large library ($\geq$ 1,000) of storm surge simulations. The results of many storm surge simulations are required to train a neural network to predict peak total water levels. Datasets of synthetic tracks, created from historical data, are preferred to develop large libraries of storm surge simulations because they represent the past conditions of tropical cyclones with thousands of tracks.

The global synthetic tropical cyclone (TC) dataset created with the Synthetic Tropical cyclOne geneRation Model (STORM), based on the International Best Track Archive for Climate Stewardship (Knapp, Diamond, et al. 2018; Knapp, Kruk, et al. 2010), extends the 38 years of real TCs to 10,000 years of synthetic TC activity. The algorithm follows a probabilistic approach, and it is composed of three main steps. First, it samples the origin of each real TC (IBTrACS, in this case), extracting information about their coordinates and month. After that, STORM creates the synthetic storm by including perturbations to the genesis coordinates. Finally, the algorithm adds the storm's minimum pressure, maximum wind speed, and radius to maximum winds. The latter steps are based on the probability distribution of the corresponding parameter on the real baseline dataset. Each storm (Figure 2.2) has information about the following parameters: year, month, TC number, time step (3-hourly), basin, latitude, longitude, minimum pressure, maximum wind speed, radius to maximum winds, category (Saffir 1974), landfall, and distance to land. For this study, we focused on the North Atlantic Ocean basin because our goal is to predict storm surge in North Carolina. The dataset provides enough information to characterize the possible scenarios of storm surge in North Carolina generated by TCs.

2.2.3 Historical storms and their perturbations

In addition to synthetic TCs, we will also use the NN to predict for historical TCs, both for their true effects and their possible perturbations. We used the CLIMADA Python package (Aznar-Siguan et al. 2023) to download the parameters of six recent historical storms (Table 2.1) that affected the North Carolina coast and to create 100 perturbations of each. CLIMADA is a software package and modeling framework designed for assessing and managing the impacts of natural disasters and climate-related risks. It is designed to analyze and simulate the potential consequences of various hazards such as tropical cyclones, floods, earthquakes, heat waves, and other extreme weather events.

To create the perturbed tracks, CLIMADA uses the random walk method defined by Kleppek et al. (2008). A one-dimensional Brownian motion (Wiener process) is applied to create a slightly perturbed track 'guided' by the original track (Gettelman et al. 2017). Random uniform values perturb the genesis point of the track in longitude and latitude. Then, each segment between two consecutive points is perturbed in direction and distance, with a correlation in time (Kropf et al. 2022). The wind speed and central pressure drop change only when the perturbation makes landfall; a decay value depending on the TC category is applied.

The historical and perturbed tracks will be input into the neural network to make probabilistic predictions, and the results will compared to observed peak water levels.

2.3 Models

2.3.1 ADCIRC

ADvanced CIRCulation (ADCIRC) is a computational framework that predicts coastal circulation and transport in 2D and 3D. It uses the finite element numerical method in space over unstructured meshes and the finite difference method in time (Luettich, Westerink,

Table 2.1: Historical storms considered in the probabilistic prediction application of the trained deep learning model.

Name	Year	Max category
Irene	2011	3
Arthur	2014	2
Matthew	2016	5
Florence	2018	4
Dorian	2019	5
Isaias	2020	1

and Scheffner 1992). For the water surface elevation, ADCIRC solves the depth-integrated continuity equation in the Generalized Wave-Continuity Equation (GWCE) form, while the velocities are obtained from the 2D depth-integrated or 3D momentum equations. It applies the continuous-Galerkin finite-element method with linear C_0 triangular elements to discretize and solve the above-mentioned equations, allowing a variable resolution so that a complex area can be represented with higher resolution. ADCIRC can be run in parallel with multiple CPUs on HPC systems; it is efficient and scales well (Dietrich, Zijlema, et al. 2011; Tanaka et al. 2011). The model can be forced with astronomical tides at the offshore boundary, wind and pressure fields to represent tropical cyclones, and river discharge upstream in estuaries. Dietrich, Trahan, et al. (2012), Hope et al. (2013), and Sebastian et al. (2014) have shown extensive validations of this model along the U.S. Gulf and Atlantic coasts.

ADCIRC simulations can have various applications, including risk analysis. ADCIRC has been used to develop extensive libraries of storm surge simulations, such as the FEMA flood map program (FEMA 2021), the South Atlantic Coast Comprehensive Study (Owensby et al. 2020, SACCS), and the North Atlantic Coast Comprehensive Study (U.S. Army Corps of Engineers 2015, NACCS). These libraries have been used to characterize tropical cyclone-drive flooding for high return period events. Another example is the automated real-time ADCIRC Surge Guidance system (ASGS; Fleming et al. 2008). This system was developed to provide local storm surge predictions for planning decisions that must be made when storms approach the coast and make landfall. Hindcast simulations is another possible application of ADCIRC. In this case, observed storm tracks are used to force the model. The goal of hindcast simulations can be model validation or to better understand the coastal hazards produced by the storm. Thomas et al. (2022) showed validations of high water marks and water level time series for Matthew in 2016 over the entire South Atlantic Bight, and for Florence in 2018 specifically in North Carolina. Extending the floodplain coverage for large storms like Matthew in 2016 and refining the mesh resolution over the floodplains, barrier islands or estuaries improves the model performance (Thomas et al. 2022).

For the atmospheric forcing to ADCIRC, the wind and pressure fields can be obtained from reanalysis products or computed from the storm track using parametric models. In this study, we will use ADCIRC to simulate a subset of tracks of the dataset obtained using STORM based on the IBTrACS for the North Atlantic Basin. To include the atmospheric forcing of the TC, we will use the parametric Holland symmetric vortex model (Holland 1980) because the wind and pressure fields can be generated using the information available in the dataset of synthetic TCs.

2.3.2 Unstructured mesh

A key input to ADCIRC is the unstructured mesh to represent the coastal environment. One example is the mesh from the Hurricane Surge On-Demand Forecast System (HSOFS; Riverside Technology and AECOM 2015), developed for doing surge and tide predictions throughout the U.S. Gulf and Atlantic coasts. HSOFS resolution near the coastline is around 500 m, which is insufficient to represent a complex coastal environment such as the NC barrier island system. A NC-specific example is the NC9 mesh (Blanton and Luettich 2008) developed for the FEMA flood insurance studies (FIS). NC9 is composed of 1,230,430 triangular elements and 622 946 nodes. Inland areas are only resolved in NC, concentrating 90% of the elements around the state, but the mesh extends across the Western North Atlantic Ocean, the Gulf of Mexico, and the Caribbean Sea. Element sizes range from 15 to 25 m near Oregon Inlet and Cape Fear River and then increase to 100 to 200 m on barrier islands and other inlets. The resolution increases in the Pamlico and Albemarle Sounds up to 1500 m in open water and then from 10 to 100 km in the deep Atlantic Ocean. Blanton and Luettich (2010) and Cyriac et al. (2018) showed extensive validations of ADCIRC simulations using the NC9 mesh. Then Thomas et al. (2022) showed the benefits of increasing the floodplain coverage. The authors merged five meshes created for the FIS in NC, South Carolina, Georgia and northeast Florida, central Florida, and south Florida to simulate hurricanes Mathew (2016) and Florence (2018). The resulting SAB mesh had 5,584,241 vertices and 11,066,018 elements, resolving the floodplains from Florida to North Carolina.

Although those meshes are well-validated for storm simulations in NC, they were developed from older datasets for coastal topography and bathymetry. For this study, we adapted the SABv3-60m mesh developed by Woodruff (2023). This mesh was designed using OceanMesh2D (Roberts, W J Pringle, and Westerink 2019) and is a 'forecast-grade' mesh, i.e. the computational burden of running a storm surge simulation on this mesh is similar to that of using a mesh used in real-time forecasting but is computationally expensive and includes high-resolution inland elements outside our study area. It has about 7 million elements and 3.5 million vertices, with the smallest resolution of about 60 m, which is adequate to resolve most of the coastal topographic and bathymetric features, such as barrier islands, estuaries, major man-made channels, and inlets. This mesh will be adapted to the storm simulations in this research, as described later in Subsection 3.2.1.

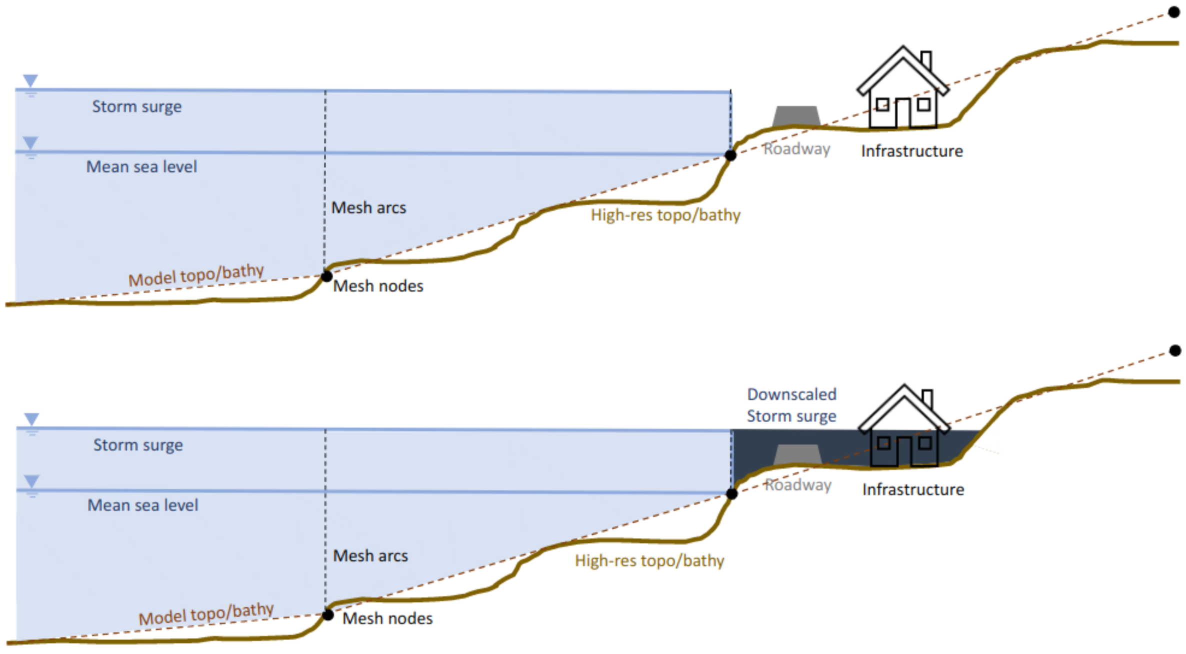
2.3.3 Kalpana

During hurricanes, emergency managers expect storm surge predictions on small scales (e.g. meters) that cannot be resolved efficiently with process-based models like ADCIRC. One option to increase the resolution of the predictions without increasing the model runtime is using a downscaling method as a post-processing step. Kalpana is a set of Python scripts to visualize ADCIRC outputs using geospatial vector formats or Google Earth KMZ files (Cyriac et al. 2018) and to downscale ADCIRC maximum water elevations (Rucker et al. 2021).

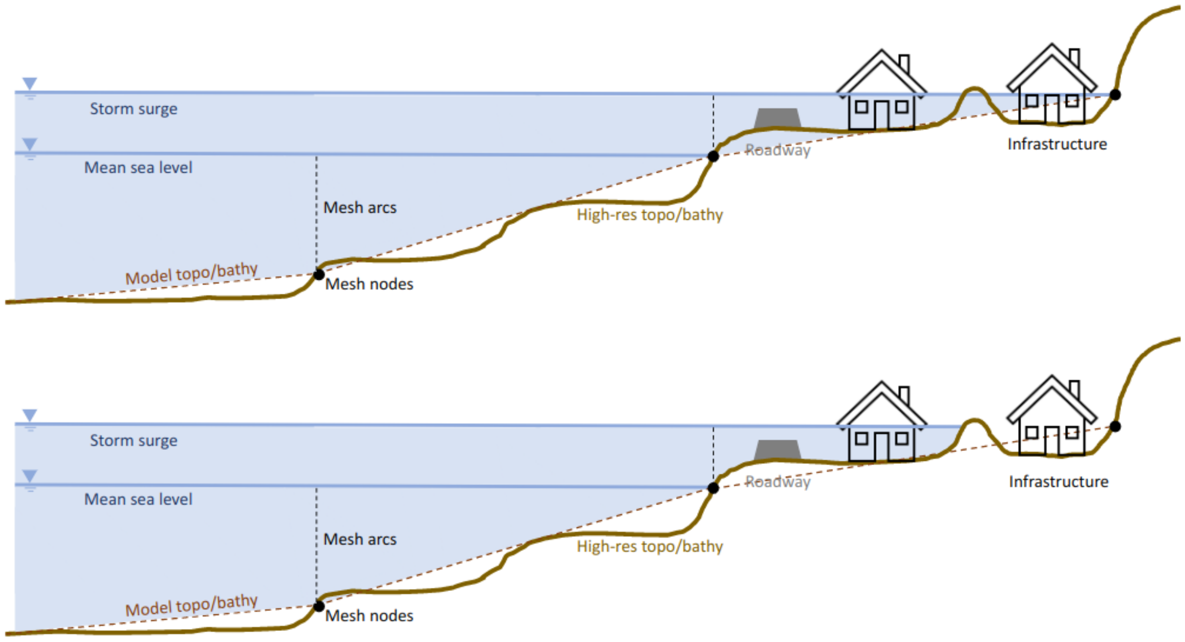
Kalpana has two methods for increasing the resolution or downscaling the peak water level: static and head loss. In this study, we focused on the static method because it is faster, requires less input to be applied, and produces accurate outputs (ibid.). The static method interpolates the maximum water level from the unstructured mesh to a higher-resolution and structured raster and then expands it horizontally until it intersects the ground surface (Figure 2.3a). The downscaling increases the resolution, allowing the storm surge to inundate areas with ground elevations below the water level. Kalpana can incorporate small-scale topographic features to reduce the extension of the peak storm surge (Figure 2.3b). The downscaling includes the topographic feature and shrinks the storm surge until the water level intersects it.

Kalpana starts by transforming the outputs of ADCIRC into a geospatial vector file. For this, the user must define a set of water level contours for Kalpana to bin the ADCIRC results into this range. Then, each area defined by two contiguous contours is transformed into a polygon. This process is done in serial, so the code takes longer when more water level contours are requested. The geospatial vector file is then rasterized with the same resolution, extent, and projection of the user-provided downscaling raster. To simplify the explanation, we call this the ADCIRC raster. The downscaling raster input must be a topography and bathymetry DEM of the site of interest with a higher resolution than the ADCIRC mesh.

For the static downscaling method, Kalpana applies nearest neighbor interpolation to fill the dry (null) cells of the ADCIRC raster with the values of the closest non-dry cells. The interpolation outputs two rasters: one is the interpolated water levels, and the other with



(a) Expansion of the inundation



(b) Reduction of the inundation

Figure 2.3: Schematized cross-shore profile of a portion of the model domain, with examples of downscaling using Kalpana's static method: (a) The water level is extended horizontally until it intersects the ground level. The upper panel shows the extent of the water level before Kalpana, and the bottom panel shows the inundation extent after downscaling. (b) The water level extent is reduced since a small topographic feature higher than the water level is included. The upper panel shows the extent of the water level before Kalpana, and the bottom panel shows the inundation extent after downscaling.

the distance from each pixel to the nearest non-dry cell. Both rasters have the same extent, resolution, and projection.

Then Kalpana removes all the filled cells on the interpolated ADCIRC raster, where the corresponding cell, on the distance raster, is larger than a user-defined input. Kalpana also corrects the inundation by removing the water from the cells where the water level is below the ground surface elevation. This is done by comparing the downscaling raster with the distanced-corrected ADCIRC raster. The last two steps may create isolated groups of cells or clumps not hydraulically connected to the ocean, which is not physically possible. Kalpana removes all the hydraulically disconnected clumps with an area below a user-defined threshold to correct this. The two user-defined inputs mentioned in this paragraph are unrelated to the mesh resolution and can vary from site to site. Thus, the benefit of using Kalpana is that it allows small-scale topographic and bathymetric features to be considered without increasing the model run time.

2.3.4 Upgrades to Kalpana

The static downscaling method and the visualization tools were modified for this study to make the code faster and compatible with Python 3 (Van Rossum and Drake 2009) and modern programming standards. Our contributions to the software modernization are:

A. *Convert ADCIRC mesh and maximum water levels to vector format.* Maximum water levels are binned, plotted as closed contours, and then saved as a geospatial vector format. In previous versions of Kalpana, each closed contour or polygon was computed and written in serial, but in the updated version, the computation of the polygons was parallelized, and the writing was optimized using GeoDataFrames. The mesh is also exported as a GIS shapefile, with the element's representative size as an attribute.

B. *Initialize the GRASS GIS environment* (GRASS Development Team 2022). In the previous version, the downscaling DEMs were loaded in serial, but the new version supports loading DEMs in parallel and has more flexibility regarding the coordinate system of the input files. Kalpana also loads and rasterizes the ADCIRC maximum water level shapefile, and in the new version, it also loads and rasterizes the mesh shapefile.

C. *Static grow.* The spatial interpolation used in the new version follows the same procedure as the previous version, but there are changes to how the growing distance is limited. Kalpana's original version removed all the cells interpolated beyond a threshold. This threshold was a user-defined input, and it was not related to the mesh resolution.

We incorporated the mesh resolution in the new version, removing all cells interpolated beyond n triangular elements, where n is a user-defined input. For this research, we used $n = 1$, i.e. cells are expanded horizontally by no more than the size of one triangular element.

D. *Grown raster post-processing.* The original and the new versions of Kalpana remove the wet cells where the ground surface is above the water level. Nevertheless, how they deal with the hydraulically disconnected cells changed drastically. Kalpana's original version removed all the clumps with an area smaller than a given threshold. This threshold is a user-defined input, and it is not related to the mesh resolution. Kalpana's new version removes the disconnected clumps if they don't intersect the non-interpolated ADCIRC maximum water level shapefile wet area.

In addition to these steps for the downscaling process, Kalpana was modernized by updating the code to Python 3.9 (or higher). It was written in Python 2.7, which was deprecated in January 2022. The old version of the Kalpana was a single Python script with nearly 2,000 lines of code. The new version is highly modularized, so now Kalpana is a set of Python modules with traceable functions. In the previous version, only GRASS GIS version 7.8 was supported; now, GRASS GIS versions 8.2 and later are supported. Lastly, the new version of Kalpana can be used in Windows and Linux, whereas before it was only supported on Linux. Kalpana is publicly available on the North Carolina State University Coastal & Computational Hydraulics Team GitHub (Kalpana 2023). The repository has a detailed instructional guide to install and run the software. The steps for the downscaling process are described in further detail in the remainder of this subsection.

A. Convert ADCIRC mesh and maximum water levels to vector format

The first step in the downscaling process is to transform the ADCIRC-predicted peak water levels from netCDF format to a GIS vector data file. To convert the ADCIRC output to a geospatial vector format, we make a contour plot of the maximum water level. The contours are then converted to polygons and then exported to a GIS shapefile.

The ADCIRC maximum water level output has a single value for each mesh vertex, i.e. a two-dimensional cloud of points. It has no time dimension because ADCIRC only saves the maximum water level for each mesh vertex. We computed isolines by binning the water level on a set of user-defined values. Then, we formatted the area defined by two consecutive isocurves as polygons. This is an iterative process. The script creates the polygon formed for each pair of adjacent contours. We parallelized this iteration, so the time to produce the maximum water level isolines is less dependent on the number of levels specified by the user.

After that, we grouped all the polygons using a two-dimensional tabular format, storing the geographic information of geometry objects. Kalpana accepts more user-defined inputs to change the vertical unit (from meters to feet, or vice-versa) of the water level isolines and to change the data's coordinate reference system (CRS). Finally, the data is saved as a GIS shapefile.

The ADCIRC mesh is also exported and saved as a GIS shapefile. The mesh elements are constructed from the information in the ADCIRC output and formatted in tabular format. Each mesh element is stored as an individual polygon (triangle). We also computed extra information like each element's area, perimeter, and minimum face length.

B. Initialize the GRASS GIS environment

Most geographic operations within Kalpana use GRASS GIS (GRASS Development Team 2022). It is necessary to create a GRASS location, which is a directory that houses all geospatial data and information associated with a specific geographic area. It represents this area with its own coordinate system, map projection, and related data, enabling users to perform geospatial analyses efficiently and manipulate geographic information within that area. This is a key step in Kalpana's workflow because all geospatial layers loaded into the location share geographical information, such as resolution and extension in case of rasters.

The first step to run the static downscaling method is to choose the downscaling DEM. This raster must have topography and a finer resolution than the ADCIRC mesh. It can have bathymetric data, but it is not mandatory. The second step is to load the downscaling DEM to the GRASS location. Kalpana accepts DEMs in multiple formats, even non-georeferenced files. The user can also provide multiple DEMs. In this case, Kalpana processes all the files and patches them together as a single raster in parallel to accelerate the process.

The next step is to load the ADCIRC maximum water level shapefile into the GRASS location. This file is rasterized, interpolating the maximum water level into the same extent and resolution of the downscaling DEM. We call it ADCIRC raster to simplify the explanation. Finally, we load the mesh shapefile and interpolate the element's representative size (perimeter / 3) into the same extent and resolution of the downscaling DEM. We call it mesh raster to simplify the explanation. Each cell of the mesh raster contains the representative size of the mesh element it lies in (Figure 2.4). In this example, the background DEM has a constant resolution of 15 m. The color of the mesh shows the representative size of each element. For example, all the DEM cells inside an element with a size of 100 m have a cell value of 100 m. The same applies to larger elements. This DEM is used in the next downscaling step.

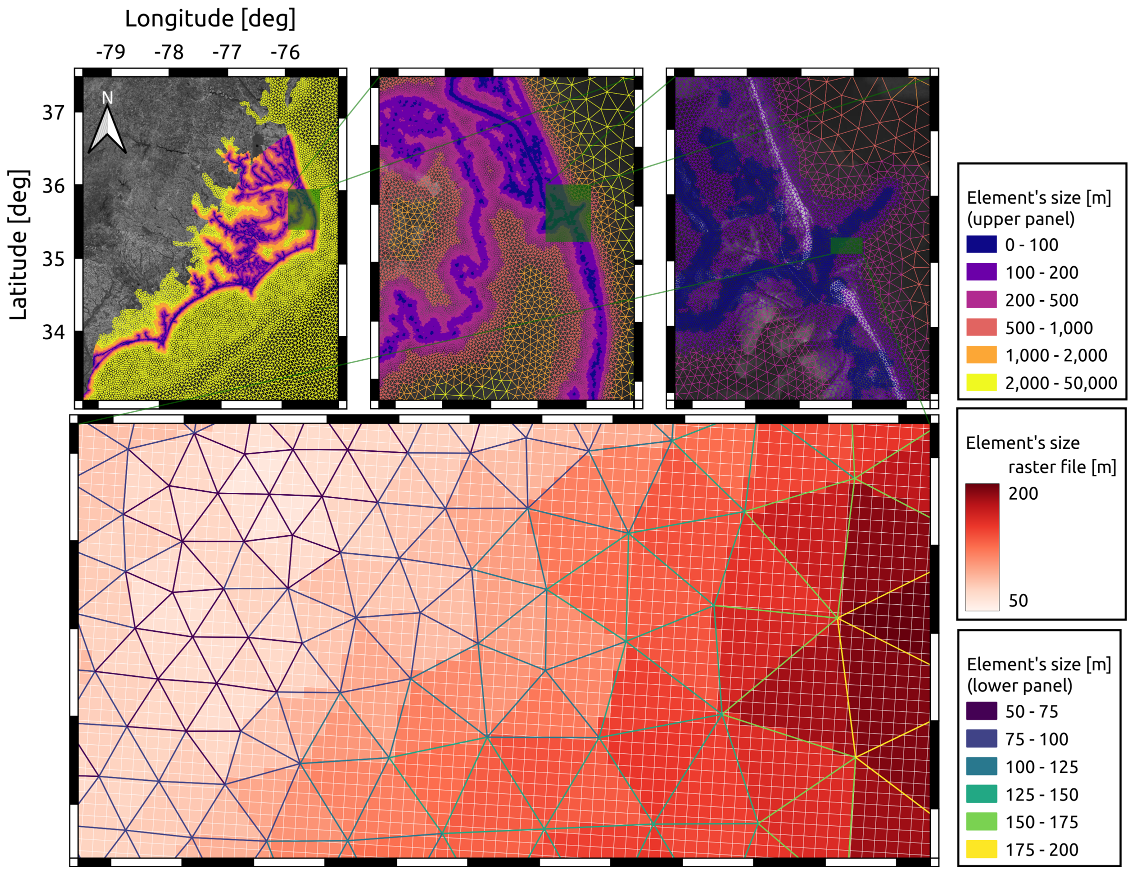


Figure 2.4: The upper panels show the SABv5 mesh with successive zoom-in views (left/wide to right/zoom) to NC and Oregon Inlet, with contour colors based on the mesh resolution. The lower panel shows the ADCIRC mesh in vector format over the raster with the elements' representative size, again with the contour colors based on the mesh resolution. Each raster cell is outlined in white. Each raster cell has a value of 1/3 of the element's perimeter it lies in. All the cells inside one element have the same value.

C. Static grow

For this step of the downscaling process, Kalpana applies nearest neighbor interpolation using the *r.grow.distance* (Larson and Clements 2008) GRASS GIS tool to spatially expand the maximum water level on the ADCIRC raster. The algorithm fills all the null DEM cells (dry cells) beyond the flooding extent with the nearest non-null cell value. It also creates a new channel in the raster — a new raster attached in the third dimension — that has the distance to the nearest non-null cell used to fill each missing value. We will refer to the filled cells in the peak surge raster as grown cells, to the distance to the nearest non-null cell as growing distance, and to the new raster as distance raster.

Then, to limit the flooding expansion, Kalpana compares the distance raster cells' value

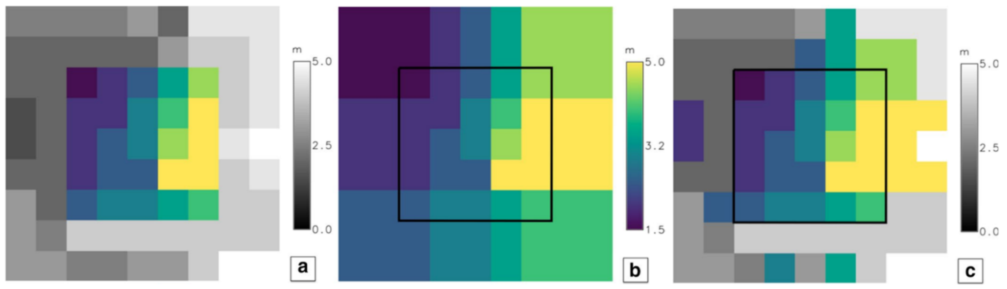


Figure 2.5: Conceptual explanation of the *r.grow* algorithm. Panel (a) shows the ADCIRC maximum water level raster (color) at the same resolution as the underlying topo/bathy DEM (greyscale). Panel (b) shows the expanded raster using the Grass GIS *r.grow* algorithm. Panel (c) shows the result of correcting the elevation with the underlying topo/bathy DEM. The grown cells are retained only if their water level exceeds the ground elevation. Adapted from Rucker et al. (2021).

to the mesh raster. It keeps only the grown cells where the growing distance is smaller than the mesh raster cells' value times a user-defined factor, which we defined as the growing factor. This step assumes that the gradient in the element's size is small for neighbor elements. Assuming a growing factor of unity, Kalpana removes all the grown cells with a growing distance larger than the mesh raster cells' value, i.e. the maximum spatial expansion corresponds to one mesh element.

The limit in the flooding expansion is one of the key differences in the new version of Kalpana. In previous versions, all cells expanded beyond a given threshold were removed. It is difficult to guess a correct value for this threshold beforehand without comparing Kalpana's outputs with measurements or an ADCIRC simulation with a much higher-resolution mesh. Furthermore, this threshold can vary within the ADCIRC domain. In the updated version, we decided to relate the threshold to the mesh resolution so it directly relates to a simulation parameter and can be spatially variable.

D. Grown raster post-processing

The first post-processing step is removing the grown cells with the ground elevation above the maximum water level. Some cells with a ground level above the water level may be wet for two reasons: (1) the mesh resolution is not fine enough to represent the bathymetric/topographic features, and (2) the expansion of the peak surge described in the previous step only takes into account horizontal distance, so water can be 'placed' in high elevation cells. To remove these 'fake' wet cells, Kalpana compares the water level of each cell with the ground level in the downscaling DEM and keeps only the cells where the water level is above the ground level (Figure 2.5).

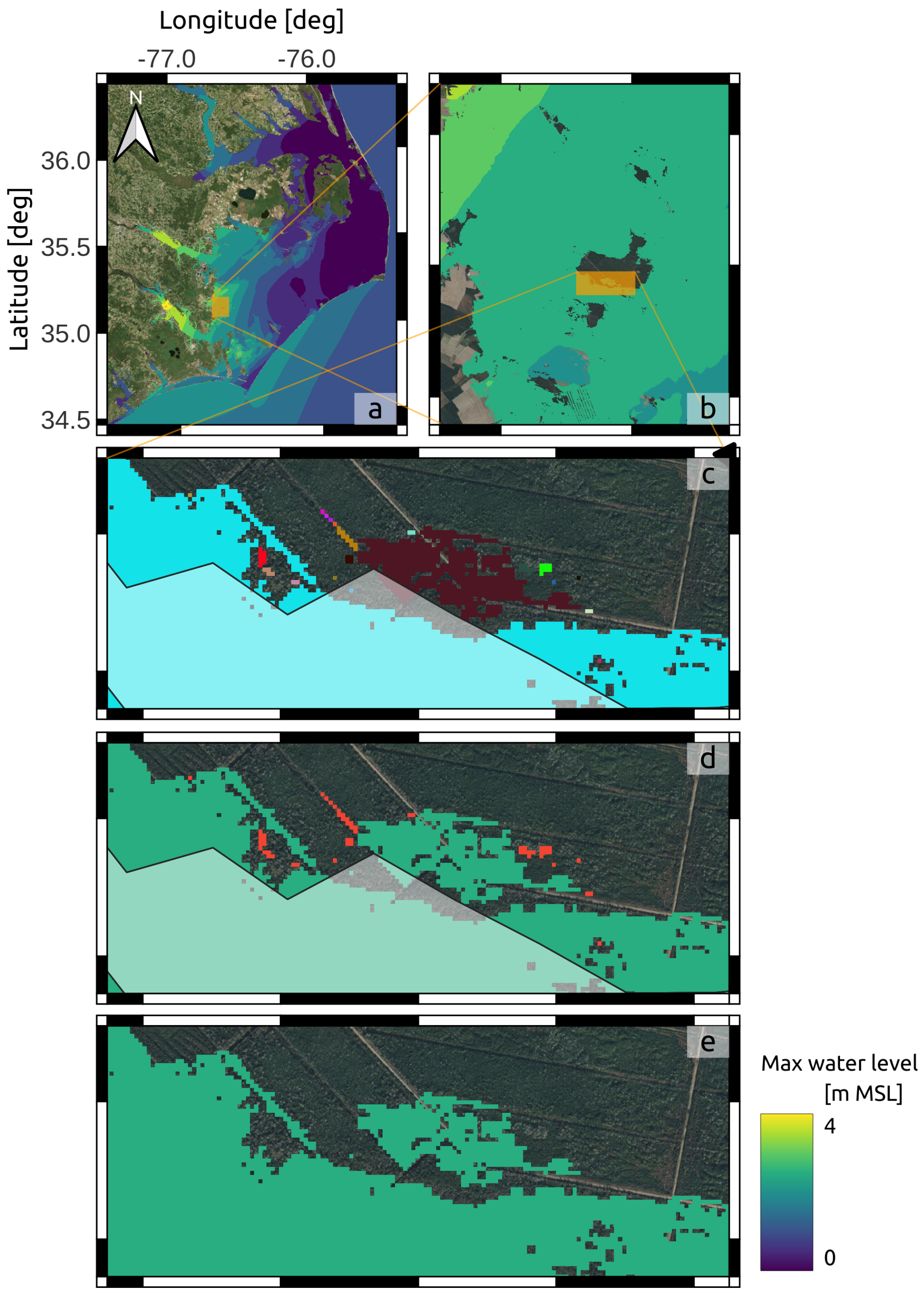


Figure 2.6: Workflow to remove hydraulically isolated cells. Panel (a) shows an ADCIRC maximum water level for a portion of the NC coast. Panel (b) shows a zoom-in view of a part of Pamlico county. Panel (c) shows the extent of the ADCIRC maximum water level in shaded white and the different clumps in randomly assigned colors. The large cyan clump corresponds to the Neuse River, which is hydraulically connected to the ocean. Panel (d) shows all clumps intersecting the ADCIRC maximum water level extent in green and the disconnected clumps that do not intersect in red. Finally, panel (e) shows the downscaled maximum water level raster corrected by hydraulic connectivity.

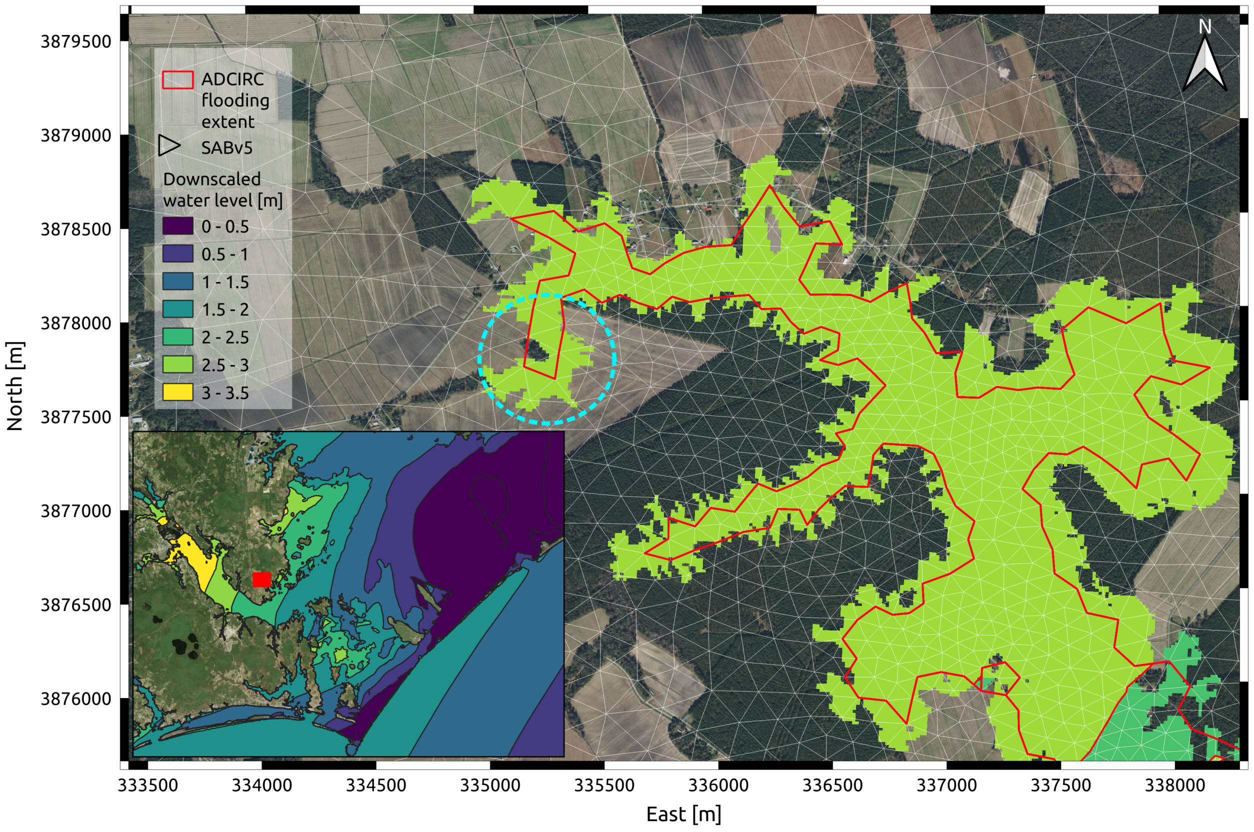


Figure 2.7: Final result of the downscaling process with Kalpana's static method. The smaller inset panel shows the maximum water level for a portion of the North Carolina Coast. The main panel shows a close-up view of the Neuse River to appreciate the downscaling effect. The red line delineates the wet/dry boundary of the raw ADCIRC peak surge. The maximum water level is expanded beyond the ADCIRC flooding extent shown in red, with a much higher resolution than the mesh. The dashed cyan circle points to a dry area inside the ADCIRC flooding extent corrected by ground surface elevation.

Removing the 'fake' wet cells may create clumps or groups of isolated cells not hydraulically connected to the ocean. In its previous version, Kalpana removed all the hydraulically disconnected clumps with areas smaller than a given threshold. This threshold was hard-coded and unrelated to any physical or numerical parameter of the ADCIRC simulation. Kalpana's new version finds all disconnected groups of cells or clumps, overlays them with the flooding extent of the ADCIRC maximum water level shapefile, and removes the ones that do not intersect (Figure 2.6). This step is another major difference from the previous version of Kalpana. It removes a hard-coded input and allows Kalpana to downscale the maximum water level in the presence of flow-blocking structures such as levees.

The final output of Kalpana's static method is a raster flood map at the same resolution of the underlying DEM (Figure 2.7). Kalpana increases the resolution outside the ADCIRC wet/dry boundary by inundating portions of originally dry elements. Also, in some parts inside the ADCIRC wet/dry boundary, Kalpana removes some initially wet cells with a ground level higher than the maximum water level.

2.4 Deep learning

Artificial intelligence (AI) is a field of computer science focused on creating systems that can perform tasks that require human-like intelligence. It involves developing algorithms and models that enable machines to analyze data, recognize patterns, and make decisions or predictions. AI encompasses various subfields like machine learning (ML), where algorithms learn from data, and deep learning (DL), which uses neural networks to simulate how the human brain processes information.

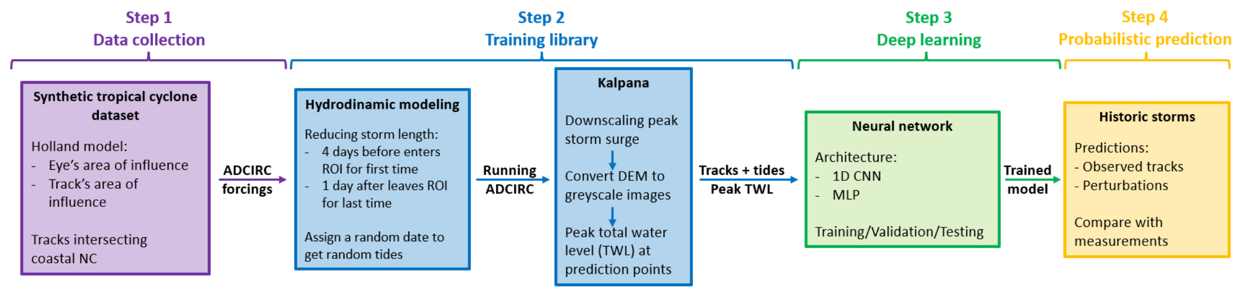
Deep learning focuses on developing and utilizing artificial neural networks inspired by the structure and functioning of the human brain's neural connections (Géron 2022). These neural networks contain multiple layers of interconnected nodes, known as neurons, arranged hierarchically. Each layer processes and extracts different levels of abstraction from the input data, enabling the network to learn patterns, features, and representations through exposure to large amounts of data. The earlier efforts in developing deep learning systems go back to the 1940s, but the technique didn't gain much popularity then due to its lack of ability to solve complex problems. In 2006, Hinton, Osindero, and Teh (2006) published a deep neural network to recognize handwritten digits that performed with incredible precision ($> 98\%$)(Géron 2022), calling the technique 'Deep Learning'. After this, deep learning gained popularity and has been applied in many fields, storm surge one of them.

There are multiple production-ready Python deep learning frameworks. For this research, we used TensorFlow (Martín Abadi et al. 2015), a widely used open-source framework for deep learning and machine learning. Developed by Google Brain, TensorFlow offers a versatile platform that simplifies the creation, deployment, and scaling of machine learning models, particularly deep neural networks.

Chapter 3

Methods

This chapter presents the implementation of a neural network to predict the peak total water level of storm-driven coastal flooding using the storm parameters and the astronomical tides as predictors. Figure 3.1 outlines the workflow to develop the deep learning model. It has four main steps: (1) selecting a dataset of synthetic tropical cyclones that could affect our study area; (2) generating a training library of process-based model simulations; (3) building, training, validating, and testing the neural network; and (4) testing the performance of the neural network with historical storms. We explain each step in detail in the following subsections.



Step 1
Data collection
Step 2
Training library
Step 3
Deep learning
Step 4
Probabilistic prediction
Synthetic tropical cyclone dataset
Holland model:
- Eye's area of influence
- Track's area of influence
Tracks intersecting coastal NC
ADCIRC forcings
Hydrodinamic modeling
Reducing storm length:
- 4 days before enters ROI for first time
- 1 day after leaves ROI for last time
Assign a random date to get random tides
Running ADCIRC
Kalpana
Downscaling peak storm surge
Convert DEM to greyscale images
Peak total water level (TWL) at prediction points
Tracks + tides
Peak TWL
Neural network
Architecture:
- 1D CNN
- MLP
Training/Validation/Testing
Trained model
Historic storms
Predictions:
- Observed tracks
- Perturbations
Compare with measurements

Figure 3.1: Schematic of the workflow followed to develop the deep learning model.

3.1 Step 1: Data collection

In this section, we describe the process of defining a dataset of synthetic tropical cyclones that represents the average and extreme conditions for North Carolina (NC). This dataset will be used in the next step as storm forcings to process-based model simulations to develop a training library.

3.1.1 Selection of impactful storms for NC

The STORM dataset of synthetic tropical cyclones has 10,000 years of data with more than 100,000 storms (Bloemendaal et al. 2020). Process-based model simulations of these storms would be both impractical, given the high requirements for computing resources, and unnecessary, given that many storms do not pass close to or cause flooding of the NC coast. Instead, we identified a subset of storms that have the potential to cause flooding in our study area.

We assume that, if a storm's hurricane-strength winds (stronger than 33 m/s) ever affect the NC coast, then that storm is 'impactful' – it has a reasonable likelihood of generating flooding. Thus each storm must be analyzed to determine if its hurricane-strength wind field ever overlaps with our study area. We created a polygon covering the main rivers, estuaries, and sounds of NC. Then we searched for storms that overlapped this polygon at any point in their track history. For each storm and time snap, we used the parametric Holland symmetric vortex model (Holland 1980, Eq. 3.1) to examine the approximate size of the storm's wind field. We found the distance to the 33 m/s isotach, which is the lower bound of the category 1 in the Saffir-Simpson scale (Saffir 1974). We solved Eq. 3.1:

$$V(r) = V_{\max} \sqrt{\left(\frac{R_{\max}}{r}\right)^{\beta} \exp\left(1 - \left(\frac{R_{\max}}{r}\right)^{\beta}\right)} \tag{3.1}$$

to find the radius r corresponding to a wind speed $V(r) = 33$ m/s, by using $R_{\max}$ as the radius to maximum wind speed $V_{\max}$ for the specific time snap, and $\beta = 1$. We defined the storm's 'area of influence' as a circle generated by the storm as the center and r as the radius. After finding the storm's area of influence for each time snap of its track, we computed the convex hull of all circles. We considered as impactful the storms with an area of influence intersecting coastal NC, for a total of 3,626 storms.

This process can be illustrated for a single storm from the dataset (Figure 3.2). This storm is initiated in the Atlantic, tracks just north of Puerto Rico and Hispaniola and east of the Bahamas, and then moves parallel to the U.S. Atlantic coast. The storm does not make landfall in the continental U.S. but it is a large (category 5) storm, and thus its wind field does overlap with the North Carolina coast. Because of this overlap, we consider this storm to be impactful and include it to be simulated for our training library.

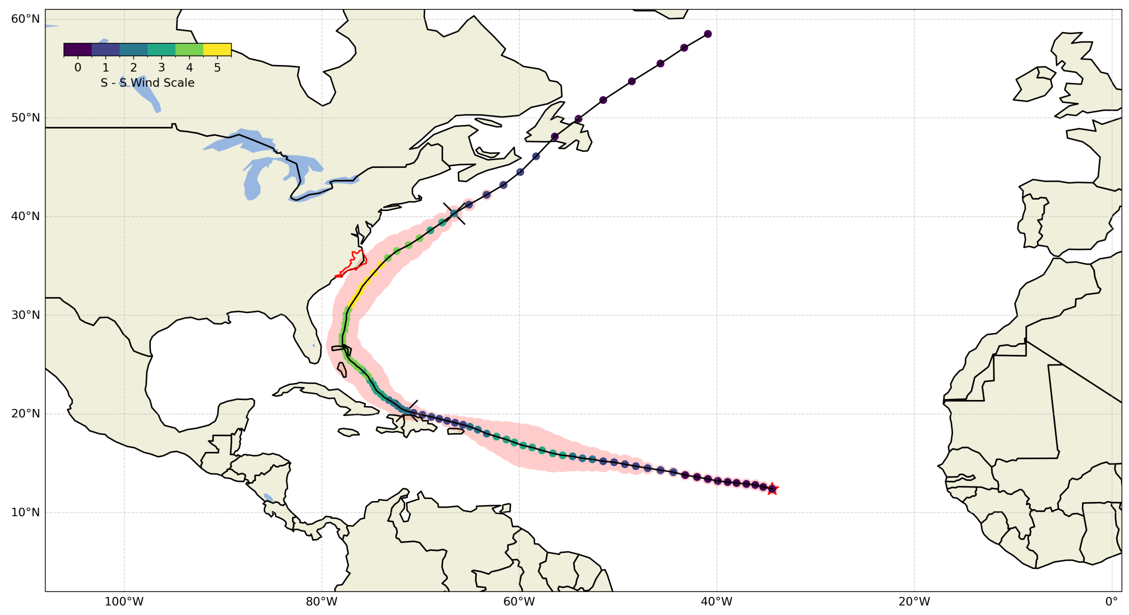


Figure 3.2: Definition of a storm's area of influence using the parametric Holland symmetric vortex model. Storm center positions are colored to indicate the category of the TC, the shaded red area corresponds to the area of influence, the red polygon represents the coastal North Carolina area, the red star shows the genesis location, and the black lines perpendicular to the track enclose the simulated portion.

3.1.2 Postprocessing of impactful storms for NC

To reduce the amount of computations required with ADCIRC, we removed and shortened storms from this data set. We did not consider storms weaker than hurricane strength (Saffir 1974) in their closest position to our study area. When the storms are close to landfall, they typically experience a decay in their wind speed. The category distribution is shifted and skewed toward weak storms in the tracks' closest position to the NC coast. We removed 1,792 storms with wind speeds below 33 m/s in their closest distance to NC. We also discarded storms shorter than 24 hours.

We also shortened storms to simulate only the portion in which they were impactful to the North Carolina coast. Before the shortening, the storms' duration ranged from 1.38 to 28.25 days, with an average and standard deviation of 6.76 and 3.72 days, respectively. We assumed that the peak storm surge would occur within the period defined by four days before the storm's area of influence overlaps coastal NC for the first time and one day after it overlaps coastal NC for the last time. Some storms could be shortened significantly. For example (Figure 3.2), several days of the storm can be discarded when the storm is far away in the Atlantic Ocean. The track-perpendicular black line over Haiti shows the beginning of

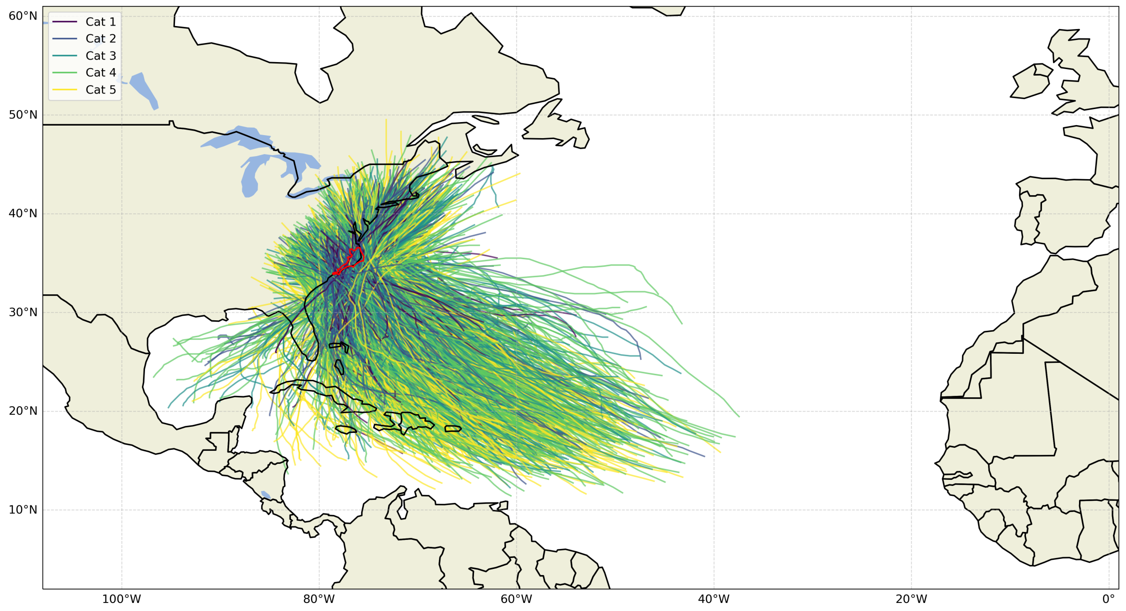


Figure 3.3: Subset of shortened impactful storms to be simulated. The line colors indicate the maximum category of the TC, and the red polygon represents the North Carolina coast.

the track portion to be simulated, and the black line in front of Massachusetts shows the end. For this storm, the duration was decreased from 6.75 to 4.88 days.

We identified a subset of 1,813 storms as impactful in North Carolina (Figure 3.3). These storms have durations ranging from 1.25 (30 hours) to 9.8 days (235 hours). This assumption allowed us to decrease the number of hours of storm to simulate by 37%, from 883,395 to 554,760 hours (Figure 3.4h).

The dataset has a large variance with many dissimilar storms. The minimum value for maximum wind speed is slightly below 33 m/s, whereas the maximum value for maximum wind speed is 83.5 m/s. The track with the largest maximum wind speed is storm 15; the peak wind speed and the lowest pressure ($\approx$ 840 hPa) occur when the storm is at its closest distance to the NC coast. This storm is an outlier because typically TCs weaken when they approach the coast; the maximum wind speeds tend to decrease (Figures 3.4a and 3.4e), thus the category distribution shifts to lower values (Figures 3.4c and 3.4g) and the pressure increases (Figure 3.4b and 3.4f). Most of the considered tracks are landfilling storms. Bypassing storms are underrepresented, being only 5% of the selected tracks. Considerable variability exists in the genesis location; some originated in the Gulf of Mexico and the Caribbean, but most are formed in the mid-Atlantic. Most storms approach NC from the southwest.

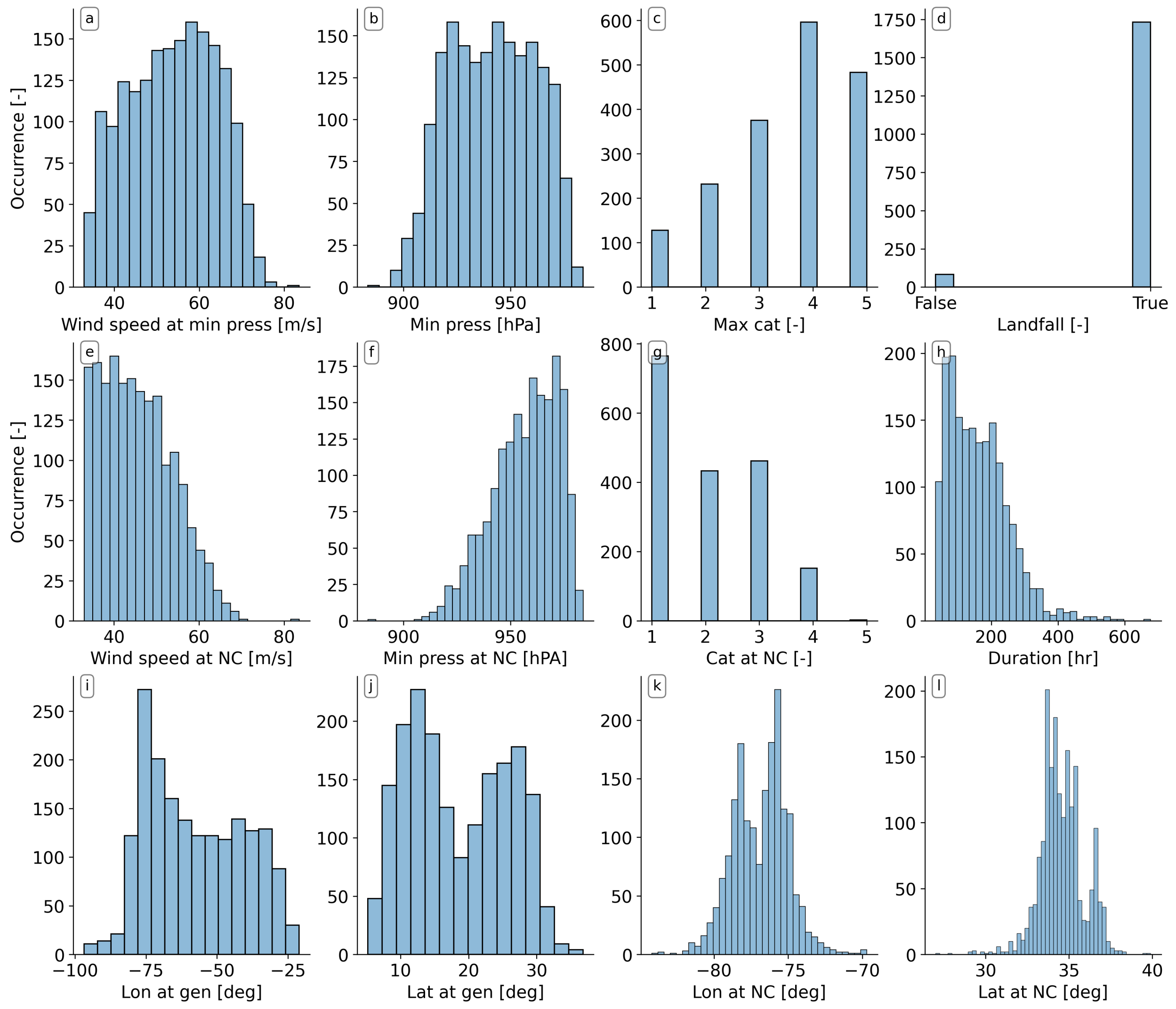


Figure 3.4: Histograms of track parameters. First row: wind speed at time snap of minimum pressure, minimum pressure, maximum category, and landfall flag. Second row: wind speed, pressure, and category at minimum distance to NC. Third row: genesis coordinates and coordinates of the storm's eye position with minimum distance to NC.

3.2 Step 2: Training library

The second step in the workflow is to develop the training library of process-based model simulations. This section describes the process of setting up and running the simulations using ADCIRC.

3.2.1 ADCIRC simulations

We automated the setup of each individual simulation using Python. An ADCIRC simulation has four main input files: (1) the unstructured mesh file (`fort.14`), (2) the nodal attributes file (`fort.13`), (3) the model parameter and periodic boundary condition file (`fort.15`), and (4) the meteorological forcing file (`fort.22`).

Unstructured mesh

The unstructured mesh is used to represent the coastal environment and solve the model's governing equations. The accuracy of the model simulations is related strongly with the mesh's quality. Woodruff (2023) showed extensive validation of the SABv3-60m mesh for hurricanes Matthew in 2016 and Florence in 2018. This mesh was developed with the most-recent nearshore bathymetry and topography as obtained from the Continuously Updated Digital Elevation Model (CUDEM) produced by the NOAA National Centers for Environmental Information (CIRES 2014). However, the SABv3-60m mesh has high-resolution inland elements for the entire South Atlantic Bight, making it too computationally expensive for this study.

To reduce the computational burden, we created a new version of this mesh by removing the elements of the floodplains and estuaries from Florida to South Carolina, and thus leaving inland mesh elements only in North Carolina. For this purpose, we developed a Python function that finds and removes the mesh elements inside a user-defined geospatial polygon, taking care of updating the boundary information and renumbering the remaining elements in ADCIRC's unstructured mesh file.

The SABv5 mesh has the same resolution as SABv3-60m, but with only 2.7 million elements and 1.4 million vertices (Table 3.1). SABv5 extends into the Western North Atlantic Ocean, the Gulf of Mexico, and the Caribbean Sea (Figure 3.5), but about 70 percent of its resolution is focused in coastal North Carolina. Element sizes range from 60 to 100 m in Oregon Inlet, Cape Fear River, and coastal areas with steep bathymetric or topographic gradients, and 100 to 200 m front and behind barrier islands. The element sizes increase in Albermarle and Pamlico Sounds up to 2,000 m in open water and from 20,000 to 50,000 m

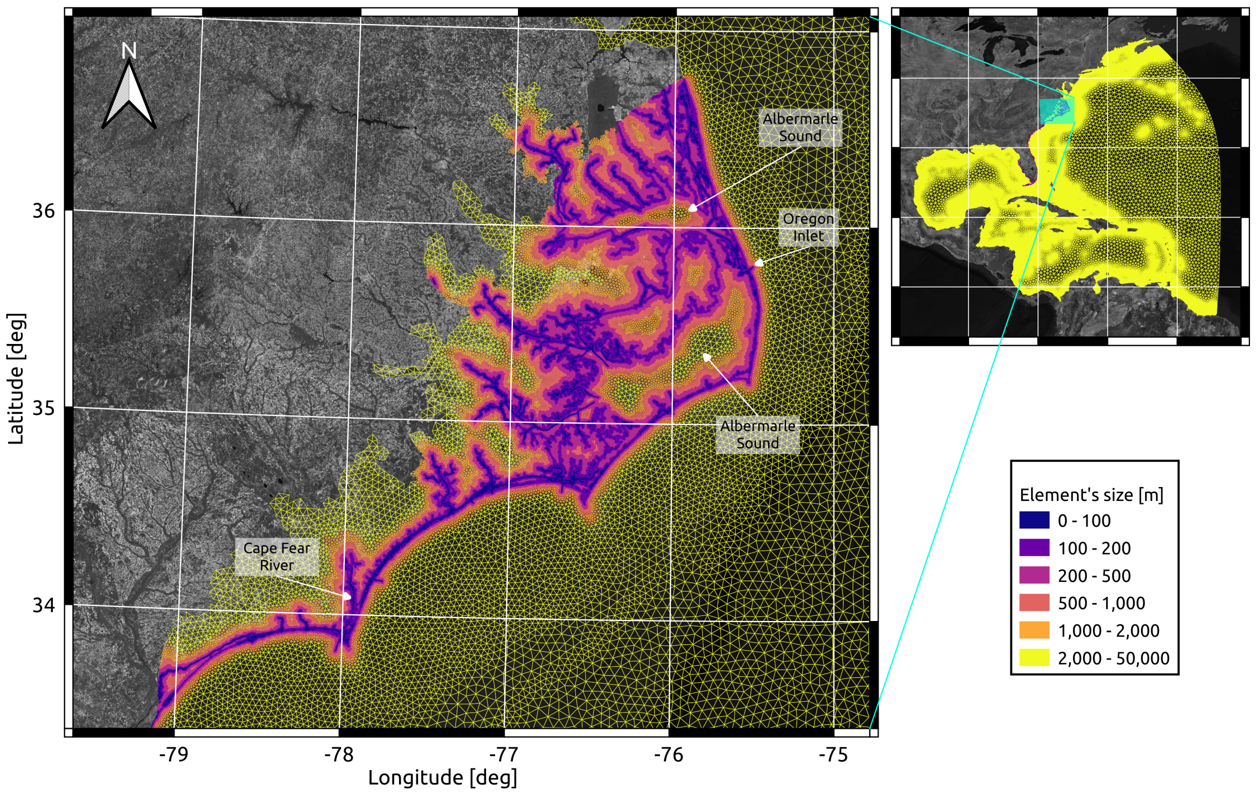


Figure 3.5: SABv5 mesh with panels: (upper right) full domain and (left) close-up view of North Carolina. The color scale indicates the average length of each triangle edge (1/3 of the triangle's perimeter).

in the open ocean. This resolution is appropriate for representing the flow along the coast, into estuaries, and floodplains (Dresback et al. 2013; Westerink et al. 2008).

Table 3.1: Main characteristics of SABv3-60m and SABv5 meshes: 'max' represents the maximum resolution, i.e. minimum element size; 'minns' represents the minimum resolution near shore, and 'min' represents the minimum resolution.

Mesh name	max [m]	minns [m]	min [m]	# of vertices	# of elements
SABv3-60m	60	100	50,000	3,531,883	6,812,980
SABv5	60	100	50,000	1,402,424	2,776,473

SABv5 uses the same nodal attributes as SABv3-60m (Woodruff 2023). In this research, we used the following mesh's vertices attributes: (1) a parameter τ_0 to control numerical damping and model stability, (2) a Manning's n coefficient for bottom friction, (3) eddy viscosity, (4) a parameter to control the advection terms in the Generalized Wave Continuity Equation (GWCE), (5) surface roughness, and (6) wind canopy coefficient. The unstructured mesh and nodal attributes are the same for all the simulations.

Model parameters and periodic boundary conditions

A key contribution of this study is that our training library will consider the dynamic interactions of tides and storm surge. However, to include tides in our ADCIRC simulations, we need each storm to correspond to a specific historical or future time period so that the tidal constituents can be applied with their correct phases and magnitudes. The synthetic storms don't have assigned dates in the STORM database, so we assigned random dates for the storms in our subset.

We defined a two-month period of representative astronomical tides in NC. We downloaded tide predictions for the six long-term tide gauges in NC from the NOAA Tides and Currents website (National Oceanic and Atmospheric Administration 2023, Figure 2.1) for the hurricane seasons (June through November) from 1991 to 2022 (Figure 3.6a) and computed the non-exceedance probability distribution (Figure 3.6b). Then, starting in January 1991, we selected consecutive two-month periods (Figures 3.6c, 3.6e, and 3.6g) and computed their non-exceedance probability distribution (Figures 3.6d, 3.6f, and 3.6h). This process allowed us to compare each individual two-month period with the full record because the non-exceedance curves were calculated using the same set of probabilities.

Then, we discarded the two-month periods where not all six gauges had data. For each station and two-month period, we computed the RMSE between the corresponding non-exceedance probability distribution (black lines in Figure 3.7) and the full record's non-exceedance probability distribution (blue lines in Figure 3.7). This allowed us to analyze, for each station and two-month period, how well the predicted tide of those two months represents the tides of the full record. Finally, for each two-month period, we averaged the RMSE values across stations and selected the two-month period with the minimum error. We selected July through August 2001 as the modeling period (orange lines in Figure 3.7). The stations-averaged RMSE was 2.2 cm.

Tides from this two-month period were then assigned randomly to each storm. For each of the 1,813 synthetic storms, we randomly selected a starting day and hour with the restriction that the full simulation must fit into the two-month period. The tracks have a 3-hourly time step, so after assigning the starting date, the full simulation has an associated date range. To enforce more stable simulations, we added a three-day ramp-up period before the storm. We extended the starting date of the tracks three days into the past, and using the ADCIRC

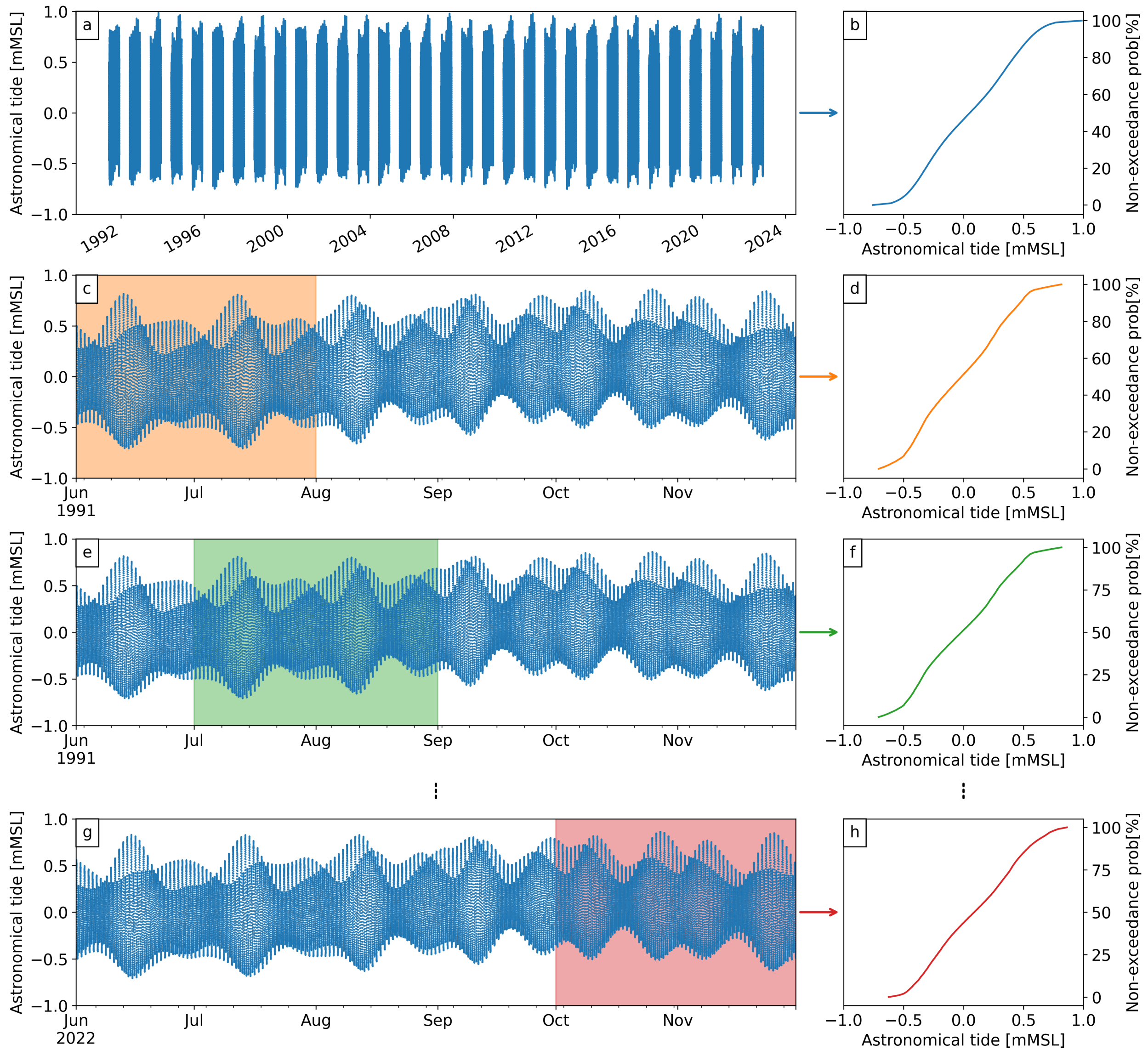


Figure 3.6: Methodology to define a two-month astronomical tide representative period at Duck NOAA tide gauge. Panel (a) shows the full record of predicted tides and panel (b) shows its non-exceedance probability distribution. Panel (c) presents the predicted tide for the 1991 hurricane season, highlighting the first two-month period in orange. The orange curve of panel (d) corresponds to the non-exceedance probability curve of the period highlighted in orange in panel (c). Panels (e) and (f) repeat the process the process for the second two-month period, and finally, panels (g) and (h) shows the process for the last consecutive two-month period.

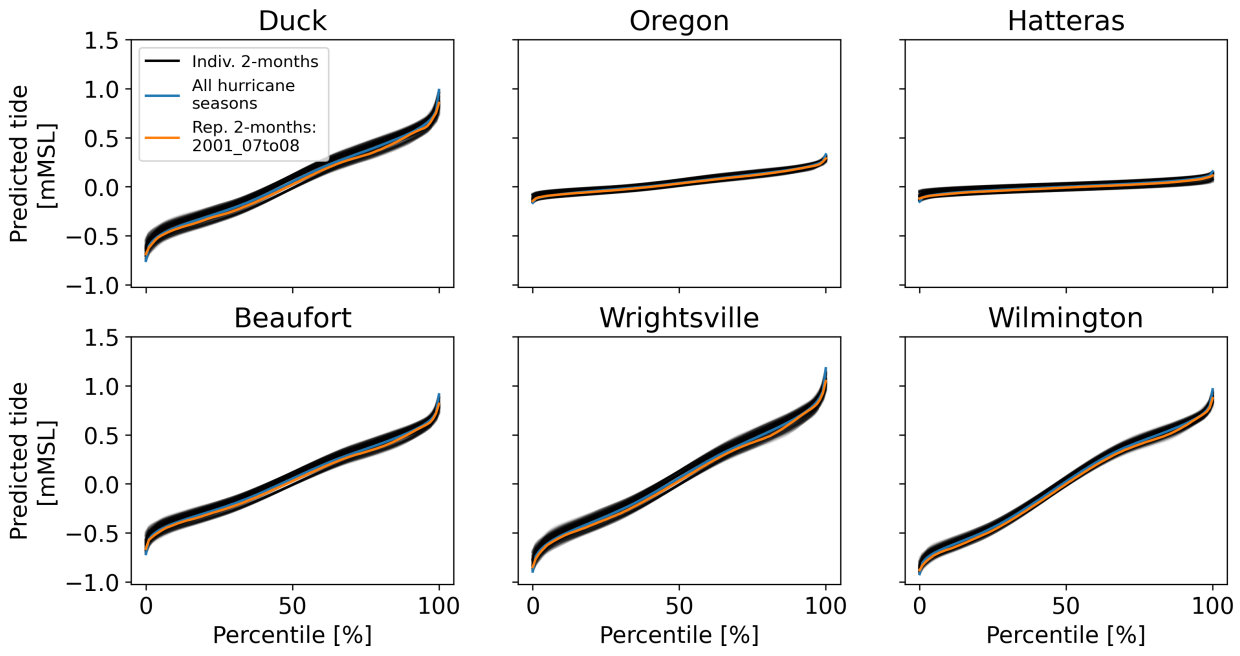


Figure 3.7: Non-exceedance curves of predicted astronomical tides. Data for the full record is plotted in blue for each station. The predicted tides' non-exceedance probability curve is presented in black for all individual two-month periods, and in orange for the selected modeling period.

built-in ramp function, we enforced a smooth transition between calm and storm winds.

With the simulation date range, the next step was to define the astronomical tide boundary condition. ADCIRC applies a user-defined number of tidal constituents to reconstruct the tidal signal in every vertex of the domain's ocean boundary (Figure 3.5). The tidal height at any location and time is given by Eq. 3.2:

$$H(t) = H_0 + \sum_{n=1}^{N} f_n H_n \cos(a_n t + (V_0 + u)_n - \kappa_n) \tag{3.2}$$

in which $H(t)$ is the height of the tide at any time t; H_0 is the mean water level above some defined datum; f_n is the node factor for adjusting H_n values for specific times; H_n is the mean amplitude of tidal constituent n; a_n is the speed of constituent n in degrees per unit time; t is time measured from some initial epoch or time; $(V_0 + u)_n$ is the equilibrium argument for constituent n at some location; and κ_n is the epoch of constituent n.

The parameters H_0, H_n, and κ_n are location-specific, thus specific to the ocean boundary location in the mesh (Woodruff 2023). The node factor f_n and the equilibrium argument $(V_0 + n)$ are time-specific, therefore related to the random starting date we assigned to each simulation. To define the astronomical tides, we used the eight principal tidal constituents: four semi-diurnal (M_2, S_2, N_2, and K_2) and four diurnal (K_1, O_1, P_1, and Q_1). We computed the factors f_n and $(V_0 + n)$ using and ADCIRC utility program (ADCIRC 2020). Using Python, we wrote a specific input file for each simulation.

The simulations use an implicit formulation to compute the complete gravity wave term, available in ADCIRC version 55 (W J Pringle et al. 2021), with a timestep of 1 s. The model parameters and periodic boundary conditions file varies across the simulations. We specified different date ranges, tidal constituents, and simulation names for each file.

Atmospheric forcing

To simulate the storm's effects, ADCIRC must represent the wind and pressure drop in the full model domain (Figure 3.5). The synthetic tracks describe the atmospheric conditions at the storm's eye, so we used an ADCIRC built-in method to generate wind and pressure fields that cover the entire model domain.

For this research, we adopted the parametric Holland symmetric Holland vortex model. This parametric model has been used widely to represent wind and pressure fields for hurricane simulations (Harper 1999). It solves the gradient wind equation, assuming that a rectangular hyperbola with two scaling parameters can approximate the surface pressure profile. Values for the scaling parameters were determined by setting $V = V_{\max}$ and $\mathrm{d}V/\mathrm{d}r = 0$ at $r = R_{\max}$ and assuming that Coriolis acceleration is negligible with the centrifugal acceleration at $r = R_{\max}$. We used this parametric wind model because it requires parameters that are available in the synthetic dataset of storms. Using Python, we wrote a specific meteorological forcing input file for each simulation following the ATCF Best Track format. To simulate the 1,813 tropical cyclones, we wrote storm-specific atmospheric forcing files following the ATCF Best Track format (Knapp, Kruk, et al. 2010).

Summary of ADCIRC simulations

We simulated 1,813 storms with a randomly assigned date between July to August 2001, enforcing a random astronomical tide. We also ran a tides-only simulation, i.e. without atmospheric forcings, for the same 2-month period. The goal of this simulation is to provide a time series of astronomical tides at any vertex of the domain.

We ran the simulations in three High-Performance Computing (HPC) systems: Stampede2 of the Texas Advanced Computing Center at the University of Texas at Austin, Anvil of Purdue University, and Hazel of North Carolina State University. We will summarize the computational resources used in Section 4.1.

3.2.2 Downscaling peak storm surge using Kalpana

In real-time storm surge forecasts, emergency managers expect flood predictions in high-resolution, so they can overlay it with maps of critical infrastructure like roads, and in short periods of time, so it can be helpful in decision making. Achieving both goals is challenging because increasing the mesh resolution also increases the model run time. And even an increase in mesh resolution may not be enough – the ADCIRC simulation results are provided as information (time series and maxima of water levels, velocities, etc.) on the vertices of the unstructured mesh, which may be much coarser than the roads or individual houses.

We use Kalpana (Section 2.3.3) as a post-processing step to increase the resolution of the peak surge predictions without increasing the model runtime. We downscaled the ADCIRC outputs into flood maps with a 15 m resolution. For this research, we used the downscaled flooding maps to extract the peak total water level at the prediction locations (Figure 2.1). In future work, we plan to extend the neural network to predict downscaled peak total water level maps as greyscale images, with the peak total water level as the pixel intensity.

For this research, we binned the ADCIRC predicted peak water levels from 0 to 10 m, every 0.5 m. The downscaled raster image has 475,541,017 pixels, covering the entire NC coast with a resolution of 15 m. The flooding extent was horizontally expanded by one triangular element. Generating each downscaling map takes around 7 minutes on an HPC using four computational cores, and the output raster file size was 3.5 GB.

3.3 Step 3: Deep learning

3.3.1 Motivation

The goal of our neural network is to map the full dimensionality of the storm track (along the time dimension for all parameters) and the astronomical tides to a single output in time and multiple outputs in space. Our model's predictors/inputs include storm parameters available in the dataset and the astronomical tide at the locations to predict. The model's output is the peak of the total water level caused by the storm. The neural network is a many-to-one architecture composed by a set of 1D CNNs and dense layers (Figure 3.8).

3.3.2 Data preprocessing

The neural network's track-related (Table 3.2) inputs include the wind speed, central pressure, and radius to maximum wind speed. We calculated the distance from the storm's eye to the prediction locations (Figure 2.1). We computed the heading direction of the storm

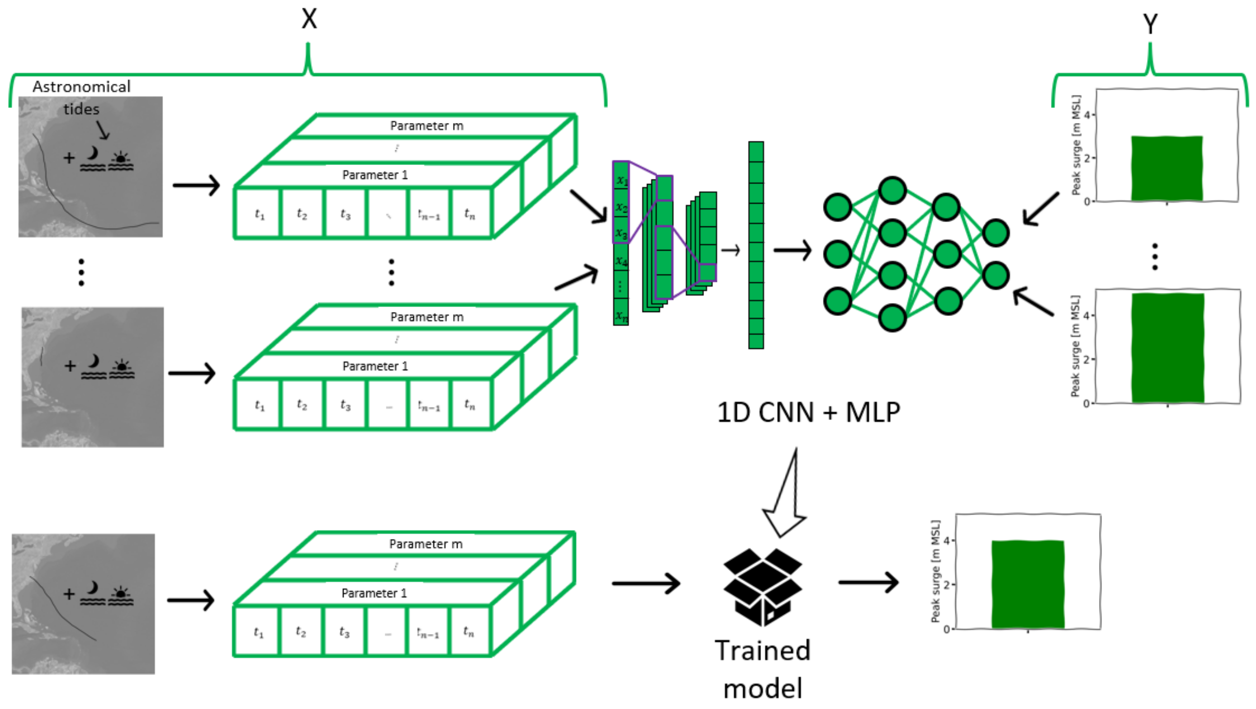


Figure 3.8: Schematic of the neural network (NN) system. The inputs of the NN are the storm track and the astronomical tide at the prediction location. The information is processed and arranged as a 2D array with the time in one dimension and the different parameters in the second dimension. Then, the 2D arrays for each storm are stacked together. The data is divided into training and testing datasets. The training data is passed through a combination of 1D convolutional neural networks (1D CNNs) and multi-layer perceptron (MLP) that learn to map the input data to the peak total water level at the station, via optimizing the NN parameters with back-propagation. Finally, the trained model predicts the peak total water level for storms and tides unseen during training.

following a non-meteorological convention, i.e. the direction toward which the storm is going. We also computed the forward speed as the distance the storm's eye covers per hour. Finally, we decomposed the heading direction and the forward speed in u and v vectors. In addition, we used the Fast Fourier Transform (FFT) algorithm to transform the wind speed, central pressure, radius to maximum wind speed, and forward speed u and v vectors into the frequency domain. We only considered the magnitude of the FFT output. The purpose is to let the NN learn frequency domain features that can help to create a better representation of the input time series (Huang, Liu, and Tseng 2018).

The last input of the NN (Table 3.2) corresponds to time series of astronomical tides at the prediction locations (Figure 2.1) and at the center of the offshore model boundary (Figure 3.5). We extracted the time series from the 2-month tide-only simulation described in section 3.2.1 for the corresponding date range of each track. This information has a 10-minute time

Table 3.2: Summary of inputs of the multitask neural network.

Input	Number of parameters
Max. sustained wind speed	1
Central pressure	1
Radius to max. wind speed	1
Forward speed u and v vectors	2
Distance from storm's eye to prediction locations	9
FFT magnitude of max. sustained wind speed	1
FFT magnitude of central pressure	1
FFT magnitude of radius to max. wind speed	1
FFT magnitude of forward speed u and v vectors	2
Offshore astronomical tide	1

step, so we computed a 1-hour average to resample the time series. The track-related data has a 3-hour time step, so we linearly interpolated it to decrease the time step to 1 hour to match the tide.

We combined the track-related resampled time series with the tides to create a single input time series per storm. We refer to it as 'input multivariate time series (IMTS)' to facilitate the explanation. As the date range covered by each IMTS may differ, we used zero padding to extend the length of the IMTS, filling the gaps with zeros. Now, all the IMTS have the same length as the longest IMTS before zero padding (Figure 3.9).

We followed two approaches for training the NN. In the first approach, we considered a specific NN for each prediction location (single model), whilst in the second approach, we trained one model (multitask model) to predict at all nine prediction locations. For the multitask model, the input array contains the distance from the storm's eye to all the prediction locations and the astronomical tides at all the locations or at the ADCIRC's offshore boundary's center. However, for each single model, only the information at the respective location will be used for training. For example, to train a model to predict at the Duck NOAA tide gauge (Figure 2.1), only the distance from the storm's eye to Duck is considered, along with the tide at Duck or the offshore tide, depending on the model's input configuration. The features sensitivity analysis addresses the definition of using the offshore or local tide (Section 4.3.2). The neural network's output is a single value in time and space (for each single model) or multiple values in space but constant in time (for the multitask model). The goal of the NN is to predict the peak total water level driven by the coastal storm. The prediction locations correspond to the six NOAA tide gauges and the Albemarle, Pamlico, and Neuse rivers. These locations have different levels of sheltering and surrounding coastal environments. Predicting at these different locations could be challenging

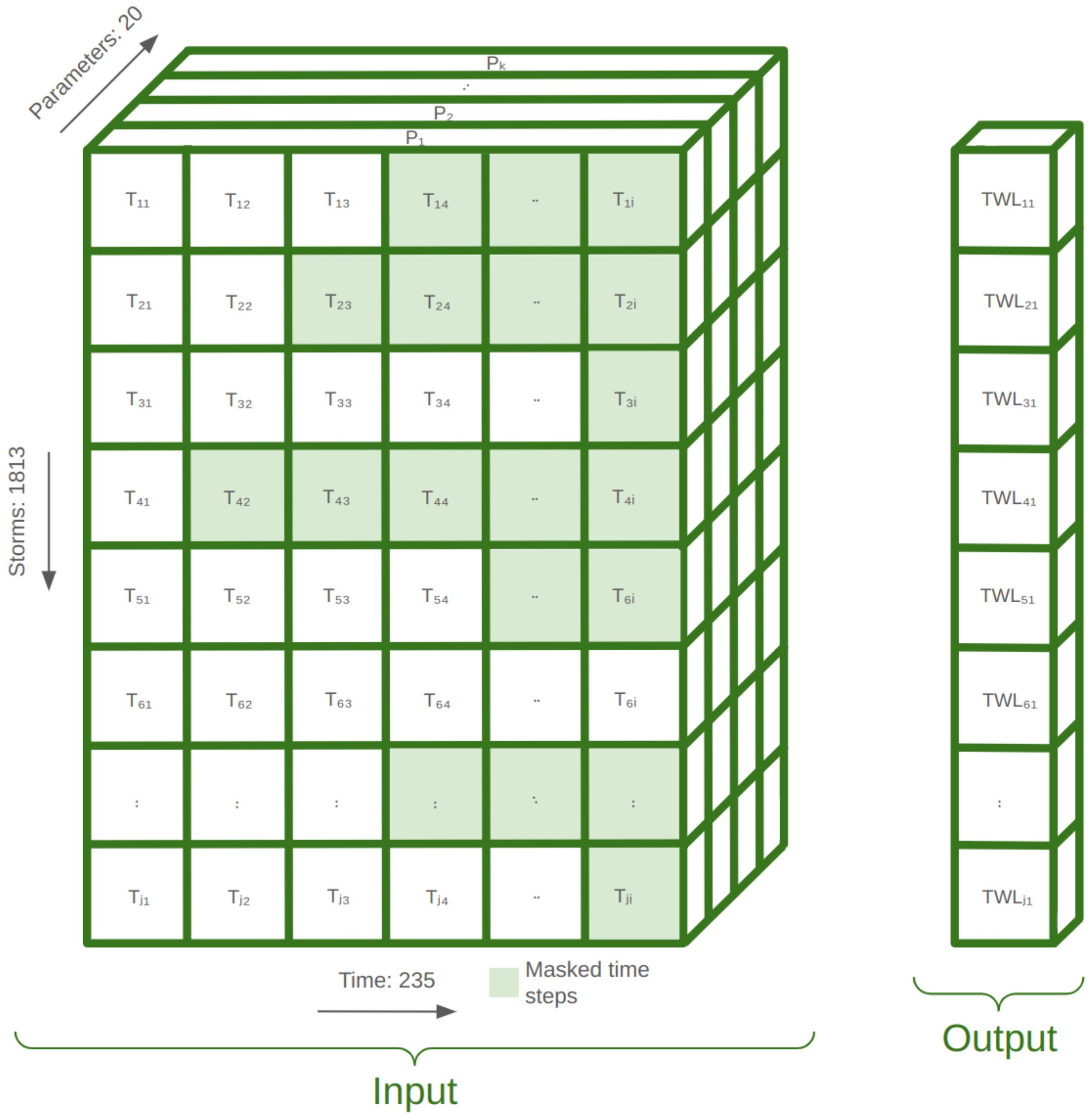


Figure 3.9: Schematic of the neural network (NN) inputs and outputs. Each row of the 3D input array corresponds to a different storm, and each column corresponds to a time snap during that storm. The input parameters are stacked in the depth dimension. Green shading represents zero-padded cells. All storms have the same amount of parameters but may have different lengths in time.

for the neural network because different physical processes may drive the flooding. Duck and Wrightsville tide gauges are on the open coast, whilst the rest have different degrees of sheltering. We didn't consider an extra point for the Cape Fear River estuary because the Wilmington tide gauge is representative of the system.

We used the *train_test_split* tool of Scikit-learn (Pedregosa et al. 2011) to divide the dataset into training, validation, and testing subsets. As recommended in Géron (2022), we used 15% of the total dataset for testing and 20% of the remaining 85% for validation. This translates to 1,233 IMTS for training, 308 for validation, and 272 for testing.

The last step in the data preprocessing is to scale it using the *StandardScaler* tool of Scikit-learn (Pedregosa et al. 2011). It standardizes the features by removing the mean and scaling to unit variance. Each parameter is scaled independently, ignoring the zero-padded values. The scaler was trained on the training and validation subset and then used to scale the testing subset. This is done to prevent the testing data from influencing the training process. By scaling the data appropriately, we prepare it for the learning process, ensuring that the model can learn effectively and generalize well to unseen storms.

3.3.3 Neural network architecture

We propose a deep learning architecture tailored for time series analysis, specifically for predicting peak total water levels from the storm track and the astronomical tides. The architecture, implemented using TensorFlow (Martín Abadi et al. 2015), is a sequential neural network consisting of convolutional and dense layers along with regularization techniques to enhance feature extraction and prevent overfitting such as batch normalization (Ioffe and Szegedy 2015) and dropout (Srivastava et al. 2014).

The NN begins with a *Masking* layer to discard the zero-padded values and ensure the learning process is not disturbed by the added zeros. Then, the architecture employs three *Conv1D* layers, each with a different number of filters (16, 32, and 64, respectively), a *kernel size* of 3 and a *MaxPooling* with a pool size of 2, activated by *Rectified Linear Unit* (Agarap 2018, ReLU) functions. The kernel slides across the input sequence, learning local patterns and relationships between neighboring data points. These filters, represented by sets of learnable weights, perform feature extraction by convolving over the input, creating feature maps that highlight relevant patterns at different levels of abstraction with the pooling function. Varying the filter sizes allows the model to learn and extract hierarchical features of different complexities. We included *Batch Normalization* layers after each 1D CNN to stabilize and accelerate training by normalizing the activation functions and reducing the internal covariate shift. *MaxPooling1D*, with a pool size of 2, are added after each batch normalization layer. The pooling layers downsample the output of the proceeding 1D CNN to retain critical features while reducing computational complexity.

The extracted features are then flattened into a single 1D vector using a *Flatten* layer. This vector is passed through two fully connected *Dense* layers with 64 and 32 neurons, respectively. We activated the layers using ReLU to introduce non-linearity. We added a 20% *Dropout* rate to randomly disable the connection of neurons between layers and prevent overfiting. Dense layers play a crucial role in learning complex patterns and relationships within the data by performing matrix multiplications between the input data and a set of learnable weights, followed by an activation function. The weights represent the strengths of connections between neurons.

Finally, in the case of the single model, it concludes with a single Dense layer activated by a ReLU function to produce one output in space. The multitask model concludes with nine Dense layers also activated by ReLUs to produce nine outputs in space. The proposed architecture is a many-to-one model because it produces a single output in time from a time-varying input.

Learnable weights in the 1DCNN and dense layers are adjusted during training through backpropagation (Rumelhart, Hinton, and R. J. Williams 1986), which minimizes the difference between predicted and ground truth outputs.

3.3.4 Neural network hyperparameters

In deep learning, parameters are the neural network's internal variables learned and optimized during training, typically weights and biases, whereas hyperparameters are external settings or configurations that dictate the behavior of the neural network during training. These settings are not learned from the data, must be specified beforehand by the user, and are fixed during training. In this subsection, we describe the main hyperparameters and what we tuned during validation.

Learning rate is the step size at which the neural network's parameters are adjusted in each step of the optimization process during training. We used a fixed value of 10^{-4} (Géron 2022). Another important hyperparameter is the batch size, which refers to the number of data samples the NN sees at each training step. Larger batch sizes can lead to faster training but require more memory and might not generalize well. Smaller batch sizes take longer to train but can sometimes converge to better solutions and generalize better. The batch size selection often involves a trade-off between computational efficiency, model convergence, and generalization performance. We adopted a value of 100, which was the largest batch size we could fit in a 5GB Nvidia Quadro P2000 GPU.

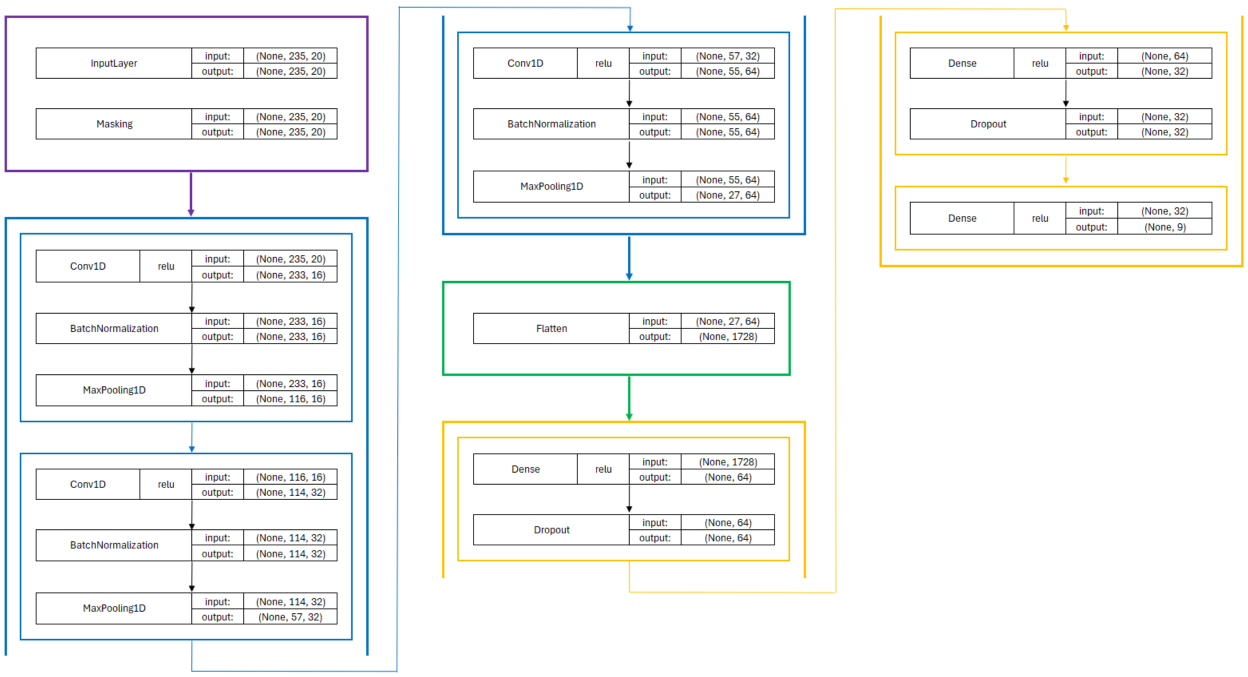

InputLayer | input: (None, 235, 20) | output: (None, 235, 20)
Masking | input: (None, 235, 20) | output: (None, 235, 20)
Conv1D | relu | input: (None, 235, 20) | output: (None, 233, 16)
BatchNormalization | input: (None, 233, 16) | output: (None, 233, 16)
MaxPooling1D | input: (None, 233, 16) | output: (None, 116, 16)
Conv1D | relu | input: (None, 116, 16) | output: (None, 114, 32)
BatchNormalization | input: (None, 114, 32) | output: (None, 114, 32)
MaxPooling1D | input: (None, 114, 32) | output: (None, 57, 32)
Conv1D | relu | input: (None, 57, 32) | output: (None, 55, 64)
BatchNormalization | input: (None, 55, 64) | output: (None, 55, 64)
MaxPooling1D | input: (None, 55, 64) | output: (None, 27, 64)
Flatten | input: (None, 27, 64) | output: (None, 1728)
Dense | relu | input: (None, 1728) | output: (None, 64)
Dropout | input: (None, 64) | output: (None, 64)
Dense | relu | input: (None, 64) | output: (None, 32)
Dropout | input: (None, 32) | output: (None, 32)
Dense | relu | input: (None, 32) | output: (None, 9)

Figure 3.10: The neural network architecture consists of a Masking layer (purple box), three blocks of 1D convolutional neural networks comprising a convolutional layer (blue blox), a batch normalization layer, and a max pooling layer, with 16, 32, and 64 filters, respectively. Then, the features are flattened (green box) and passed through 3 dense layers of 64, 32, and 1 neuron, with 20% dropout each (yellow box). ReLU is the activation function in all 1D CNNs and dense layers.

In the validation (Section 4.2.1), we analyzed the hyperparameters of loss, optimizer, augmentation size, number of epochs, and the NN inputs. Loss is a mathematical function to quantify the difference between the real and predicted values by the neural network. It measures how well the NN fits the data to guide the optimization process. For this study, we used the mean squared error (MSE):

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{3.3}$$

and the Huber loss (Huber 1992):

$$\text{Huber} = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{if } |y - f(x)| \le \delta \\ \frac{1}{2}\delta|y - f(x)| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases} . \tag{3.4}$$

MSE is the average of the square of the biases and ensures the trained model does not have outlier predictions with huge errors. It puts a larger weight on these errors due to the squaring. Huber loss combines the MSE and mean absolute error (MAE):

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |Y_i - \hat{Y}_i|. \tag{3.5}$$

Equation 3.4 means that for a loss value less than δ the MSE is used, and for a greater loss, MAE is applied. We used a δ value of 1, the default value in TensorFlow (Martín Abadi et al. 2015). Using MAE for larger loss values mitigates the weights we put on outliers, and using MSE for the smaller loss values maintains a quadratic function near the center. Huber and MSE losses are well-suited to predict the extremes or outliers of the training data.

In deep learning, optimizers are algorithms that tune the neural network's weights and biases during training to minimize the loss function. For this study, we tested the Root Mean Square Propagation (RMSProp; ibid.) and the Adaptative Moment Estimation (Adam Kingma and Ba 2014). RMSProp aims to improve the limitations of using a fixed learning rate by adapting it for each parameter individually. It computes exponentially weighted moving averages of the squared gradients for each parameter. By dividing the current gradient by the square root of this moving average, RMSProp normalizes the learning rates, which helps to control the magnitude of parameter updates. This adaptive adjustment mitigates the vanishing or exploding gradient issues, leading to more stable and efficient convergence during neural network training. Adam leverages the gradients' first (mean) and second moments (variance) to adjust the learning rates for each parameter during adaptive optimization. It maintains exponential decaying averages of past gradients (first moment) and past squared gradients (second moment). By incorporating these estimates, Adam computes adaptive learning rates for individual parameters, allowing for more effective convergence by mitigating the effects of noisy gradients, adapting to different parameter scales, and providing efficient updates during training. RMSProp and Adam differ in estimating the learning rate and the momentum term. Adam tends to show a faster convergence and robustness to various hyperparameters due to its additional momentum term, whereas RMSProp provides stable performance and simplicity in its implementation by solely adjusting learning rates based on squared gradients. RMSProp and Adam are two recommended optimizers when working with regression problems in deep learning.

The number of epochs refers to how many times the training set is passed through the neural network to update the model's parameters with the optimizer. We trained the NN for 1,000 epochs.

Finally, we did feature engineering to define the NN inputs. We validated the model using the astronomical tide at the prediction location (referred hereafter as local tide) and the astronomical tide at the ADCIRC boundary (offshore tide). We also validated the model using the coordinates of the storm's eye instead of the distance to the prediction point, not considering the FFT crafted inputs and only with the FFT inputs and not the storm parameters. From a storm surge forecaster's perspective, it is easier to estimate the offshore tide than the local tide. In the local prediction, the tide interacts with the nearshore bathymetry and coastline, so non-linearities play a significant role. On the other hand, offshore tides can be predicted using Equation 3.2. We aim to create the simplest framework that gives accurate enough results to be extended later on to predict spatially continuous 2D maps of peak total water levels. In Section 4.2 we included only the model results that we consider can be extended to predict 2D maps in the future, not necessarily the model with more accurate predictions. The trade-off between input simplicity from a forecast system perspective and model accuracy is addressed in Section 4.3. It is important to mention that the hyperparameter tuning and model validation was done for the single model trained to predict at the Beaufort NOAA tide gauge (Figure 2.1).

3.3.5 Data augmentation

Data augmentation is the process of increasing the size of the dataset by modifying the existing data. It is used in object recognition problems, e.g. when rotating an image of a dog, the label of the new image is still a dog. In regression problems, it is more complicated; sometimes, Gaussian noise is applied to change the input and outputs slightly. As far as we know, data augmentation has not been applied to storm surge prediction. One of the main reasons is that storm surge prediction is a highly non-linear problem, e.g. a 10% increase of the storm's wind speeds may not translate into a 10% increase in the storm surge.

We augmented our input data by creating new combinations of storms and tides (Figure 3.11). We assumed that there is a linear interaction between the storm surge and the astronomical tide – this assumption is imperfect, but it is a better estimation than adding random Gaussian noise because the superposition of tides and surge can be done with reasonable superposition (Irish, D T Resio, and Cialone 2009), and it allowed us to significantly increase the size of our input data. For each storm, using the two-month, tides-only ADCIRC simulation, we removed the astronomical tide from the total water level time series to get a surge-only time series. Then, we combined the surge-only time series with a randomly selected series of astronomical tides to get a new total water level time series. This process creates a new set of inputs by changing the astronomical tide time series and a new output because the new total water level time series may have a new peak value. This process can be repeated as many times as we want per storm, which we defined as the augmentation size. We validated the model using the original dataset and augmentation sizes of 10, 20, 50,

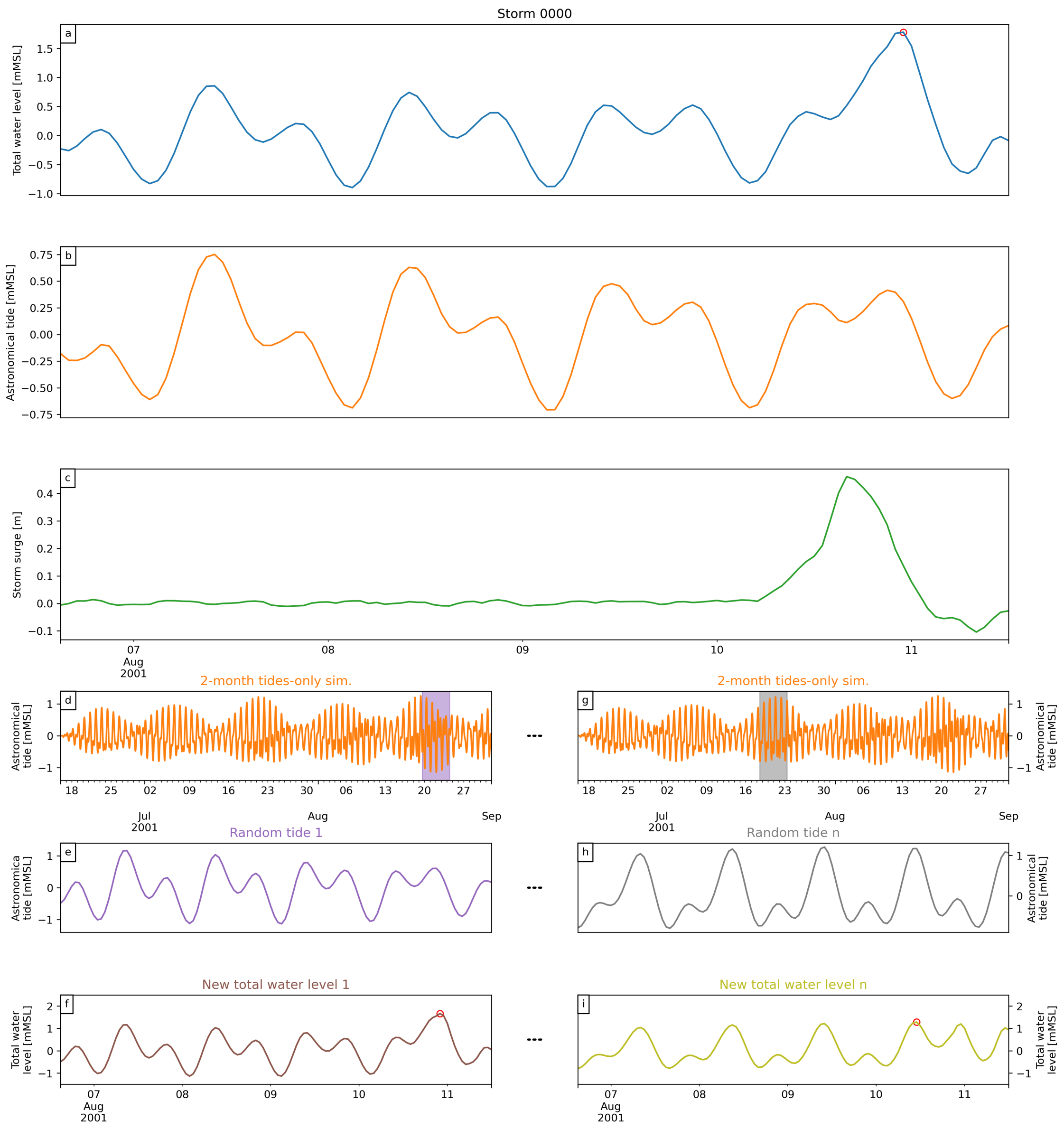


Figure 3.11: Example of the data augmentation methodology for storm 0. Panel a shows the ADCIRC-predicted total water level time series at Duck, with a red circle to indicate the total water level value to be predicted by the neural network; panel b shows the astronomical tide time series obtained from the 2-month tide-only simulation at Duck; and panel c shows the storm surge time series (computed as the difference of total water level and astronomical tide). Panel d shows the 2-month astronomical tide time series, highlighting a randomly selected period of the same duration as the storm track; panel e shows a zoom-in view of the randomly selected astronomical tide; and the time series shown in panel f corresponds to the addition of the storm surge and randomly selected astronomical tide time series, with a red circle to indicate the new total water level value to be predicted. Panels g, h, and i are the iteration number n of the methodology, which can be repeated as many times as we want. The augmentation size is analyzed in subsection 4.3.2 as one of the neural network's hyperparameters.

Table 3.3: Dataset sizes for different augmentation sizes.

Augmentation size	Total dataset	Training set	Validation set	Test set
0	1,813	1,233	308	272
10	19,943	13,563	3,388	2,992
20	38,073	25,893	6,468	5,712
50	92,463	62,883	15,708	13,872
100	183,113	124,533	31,108	27,472

and 100. Table 3.3 shows the training and validation sets size for each augmentation size we tried.

We note that the input data represent the full range of possible storm conditions because the dataset used to develop the training library is based on the last 38 years of real storms, so it is not tailored to represent only extreme events. This behavior can be seen in histograms of the peak total water levels as observed at all prediction locations (Figure 3.12), where the mode of the peak total water levels is close to 1 m, but the tails extend up to 4 m. This translates into an unbalanced dataset, biased towards low peak total water levels.

3.3.6 Metrics

For the validation process and choosing the best combination hyperparameters, we used the mean bias error (MBE):

$$\mathrm{MBE} = \frac{1}{N}\sum_{i=1}^{n}(Y_i - \hat{Y}_i). \tag{3.6}$$

This metric computes the average difference between the true values (Y_i) and the predicted values ($\hat{Y}_i$), keeping track of the sign of the difference. It is important to know if the model tends to over- or underpredict because, in real-time forecasting of storm surge or any natural hazards, underpredictions can impact safety measures taken to keep people safe. We also compute the MBE for the validation's largest 10% of true values because the NN needs to be very accurate for extreme values. We also used the root mean squared error (RMSE):

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{N}} \tag{3.7}$$

to study the model performance over the test set.

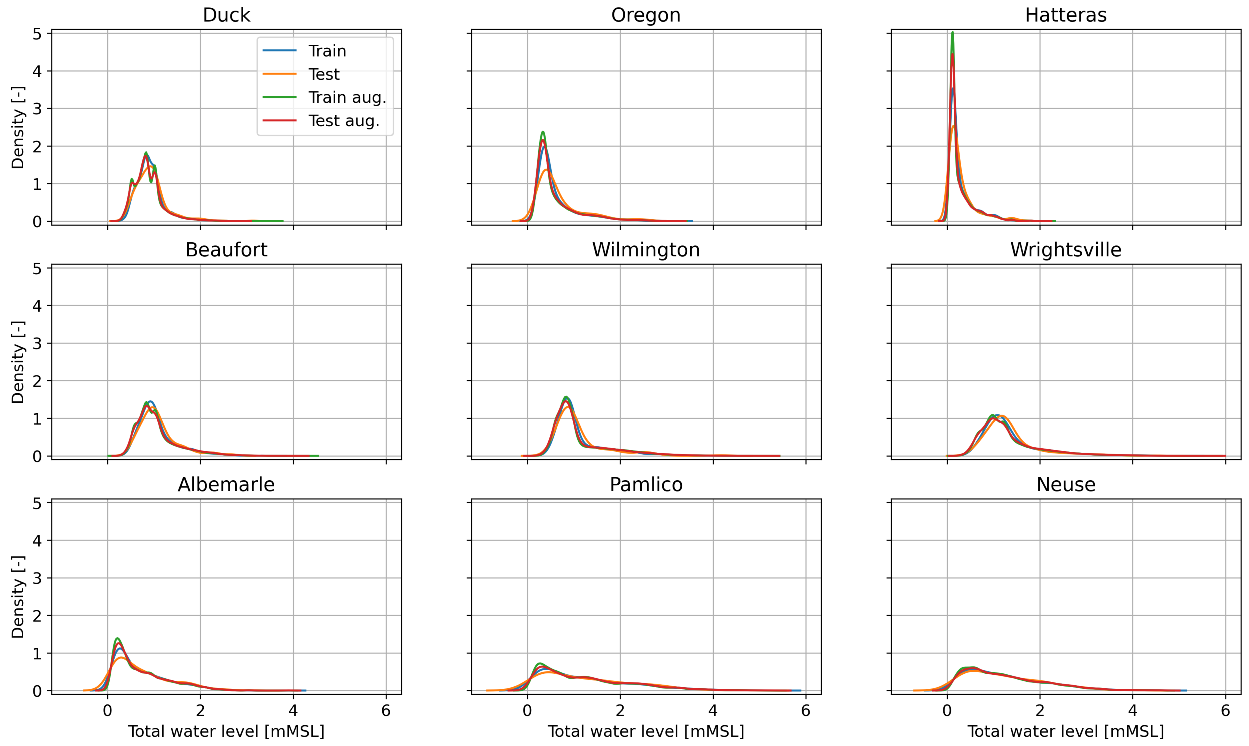


Figure 3.12: Kernel density plots of the peak total water level distribution for the augmented and non-augmented datasets, and test and train sets of each.

3.4 Step 4: Probabilistic prediction

Once the neural network was validated and tested, we used it to predict the peak total water level for a set of real historical storms (Table 2.1). From the National Oceanic and Atmospheric Administration (2023), we downloaded the peak total water level measured for the date range of the storms at the NOAA tide gauges of Duck and Wrightsville. We selected the open-ocean Duck and Wrightsville NOAA tide gauges only because the other tide gauges measurements may have been influenced by processes not included in our ADCIRC simulations like rainfall or river discharge.

The STORM dataset used to develop the training library is based on the Best Track Archive, which includes the historical storms we are considering, so we expect the NN to perform well in predicting the peak total water level observed during each storm. We considered the ‘true’ value as the maximum total water level of the NOAA tides observation during the storm’s date range.

Taking advantage of how fast the NN predicts, we predicted the peak total water level for 100 perturbations computed using CLIMADA (e.g. Figure 3.13) of the historical tracks. It

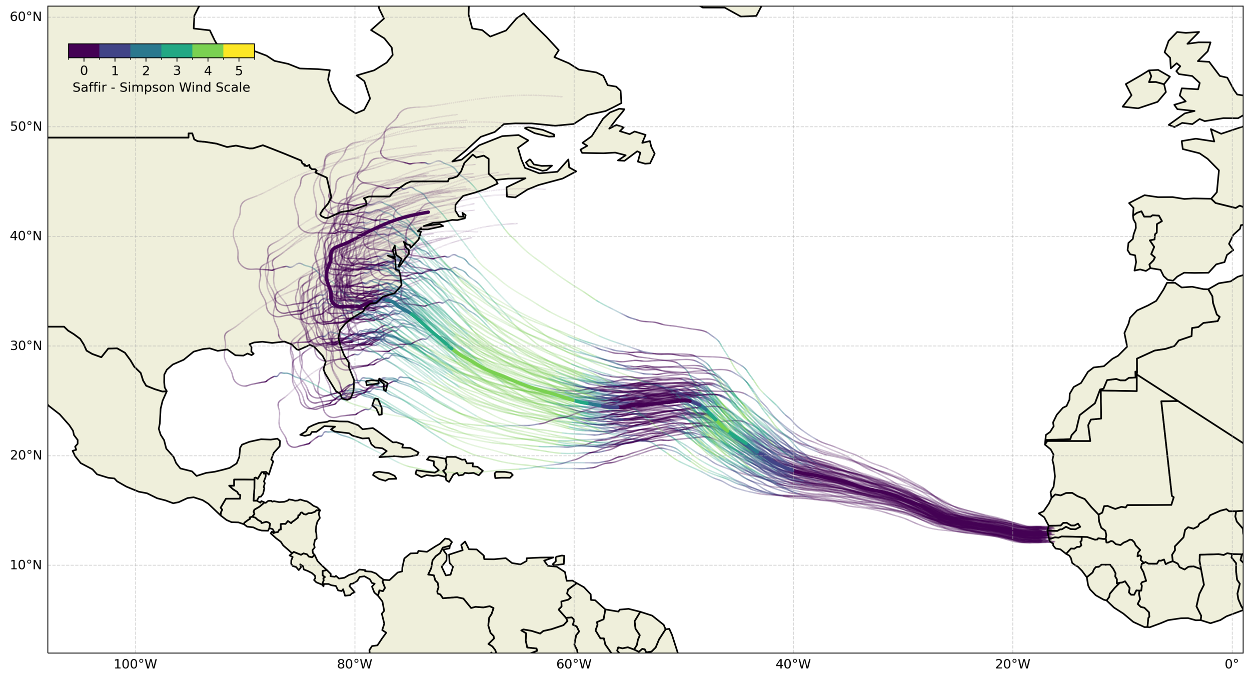


Figure 3.13: Florence's track perturbations computed using CLIMADA's random walk algorithm. The observed track is shown with a wider line. The colors indicate the storm category.

is worth mentioning that only the track's coordinates are perturbed before the storms make landfall, but in case of landfall, the maximum wind speed and the minimum pressure are also modified with a decay parameter. CLIMADA's track perturbation tool has been successfully used in other studies (Gettelman et al. 2017; Gevecke 2022; Meiler, Ciullo, et al. 2023) and provides a simple framework to develop a probabilistic distribution of tracks (Meiler, Vogt, et al. 2022). The exercise aims to see if the measured peak total water level is contained in the probabilistic prediction interval defined by NN's output for the 100 perturbations.

Chapter 4

Results and Discussion

First for the training library, we present the results of one ADCIRC simulation to illustrate the predictions of total water levels in coastal North Carolina (NC), and then we present the statistical results obtained from aggregating all the downscaled peak total water level maps across all storms. Then for the neural network development, we show the validation and testing of the multitask neural network to predict peak total water levels at the six NOAA tide gauges and at three NC rivers, and then we use the neural network to make probabilistic predictions of observed and perturbed scenarios of historic storms. We discuss the similarities and differences of our NN with others available in the literature; we present a sensitivity analysis of hyperparameters, augmentation methods, and features. Finally, we discuss possible applications of a probabilistic prediction framework.

4.1 ADCIRC

For running the ADCIRC simulations, we used a total of 1,337,444 CPU-hours on three HPC systems: TACC Stampede2, Purdue Anvil, and NCSU Hazel. We ran the simulations on 128, 192, or 256 cores, depending on the system and available computing resources. The library of simulations has about 17 terabytes of data. The storm input data and the library of downscaled peak total water level maps will be publicly available to other researchers. The wall clock runtime ranged from 1.2 to 33.3 hours per simulation, averaging 3.7 hours. The histogram is skewed toward short runtimes (Figure 4.1), and 98 percent of the simulations had a runtime below 10 hr. Only 11 simulations required longer than 20 hr.

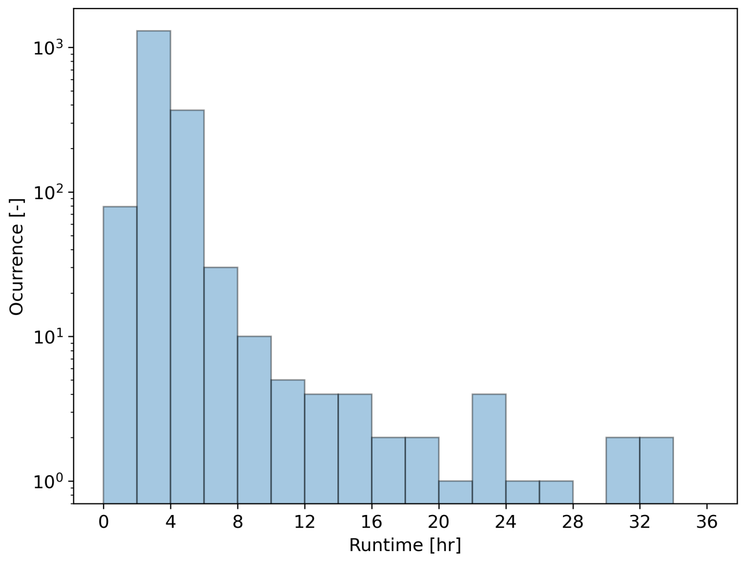


Figure 4.1: Histogram of ADCIRC simulations' runtime.

4.1.1 Individual storms

In this subsection, we present the simulation results of storm 0, focusing on the water level predictions at the locations of the NOAA tide gauges at Duck and Wrightsville. Storm 0 is one of the extreme tropical cyclones we simulated. It is the most unique storm in the subset affecting NC, and thus its track and other parameters should cause it to generate an interesting response in the coastal waters of NC.

Storm 0 has a duration of 6 days; it forms about 700 km northeast of Barbados in the Atlantic Ocean as a tropical depression. The maximum sustained wind speed is 56 m/s (category 3) with a minimum pressure of 939 hPa, and it occurs when the storm is in the open ocean between Bermuda and the coast of North Carolina. We analyzed its evolution and maximum water levels (Figure 4.2). The maximum water level output corresponds to the peak water level on each mesh vertex at any time during the entire simulation, i.e. only the highest value is reported. Half a day before landfall (Figure 4.2, top row), the storm was still in category 3, but the NC coast was unaffected because the storm center was 220 km away. It made landfall with a category-2 status, pushing water from the sounds to the Neuse, Pamlico, and Albemarle rivers (Figure 4.2, middle row). The total water level exceeded 3.5 m and 4.0 m in the Pamlico and Neuse river estuaries, respectively. Athough the storm passed over Wrightsville Beach, the maximum flooding occurred near Pamlico Sound and the Neuse and Pamlico rivers. As these locations are rightwards of the storm's eye and relatively close, strong winds push water inland.

We analyzed time series of track parameters and water levels at Duck and Wrightsville

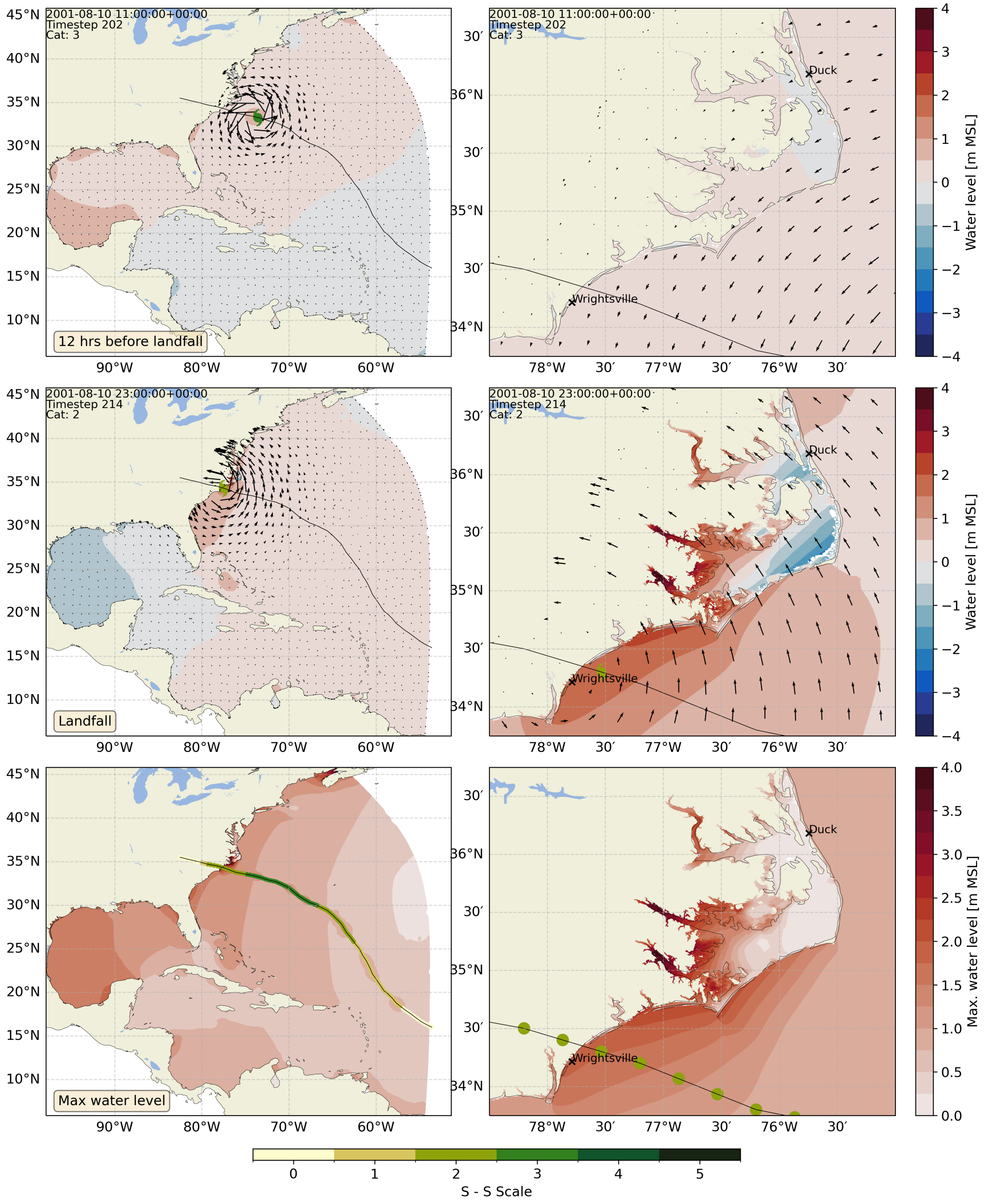
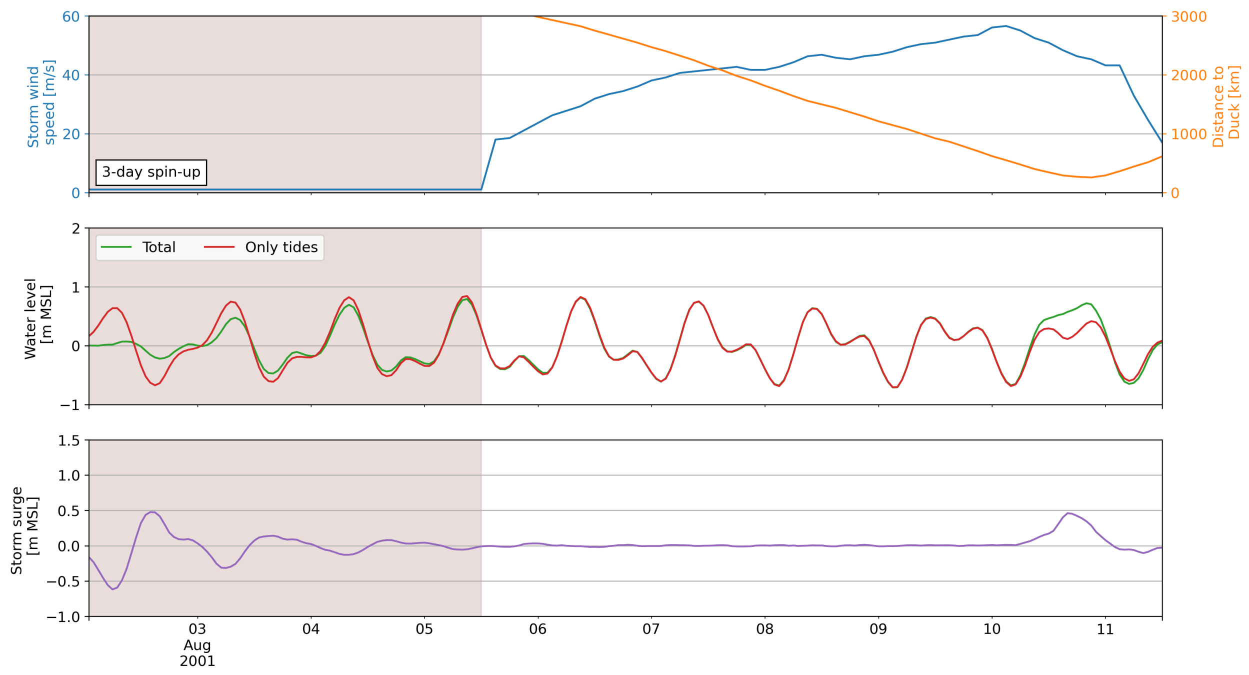
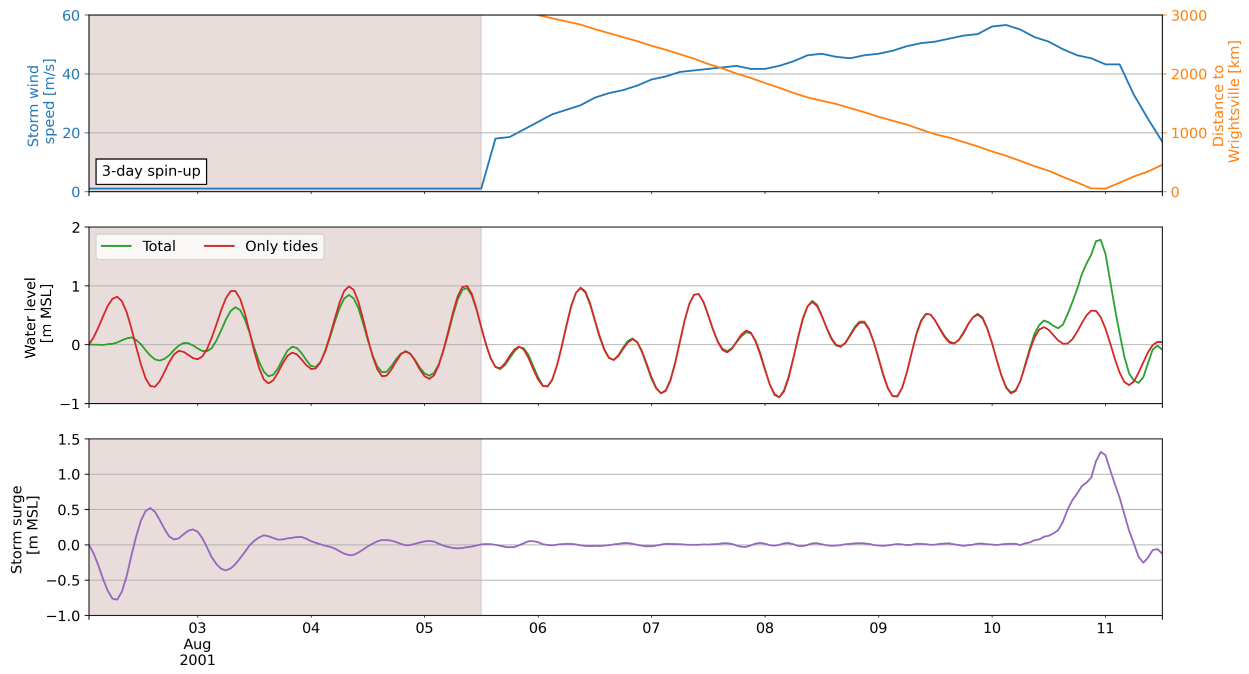


Figure 4.2: Maps of the total water levels from Storm 0 at: (top row) 12 hours before landfall, (middle row) landfall, and (bottom row) maximum across the simulation; and for zoom levels of: (left column) full domain, and (right column) North Carolina. The black line shows the storm track, and the color of the marker indicates the category of the storm on the Saffir-Simpson scale. Duck and Wrightsville tide gauges are highlighted.



(a) Duck



(b) Wrightsville

Figure 4.3: Time series of storm 0 track parameters and water levels at the (a) Duck and (b) Wrightsville NOAA gauge locations. For each location, the parameters are shown as: (top) maximum wind speed (blue) and distance to Wilmington or Duck (orange) along the track, (middle) total water level time series (green) and the astronomical tide (from the 2-month tide-only simulation, red), and (bottom) surge-only time series defined as the difference between the total water level and the astronomical tide.

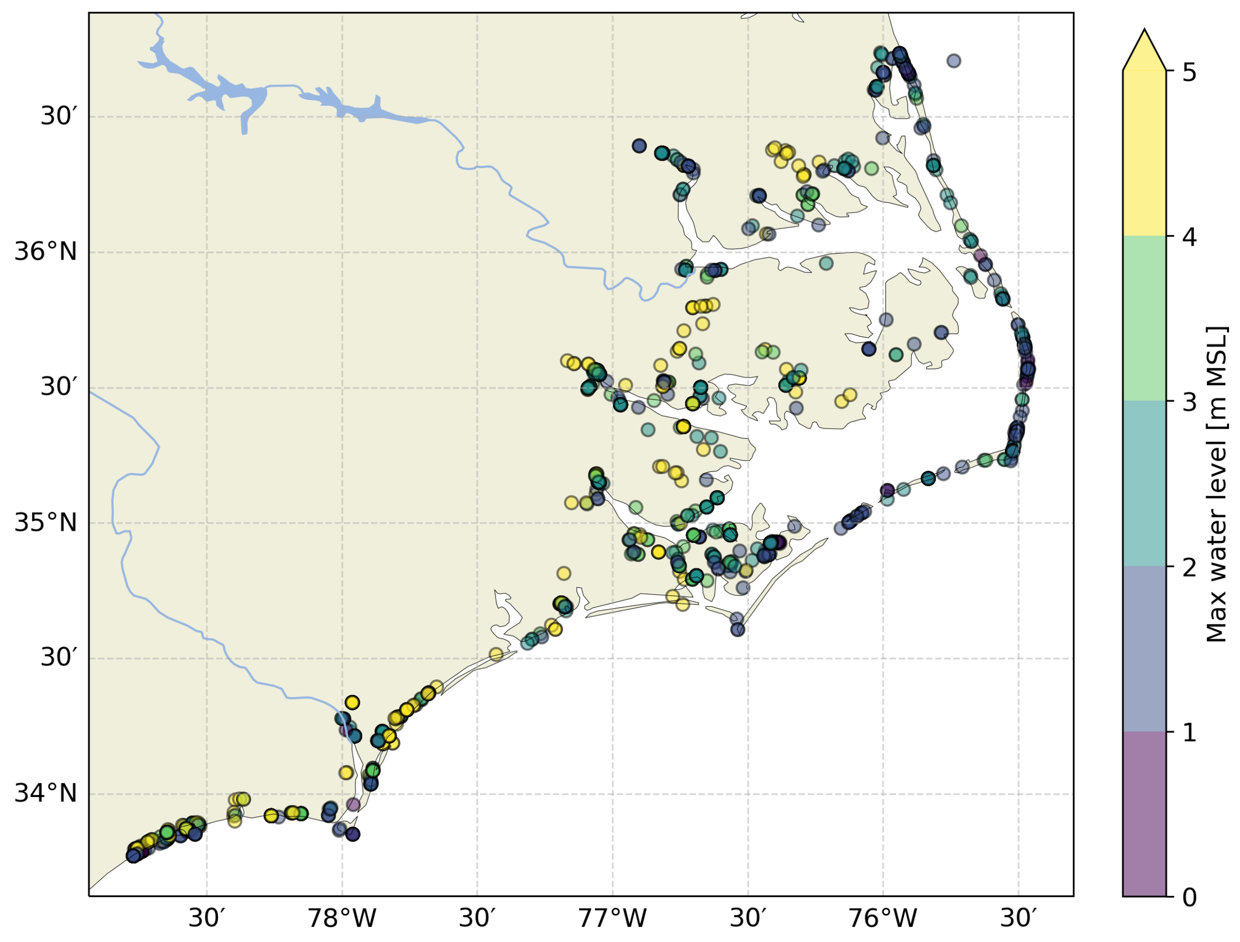


Figure 4.4: Location and value of the highest total water level generated by each storm on the NC coast. The color map indicates the value of the total water level.

(Figure 4.3). It is worth remembering that we also ran a tides-only simulation (without atmospheric forcing) for the same period, so we have astronomical tide predictions anywhere in the domain for the date ranges of all storms. The astronomical tide is fairly similar at both sites, but as the track passes much closer to Wrightsville than Duck (Figure 4.3), the total water level is about 1.8 m (MSL) at Wrightsville compared to 0.7 m (MSL) at Duck. The difference in the storm surge (about 1.5 m to 0.5 m) is in part because the Duck tide gauge is sheltered from the storm by the Outer Banks. The storm size also plays a role in the difference; both stations are about 270 km apart, and the radius to maximum wind speed at landfall is 40 km.

For each of the 1,813 maximum water level outputs, we obtained the location and value of the highest maximum water level (Figure 4.4). Most of the highest water levels, i.e. $\geq$ 4 m (MSL), are located in estuaries or the NC floodplains. For a large storm surge to be developed, the water must be piled up to a geographic feature such as a dune. For all storms, the maximum total water levels range from 0.6 to 9.6 m (MSL), with a median of 2.8 m (MSL). The maximum total water level of storm 0 (Figure 4.2) is 6.5 m (MSL) and occurred in a creek along the Pamlico River.

4.1.2 Aggregated downscaled maps

We downscaled the maximum water level output with Kalpana to produce high-resolution maps of peak total water levels. Each downscaled map has the same resolution and extent as the downscaling DEM we used for running Kalpana. We used a 3.5 GB downscaling DEM (Figure 2.1) that covers the entire NC coast with a resolution of 15 m. It is not feasible to work with 1,813 $\times$ 3.5 GB of data, so we exported the downscaled maps (raster files) as grayscale images with the peak total water level as the pixel intensity. For the downscaling, we binned the peak water levels from 0 to 10 m with intervals of 0.5 m. To transform the raster, which may have decimal numbers given the binning intervals, we amplified the value of each pixel by a factor of 2. This way, we forced the raster to have only integers so we could save it as a PNG image using an 8-bit integer representation. This drastically reduced the size of each downscaled map from 3.5 GB to about 2 MB. We created a world file with all the necessary georeferencing geospatial data to retain the raster's geographic information. This file allows the PNG images to be loaded into any geospatial software, such as QGIS or ArcGIS, and be displayed in the correct location.

Using Python, we load each grayscale image and divide the pixel intensities by 2. Then we combined the 1,813 maps to obtain statistical results. The first statistics we computed were the average and maximum (Figure 4.5) for each pixel through the entire library of downscaled peak total water level maps. In some areas, there are notable differences between the two maps. The average map has significantly smaller values on the sounds and estuaries, below 1 m (MSL), which means that only a few simulations generated a severe surge in those locations. Around Surf City, the difference exceeds 4.5 m. In the Neuse, Pamplico, and Albemarle estuaries, the difference is between 2 and 4 m, and on the sound side of Rodanthe Island, it is close to 3 m. The area between the Pamlico River and Albermarle Sound shows similar values in both maps, more than 2 m (MSL), indicating that this zone is more prone to flooding by storms that may not be considered extreme. The same occurs north of Albermarle Sound and between the Neuse and Pamlico Rivers, with yellow-colored areas on both maps.

Because each storm is equally probable (Bloemendaal et al. 2020), and assuming that our dataset is representative of the real conditions of tropical cyclones on the NC coast, it

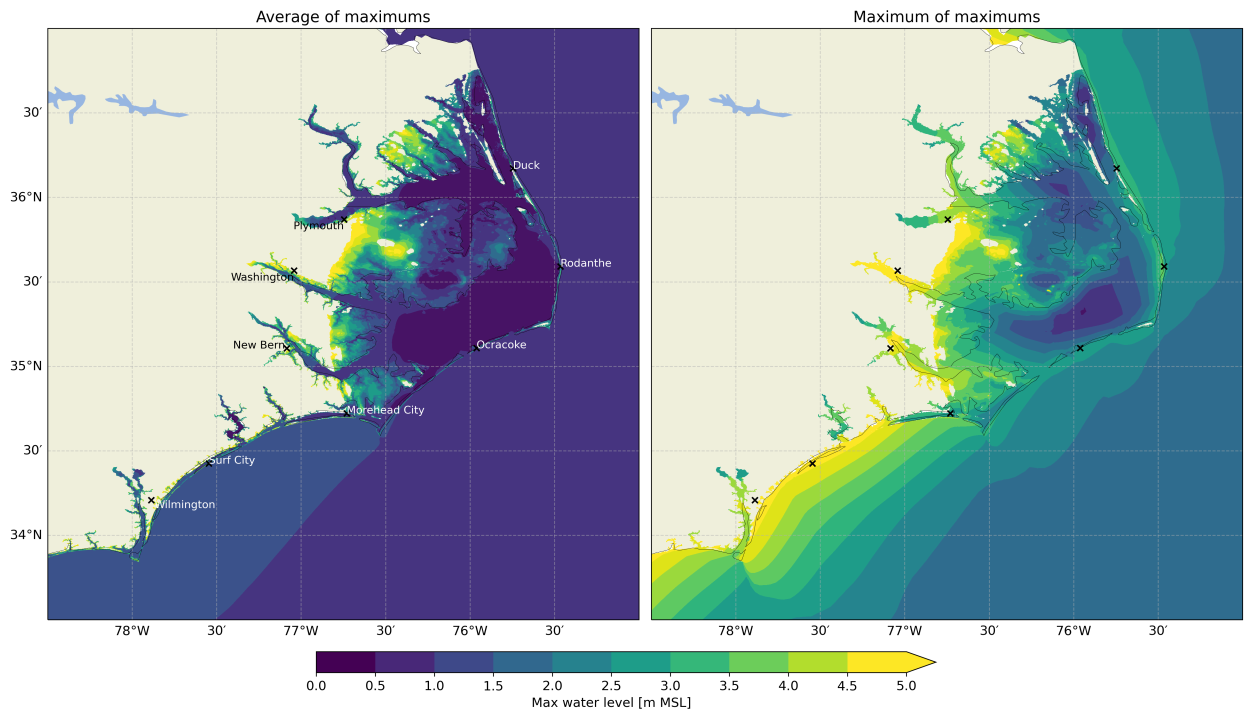


Figure 4.5: Total water levels in North Carolina, showing: (left) average of maximums, i.e. at the downscaling DEM scale, the average peak total water level for each raster cell within all downscaled maps; and (right) maximum of maximums, i.e. the maximum total water level for each raster cell within all downscaled maps. North Carolina's coastal towns are labeled.

is possible to define non-exceedance probabilities anywhere in the domain. Non-exceedance probability curves relate the magnitude of a variable and the probability that this variable exceeds a certain threshold. For example, at New Bern, the probability of having a peak total water level exceeding 2 and 3 m (MSL) is 14.7 and 2.4 percent, respectively (Figure 4.6).

We have the data to compute the exceedance probability on any points in the domain. So, if we define a peak total water threshold, we can analyze the probability of exceeding it anywhere in the domain. As an example, we computed the non-exceedance probability of 1 m (MSL) (Figure 4.7, left) and 2 m (MSL) (Figure 4.7, right). Most of the inner shelf from Morehead City to southwest has a probability larger than 50 percent of experiencing a total water level above 1 m MSL. The same happens in the Cape Fear, Neuse, and Pamlico estuaries, whereas the probability is close to 30 percent in the Albermarle River. Carteret County is the floodplain area most likely to experience 1 m (MSL) of flooding, with probabilities exceeding 40 percent. This is mainly because it is a low-lying region (Figure 2.1). Only a handful of coastal towns are prone (probabilities $\geq$ 20 percent) to experience 2 m (MSL) of flooding: Washington (Pamlico River), New Bern (Neuse River), Carteret County,

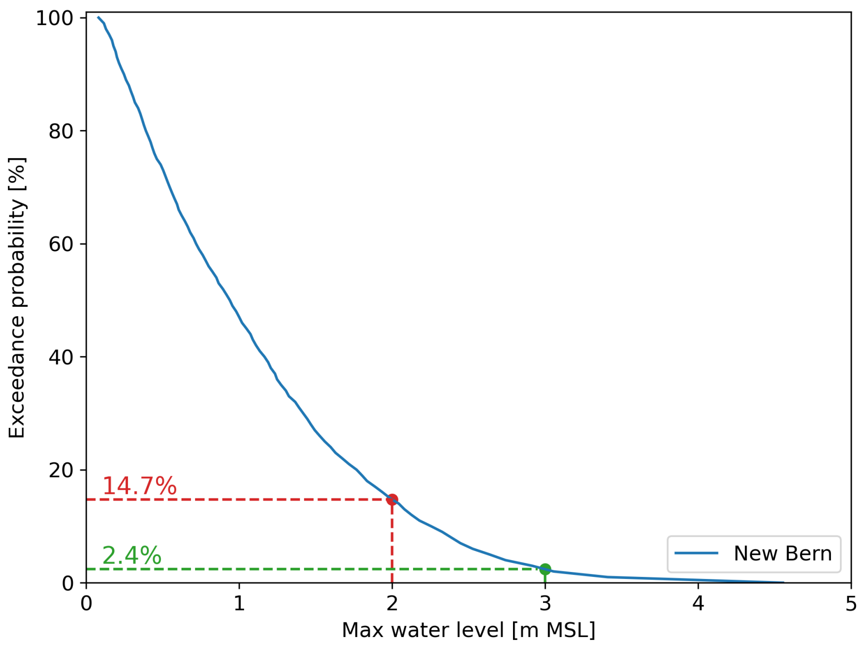


Figure 4.6: Peak total water level non-exceedance probability curve at New Bern, NC, highlighting the probability of exceeding peak total water level thresholds of 2 and 3 m (MSL).

and the coast between Surf City and Cape Fear River mouth.

4.2 Neural network

We aim to define a NN that can be extended to predict 2D maps of TC-driven peak total water levels. This aim then leads to a preference to have the same architecture, hyperparameters, and datasets for all locations across the NC coast. The hyperparameter tuning was done for the single model at the Beaufort NOAA tide gauge, whereas the feature engineering was done for the multitask model (only for the six NOAA tide gauge locations). In this section, we present the results of the multitask model to predict the total water levels at the six NOAA tide gauge locations, as well as locations in the Albemarle, Pamlico, and Neuse rivers. The input considers 20 input time series, including the offshore tide; distances from the storm's eye to the nine prediction locations; track parameters of wind speed, minimum pressure, radius to maximum wind speed, u and v vectors of the forward speed, and the FFT's magnitude of the storm parameters. This model is accurate, and it has the potential to be adapted in future work to predict maps.

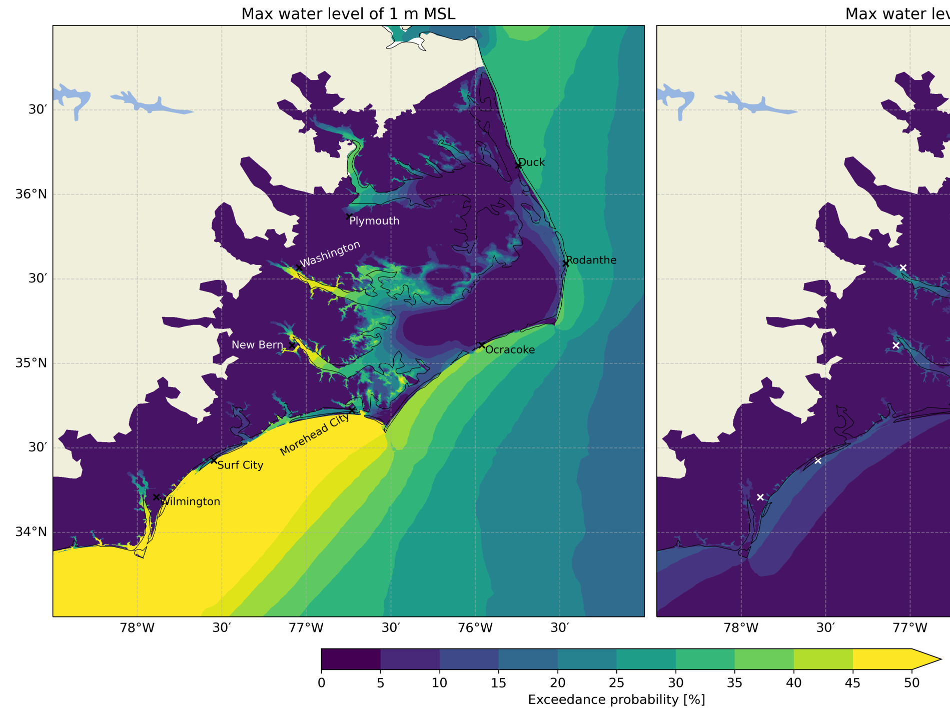


Figure 4.7: Probability map of exceeding a given peak total water level threshold. The left map shows the probability of exceeding 1 m of total water level at the downscaling DEM scale. The right map shows, similarly, the probability of exceeding 2 m of total water level. North Carolina's coastal towns are labeled.

4.2.1 Validation

The network parameters (biases and weights) are initialized at the beginning of training using the initialization scheme Xavier (Glorot and Bengio 2010). Then the network propagates the inputs through the forward pass to compute the loss, comparing the predictions and the true training set dependent variables. The selected optimizer, RMSprop, minimizes the loss with backpropagation to update the network parameters. During the model selection stage, at the end of each epoch, the neural network (NN) predicts the outputs for the independent variables in the validation set after all the training data has passed through the network. This process is repeated for the number of epochs specified by the user for training, which we set to 1,000. For validating the NN, we used a randomly selected 20 percent (15,708 storm-tide combinations) of the data as the validation set and the remaining 80 percent (62,883) storm-tide combinations for training. To prevent overfitting, we saved the model state (parameters and hyperparameters) for the epoch with the lowest validation loss .

The validation curve (Figure 4.8, top panel) shows how the training and validation loss evolves over the epochs. Both curves start decreasing slowly after 200 epochs to become

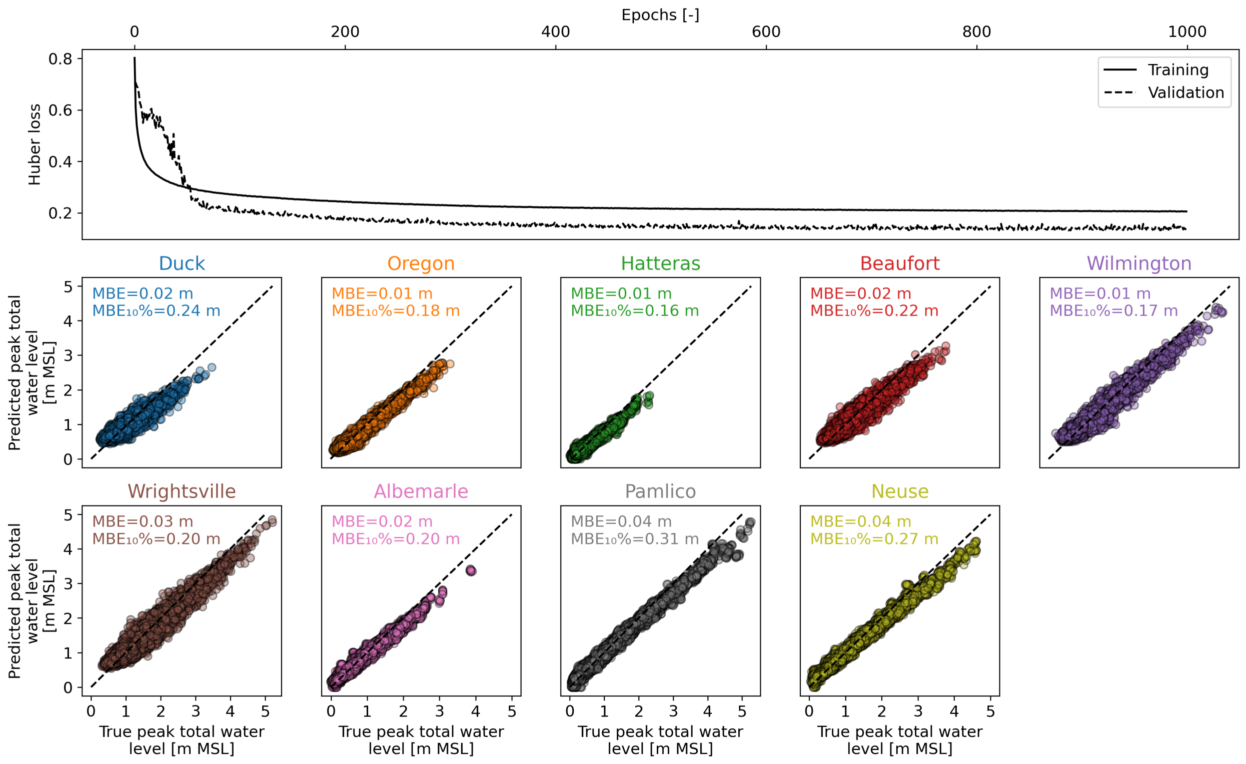


Figure 4.8: Validation of the multitask model, with panels showing: (top) training (continuous) and validation (dashed) curves for predicting at the nine prediction stations, and (middle, bottom) scatter plots of the validation set's true peak total water level on the horizontal axis and the model's predictions on the vertical axis. The dashed black line is the 1:1 curve.

stable after 400 epochs. At epoch 950, the model state with the lowest validation loss was saved. Scatter plots (Figure 4.8, middle and bottom rows) comparing the validation set's true values and the network predictions at epoch 950 show a good agreement for all nine prediction locations, especially looking at the metric of the full record. The MBE ranges from 1 to 4 cm. Pamlico and Neuse River locations have the largest mean bias error (MBE), which is 4 cm. The validation error for larger true peak total water levels is greater, which is expected given that the larger peak total water levels are underrepresented (Figure 3.12). Hatteras has the lowest $\mathrm{MBE}_{10\%}$ and the lowest peak total water levels. The largest $\mathrm{MBE}_{10\%}$ occurs at the Pamlico River, the location with the largest peak total water level. In general, the neural network performs very well for the validation set.

4.2.2 Testing

After the neural network was validated, we trained it without a validation set. We used the hyperparameters and inputs defined during the model selection stage (Table 4.1). The performance of the NN on unseen data is assessed during the testing stage. In this case, we used the full training set (78,591 storm-tide combinations) for training and tested the predictions comparing against the peak total water levels from a test set (13,872 storm-tide combinations).

Table 4.1: Multitask neural network's final hyperparameters.

Hyperparameter	Value
Learning rate	10^{-4}
Batch size	100
Epochs	950
Optimizer	RMSprop
Loss	Huber
Augmentation size	50
Input tide time series	Offshore
Storm location	Distance
Storm parameters	Raw and FFT

We quantified the model's overall performance using the root mean squared error (RMSE) and the $\text{MBE}_{10\%}$ for the extremes. The RMSE ranges from 8 cm at Hatteras to 19 cm at Pamlico River (Figure 4.9). The overall performance of the NN is good; at Wrightsville, the RMSE corresponds to 3% of the largest true peak total water level. The $\text{MBE}_{10\%}$ ranges from 15 cm at Wilmington to 43 cm at Pamlico River. Real (black edge dots) and augmented storms follow the same trend, thus the model does not perform worse in predicting the peak total water level of augmented storms. These results support the assumption to augment the data and show that the data augmentation process does not incorporate an extra source of error. The results highlight the NN's capability to predict the peak total water level with good agreement in places with different coastal process dynamics.

To better analyze the NN's errors, we binned the bias between the test set's true peak total water level and the predictions (Figure 4.10). The NN tends to underestimate the extremes, and this underestimation gets larger for high true peak total water levels. The magnitude

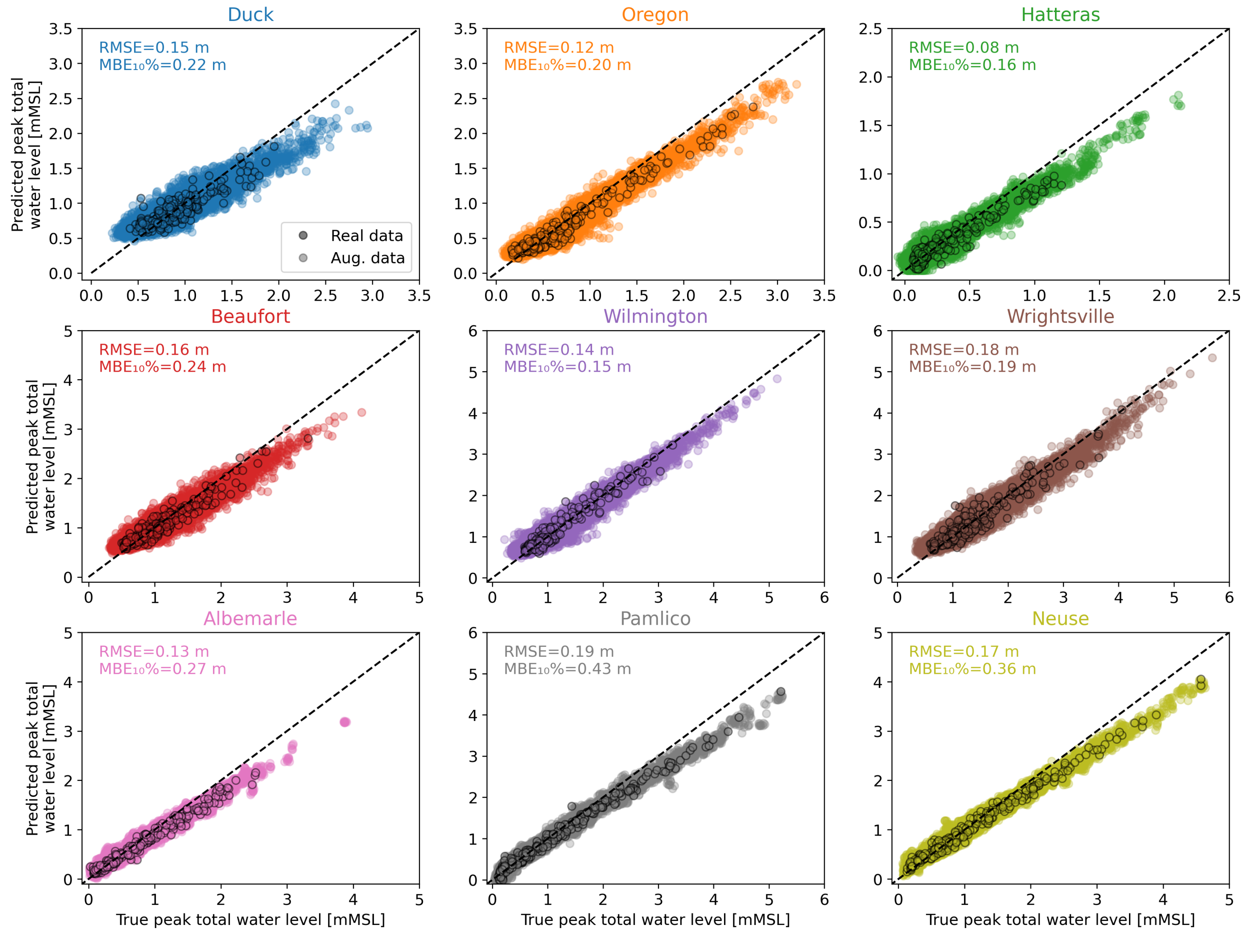


Figure 4.9: Multitask model's predictions for the testing set at the nine prediction locations. Each panel shows the RMSE and $MBE_{10\%}$ metrics. Black edge dots correspond to real storms, whilst others are augmented data. The horizontal axis shows the true values and the vertical axis shows the predicted values. Scatter plots do not share the same axis ranges.

of the bias is relatively stable across all stations for the same bin. The nine locations we are analyzing have different levels of sheltering and surrounding coastal environments. For example, Duck and Wrightsville are on the open coast, whereas Wilmington, Neuse, Pamlico, and Albemarle are upstream in river estuaries, so flooding drivers may differ. Moreover, the Oregon location is inside a marina on the sound side of a barrier island, and Beaufort is located in a complex coastal environment surrounded by islands and shallow areas. Having a similar bias distribution across stations for the same bins means that the neural network was capable of learning the different processes driving the flooding in all analyzed areas.

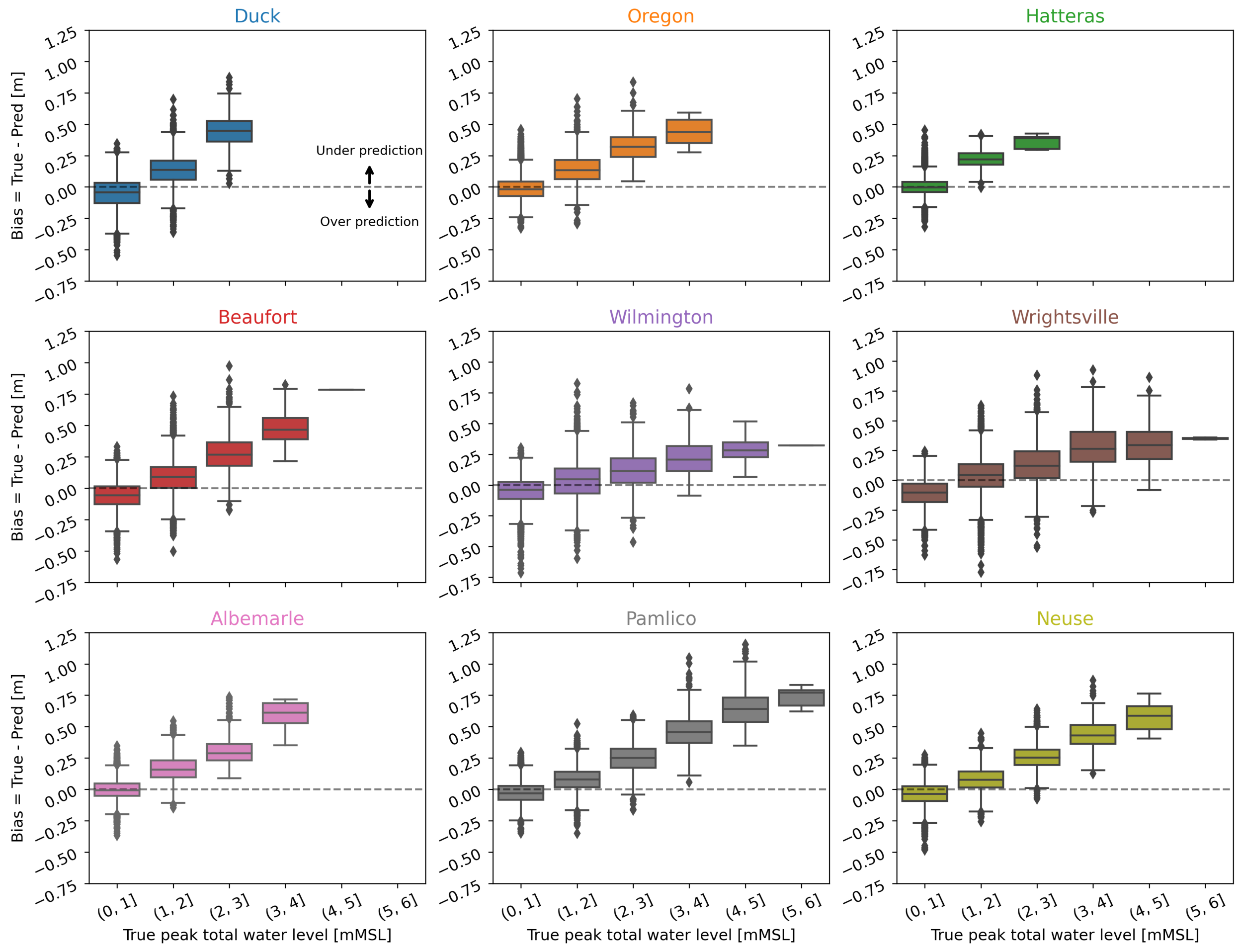


Figure 4.10: Box plot of the bias for binned test set's true peak total water level at NOAA tide gauge locations. The box corresponds to the interquartile range (IQR), $Q_3 - Q_1$. The bottom of the box is the 25 percentile (Q_1), and the upper limit is the 75 percentile (Q_3). The line inside the box is the median (Q_2). The lines outside the box are the *Maximum*: $Q3 + 1.5IQR$, and the *Minimum*: $Q1 - 1.5IQR$. The points above the maximum or below the minimum are outliers. All box plots share axis ranges. Some stations don't have water levels for all bins, e.g. there are no peak water levels above 3 m (MSL) at the Duck NOAA tide gauge location.

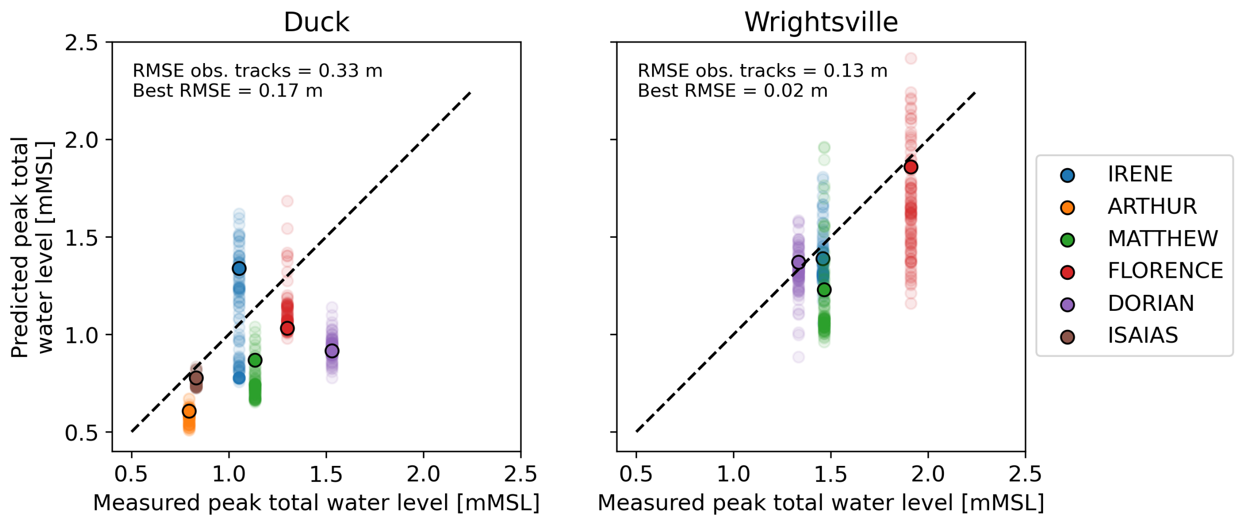


Figure 4.11: Scatter plots comparing the observed peak total water level of historic storms and the probabilistic prediction done by the NN for the observed and perturbed tracks. The horizontal axis shows the observed values and the vertical axis shows the predicted values. Scatter plots share the same axis ranges. Black edge dots correspond to the historical track prediction, whilst shaded dots are the perturbed track predictions. For each storm, all the predictions have the same horizontal value because there is one observed peak total water level. The black dashed line is the 1:1 curve.

4.2.3 Probabilistic prediction

After the neural network was validated and tested, we used it to consider the storm track uncertainty, which was one of the motivations for this research. In this section, we forced the NN with observed and multiple perturbations of real storm tracks. We predicted the total water level for the historic storms (Table 2.1) and perturbations created using CLIMADA (Aznar-Siguan et al. 2023), as described in Section 2.2.3. To address the performance of the probabilistic prediction, we compared the range of peak total water levels to the peak water level measured during the date range of each storm (Figure 4.11). We did this exercise at Duck and Wrightsville for six recent historic storms (Table 2.1); observations were unavailable for Arthur (2014) and Isaias (2020) at Wrightsville.

At Wrightsville, the predictions for the observed track of Irene (2011), Dorian (2019), and Florence (2019) show a good agreement with the measurements, whereas Matthew (2016) is underestimated. Nevertheless, the probabilistic prediction interval encloses the 1:1 line (perfect prediction). The RMSE between the prediction of the observed track and the measured values is 0.13 m. The neural network performs worse at Duck. The probabilistic prediction interval does not enclose the measured peak water level for Arthur (2014), Dorian (2019), and Matthew (2016). The peak total water level predicted for Irene (2011) is overestimated by approximately 20 cm, but the observed value is included in the probabilistic prediction interval. The RMSE between the observed track predictions and the measured values is 0.33 m. At both stations, the RMSE is similar to those obtained by high-fidelity ADCIRC simulations (Bilskie et al. 2022).

The errors we observed can be due to processes we did not include in the training library's development that are captured by the in-situ measurements. We didn't include far-field winds in our ADCIRC simulations, and they can increase the water level many hours before the storm is about to make landfall. The parametric wind model can contribute to the errors because it assumes a symmetric storm wind field. We adopted it because all its necessary inputs were available in the synthetic track dataset.

There is considerable variance in the probabilistic prediction interval width. For some storms like Arthur (2014), the peak total water levels predicted for all perturbations are close to the prediction for the observed track, whilst for Irene (2011), there is a maximum difference close to 1 m. Arthur (2014) and Isaias (2020) were relatively weak storms. The peak total water level probably has a large astronomical tide contribution, so perturbing the track does not affect much. Matthew (2016) and Dorian (2019) tracked shore-parallel from FL to NC. Most perturbed tracks are displaced either more offshore or inland almost parallel to the original track. At the Duck NOAA tide gauge, there is no considerable difference in the predicted peak total water level for the different perturbations, and this is mostly due to the sheltering that the Outer Banks provides. The perturbed tracks that shifted inland hit Duck more directly, but as they tracked inland, they suffered a considerable decay in the wind speed and pressure. There is a high variance in the predicted peak total water levels at Wrightsville. The perturbations that shifted offshore hit this location without any intensity decay. Florence (2018) originated near the coast of Africa, very far away from NC. A slight perturbation at this distance made the perturbations track away from the study site. Irene (2011) also shows high variance at both analyzed stations because the storm tracked crossing the Outer Banks affecting most of the NC coast.

When we developed the training library, the minimum ADCIRC runtime was 1.2 hours (Figure 4.1) for a 4.75-day simulation using 256 cores. Considering this runtime and aiming to simulate the six historic storms along with the 99 perturbations, we would need 184,320 CPU hours (256 cores $\times$ 1.2 hours $\times$ 600 storms). The wall clock time of this exercise depends on the available cores. For example, with 1,536 cores (6 $\times$ 256) to run six simulations in parallel, we would need five days (120 hours) to simulate all the perturbations. And that would be for just one forecast advisory, with another advisory to follow in 6 hr. This is not feasible in real-time forecasting. The neural network took 3.4 hours to train on a 64-bit

Intel® Xeon(R) Silver 4114 CPU 40 cores 2.20 GHZ desktop computer with a 5GB Nvidia Quadro P2000 GPU and 6 seconds to predict the peak total water level for the 600 tracks at both stations. We recognize that an ADCIRC simulation provides more comprehensive information, such as spatial and time-varying results, but we want to highlight how fast the neural network predicts. To the author's knowledge, this is the first attempt to predict peak total water levels for an ensemble of storm tracks using neural networks.

4.3 Discussion

4.3.1 Neural network

The performance of our neural network is similar to others available in the literature (Lee et al. 2021; Pachev et al. 2023).

Even when the goal is similar, predicting peak storm surge (or total water level in our case), significant differences exist between the three frameworks. Lee et al. (2021) predicted peak storm surge (without considering astronomical tides) using a portion of 39 hours of the storm track. One of the advantages of their framework is that it combines K-means, PCA, and spatial interpolation with the neural network. They implemented individual NNs for a set of points in the Chesapeake Bay and then upsampled the predictions spatially to several points. Unlike our NN, the Lee et al. (ibid.) framework does not include the interaction between storm surge and astronomical tides, the full storm track is not considered, and multiple neural networks are required to predict at the different locations, increasing the training burden.

Pachev et al. (2023) implemented a two-step framework to classify the prediction locations as wet or dry and then to predict the peak surge or total water level at wet points. They used two study sites, the Texas coast, where astronomical tides were not considered, and the coast of Alaska, where they did consider the interaction between the storm surge and the astronomical tides. This framework differs from ours in two main aspects. First, the temporal component of the storm track is considered through statistical parameters: minimum, mean, and maximum values. Second, the storm track is not considered as input for the NN. They interpolated the storm's wind and pressure field into the ADCIRC mesh. So for each storm, they had a set of (X_i, y_i) points near the coast (for the Texas application). For each point, the X_i input vector considers the atmospheric time statistics and local statistics of the surrounding topography or bathymetry. The y_i corresponds to the peak storm surge (or peak total water level for Alaska) at the mesh vertex. As they don't have a temporal component, this framework considers a Multi-Layer Perceptron (dense layers) to directly predict the peak surge (or total water level for Alaska) at many locations.

Lee et al. (2021) and Pachev et al. (2023) frameworks have the advantage of predicting at many locations with a good performance when we currently predict at nine. Our framework was designed to be extended to predict flooding maps in future research.

Of particular interest is the underprediction for extreme values (i.e. the highest peaks in the total water levels). Our NN's underpredictions are consistent with the performance for extreme values in other studies. This degradation can be related to the imbalances in the training libraries, even when libraries were specifically developed to analyze flooding for high return period (extreme) events as training datasets. In our case, we trained the NN with a dataset tailored to represent the average and extreme conditions of TCs in NC. The errors we obtained for extreme values were similar to those in Lee et al. (2021) and Pachev et al. (2023).

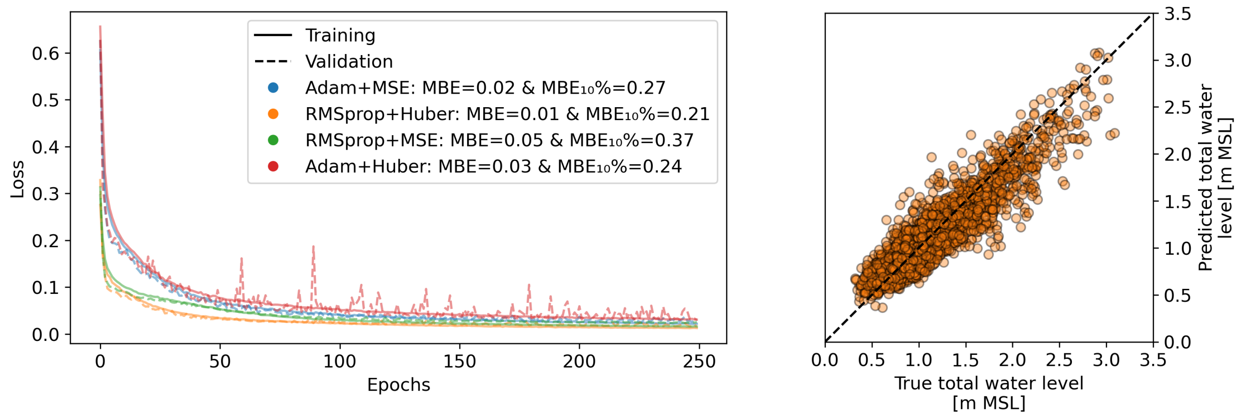
4.3.2 Model sensitivity

Hyperparameters

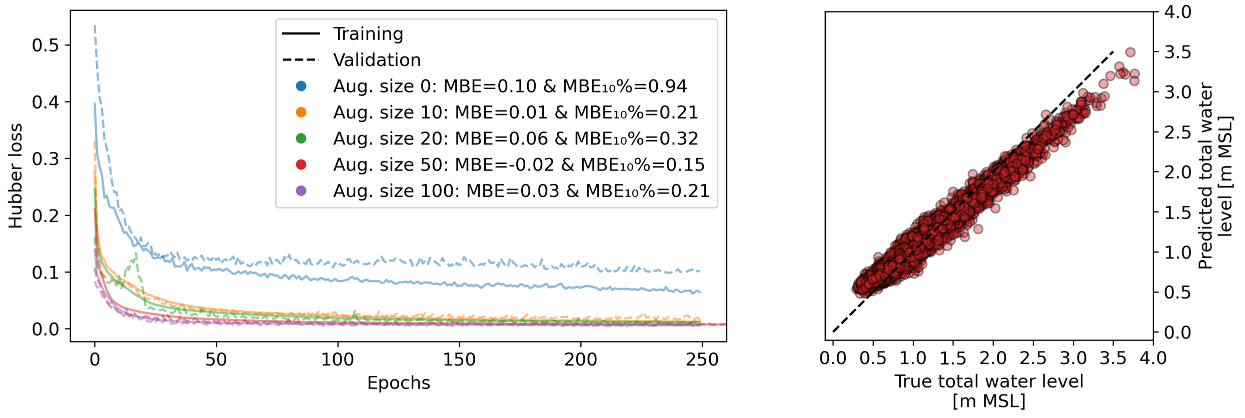
All neural networks are sensitive to their hyperparameters, and here we use a simplified version of our NN (single model) to analyze its sensitivity to the following hyperparameters: optimizer, loss, and augmentation size. We did the hyperparameter sensitivity only for the validation stage, i.e. without predicting for data unseen during training and using the single model for Beaufort. This prediction location is in a complex coastal environment, so it should be a challenging task to predict its peak total water levels. For hyperparameter selection, we considered the following input features: local tide, track parameters, and FFT of the track parameters.

We analyzed the combinations of Adam or RMSprop optimizers and MSE or Huber losses (Figure 4.12a). Using RMSprop optimizer with Huber loss gives the lowest MBE (0.01 m) and $\mathrm{MBE}_{10\%}$ (0.21 m) errors. Huber loss is less sensitive to outliers than MSE because it combines absolute and squared terms that contribute to a smoother optimization. Adam is typically preferred over RMSProp because of its adaptive learning rate for each parameter and robustness. Nevertheless, RMSprop may outperform Adam optimizer in specific cases, so experimenting with both optimizers is recommended (Géron 2022). In our specific problem, RMSprop performed slightly better, with a 3 cm lower $\mathrm{MBE}_{10\%}$ metric. The difference is a 12% of the error obtained using Adam.

We also analyzed the NN performance for different augmentation sizes. It is important to note that the validation set differs for all experiments because the datasets' sizes differ.



(a) Hyperparameters: optimizer and loss.



(b) Hyperparameters: augmentation size.

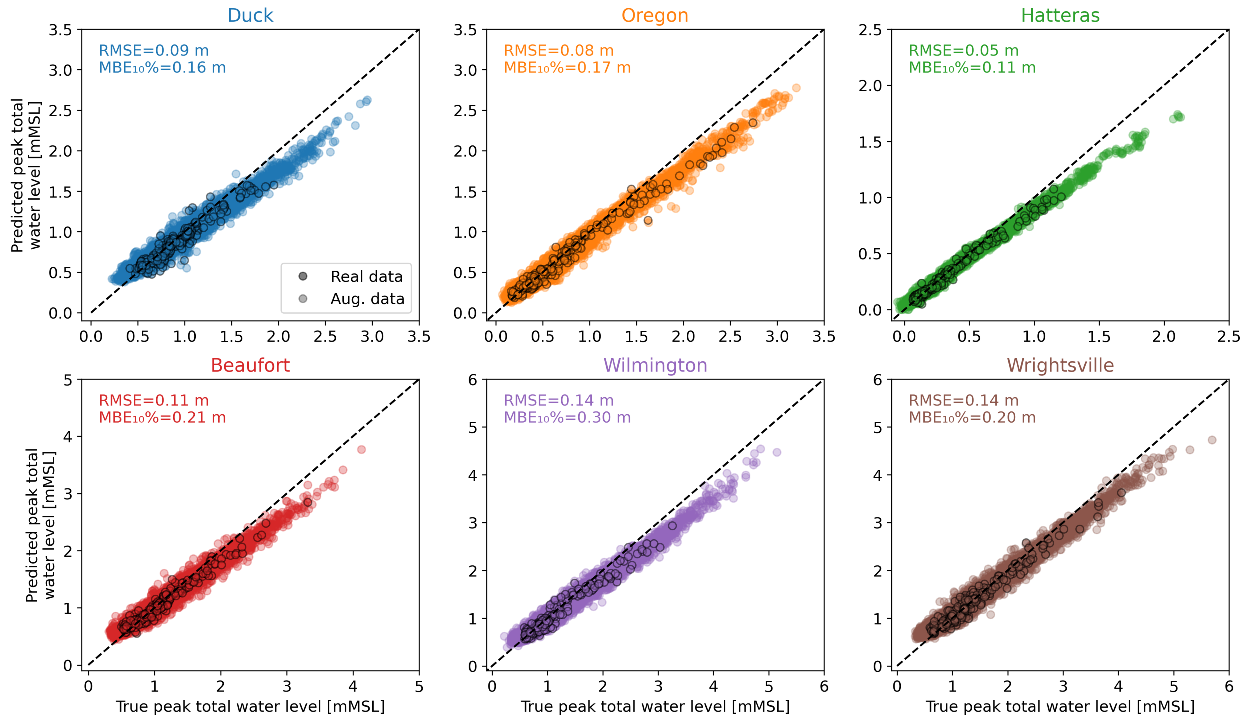
Figure 4.12: NN sensitivity to hyperparameters, with panels showing: (left) training and validation curves for different model's hyperparameters configuration at Beaufort, and (right) scatter plot of the validation set's true and predicted values using the selected hyperparameters; and subfigures showing sensitivity to (a) optimizer and loss, and (b) augmentation size. MBE corresponds to the mean bias error, and $\text{MBE}_{10\%}$ is the mean bias error of the validation set's largest 10% true values.

Our augmentation scheme (Section 3.3.5) varies only 1 of 20 columns in the input vector, so augmenting a storm many times may create very different targets with slightly different inputs. As far as we know, this is the first attempt to apply data augmentation for storm surge surrogate models. We hypothesize that other researchers didn't have the need because they used low-variance datasets for training. Lee et al. (2021) used a dataset of 1,050 storms (landfilling and bypassing), and Pachev et al. (2023) used a dataset of 446 landfilling storms for training the NN for the Texas coast. We validated the model using the augmentation sizes 10, 20, 50, and 100, and with no augmentation (Figure 4.12b). The NN performed poorly when using non-augmented storms. The MBE and $MBE_{10\%}$ were 10 and 94 cm respectively. We hypothesize that 1,813 storms are not enough for our proposed model given the dataset's high variance; it represents both the average and extreme conditions of tropical cyclones in NC. The validation results showed that more data is not always better; the errors for an augmentation size of 10 were 1 cm (MBE) and 21 cm ($MBE_{10\%}$) whereas 6 cm (MBE) and 32 cm ($MBE_{10\%}$) for an augmentation size of 20. For an augmentation size of 50, the errors were -2 cm (MBE) and 15 cm ($MBE_{10\%}$) whilst for an augmentation size of 100, the errors were 3 cm (MBE) and 21 cm ($MBE_{10\%}$). An augmentation size of 50 gives a slightly worse overall MBE than an augmentation size of 10 but negative (-2 vs 1 cm). Framing the problem to a real-time forecasting scenario, we prefer the slight overestimation. Regarding the $MBE_{10\%}$, an augmentation size of 50 gives a 6 cm more precise model. We hypothesize that more data is not always better because we are varying only 1 of 20 input time series in our augmentation procedure, and the output can differ in 1 m given Beaufort's tides range (National Oceanic and Atmospheric Administration 2023). When the augmentation size is too large, the NN may see data points with a similar input but a very different output.

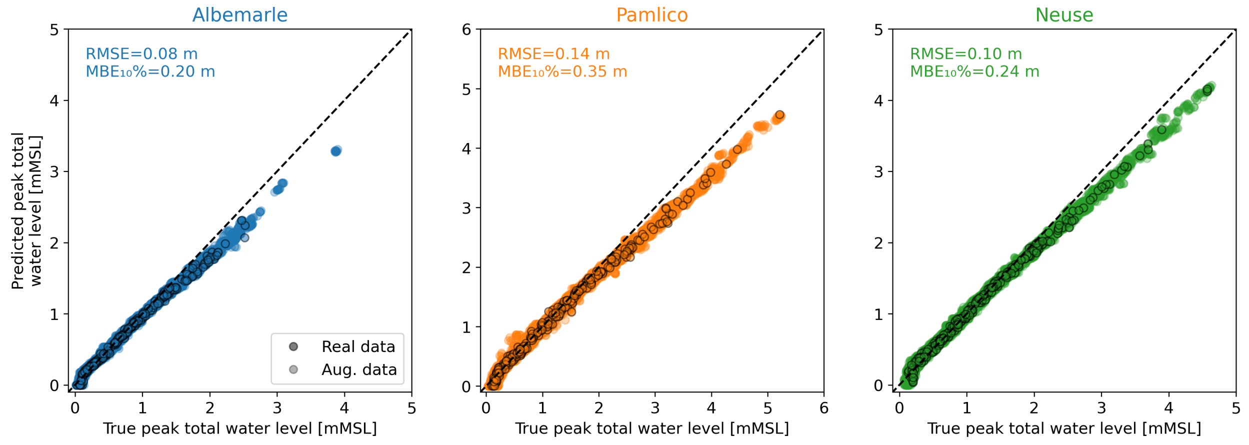
We tested the individual models' performance for all six NOAA tide gauge locations and the Albemarle, Pamlico, and Neuse river locations using RMSprop optimizer, Huber loss, and an augmentation of 50 (Figure 4.13). The overall RMSE ranges from 5 cm at Hatteras to 35 cm at Pamlico River. Even when this NN setup has smaller errors than the model presented in Section 4.2 ($MBE_{10\%}$ of 43 cm at Pamlico River), we discarded the individual model approach. The multitask model approach takes us one step closer of having an NN architecture to predict 2D spatially continuous maps of peak total water levels.

Augmentation method

We showed how the bias increases for larger true peak total water levels (Figure 4.10), which is mostly because our dataset is imbalanced (Figure 3.12). The same behavior was observed by Lee et al. (2021) and Pachev et al. (2023). To address this, we tried two different augmentation approaches focusing on oversampling large storms. It is worth noting that this



(a) NOAA tide gauge locations.



(b) NC river locations.

Figure 4.13: Predictions of individual NNs (single models) for the testing set at the different NOAA tide gauge locations and NC rivers. Each panel shows the RMSE and $\mathrm{MBE}_{10\%}$ metrics. Black edge dots correspond to real storms, whilst others are augmented data. The horizontal axis shows the true values and the vertical axis shows the predicted values. Scatter plots do not share the same horizontal or vertical axis values.

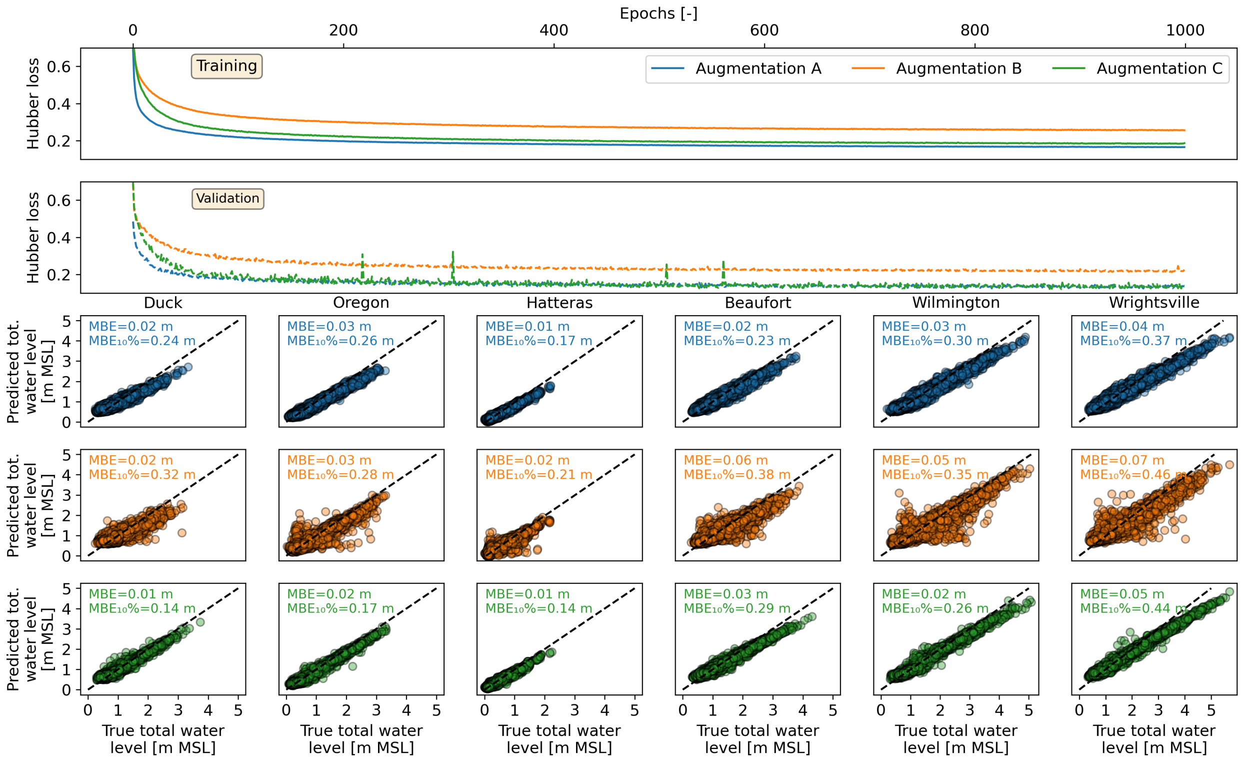


Figure 4.14: Sensitivity to augmentation method, with panels showing: (top two rows) training and validation loss curves for the augmentation method (Section 3.3.5) and two variations; and (bottom three rows) scatter plots to compares the NN predictions and the validation set's true peak total water levels. The colors across the subplots are consistent. It is important to mention that each case's validation set was randomly selected so they may differ.

validation was done for the multitask model to predict only at the six NOAA tide gauge locations (Figure 4.14). The augmentation A augments all storms 50 times by assigning a randomly selected astronomical tide. This method was explained in detail in Section 3.3.5. In augmentation B, for each prediction location, we sorted the storms in terms of the peak storm surge (difference between peak total water level and astronomical tides). In this case, we augmented only the storms with a peak surge within the largest 10% by assigning a randomly selected astronomical tide to each storm surge time series. In augmentation C, we augmented all storms 25 times by aligning the peak storm surge with a randomly selected diurnal high tide. This method differs from A and B because we constrain the peak storm surge to occur simultaneously with a high tide value. We found the daily peaks of the astronomical tide from the two-month tide-only simulation at each prediction station. Augmentation B (orange) had the worst performance; it didn't improve the performance of the extremes and added considerable noise. Both error metrics, MBE (4 cm on average) and MBE$_{10\%}$ (33 cm

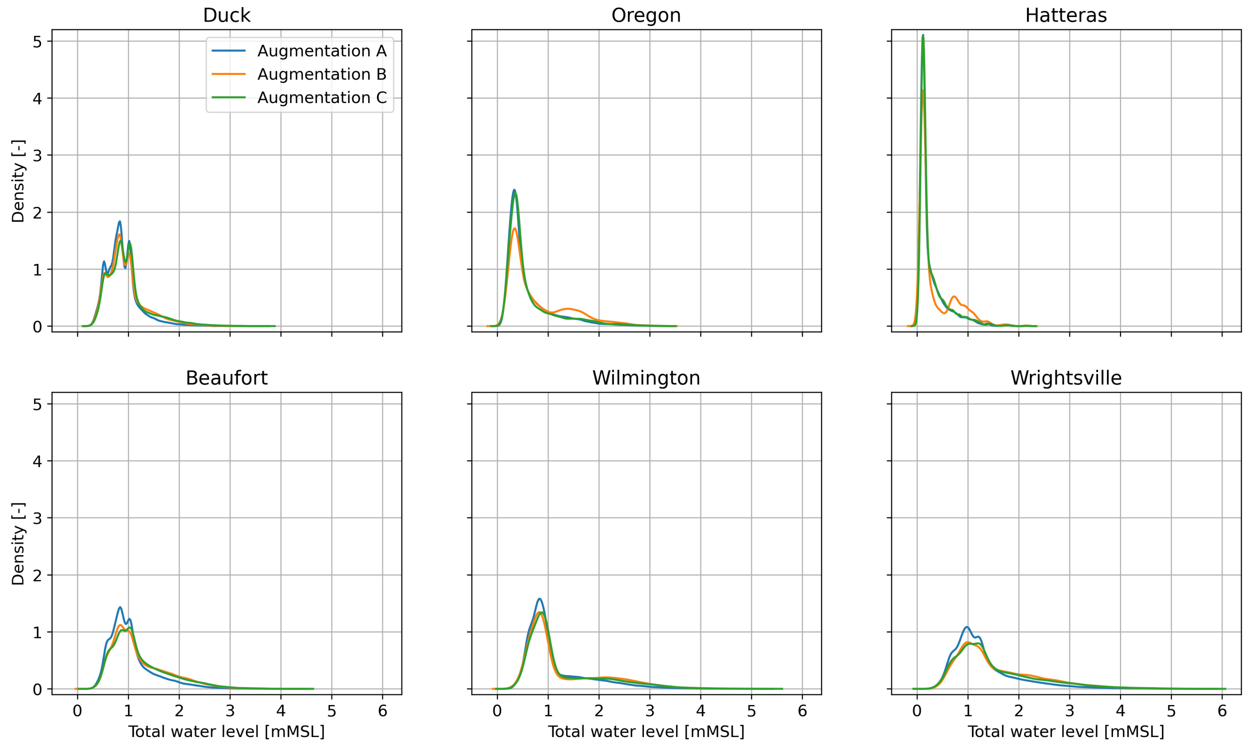


Figure 4.15: Kernel density plots of the peak total water level distribution combining the train and test sets for the different augmentation methods.

on average), are larger or equal to Augmentation A (2.5 and 26 cm on average). On average, Augmentation C (MBE of 2.3 cm and $\text{MBE}_{10\%}$ of 24 cm) method performs better than the Augmentation A (MBE of 2.5 cm and $\text{MBE}_{10\%}$ of 26 cm) method across stations (MBE: 2.3 vs 2.5 cm,). Nevertheless, we discarded it because its implementation is more convoluted and because the $\text{MBE}_{10\%}$ metric worsened by 7 cm at Wrightsville, the station with the larger peak total water levels.

The histograms (Figure 4.15) show that none of the proposed augmentation methods balanced the dataset. The goal of trying different augmentation methods was to reduce the variance in our dataset and to force it to be more representative of the extreme values such as the datasets used in Lee et al. (2021) and Pachev et al. (2023). Balancing the dataset for all stations is challenging. Coastal NC is very large; storms that may create significant flooding at Wrightsville may not be severe at Duck. So, when augmenting extreme events for Duck, there is a high chance that more average storms will be added to the dataset at Wrightsville. This challenge arises from our decision to use the same model architecture, hyperparameters, and dataset for the full study site.

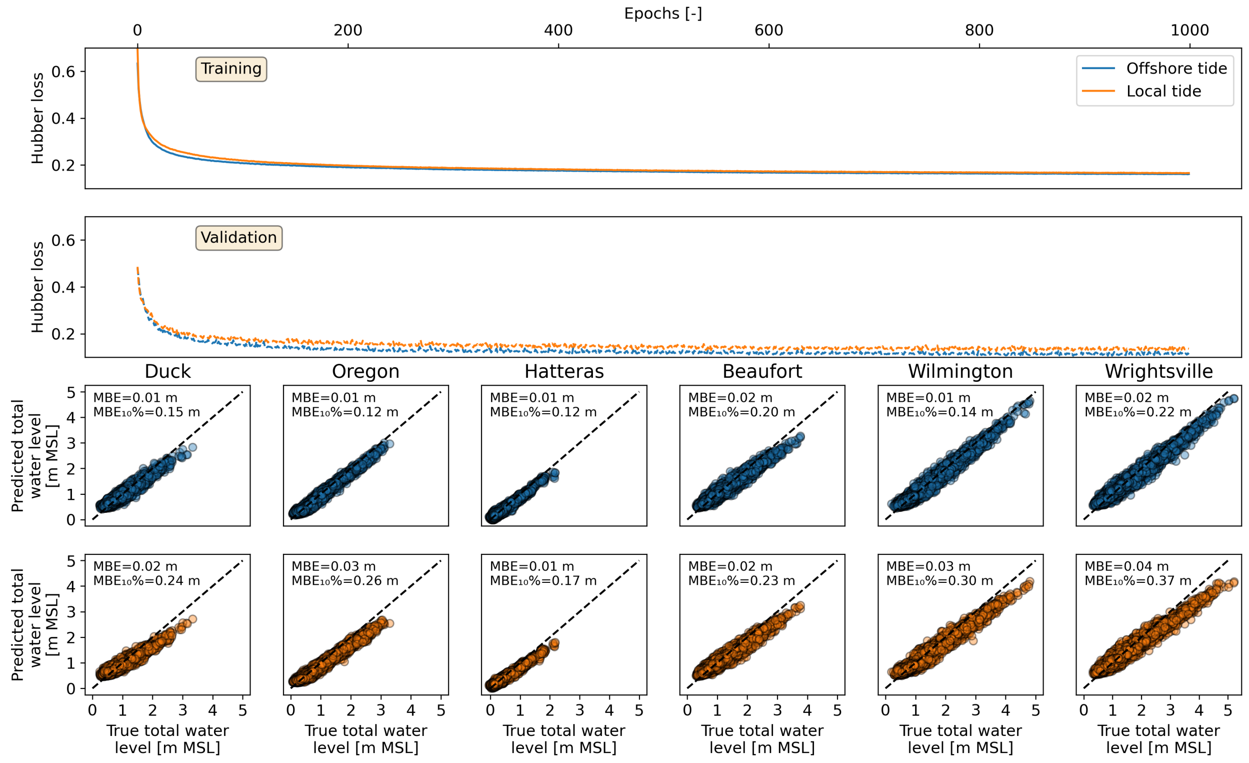


Figure 4.16: Sensitivity to input tide location (local or offshore), with panels showing: (top two rows) NN's training and validation curves; and (bottom three rows) scatter plots comparing the NN predictions and the validation set's true peak total water levels. The colors across the subplots are consistent. It is important to mention that each case's validation set was randomly selected so they may differ.

Feature engineering

So far, the validations have considered local astronomical tides as inputs. Local astronomical tides in a location such as Beaufort are not easy to estimate, and a hydrodynamic model or long-term measurements may be required. The tides interact with the bathymetry and coastline in nearshore areas, so non-linearities play a significant role. On the other hand, offshore tides can be predicted using Equation 3.2 with constituents from global tide models such as DTU10 (Cheng and Andersen 2010). We aim to create the simplest framework that gives accurate enough results to be extended later on to predict spatially continuous 2D maps of peak total water levels so offshore tides are preferred, even when in sheltered areas offshore tides do not fully explain local tides.

The NN considering the offshore tide performs better on the randomly selected validation set (Figure 4.16). In this case, the multitask model predicts on all six NOAA tide gauges so when using the local tide as input, it requires six local tide time series. In contrast, only one tide time series is required when using the offshore tide as input, independent of how many output points the model has. Pachev et al. (2023) considered the astronomical tides as one of the NN's inputs for the Alaska application. They used the amplitude and phase of the eight major tidal constituents obtained from ADCIRC simulations without the atmospheric forcing on each mesh vertex. This makes this framework difficult to use in real-time forecasting because a tides-only simulation will be required to get the tide-related necessary inputs to force the NN. Adopting the offshore tide simplifies the model because fewer inputs are required and also because it is easier to obtain.

In general, having a neural network with fewer inputs is preferred. Here, we explore different input combinations we tried for training the multitask neural network model. The definitive multitask NN presented in section 4.2 considers the distance from the storm's eye to the nine points to predict, some storm track parameters, and its FFT magnitude as inputs. We tried removing the FFT inputs (five parameters) to simplify and reduce the input dimensionality. Additionally, we trained the model including the coordinates of the storm's eye as the only indicator of the storm's position and proximity to the points to predict. This also aims to reduce the input dimensionality because only the eye's lat/lon coordinates are required; in summary, if the model predicts at n points, considering the storm eye's coordinates reduces the input dimensionality in $n - 2$ variables. We showed (Section 2.3.1) that in some cases there is a clear correlation between the storm's proximity to the prediction location and the peak total water level experienced at that point (Figure 4.3). When considering only the storm's eye coordinates, the NN is unaware of where the point to predict is located, so it doesn't have any proximity information.

We compared the multitask model performance (Figure 4.17) on a randomly selected validation set for three input configurations: (1) including distance and FFT magnitudes, (2) using eye's coordinates and FFT magnitudes, and (3) using eye's coordinates and without FFT magnitudes. The NN considering distances and FFT inputs performs better. On average across all six stations, the MBE and $\text{MBE}_{10\%}$ metrics are 2 and 10 cm lower for the definitive multitask NN, respectively.

Including the FFT inputs improves the model performance and it is relatively direct to compute. There are multiple ways to compute the Fast Fourier transform of time series using Python. On the other hand, considering the distances instead of the storm's eye coordinates makes the extension of the NN to predict maps more complex. In the case of predicting a 2D spatially continuous map, it is not feasible to include the storm's eye distance to all pixels. The downscaled maps we produced have 475,541,017 pixels, so the amount of memory to process input time series with ≈ 475 million parameters would be tremendous. A set of representative points would need to be defined beforehand to sort this issue. For example, we

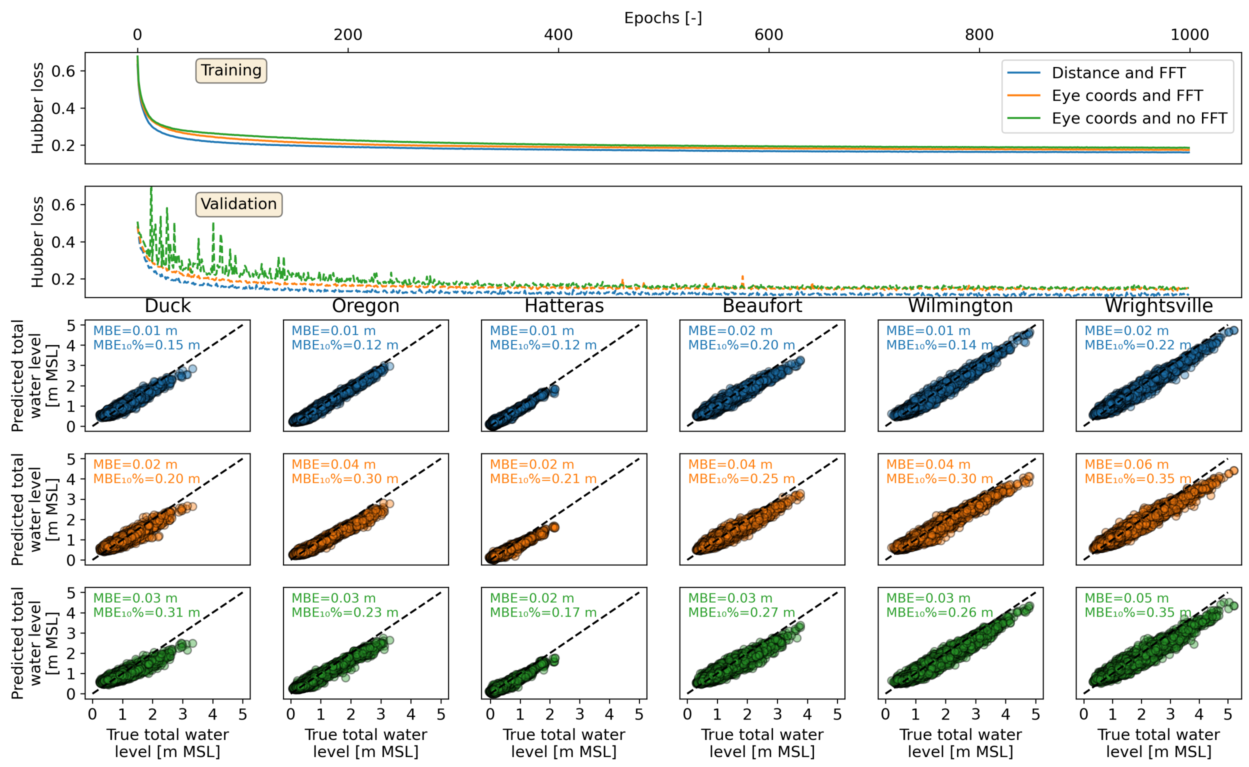


Figure 4.17: Sensitivity to input combinations, with panels showing: (top two rows) NN's training and validation curves; and (bottom three rows) scatter plots comparing the NN predictions and the validation set's true peak total water levels. The colors across the subplots are consistent. It is important to mention that each case's validation set was randomly selected so they may differ.

can use the information from the map showing the location and value of the peak total water level for each simulation (Figure 4.4) as input for a clustering technique such as K-Means (Lloyd 1982; MacQueen et al. 1967). We can compute the distances from the storm's eye to the clusters and use those time series as inputs for the neural network.

4.3.3 Probabilistic prediction framework

Having a framework capable of predicting peak total water level in seconds would allow forecasters to predict flooding for several perturbations of the storm track, thus addressing track uncertainty. The NHC issues advisories with the consensus track and multiple scenarios that define a cone of uncertainty. Figure 4.18 shows the uncertainty cones for multiple advisories of Ian in 2022. With process-based hydrodynamic models like ADCIRC, it may not be feasible to simulate all storms in the ensemble, whereas with a neural network, the peak total water level can be predicted for the multiple storms that define the uncertainty

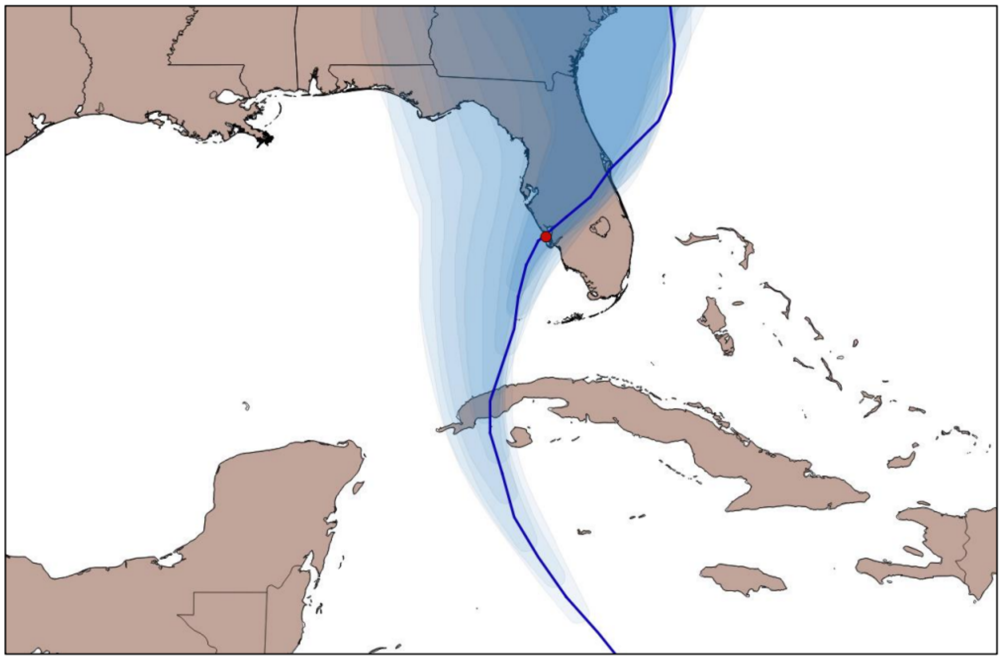
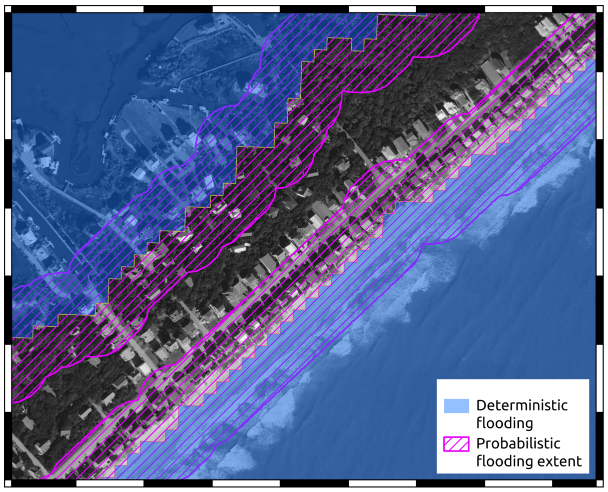


Figure 4.18: Best track of Ian (2022) in solid blue with forecasted uncertainty cones from 00 UTC 26 September to 06 UTC 28 September 2022 Bucci et al. (2023).

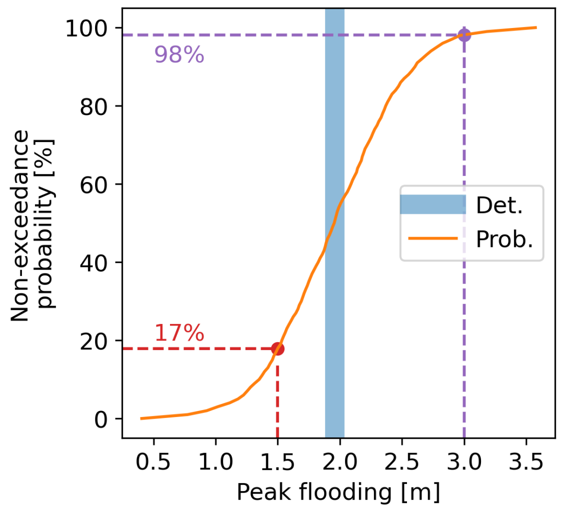
cone.

Two possible outcomes of a probabilistic prediction framework are flooding extent maps with uncertainty (Figure 4.19a) and non-exceedance curves of peak total water level at specific points (Figure 4.19b). In the first example (left), multiple inundation extents can be predicted to incorporate uncertainty. The magenta hatched area indicates the area defined by the probabilistic prediction of inundation extent. This area incorporates all the inundation extents predicted for the different track perturbations. It assumes that all storms are equally probable. The blue area shows the single result of a deterministic framework. The second example shows a non-exceedance curve defined by predicting peak flooding for multiple scenarios and the deterministic as a vertical line. The deterministic result indicates a peak flooding of 2 m, whilst the probabilistic framework gives the probability associated with different flooding levels. For instance, there could be probabilities of 17 percent and 98 percent of having water levels below 1.5 and 3 m, respectively. A probabilistic framework may improve hazard communication and reduce the errors in flooding predictions.

Communicating uncertainty to the public is challenging, but they want information about it during extreme weather events such as tropical cyclones (Morss, Demuth, and Lazo 2008). The main challenge is communicating uncertainty in a good way so that users can accurately interpret risks and take appropriate actions (Colle et al. 2021); otherwise, the forecast does



(a) Probabilistic 2D result



(b) Probabilistic result at stations

Figure 4.19: Examples of results from a probabilistic prediction system. Left: map with the inundation extent of a deterministic modeling framework (blue) and the area defined by the possible inundation extent (pink hatched) given by a probabilistic framework. Right: Non-exceedance curve of peak flooding at a point. The blue vertical line shows the deterministic peak flooding, whereas the orange curve gives the probability of non-exceeding flooding thresholds. In this example, there is a probability of 17 percent of non-exceeding 1.5 m and a 98 percent of non-exceeding 3.0 m of peak flooding. Both examples correspond to a demonstration of possible outcomes of a probabilistic prediction framework using synthetic data.

not have intrinsic value (Murphy 1993). A probabilistic prediction framework may require forecasters to interact with social science researchers to define communication strategies. The main goal of a real-time forecasting system is to provide information for emergency managers and decision-makers so they can plan safety measures; if the probabilistic framework does not contribute to this end, it may not be useful.

Chapter 5

Conclusions and Future Work

Flooding during tropical cyclones (TC) is a severe hazard for coastal communities. Providing accurate, high-resolution, and timely predictions of TC-driven flooding is critical for management during and between storms. This research developed a neural network to predict peak total water levels from TC-driven coastal flooding for North Carolina (NC), considering astronomical tides and storms of any duration as inputs, and trained with a dataset representative of the extreme and average conditions. We simulated 1,813 tropical cyclones using ADCIRC from a dataset of synthetic tracks created with a fully probabilistic model to develop the training library. Our dataset represents the average and extreme tropical cyclone conditions in NC. We used this data to train, validate, and test a neural network composed of 1D convolutional and dense layers to make point-wise predictions of peak total water levels using the entire storm track and astronomical tides as inputs. The neural network's capability of making predictions in seconds allowed us to demonstrate a probabilistic approach in which flooding was predicted for several perturbations of historical storm tracks. This work takes us one step closer to having a neural network framework for making probabilistic predictions of spatially continuous maps of peak coastal flooding for NC.

The major takeaways of the study are:

- *A training library of storm-tide simulations was developed using unique and realistic storms.* From a dataset of synthetic tracks based on the historical record, we identified a subset of storms with likelihood of generating flooding in NC. We used ADCIRC to simulate the 1,813 storms, and then we used Kalpana to downscale the peak storm surge outputs to produce high and constant resolution maps of coastal NC, including our prediction locations. From this training library, the aggregated maps can quantify the magnitude and locations of the worst flooding, e.g. the high probabilities (greater than 50 percent) of total water levels higher than 1 m along the open coast south of

Cape Lookout and in the Pamlico and Neuse River estuaries (Figure 4.7).

– *A deep learning model was developed with 1D CNNs and dense layers to predict at multiple stations simultaneously.* The many-to-one neural network predicts peak total water levels (single value in time) at the prediction locations provided for training, using the full storm track and offshore astronomical tide as inputs. The multitask NN to predict at the nine prediction locations considers 20 inputs. Five inputs relate to the storm track: wind speed, minimum pressure, radius to maximum wind speeds, and u and v vectors of the forward speed, along with their FFT magnitude. It also considers the distance from the storm's eye to each prediction point and the offshore astronomical tide. The multitask NN has 122,009 trainable parameters and took less than 4 hours to train in a standard desktop workstation.

– *The dataset was augmented, increasing the amount of data from 1,813 to 92,463 storm-tide combinations.* Assuming that the astronomical tides and the storm surge were independent, we computed the storm surge as the difference between the total water level and the astronomical tide. We augmented 50 times each of the 1,813 storm surge series by adding a randomly selected astronomical tide.

– *The NN performed well for all nine prediction stations.* We reserved 15 percent of the data for testing, and then split the remaining 85 percent by using 80 percent for training, and 20 percent for validation. The average mean bias error (MBE) was 2 cm for the validation set, and the $\text{MBE}_{10\%}$ was 22 cm. For data unseen during training, the averaged RMSE and $\text{MBE}_{10\%}$ were 15 and 25 cm, respectively. The performance of the NN is good and similar to other works available in the literature. The model performed similarly to predict the peak total water across the NC's coast. The largest RMSE (19 cm) occurs at Neuse River whereas the minimum RMSE (8 cm) occurs at Hatteras.

– *The NN's bias varies almost linearly with the magnitude of the peak water level.* There is a clear relationship between the NN's bias and the magnitude of the true peak total water level, which is related to how imbalanced the training dataset is. The augmentation method used contributed to having an imbalanced dataset. Given the size of the study site, it was challenging to oversample the extreme storms in all stations simultaneously. An extreme storm at Duck may not be extreme at Wrightsville.

– *A potential use for the neural network was demonstrated via probabilistic predictions with perturbations of recent historic storm tracks.* We predicted a range of peak water levels at the Wrightsville NOAA tide gauge for Irene in 2011, Matthew in 2016, Florence in 2018, and Dorian in 2019. At Duck, we also predicted for Arthur in 2014 and Isaias in 2020. The RMSE of the DL-predicted peak total water level for the observed track is 33 cm at Duck and 13 cm at Wrightsville. The accuracy is slightly below that obtained with high-fidelity models but a much shorter time. The peak total water level intervals defined by the probabilistic predictions may help improve hazard communication during real-time forecasting.

In future research, the library of high-resolution maps of TC-driven peak coastal flooding can be used for risk analysis to determine how exposed NC's coastal towns are to flooding. Flooding hotspots can be identified, and it can motivate the implementation of smaller-scale hydrodynamic models to test mitigation measures. More sophisticated recurrent neural networks, such as Long Short-Term Memory (LSTM) units, may be tested in future research to improve the NN performance, specifically on extreme values. Also, other augmentation schemes focusing on oversampling the extremes need to be developed to decrease the $\mathrm{MBE}_{10\%}$ metric. In real-time forecasting, it is important not to underpredict high peak total water levels. It may be worth increasing the training library only with extreme storms.

We included the probabilistic prediction to show a possible neural network application. It may be worth using a more sophisticated method to create the historic track perturbations, varying the wind speed, pressure drop, or radius to maximum winds, not only the eye's coordinates. In future research, this will allow to quantify the performance of the probabilistic prediction framework. Another possible application of this framework is to analyze how the width of the probabilistic prediction interval varies with the lead time in the NHC advisories. This will allow understanding how the uncertainty in the NHC advisories evolves during the forecasts of real events.

Currently, the NN predicts a fixed number of stations provided for training. The framework will be expanded to predict 2D spatially continuous maps of peak total water level for the NC coast. We will replace the dense layers in our NN architecture with a set of transposed 2D convolution layers. Predicting maps is of considerable importance to implement our NN in real-time forecasting. The flooding hotspots may vary from storm to storm, so having an NN capable of predicting flooding for the entire NC coast is preferred.

BIBLIOGRAPHY

A Latto, A Hagen and R Berg (2021). *Tropical Cyclone Report, Hurricane Isaias (AL092020), 30 July – 4 August 2020.* Tech. rep. National Hurricane Center.

ADCIRC (2020). *ADCIRC Utility Programs.* `https://adcirc.org/home/related-software/adcirc-utility-programs/`. [Retrieved 11 May 2020].

Adeli, Ehsan et al. (2022). "An advanced spatio-temporal convolutional recurrent neural network for storm surge predictions". In: *arXiv preprint arXiv:2204.09501.*

Agarap, Abien Fred (2018). "Deep learning using rectified linear units (relu)". In: *arXiv preprint arXiv:1803.08375.*

Ayyad, Mahmoud, Muhammad R Hajj, and Reza Marsooli (2022). "Artificial intelligence for hurricane storm surge hazard assessment". In: *Ocean Engineering* 245, p. 110435.

Aznar-Siguan, Gabriela et al. (Sept. 2023). *CLIMADA-project/climada_python: v4.0.1.* Version v4.0.1. DOI: `10.5281/zenodo.8383171`. URL: `https://doi.org/10.5281/zenodo.8383171`.

Backstrom, Joni T, Carlos Loureiro, and Devon O Eulie (2022). "Impacts of Hurricane Matthew on adjacent developed and undeveloped barrier islands in southeastern North Carolina". In: *Regional Studies in Marine Science* 53, p. 102391.

Bai, Long-Hu and Hang Xu (2022). "Accurate storm surge forecasting using the encoder–decoder long short term memory recurrent neural network". In: *Physics of Fluids* 34.1, p. 016601.

Berg, R (2015). *Tropical Cyclone Report, Hurricane Arthur, 1 - 5 July 2015.* Tech. rep. National Hurricane Center.

Bilskie, M V et al. (2022). "Real-Time Simulated Storm Surge Predictions during Hurricane Michael (2018)". In: *Weather and Forecasting* 37, pp. 1085–1102.

Blanton, B O and R A Luettich (2008). *North Carolina Coastal Flood Analysis System: Model Grid Generation.* Tech. rep. TR-08-05. Renaissance Computing Institute.

— (2010). *North Carolina Floodplain Mapping Program, Coastal Flood Insurance Study: Water Level Validation Study.* Tech. rep. TR-10-06. Renaissance Computing Institute.

Bloemendaal, N et al. (2020). "Generation of a global synthetic tropical cyclone hazard dataset using STORM". In: *Scientific Data* 7.40. DOI: `10.1038/s41597-020-0381-2`.

Bucci, L et al. (2023). *Tropical Cyclone Report for Hurricane Ian, 23 September – 30 September 2022.* Tech. rep. National Hurricane Center.

Cangialosi, J P (2023). *NHC Forecast Verification Report, 2022 Hurricane Season.* Tech. rep. National Hurricane Center.

Chao, Wei-Ting and Chih-Chieh Young (2022). "Accurate storm surge prediction with a parametric cyclone and neural network hybrid model". In: *Water* 14.1, p. 96.

Chen, Kuo et al. (2022). "Storm surge prediction based on long short-term memory neural network in the East China Sea". In: *Applied Sciences* 12.1, p. 181.

Cheng, Yongcun and Ole Baltazar Andersen (2010). "Improvement in global ocean tide model in shallow water regions". In: *Poster, SV* 45, pp. 1–68.

Cialone, M A et al. (2017). "Coastal-Storm Model Development and Water-Level Validation for the North Atlantic Coast Comprehensive Study". In: *Journal of Waterway, Port, Coastal, and Ocean Engineering* 143 (5).

CIRES (2014). *Continuously Updated Digital Elevation Model (CUDEM) - 1/9 Arc-Second Resolution Bathymetric-Topographic Tiles.* `https://coast.noaa.gov/htdata/raster2/elevation/NCEI_ninth_Topobathy_2014_8483/`. Accessed 15 July 2021.

Colle, Brian A et al. (2021). "Improving communication of uncertainty and risk of high-impact weather through innovative forecaster workshops". In: *Bulletin of the American Meteorological Society* 102.7, E1424–E1430.

Cyriac, R et al. (2018). "Variability in Coastal Flooding Predictions due to Forecast Errors during Hurricane Arthur (2014)". In: *Coastal Engineering* 137, pp. 59–78.

Dawson, Clinton N et al. (2021). "ADCIRC Simulation of Synthetic Storms in the Gulf of Mexico". In.

Di Liberto, Tom et al. (2011). "Verification of a multimodel storm surge ensemble around New York City and Long Island for the cool season". In: *Weather and Forecasting* 26.6, pp. 922–939.

Dietrich, J C, C J Trahan, et al. (2012). "Surface Trajectories of Oil Transport along the Northern Coastline of the Gulf of Mexico". In: *Continental Shelf Research* 41 (1), pp. 17–47.

Dietrich, J C, M Zijlema, et al. (2011). "Modeling Hurricane Waves and Storm Surge using Integrally-Coupled, Scalable Computations". In: *Coastal Engineering* 58, pp. 45–65.

Dresback, K M et al. (2013). "Skill assessment of a real-time forecast system utilizing a coupled hydrologic and coastal hydrodynamic model during Hurricane Irene (2011)". In: *Continental Shelf Research* 71, pp. 78–94.

Emanuel, K (2021). "Response of global tropical cyclone activity to increasing CO 2: Results from downscaling CMIP6 models". In: *Journal of Climate* 34.1, pp. 57–70.

FEMA (2021). *Flood Risk Study Engineering Library.* `https://hazards.fema.gov/wps/portal/frisel`. [Retrieved 4 February 2021].

Fleming, J et al. (2008). "A Real Time Storm Surge Forecasting System Using ADCIRC". In: *Estuarine and Coastal Modeling (2007)*, pp. 893–912.

Flowerdew, Jonathan, Kevin Horsburgh, and Ken Mylne (2009). “Ensemble forecasting of storm surges”. In: *Marine Geodesy* 32.2, pp. 91–99.

Géron, Aurélien (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow.* " O'Reilly Media, Inc.".

Gettelman, A et al. (2017). “Electronic Supplement to: Projections of Future Tropical Cyclone Damage with a High Resolution Global Climate Model”. In.

Gevecke, Anna (2022). “Comparison of today's vs. future tropical cyclone impacts using different sources of storm tracks in CLIMADA”. MA thesis. ETH Zurich.

Glorot, Xavier and Yoshua Bengio (2010). “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, pp. 249–256.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning.* MIT press.

GRASS Development Team (2022). *Geographic Resources Analysis Support System (GRASS GIS) Software, Version 7.2.* Open Source Geospatial Foundation. URL: `http://grass.osgeo.org`.

Hagen, S C, J J Westerink, and R L Kolar (2000). “One-dimensional finite element grids based on a localized truncation error analysis”. In: *International Journal for Numerical Methods in Fluids* 32, pp. 241–261.

Harper, Bruce A (1999). “Numerical modelling of extreme tropical cyclone winds”. In: *Journal of Wind Engineering and Industrial Aerodynamics* 83.1, pp. 35–47. ISSN: 0167-6105. DOI: `https://doi.org/10.1016/S0167-6105(99)00059-8`. URL: `https://www.sciencedirect.com/science/article/pii/S0167610599000598`.

Hashemi, M R et al. (2016). “An efficient artificial intelligence model for prediction of tropical storm surge”. In: *Natural Hazards* 82 (1), pp. 471–491.

Hersbach, Hans et al. (2020). “The ERA5 global reanalysis”. In: *Quarterly Journal of the Royal Meteorological Society* 146.730, pp. 1999–2049.

Hinton, Geoffrey E, Simon Osindero, and Yee-Whye Teh (2006). “A fast learning algorithm for deep belief nets”. In: *Neural computation* 18.7, pp. 1527–1554.

Hodges, Kevin, Alison Cobb, and Pier Luigi Vidale (2017). “How well are tropical cyclones represented in reanalysis datasets?” In: *Journal of Climate* 30.14, pp. 5243–5264.

Holland, R W (1980). “An Analytic Model of the Wind and Pressure Profiles in Hurricanes”. In: *Monthly Weather Review* 108, pp. 1212–1218.

Hope, M E et al. (2013). “Hindcast and Validation of Hurricane Ike (2008) Waves, Forerunner, and Storm Surge”. In: *Journal of Geophysical Research: Oceans* 118, pp. 4424–4460.

Huang, Huai-Shuo, Chien-Liang Liu, and Vincent S. Tseng (2018). “Multivariate Time Series Early Classification Using Multi-Domain Deep Neural Network”. In: *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 90–98. DOI: 10.1109/DSAA.2018.00019.

Huber, Peter J (1992). “Robust estimation of a location parameter”. In: *Breakthroughs in statistics: Methodology and distribution*. Springer, pp. 492–518.

Ioffe, Sergey and Christian Szegedy (2015). “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *CoRR* abs/1502.03167. arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167.

Irish, J L, D T Resio, and M A Cialone (2009). “A surge response function approach to coastal hazard assessment, Part 2: Quantification of spatial attributes of response functions”. In: *Natural Hazards* 51 (1), pp. 183–205.

Jelesnianski, C P, J Chen, and W A Shafer (1992). *SLOSH: Sea, Lake, and Overland Surges from Hurricanes*. Tech. rep. Silver Spring, Maryland: U.S. Dep. of Commer., Natl. Oceanic and Atmos. Admin., Natl. Weather Serv.

Jia, P and M Li (2012). “Circulation dynamics and salt balance in a lagoonal estuary”. In: *Journal of Geophysical Research - Oceans* 117, p. C01003.

Johnston, Jeremy et al. (2021). “Projecting the effects of land subsidence and sea level rise on storm surge flooding in Coastal North Carolina”. In: *Scientific Reports* 11.1, p. 21679.

Joyce, BR et al. (2019). “US IOOS coastal and ocean modeling testbed: Hurricane-induced winds, waves, and surge for deep ocean, reef-fringed islands in the Caribbean”. In: *Journal of Geophysical Research: Oceans* 124.4, pp. 2876–2907.

Kalpana (2018). https://github.com/ccht-ncsu/Kalpana/. [Retrieved 16 July 2018].

— (2023). https://github.com/ccht-ncsu/Kalpana/. [Retrieved 27 December 2023].

Kim, S-W et al. (2015). “A time-dependent surrogate model for storm surge prediction based on an artificial neural network using high-fidelity synthetic hurricane modeling”. In: *Natural Hazards* 76 (1), pp. 565–585.

Kim, Sooyoul, Yoshiharu Matsumi, et al. (2016). “A real-time forecast model using artificial neural network for after-runner storm surges on the Tottori coast, Japan”. In: *Ocean Engineering* 122, pp. 44–53.

Kim, Sooyoul, Shunqi Pan, and Hajime Mase (2019). “Artificial neural network-based storm surge forecast model: Practical application to Sakai Minato, Japan”. In: *Applied Ocean Research* 91, p. 101871.

Kingma, Diederik P and Jimmy Ba (2014). “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980*.

Kleppek, Sabine et al. (2008). "Tropical cyclones in ERA-40: A detection and tracking method". In: *Geophysical research letters* 35.10.

Knapp, Kenneth R, Howard J Diamond, et al. (2018). "International best track archive for climate stewardship (IBTrACS) project, version 4". In: *NOAA National Centers for Environmental Information* 10.

Knapp, Kenneth R, Michael C Kruk, et al. (2010). "The international best track archive for climate stewardship (IBTrACS) unifying tropical cyclone data". In: *Bulletin of the American Meteorological Society* 91.3, pp. 363–376.

Knutson, T et al. (2020). "Tropical Cyclones and Climate Change Assessment: Part II: Projected Response to Anthropogenic Warming". In: *Bulletin of the American Meteorological Society* 101 (3), E303–E322. DOI: `doi.org/10.1175/BAMS-D-18-0194.1`.

Kopp, Robert E et al. (2015). "Past and future sea-level rise along the coast of North Carolina, USA". In: *Climatic Change* 132, pp. 693–707.

Kropf, C. M. et al. (2022). "Uncertainty and sensitivity analysis for probabilistic weather and climate-risk modelling: an implementation in CLIMADA v.3.1.0". In: *Geoscientific Model Development* 15.18, pp. 7177–7201. DOI: `10.5194/gmd-15-7177-2022`. URL: `https://gmd.copernicus.org/articles/15/7177/2022/`.

Kyprioti, A P, E Adeli, et al. (2021). "Probabilistic Storm Surge Estimation for Landfalling Hurricanes: Advancements in Computational Efficiency Using Quasi-Monte Carlo Techniques". In: *Journal of Marine Science and Engineering* 9.12, p. 1322. DOI: `https://doi.org/10.3390/jmse9121322`.

Kyprioti, A P, C Irwin, et al. (2023). "Spatio-temporal storm surge emulation using Gaussian Process techniques". In: *Coastal Engineering* 180, p. 104231.

Kyprioti, A P, A A Taflanidis, et al. (2021). "Storm hazard analysis over extended geospatial grids utilizing surrogate models". In: *Coastal Engineering* 168, p. 103855.

L A Avila S R Stewart, R Berg and A B Hagen (2020). *Tropical Cyclone Report, Hurricane Dorian (AL052019), 24 August – 7 September 2019*. Tech. rep. National Hurricane Center.

Larson, M and G Clements (2008). *r.grow*. Open Source Geospatial Foundation. URL: `https://grass.osgeo.org/grass78/manuals/r.grow.html`.

Lau, Yui-Yip et al. (2022). "A review of historical changes of tropical and extra-tropical cyclones: a comparative analysis of the United States, Europe, and Asia". In: *International journal of environmental research and public health* 19.8, p. 4499.

LeCun, Yann, Yoshua Bengio, et al. (1995). "Convolutional networks for images, speech, and time series". In: *The handbook of brain theory and neural networks* 3361.10, p. 1995.

Lee, J-W et al. (2021). “Rapid prediction of peak storm surge from tropical cyclone track time series using machine learning”. In: *Coastal Engineering* 170, p. 104024.

Lloyd, Stuart (1982). “Least squares quantization in PCM”. In: *IEEE transactions on information theory* 28.2, pp. 129–137.

Luettich, R A and J J Westerink (2004). *Formulation and Numerical Implementation of the 2D/3D ADCIRC Finite Element Model Version 44.XX*. `https://adcirc.org/files/2018/11/adcirc_theory_2004_12_08.pdf`.

Luettich, R A, J J Westerink, and N W Scheffner (1992). *ADCIRC: An Advanced Three-Dimensional circulation model for shelves coasts and estuaries, report 1: Theory and methodology of ADCIRC-2DDI and ADCIRC-3DL*. Tech. rep. United States Army Corps of Engineers.

MacQueen, James et al. (1967). “Some methods for classification and analysis of multivariate observations”. In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA, pp. 281–297.

Martín Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: `https://www.tensorflow.org/`.

Martinez, Andrew B (2020). “Forecast accuracy matters for hurricane damage”. In: *Econometrics* 8.2, p. 18.

Meiler, Simona, Alessio Ciullo, et al. (2023). “Uncertainties and sensitivities in the quantification of future tropical cyclone risk”. In: *Communications Earth & Environment* 4.1, p. 371.

Meiler, Simona, Thomas Vogt, et al. (2022). “Intercomparison of regional loss estimates from global synthetic tropical cyclone models”. In: *Nature Communications* 13.1, p. 6156.

Mel, Riccardo and Piero Lionello (2014). “Storm surge ensemble prediction for the city of Venice”. In: *Weather and forecasting* 29.4, pp. 1044–1057.

Morss, Rebecca E, Julie L Demuth, and Jeffrey K Lazo (2008). “Communicating uncertainty in weather forecasts: A survey of the US public”. In: *Weather and forecasting* 23.5, pp. 974–991.

Murphy, Allan H (1993). “What is a good forecast? An essay on the nature of goodness in weather forecasting”. In: *Weather and forecasting* 8.2, pp. 281–293.

Nadal-Caraballo, N C and J A Melby (2014). *North Atlantic Coast Comprehensive Study Phase I: Statistical Analysis of Historical Extreme Water Levels with Sea Level Change*. Tech. rep. U.S. Army Corps of Engineers, Engineer Research, and Development Center.

National Centers for Environmental Information (2023). *U.S. Billion-Dollar Weather and Climate Disasters (2023)*. `https://www.ncei.noaa.gov/access/billions/`. [Retrieved 6 April 2023]. DOI: `https://doi.org/10.25921/stkw-7w73`.

National Oceanic and Atmospheric Administration (2011). *Hurricane Irene*. Tech. rep.

— (2023). *Tides and Currents*. https://tidesandcurrents.noaa.gov/. [Retrieved 23 May 2023].

North Carolina Emergency Management (2014). *North Carolina Sea Level Rise Impact Study*. https://fiman.nc.gov/. [Retrieved 15 January 2024].

Oceanic, National and Atmospheric Administration (2020). *Data Access Viewer*. https://coast.noaa.gov/dataviewer/. [Retrieved 20 October 2020].

Owensby, M et al. (2020). *Calibration and Validation of the South Atlantic Domain Model Setup for the South Atlantic Coastal Study (SACS)*. Tech. rep. US Army Corps of Engineers, p. 210.

Pachev, Benjamin et al. (2023). "A framework for flexible peak storm surge prediction". In: *Coastal Engineering* 186, p. 104406.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Plumlee, M et al. (2021). "High-fidelity Hurricane Surge Forecasting using Emulation and Sequential Experiments". In: *Annals of Applied Statistics* 15 (1), pp. 460–480.

Pringle, W J et al. (2021). "Global storm tide modeling with ADCIRC v55: unstructured mesh design and performance". In: *Geoscientific Model Development* 14 (2), pp. 1125–1145.

Pringle, William J et al. (2023). "Efficient Probabilistic Prediction and Uncertainty Quantification of Tropical Cyclone-driven Storm Tides and Inundation". In: *Artificial Intelligence for the Earth Systems*, pp. 1–40.

Rasmussen, C E and C K Williams (2006). *Gaussian processes for machine learning*. 39. MIT Press.

Resio, Donald T et al. (2007). "White paper on estimating hurricane inundation probabilities". In.

Riverside Technology and AECOM (2015). *Mesh Development, Tidal Validation, and Hindcast Skill Assessment of an ADCIRC Model for the Hurricane Storm Surge Operational Forecast System on the US Gulf-Atlantic Coast*.

Roberts, K J, W J Pringle, and J J Westerink (2019). "OceanMesh2D 1.0: MATLAB-based software for two-dimensional unstructured mesh generation in coastal ocean modeling". In: *Geoscientific Model Development* 12 (5), pp. 1847–1868.

Rucker, C A et al. (2021). "Downscaling of Real-Time Coastal Flooding Predictions for Decision Support". In: *Natural Hazards* 107, pp. 1341–1369. DOI: 10.1007/s11069-021-04634-8.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1986). “Learning representations by back-propagating errors”. In: *nature* 323.6088, pp. 533–536.

Saffir, HS (1974). “Saffir-Simpson Scale for Hurricanes”. In: *Herbert S. Saffir Consulting Engineers, Coral Gables, Florida*, pp. 21–23.

Saginor, Jesse and Yue Ge (2017). “Do hurricanes matter? A case study of the residential real estate market in Brunswick County, North Carolina”. In: *International Journal of Housing Markets and Analysis* 10.3, pp. 352–370.

Schenkel, Benjamin A and Robert E Hart (2012). “An examination of tropical cyclone position, intensity, and intensity life cycle within atmospheric reanalysis datasets”. In: *Journal of Climate* 25.10, pp. 3453–3475.

Sebastian, A G et al. (2014). “Characterizing Hurricane Storm Surge Behavior in Galveston Bay using the SWAN+ADCIRC Model”. In: *Coastal Engineering* 88, pp. 171–181.

Smith, A et al. (2020). *U.S. Billion-Dollar Weather and Climate Disasters 1980-2019*. Tech. rep. NOAA National Centers for Environmental Information (NCEI).

Srivastava, Nitish et al. (2014). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56, pp. 1929–1958. URL: `http://jmlr.org/papers/v15/srivastava14a.html`.

Stewart, S R (2017). *Tropical Cyclone Report for Hurricane Matthew, 28 September - 9 October 2016*. Tech. rep. National Hurricane Center.

Stewart, S R and R Berg (2019). *Tropical Cyclone Report for Hurricane Florence, 31 August – 17 September 2018*. Tech. rep. National Hurricane Center.

Taflanidis, A A et al. (2013). “Rapid assessment of wave and surge risk during landfalling hurricanes; probabilistic approach”. In: *Journal of Waterway Port Coastal and Ocean Engineering* 139 (3), pp. 171–182.

Tanaka, S et al. (2011). “Scalability of an Unstructured Grid Continuous Galerkin Based Hurricane Storm Surge Model”. In: *Journal of Scientific Computing* 46, pp. 329–358.

Taylor, A A and B Glahn (2008). *Probabalistic guidance for hurricane storm surge*. Tech. rep. Meteorological Development Laboratory Office of Science and Technology National Weather Service.

Thomas, A et al. (2022). “Effects of Model Resolution and Coverage of Storm-Driven Coastal Flooding Predictions”. In: *Journal of Waterway, Port, Coastal, and Ocean Engineering* 148 (1).

Tiggeloven, Timothy et al. (2021). “Exploring deep learning capabilities for surge predictions in coastal areas”. In: *Scientific reports* 11.1, p. 17224.

U.S. Army Corps of Engineers (2015). *North Atlantic Coast Comprehensive Study: Resilient Adaption to Increasing Risk*. Tech. rep. U.S. Army Corps of Engineers.

USGS (2020). *TNM Download*. URL: `http://https://viewer.nationalmap.gov/basic/` (visited on 02/15/2020).

Van Rossum, Guido and Fred L. Drake (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace. ISBN: 1441412697.

Vijayan, Linoj et al. (2023). "Improving the accuracy of hurricane wave modeling in Gulf of Mexico with dynamically-coupled SWAN and ADCIRC". In: *Ocean Engineering* 274, p. 114044.

Wang, Bao et al. (2021). "Multi-step ahead short-term predictions of storm surge level using CNN and LSTM network". In: *Acta Oceanologica Sinica* 40, pp. 104–118.

Westerink, J J et al. (2008). "A Basin to Channel Scale Unstructured Grid Hurricane Storm Surge Model Applied to Southern Louisiana". In: *Monthly Weather Review* 136 (3), pp. 833–864.

Wold, Svante, Kim Esbensen, and Paul Geladi (1987). "Principal component analysis". In: *Chemometrics and intelligent laboratory systems* 2.1-3, pp. 37–52.

Woodruff, J L (2023). "Subgrid Corrections in Storm-Driven Coastal Flooding". PhD thesis. North Carolina State University.