IMPLEMENTATION OF A HYDRODYNAMIC MODEL FOR SALINITY IN NUECES AND CORPUS CHRISTI BAYS

By

JAMES C. DAVIS

MAY, 2013

A Thesis Paper Submitted
In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

for
The Graduate Committee
Environmental Science Program
Texas A&M University-Corpus Christi
Corpus Christi, Texas

Approved: ______________________________ Date: ______________
Dr. P. Tissot, Chairperson

____
Dr. G. Jeffress, Member

____
Dr. J. Gibeaut, Member

____
Dr. F. Pezold, Dean
College of Science and Technology

Format Journal: Ocean Modelling

Abstract

The purpose of this research was to implement and calibrate a hydrodynamic model for Nueces Bay, Texas and use said model to predict how salinity and circulation in the bay could be affected by sea level rise and changes in freshwater input. The Nueces Bay estuary is fed fresh water from the Nueces River and connects to Corpus Christi Bay, which connects to the Gulf of Mexico. Salinities in Nueces Bay range from near 0 PSU at times of high riverine flow to over 50 PSU during drought periods. The model selected is the Finite Volume Coastal Ocean Model (FVCOM), an unstructured grid, finite-volume, three-dimensional primitive equation ocean model developed for the study of coastal oceanic and estuarine circulation.

An unstructured triangular mesh was created for the bay and bathymetry was interpolated to the mesh. Forcing inputs included water level, water temperature, salinity, wind speed and direction, and the flow rate of the Nueces River. Several software utilities were created to facilitate set-up and testing of the model. Predictions were compared with measured water levels and salinity at several locations in the study area for years 2008 & 2010.

The model produced good results for water level predictions with a mean absolute error of 6.5 cm over the test period. The model also produced overall realistic currents and salinity variations in Corpus Christi Bay with a mean absolute error of 1.7 psu at Ingleside. However, the model predicted salinity poorly in Nueces Bay with a mean absolute error greater than 6 psu at all stations and a maximum absolute error of greater than 20 psu.

While its initial goal was to investigate the impact of sea level rise on salinity levels, the study focused instead on model performance for salinity predictions in Nueces Bay. The investigation revealed that while freshwater from the Nueces River was entering the system at the correct rate, the model was not accurately reflecting salinity response in cells further down river and ultimately in Nueces Bay.

Table of Contents

List of Figures

List of Tables

<u>Acknowledgements</u>

There are many people who have contributed in one way or another to this thesis and I would like to thank a few of them here. First, I would like to especially thank my committee members: Dr. Philippe Tissot, Dr. Gary Jeffress, and Dr. James Gibeaut. I would also like to thank Dr. Darek Bogucki for serving on the first version of my committee and for his excellent course on oceanography.

I was extremely privileged to have the opportunity to complete this graduate program under the tutelage of Dr. Philippe Tissot. His wealth of knowledge of physical processes, unending patience with my exceeding long time scale for completion, and gentle nudges when I've needed them have all been invaluable along the way. I consider him my colleague and my friend and will always value his input.

I owe a lot to Dr. Gary Jeffress, Director of the Conrad Blucher Institute for Surveying and Science (my former employer), for his support and for providing me with a job that made it possible to complete my graduate work at the same time. He was a pleasure to work under and I have the deepest respect for him and consider him my friend as well.

I would also like to thank Dr. James Gibeaut, Director of the Gulf of Mexico Research Initiative Information and Data Cooperative (my current employer) not only for offering me a chance to advance my career, but for jumping aboard my committee as a fourth quarter substitution. He has been a pleasure to work for and has graciously allowed me time to complete my graduate work. I look forward to many years of doing great work under his direction.

I have a huge amount of gratitude for everyone at the Conrad Blucher Institute (CBI). A big thanks goes out to Sergey Reid for all of his assistance with creating grids, for great conversation, and for pushing on me when I needed it. I want to thank, in no particular order, Gina, James, Kat, Larry, Lucy, Dee, Dave, Dominic, Richard, Rick, Robin, Rose, Joe, Jeannine, Julien, Tommy, Stacey, Zack, Kelby, Greg, Dan, and all the CBI student workers for the many years of great laughs, conversation, and support. I especially want thank my former boss, Scott Duff, for making me crazy (in a good way) and my former suite-mates and forever friends, Niall Durham and Sara Ussery, for keeping me sane.

I would like to thank my parents, Jim and Susan Davis, for giving me a strong foundation that I could build upon to get where I am today. Their love and support through this thesis work (and all the trials and tribulations of life that have occurred along the way) has been wonderful.

My final acknowledgements go out to those whom the time and energy put towards this thesis has affected the most: my amazing wife, Stephanie, and our son, Caffall. Without their unending love and support, I simply could not have complete this work. I owe them so much and it will take the rest of my lifetime to repay them. I look forward to many many days where I am not preoccupied with “the thesis” and am fully available to be the husband and father I want to be and that they deserve. Steph and Caffall, I love you both so very much!

Introduction

Nueces Bay, just to the north and bordering the city of Corpus Christi, TX, is part of an estuary that includes the Nueces River and a bay system that includes Corpus Christi Bay. This estuary and bay system provide habitat for many kinds of marine life, including several economically important species. These species rely on the fresh water from the Nueces River to create the reduced salinity environment that is an essential nursing ground for their larvae.

Nueces Bay has seen major alteration of its natural fresh water inflows due to creation of reservoirs upstream on the Nueces and Frio rivers. The creation of these reservoirs, especially the Choke Canyon Reservoir on the Frio River, has been highly controversial. The Choke Canyon project was cleared to go forward in the late 1970's when an agreement was made to release as much as 151,000 acre-feet of fresh water per year in to Nueces Bay (Blackburn, 2004). Unfortunately, this plan was never implemented. After the reservoir was completed in 1984, freshwater inflows to Nueces Bay decreased drastically.

In 1990, a suit was filed by a seafood industry group in the Corpus Christi Bay area after noticing a substantial decline in shrimp catch. The suit demanded the mitigation water that had been promised to be released in to Nueces Bay. Eventually, a new plan was drawn up which involved a “pass-through” system that released as much water as came in to the reservoir up to a certain point. The pass-through requirement varied depending on the time of year in an attempt to mimic the natural, historical flow patterns of the river. The final “Agreement in Principle”, signed in 1997, included a substitution of treated wastewater for pass-throughs when the reservoir fell below 50 percent capacity. It also called for increasingly stringent water restrictions for the City of Corpus Christi when the reservoir fell below 50, 40, and 30 percent capacities.

Fresh water is a very valuable and increasingly scarce resource in South Texas. The population of Corpus Christi and other communities in the Coastal Bend continues to grow and with it the demand for water for drinking, lawn watering, etc. Local farming and industrial operations as well as power plants also require fresh water. These anthropogenic needs are in direct competition for water with the Nueces estuary ecosystem. This presents a need to model salinity in Nueces Bay and understand how it will be affected by sea level rise in the coming century. If sea level rise will cause an increase in salinity, there will be a need for more fresh water to be released to maintain salinity levels at acceptable levels. This will be something that decision-makers will need to consider when planning for the future.

Impact of Sea Level Rise

Future projected sea level rise is a topic of much debate. Sea levels have been estimated by the International Panel on Climate Change (IPCC) to rise globally by as much as 59 cm by 2100 (Bindoff et. al., 2007). Rahmstorf's semi-empirical approach to projecting sea level rise (Rahmstorf, 2007) estimates global sea level could rise by as much as 140 cm by 2100. The latest NOAA report on sea level rise (Parris et. al., 2012) found with 90% confidence that global sea levels will rise by at least 20 cm but no more than 200 cm by 2100. This study considers a range of sea level rise estimates from the overly conservative estimate based on sea level rise in the twentieth century to the upper end predicted by the IPCC.

This rise in water level will have a profound effect on Nueces Bay. Marshes will be converted to open water and tidal flats will become permanently flooded. Sea level rise could also cause salt water to intrude farther up the Nueces River as has been predicted in other estuaries (Yang and Zhu, 1993). To understand the effect sea level rise will have on salinity in Nueces Bay, it needed to be modeled and have these scenarios tested. The year 2100 has been used as a target for prediction in many climate change and sea level rise studies (Bindoff et. al., 2007) (Uddameri, 2007). As such, this study also set 2100 as the target for prediction.

Modeling Estuaries

Estuaries are complex systems that can be difficult to model accurately. There is a wide range of modeling techniques from simple box models to complex hydrodynamic models using unstructured grids. For example, Hardisty (2007) presents a relatively simple model that can be implemented in Microsoft Excel. This model represents the width and depth of the estuary based on their relationship to the distance from the mouth and temperature and salinity based on the Gaussian distribution. While this model is effective for a high-level understanding of an estuarine system, it cannot model things such as circulation and the three dimensional nature of a stratified estuary. For this, a gridded model based on the primitive equations is required. There are several models available that fit this description, including ROMS (Regional Ocean Modeling System) (Shchepetkin and McWilliams , 2005), OHSU SELFE (Zhang and Baptista, 2008), and ADCIRC (Luettich et. al., 1992).

In recent years, estuarine systems have been modeled successfully (Chen et. al., 2008), (Yang and Khangaonkar, 2008), (Bek et. al., 2010) with the Finite-Volume Coastal Ocean Model (FVCOM), developed by Chen et. al. (2003). This model employs the finite-volume method of numerically solving the primitive equations, which combines the geometric flexibility of finite-element methods (unstructured grid) with finite-difference methods for simple discrete computation. Chen et. al. (2008) used FVCOM to model the Satilla River Estuary in Georgia. They found that a mass-conservative unstructured-grid model was required to accurately and efficiently simulate tidal flow and flushing in a geometrically complex estuarine system. They suggest a model that fails to resolve the complex coastal geometry of tidal creeks, barriers and islands can generate unrealistic flow and water exchange and thus predict the wrong dynamics for an estuary. Yang and Khangaonkar (2008) used FVCOM to model the Skagit River Estuary of Puget Sound in Washington. This estuary contains large areas of tidal flats and they found FVCOM's ability to handle wetting and drying essential in modeling it successfully. They also found that FVCOM successfully modeled the significant effect on currents in the intertidal zone from strong wind events. This is important for the Nueces Estuary as well because the Coastal Bend area is regularly subject to high winds. Bek et. al. (2010) used FVCOM to model a shallow coastal lake in Egypt that supports fish farm production. Like the Nueces Estuary, this is an engineered system where humans, rather than nature, are in control of the fresh water inflows. They concluded that their FVCOM-based model could be used to investigate the potential effects of modifying these inflows to improve water quality. Each of these estuaries are similar to the Nueces Estuary. The success of FVCOM as a tool for modeling them suggests that it is well suited for modeling the Nueces Estuary.

Study Area

Figure 1 shows a regional map indicating the location of Nueces Bay on the coast of Texas. Figure 2 shows the Nueces/Corpus Christi Bay System with Nueces Bay bordered by Corpus Christi Bay to the East, the Nueces River Delta to the West, the City of Portland and unincorporated farmland to the North, and the City of Corpus Christi to the South. The southern border consists mostly of industrial complexes along the Corpus Christi Ship Channel. Figure 3 shows a map of Nueces Bay and the salinity and water level stations from which data was used in this study.

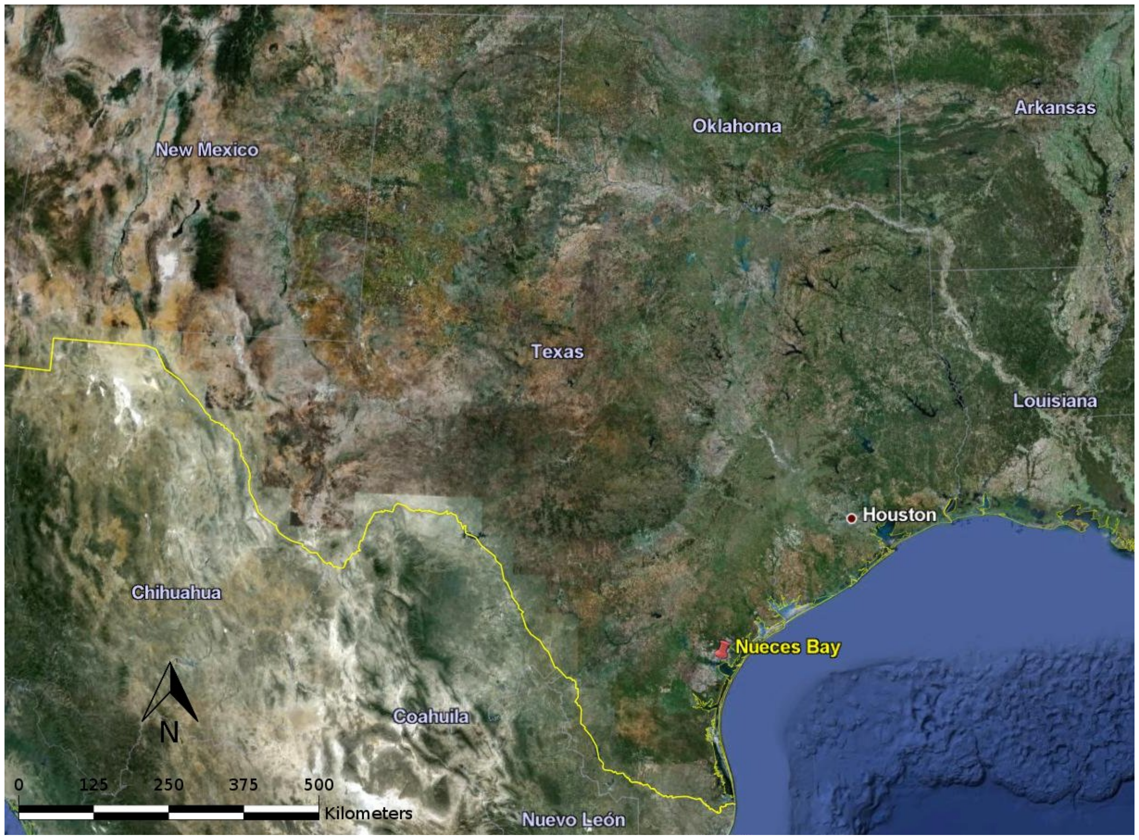

Figure 1: Regional Map Showing Location of Study Area

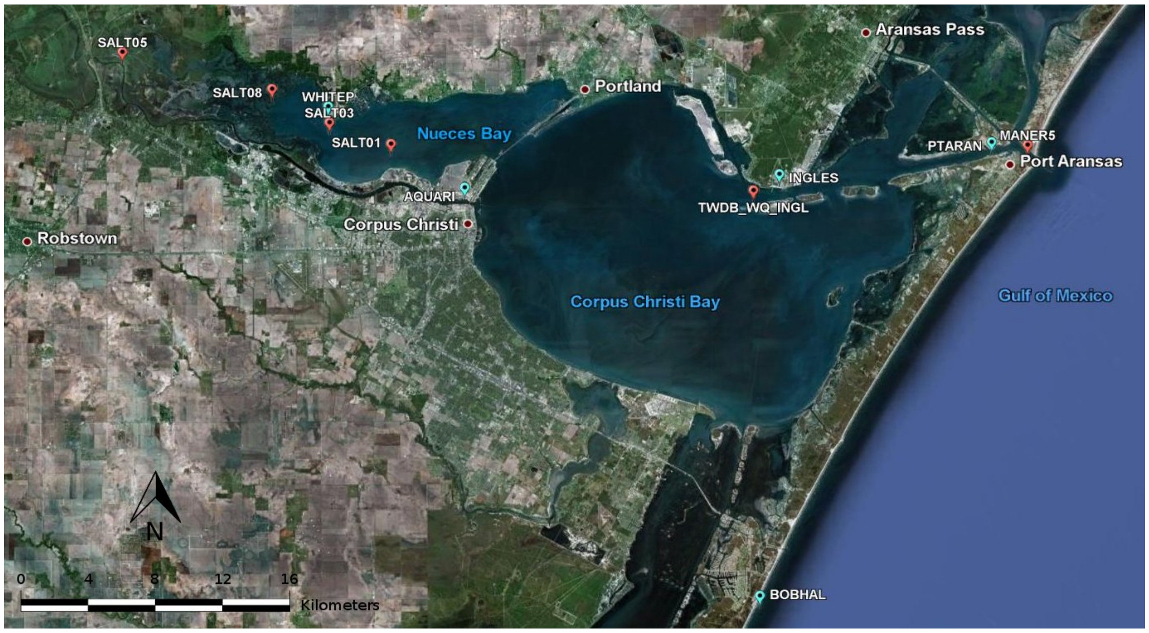

Figure 2: Map Showing Nueces/Corpus Christi Bay System and All Stations

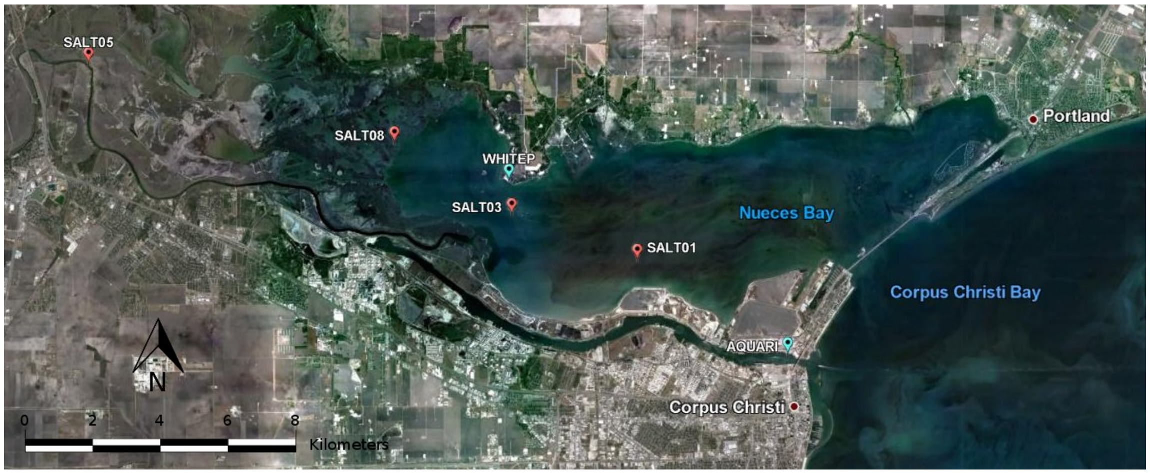

Figure 3: Map of Nueces Bay Showing Salinity and Water Level Stations

Nueces Bay is relatively small, measuring approximately 18 km long by 6 km wide and running generally East-West. It is supplied with fresh water via the Nueces river, which has a watershed of approximately 26,700 km^2. The Nueces River is also supplied water from the Frio River with a watershed of approximately 14,200 km^2. The

Nueces River is not very large due to the semi-arid nature of the region that comprises these watersheds. The average stream flow at the USGS gauge at Calallen for 2000-2009 was 22 m^3/s. The Nueces River empties into the Nueces Bay on the West end. This west end of the bay is composed of the Nueces River Delta which consists mainly of salt marsh and tidal channels. At its East end, the approximately 2.5 km wide mouth of Nueces Bay opens to Corpus Christi Bay. This mouth is somewhat narrower than the width of Nueces Bay (approximately 5 km at the East end), restricting the exchange of water between the two bays.

Corpus Christi Bay connects to Redfish and Aransas Bays to the North, the Laguna Madre to the South, and to the Gulf of Mexico at Port Aransas. The Corpus Christi Ship Channel has been cut into the southern shore of Nueces Bay and many industrial and petrochemical complexes flank the shoreline. An effort has been made to reconnect the river with its historical delta. This effort, called the Rincon Bayou Project, only feeds fresh water to the Nueces River Delta during times of high river flow (Blackburn, 2004). The Nueces estuary is classified as a normal, alluvial, delta type estuary where salinity decreases gradually from sea salinity to river water salinity and sediments are deposited from the sea and the river input, with river sediments forming a delta where the river enters the bay (Savenije, 2005).

As can be seen in Figure 4, most of the time the flow of the Nueces River is quite low. It is punctuated by high-flow events during times of heavy precipitation in the watershed. There are several dry periods with very low flow throughout the year, especially in the mid-nineties.

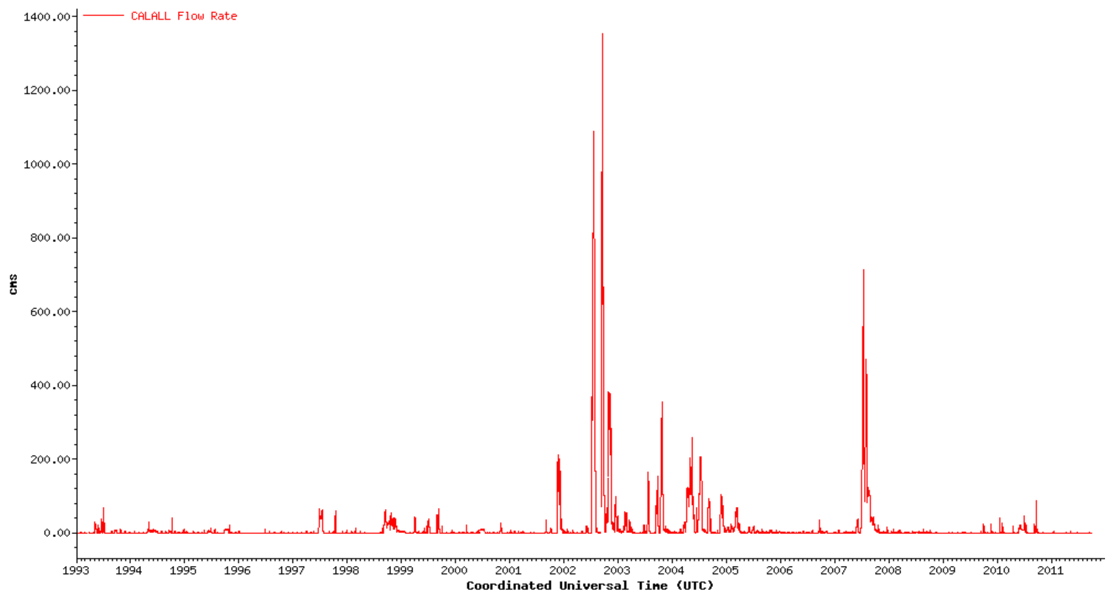

Figure 4: Nueces River Flow in Cubic Meters per Second – 1993-2011

Figure 5 shows the approximate salinity of the area of the Gulf of Mexico near Port Aransas for 2008. The graph shows a clear seasonal variation with salinity steadily increasing through the warmer late spring and summer months followed by a decrease in salinity in the fall.

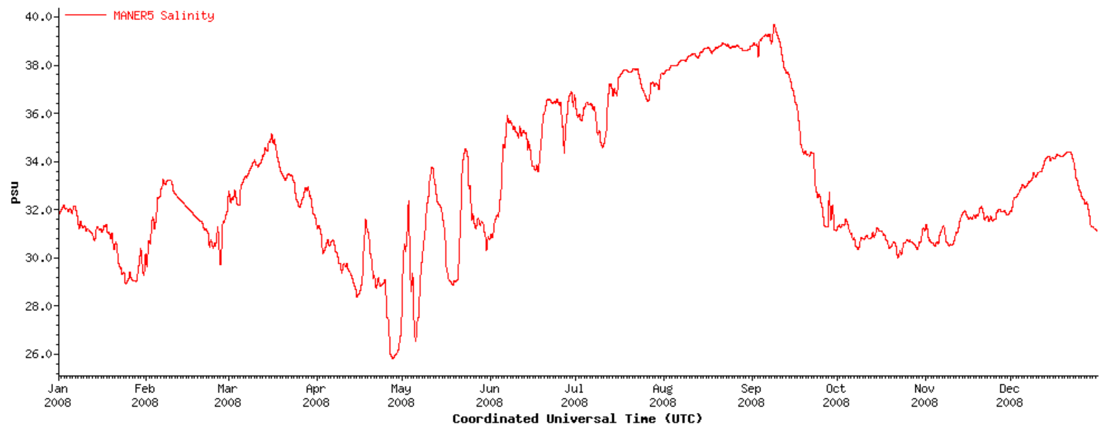

Figure 5: Approximate Salinity of the Gulf of Mexico near Port Aransas (2008)

Materials and Methods

Nueces Bay was modeled using the Finite-Volume Coastal Ocean Model (FVCOM). An unstructured, triangular grid was created using the Surface-water Modeling System (SMS), distributed by Aquaveo LLC. Salinity and water level data were retrieved from the Conrad Blucher Institute for Surveying and Science (CBI) and stream flow data were retrieved from the USGS. The steps of grid creation though running the model, including the selection of model boundaries and forcings, are described below.

Grid Creation

The Surface-water Modeling System (SMS) was used to create the model grid for Nueces Bay. First, shoreline data from NOAA (Signel, 2010) was imported in to SMS as a starting point for the map. As part of the west end of the bay and the Nueces River were missing, they needed to be estimated based on aerial photography. To assist in this process, the map was overlaid on a Geo-rectified aerial photograph (Figure 6). Bathymetric data was also used to help determine the shoreline boundary. In order for the grid to be created properly, the shoreline map needed to be simplified to be a reasonable approximation of the actual shoreline. Map nodes needed to be spaced such that smaller triangles would be created in areas where more detail was necessary and larger triangles would be created in other areas. Additionally, the portion of the ship channel that is carved in to the south shore was removed because it does not contribute to water exchange with Nueces Bay. In order to have a fine enough grid for the model to run properly in the narrow river channel, additional nodes needed to be added. The final grid was then created using SMS (Figure 7). The resulting grid contains cells that range from areas of 1,470 - 161,200 m^2. Figure 8 shows this final grid overlaid on satellite imagery of the study area.

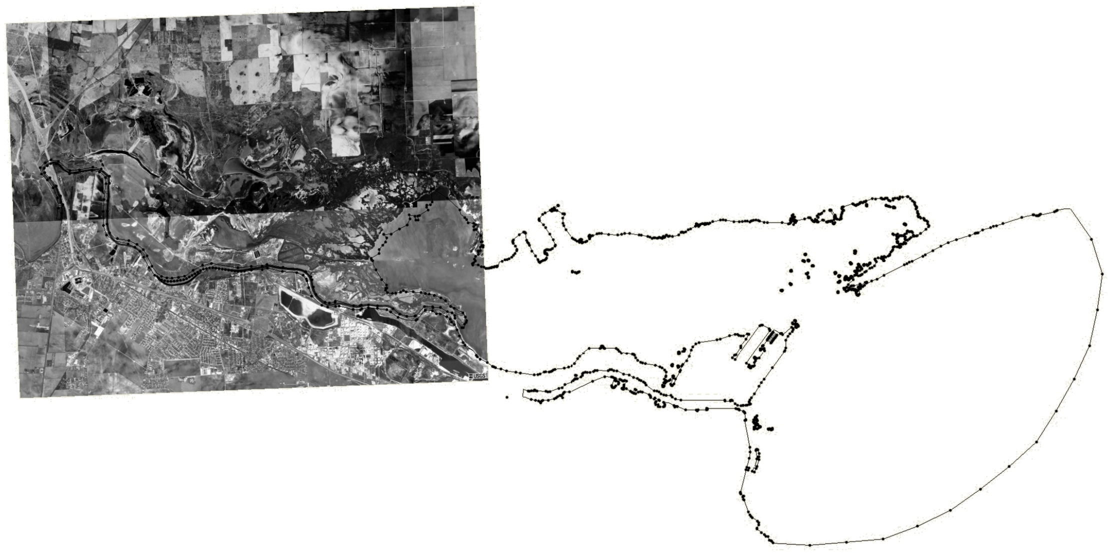

Figure 6: Aerial Imagery of Nueces River Delta (TerraServer, 2010)

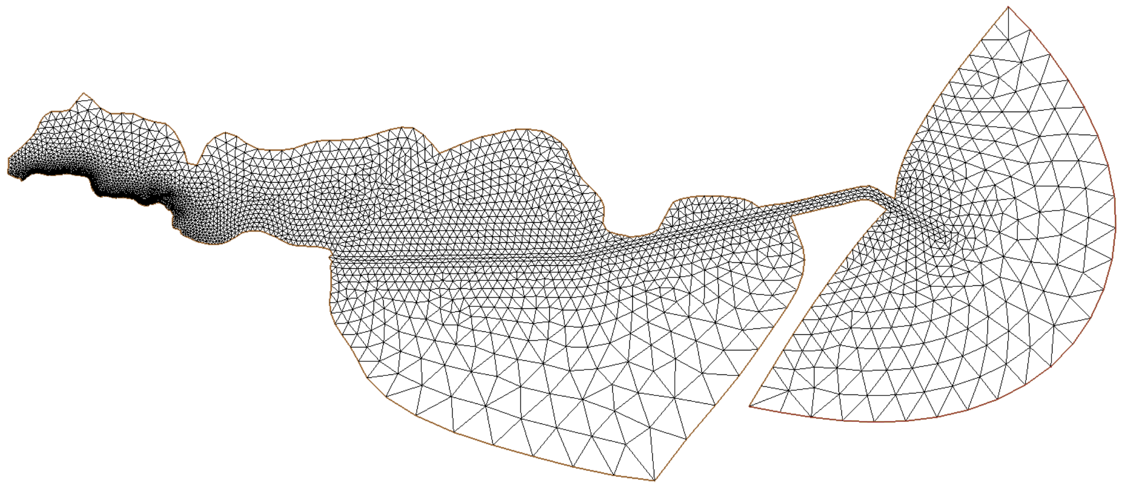

Figure 7: Final Grid Created with SMS

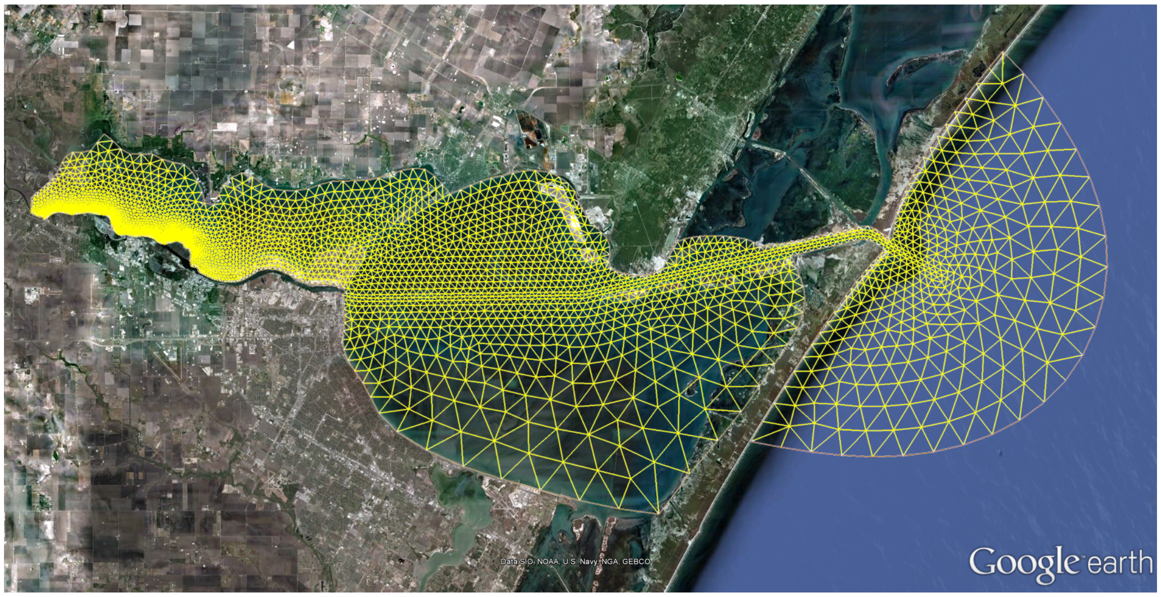

Figure 8: Final Grid Overlaid on Satellite Imagery

<u>Bathymetric Data</u>

High resolution (1/3 arc-second) NOAA Tsunami Inundation Digital Elevation Models (Taylor et. al., 2008) were obtained from NOAA's National Geophysical Data Center for the Nueces Bay region. These models combine data from a variety of sources, including ground surveys and LIDAR, and include coastal land area and bathymetry. Unnecessary land area outside of Nueces Bay was trimmed away to reduce the data set size. The data was then multiplied by -1 to reverse the signs. This is necessary because FVCOM expects bathymetry to be positive. The trimmed and shifted bathymetry data is shown in Figure 9 with a detail of Nueces Bay shown in Figure 10. The bathymetry data was then interpolated to the grid created in the previous section (Figure 11). Figure 12 shows a detail of Nueces Bay with the interpolated bathymetric data.

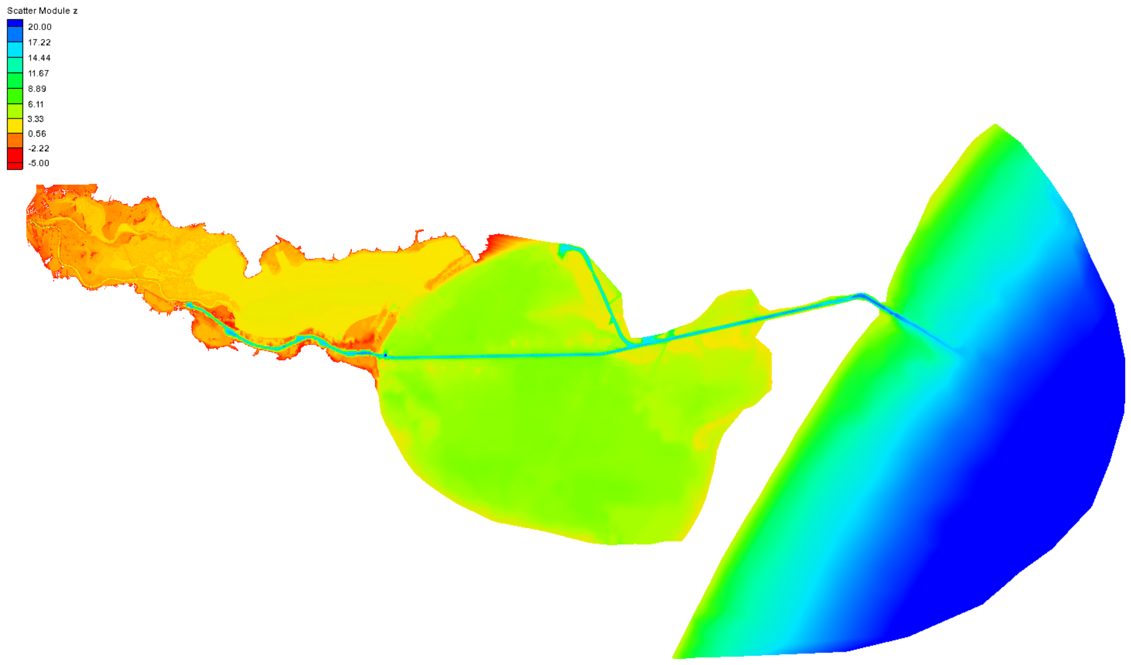

Figure 9: Bathymetry Data (in meters below MSL)

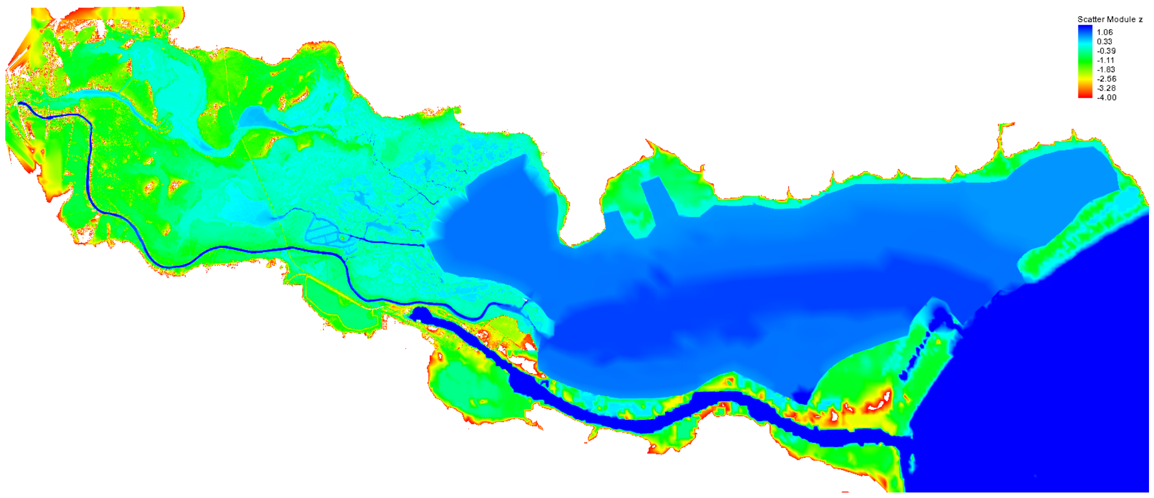

Figure 10: Detail of Nueces Bay Bathymetry Data

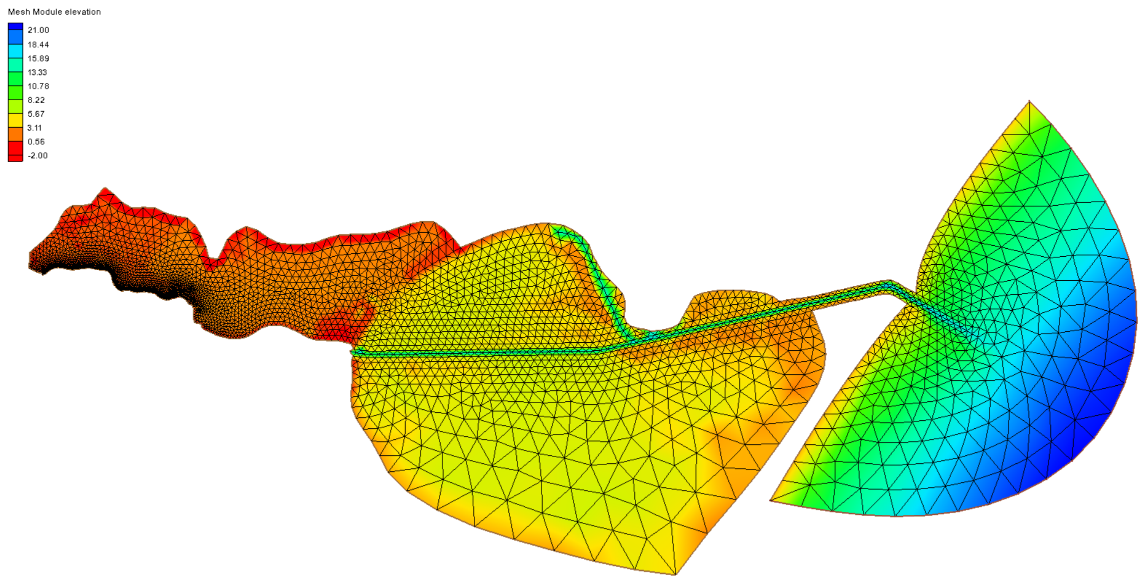

Figure 11: Final Grid with Interpolated Bathymetry Data

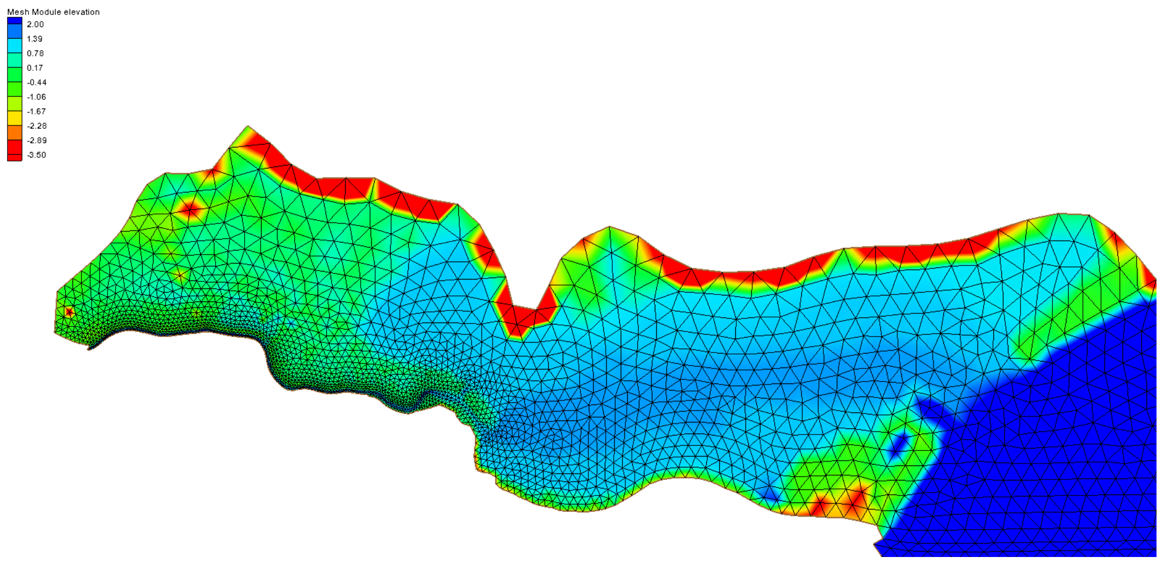

Figure 12: Detail of Nueces Bay Final Grid with Interpolated Bathymetry Data

One of the most difficult challenges encountered during the development of this model was obtaining a good bathymetric data set. It was discovered early on that the NOAA Tsunami Inundation bathymetric data for Nueces Bay was of poor quality (likely because it is based on very little measured data for this location). Further research did not uncover any other easily obtainable bathymetric data set for Nueces Bay. The TWDB (Texas Water Development Board) provided their model grid upon request, which included realistic bathymetric data for Nueces Bay. Unfortunately, this dataset was not very dense. The model could likely be improved by using better bathymetric data for Nueces Bay.

The Harte Research Institute Coastal and Marine Geospatial Lab provided a very dense (1 meter resolution) bathymetric/topographic data set for the Nueces River Delta. This data set included detailed bathymetric data for the Nueces River and the Rincon Bayou. This allowed the creation of a very accurate grid for this area.

The integration of these three data sets presented several challenges. Firstly, each of the data sets used a different vertical datum. The NOAA Tsunami Inundation data was referenced to mean high water (MHW), while the TWDB data was relative to mean sea level (MSL) and the Nueces Delta data to the North American Vertical Datum of 1988 (NAVD88). This required conversion to a common datum. MSL was chosen as the common datum because the IPCC sea level rise predictions refer to a rise in mean sea level. The conversion of the NOAA Tsunami Inundation data from MHW to MSL and the Nueces Delta data from NAVD88 to MSL was done using the VDATUM tool (version 2.3.3) provided by NOAA (http://vdatum.noaa.gov). This tool performs conversions between several vertical datums for a specific area. The NOAA Tsunami Inundation data set covered two of these areas and thus had to be split, converted separately, and merged back together.

The second challenge was how to handle the locations where data sets overlapped. In cases where minor discrepancies were found, a choice had to be made as to which data set most realistically represented the actual bathymetry. One particularly troublesome area was the boundary between Nueces and Corpus Christi Bays. The initial selection of the NOAA Tsunami Inundation data set for this area proved to artificially close off Nueces Bay, preventing exchange of the water as water levels rose and fell in Corpus Christi Bay. This problem was discovered by analyzing model predictions of water levels and salinity within Nueces Bay. The model was incorrectly predicting almost no change in water level or salinity prior to the correction. An analysis of the relationship between measured salinity and water level showed that salinity in Nueces Bay depends on water level, rising as water levels rise and water of higher salinity is pushed into the bay and vice versa. Visual inspection of water level and currents using SMS showed no water exchange taking place between Nueces and Corpus Christi Bays. A closer look showed that most of the bathymetry for this area was erroneously higher than MHW. This was corrected by using the TWDB data instead, which more accurately represented the bathymetry in this area because it is based on field measurements and local boaters' experience. The updated model allows water exchange between the Bays.

<u>Experimental Grid Adjustment</u>

Once the original grid set up as described above was used in the model, it was found that some adjustments were needed for the model to run without numeric instability. The necessary adjustments were determined experimentally.

Cell size gradient was determined to be important in areas where current velocities could become high due to tidal and/or wind forces. These areas required a gradual change from large cells to small cells. It was found that there were different tolerances for water level and salinity by disabling salinity calculations. There were cases where the model would remain stable when salinity was disabled, but would crash for salinity. In general, water level has a higher tolerance for steeper cell size gradients.

A technique was developed for diagnosing areas of instability that involved removing parts of the grid until the instability disappeared. Once the area of instability was found, the cell size gradient was adjusted until the model would run successfully.

<u>Project Data</u>

Salinity data for model set-up and verification were retrieved from the Conrad Blucher Institute for Surveying and Science (CBI), which maintains several salinity stations in Nueces Bay (Figure 3). Water level data at Bob Hall Pier (Figure 2) for forcing the model at the Gulf open boundary were also obtained from CBI. Salinity data for the Gulf open boundary were obtained from the Mission Aransas National Estaurine Research Reserve (MANERR) Port Aransas water quality station (operated by CBI) (Figure 2). Stream flow data were retrieved from the USGS, which maintains a flow gauge at Calallen on the Nueces River. Data from other stations were used to assess the performance of the model, including: water levels from the Port Aransas, Ingleside and Aquarium stations (Figure 2) and salinity from the SALT01, SALT03 and SALT08 stations (Figure 3).

The salinity at the open boundary in the Gulf of Mexico was approximated by starting with the salinity data series at the MANERR Port Aransas station (Figure 13) and removing the data that corresponds to ebb tides, leaving only salinity for water flowing in to the Port Aransas Ship Channel from the Gulf of Mexico. The missing data were then interpolated to create a full salinity series (Figure 14), which was used to force the model at the open boundary in the Gulf of Mexico.

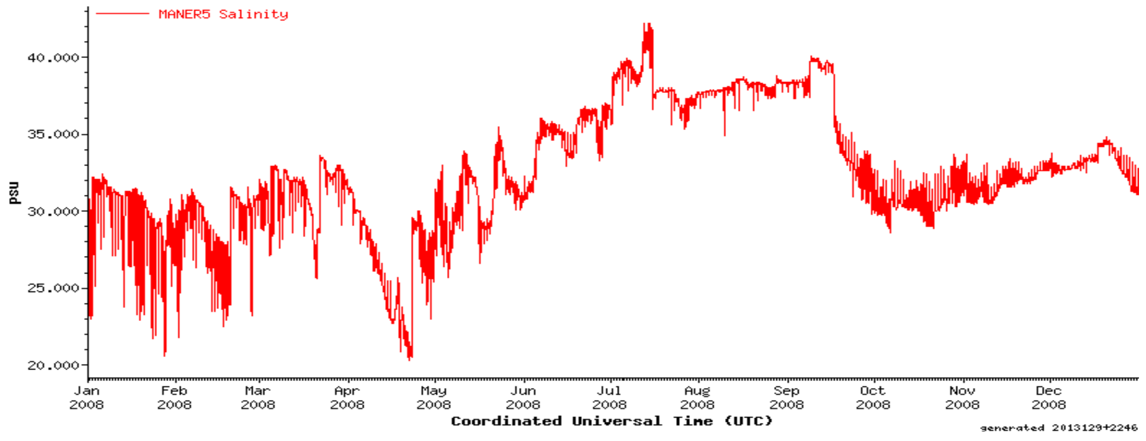

Figure 13: MANERR Port Aransas Original Salinity

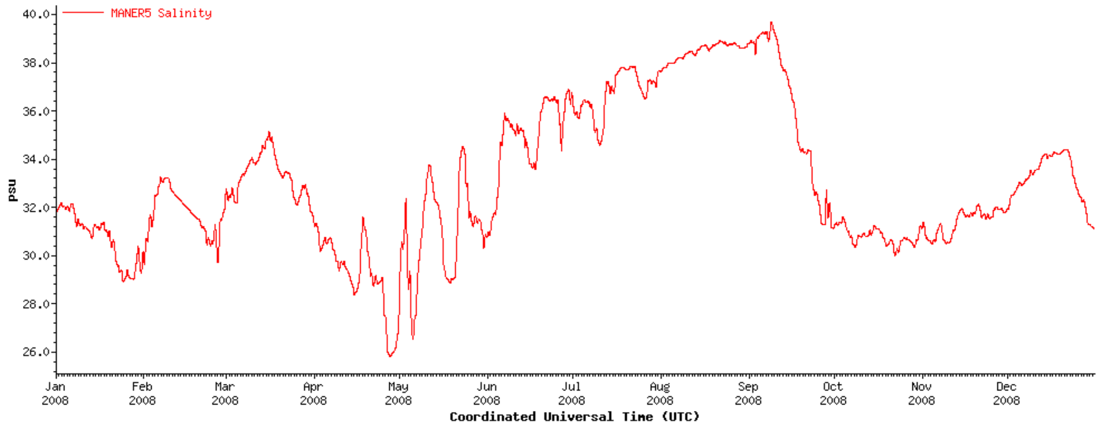

Figure 14: MANERR Port Aransas Salinity (flood tide removed)

The model was set up using data from January through December 2008 and verified using data from August through October 2010. These periods were selected for several reasons including completeness of data sets and inclusion of large freshwater inflow events.

The model set-up dataset has input salinities ranging from 20.3 psu to 42.2 psu and assessment salinities ranging from 3.8 psu at SALT08 in the Nueces River delta to 44.4 at SALT01 in the middle of Nueces Bay. River flow for the model set-up period ranged from 0 to 12 cms. Table 1 lists information about missing data and minimum and maximum values for model input data for the set-up period while Table 2 shows the same information for the assessment data.

	% missing	min	max
Primary Water Level (m above MSL) Bob Hall Pier	0.2	-0.893	1.362
Water Temperature (°C) MANERR Station #5 (Port Aransas)	11.1	10.3	30.4
Salinity (psu) MANERR Station #5 (Port Aransas)	11.1	20.3	42.2
Wind Speed (m/s) Bob Hall Pier	1.4	0	21.3
Wind Direction (degrees from North) Bob Hall Pier	1.4	0	360
Wind Speed (m/s) Port Aransas	34.8	0	17.2
Wind Direction (degrees from North) Port Aransas	34.8	0	360
Flow Rate (cms) USGS Flow Gage - Nueces River at Calallen, TX	0.0	0	12.18
Water Temperature (°C) SALT05 (Nueces River)	36.4	11.2	33.5

Table 1: Model Set-up Input Data

	% missing	min	max
Primary Water Level (m above MSL) Texas State Aquarium	0.4	-0.492	0.939
Primary Water Level (m above MSL) Ingleside	0.0	-0.576	0.971
Primary Water Level (m above MSL) Port Aransas	0.3	-0.739	1.128
Salinity (psu) SALT01 (Nueces Bay)	13.4	9.9	44.4
Salinity (psu) SALT03 (Nueces Bay)	8.1	10.7	36.5
Salinity (psu) SALT08 (Nueces Delta)	18.8	3.8	41.8

Table 2: Model Set-up Assessment Data

Figures 15, 16, and 17 show plots of salinities, Nueces Bay river flow, and water levels, respectively, for the model set-up period.

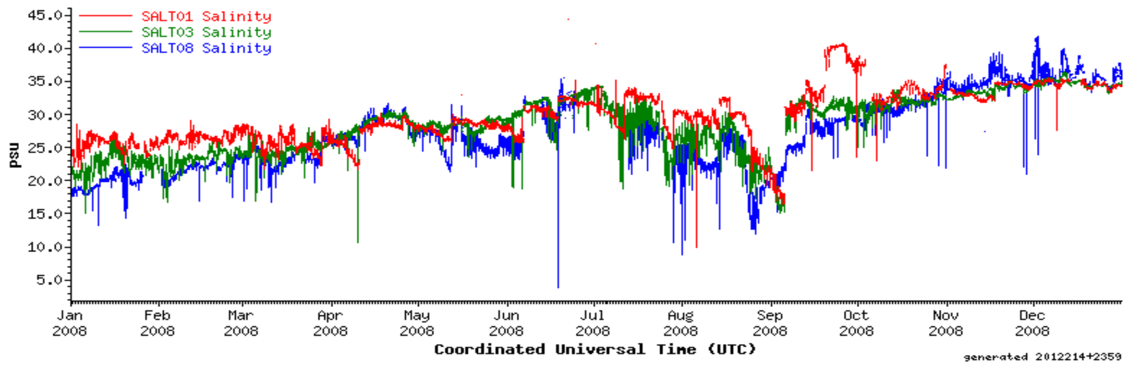

Figure 15: Model Set-up Salinities

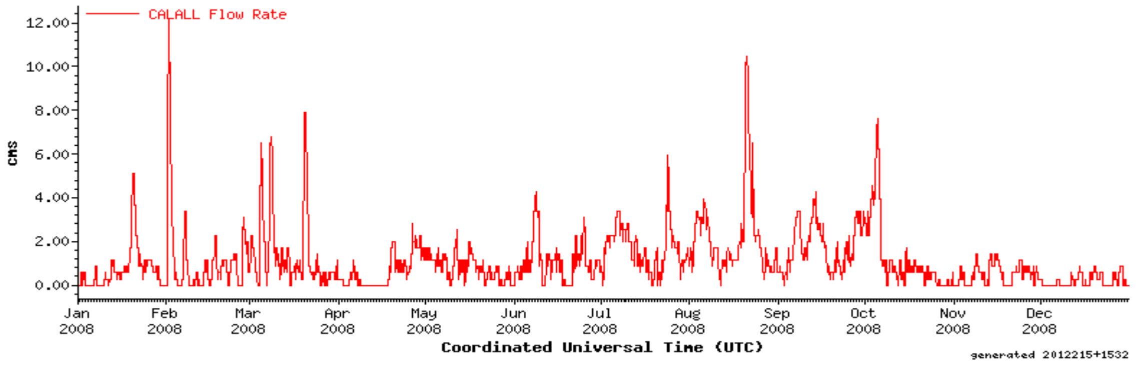

Figure 16: Model Set-up Flow Rate (Nueces River)

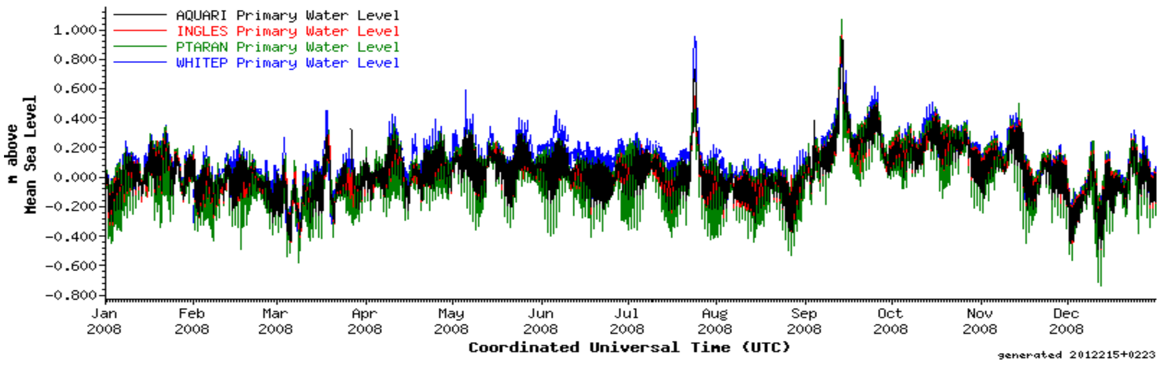

Figure 17: Model Set-up Water Levels

The verification data set has a broader range of input salinities ranging from 12.9 psu to 37.8 psu and assessment salinities ranging from 0 psu to 32.7 psu at SALT01 in the middle of Nueces Bay. River flow for the verification period ranged from 0 to 87.8 cms. Tables 3 and 4 list information about missing data and minimum and maximum values for model input and assessment data for the verification period.

	% missing	min	max
Primary Water Level (m above MSL) Bob Hall Pier	0.0	-0.407	0.976
Water Temperature (°C) MANERR Station #5 (Port Aransas)	4.8	23.7	31.8
Salinity (psu) MANERR Station #5 (Port Aransas)	4.8	12.9	37.8
Wind Speed (m/s) Bob Hall Pier	0.0	0.0	19.9
Wind Direction (degrees from North) Bob Hall Pier	0.0	0	360
Wind Speed (m/s) Port Aransas	1.7	0.0	14.2
Wind Direction (degrees from North) Port Aransas	1.7	0	360
Flow Rate (cms) USGS Flow Gage - Nueces River at Calallen, TX	0.0	0	87.8
Water Temperature (°C) SALT05 (Nueces River)	12.1	20.8	33.8

Table 3: Model Verification Input Data

	% missing	min	max
Primary Water Level (m above MSL) Texas State Aquarium	0.0	-0.167	0.650
Primary Water Level (m above MSL) Ingleside	0.3	-0.266	0.684
Primary Water Level (m above MSL) Port Aransas	0.4	-0.413	0.701
Salinity (psu) SALT01 (Nueces Bay)	2.6	0.0	32.7
Salinity (psu) SALT03 (Nueces Bay)	7.3	2.0	29.9
Salinity (psu) SALT08 (Nueces Delta)	8.6	1.6	29.6

Table 4: Model Verification Assessment Data

Figures 18, 19, and 20 show plots of salinities, Nueces Bay river flow, and water levels, respectively, for the verification period.

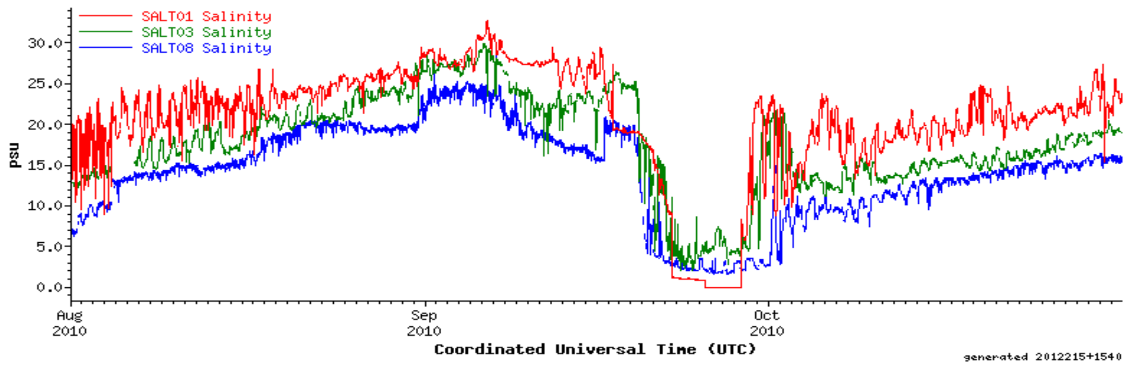

Figure 18: Verification Salinities

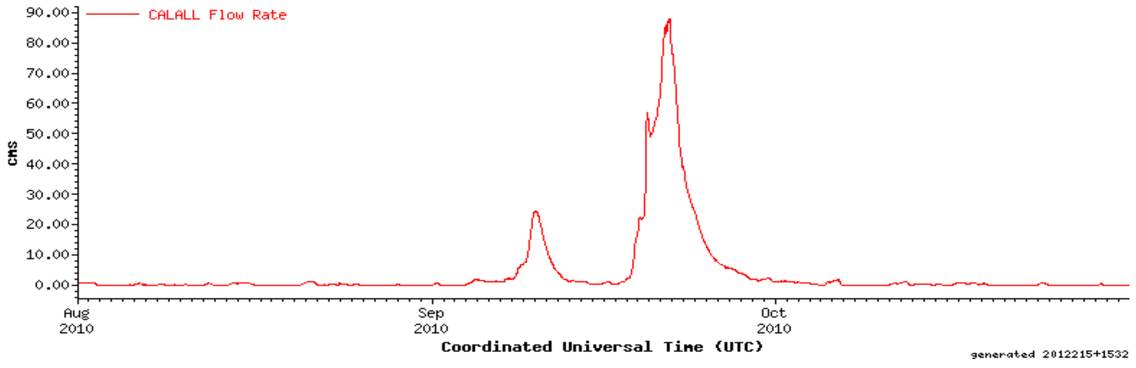

Figure 19: Verification Flow Rate (Nueces River)

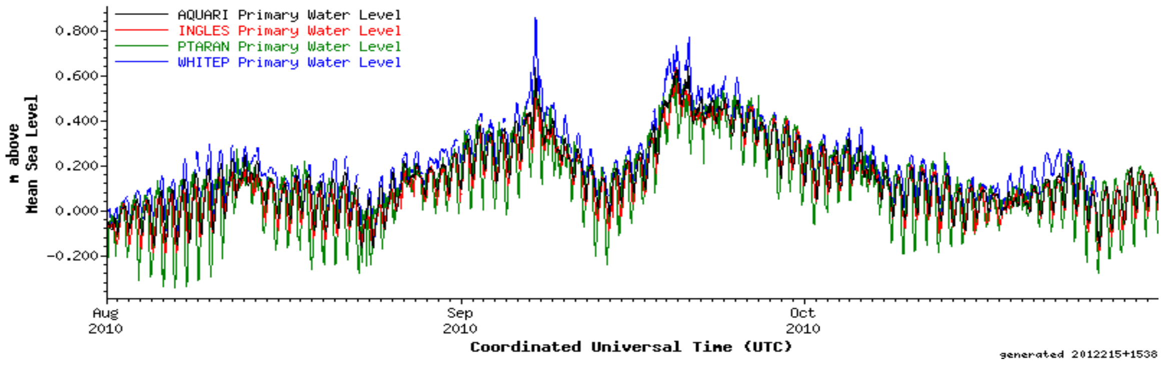

Figure 20: Verification Water Levels

<u>FVCOM</u>

The Finite-Volume Coastal Ocean Model, developed by Chen et. al. (2003), was used for this model. It is an unstructured grid, finite-volume, three-dimensional primitive equation ocean model developed for the study of coastal oceanic and estuarine circulation. The primitive equations for momentum, continuity, temperature, salinity, and density are:

Momentum:

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}+w\frac{\partial u}{\partial z}-fv=-\frac{1}{\rho_0}\frac{\partial P}{\partial x}+\frac{\partial}{\partial z}(K_m\frac{\partial u}{\partial z})+F_u$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}+w\frac{\partial v}{\partial z}+fu=-\frac{1}{\rho_0}\frac{\partial P}{\partial y}+\frac{\partial}{\partial z}(K_m\frac{\partial v}{\partial z})+F_v$$

$$\frac{\partial P}{\partial z}=-\rho g$$

Continuity:

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}=0$$

Temperature:

$$\frac{\partial T}{\partial t}+u\frac{\partial T}{\partial x}+v\frac{\partial T}{\partial y}+w\frac{\partial T}{\partial z}=\frac{\partial}{\partial z}(K_h\frac{\partial T}{\partial z})+F_T$$

Salinity:

$$\frac{\partial S}{\partial t}+u\frac{\partial S}{\partial x}+v\frac{\partial S}{\partial y}+w\frac{\partial S}{\partial z}=\frac{\partial}{\partial z}(K_h\frac{\partial S}{\partial z})+F_S$$

Density:

$$\rho=\rho(T,S)$$

Variables used in these equations:

(x, y, z) – east, north, and vertical axes in Cartesian coordinates

(u, v, w) – velocity components in the x, y and z directions
(F_u, F_v) – horizontal momentum diffusivity terms in the x and y directions
F_T – horizontal thermal diffusion term
F_S – horizontal salt diffusion term
K_m – vertical eddy viscosity coefficient
K_h – thermal vertical eddy diffusion coefficient
ρ – density
P – pressure
T – temperature
S – salinity
f – Coriolis parameter

These equations are closed mathematically using the Mellor-Yamada level-2.5 turbulence closure model (Mellor and Yamada, 1982). This model approximates mixing due to turbulence based on length scale of the boundary layer. FVCOM makes use of a simplification by (Galperin et. al., 1988) which removes a slight inconsistency in scaling analysis so that S_m and S_h depend only on G_h. The equations for Galperin's simplification of MY-2.5 are:

$$K_m = qlS_m$$

$$K_h = qlS_h$$

$$S_m = \frac{0.4275 - 3.354\mathrm{G}_h}{(1 - 34.676\mathrm{G}_h)(1 - 6.127\mathrm{G}_h)}$$

$$S_h = \frac{0.494}{1 - 34.676\mathrm{G}_h}$$

$$G_h = \frac{l^2 g}{q^2 \rho_0} \rho_z$$

F_u and F_v represent the terms for horizontal momentum diffusion in the x and y direction. F_T, and F_S represent the terms for thermal and salinity diffusion. These terms are of the form:

$$F_u \approx \frac{\partial}{\partial x}[2A_m H \frac{\partial u}{\partial x}] + \frac{\partial}{\partial y}[A_m H (\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})]$$

$$F_v \approx \frac{\partial}{\partial x}[A_m H (\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})] + \frac{\partial}{\partial y}[2A_m H \frac{\partial u}{\partial y}]$$

$$F_T \approx [\frac{\partial}{\partial x}(A_h H \frac{\partial}{\partial x}) + \frac{\partial}{\partial y}(A_h H \frac{\partial}{\partial y})]T$$

$$F_S \approx [\frac{\partial}{\partial x}(A_h H \frac{\partial}{\partial x}) + \frac{\partial}{\partial y}(A_h H \frac{\partial}{\partial y})]S$$

Where H is the bottom depth relative to z = 0. A_m and A_h are the horizontal momentum and thermal diffusion coefficients. These coefficients can be set constant or use the Smagorinsky eddy parameterization method (Smagorinsky, 1963). The Smagorinsky method defines the formula for the horizontal momentum diffusion coefficient as:

$$A_m = 0.5 C \Omega^u \sqrt{(\frac{\partial u}{\partial x})^2 + 0.5(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial y})^2}$$

Where C is a constant and Ω^u is the area of the individual momentum control element. A_m varies with the model resolution, decreasing as the grid size is reduced.

For temperature and salinity, a similar formula is used:

$$A_h = \frac{0.5 C \Omega^\zeta}{P_r} \sqrt{(\frac{\partial u}{\partial x})^2 + 0.5(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial y})^2}$$

Where C is a constant, Ω^ζ is the area of the individual tracer control element, and P_r is the Prandtl number. A_h is proportional to the area of the individual tracer control element and the horizontal gradient of the tracer concentration.

Formatting Data for FVCOM

FVCOM requires that its input data be in a specific format. Several scripts were written to facilitate this transformation. First, a program called “getdata” was written to automate retrieval of data from CBI (see Appendix A). Another program, called “2dm2fvcom” was written to convert the 2dm grid file created by SMS to a grid file formatted for FVCOM (see Appendix B). This program also calls getdata to retrieve the input water level, stream flow, and salinity data and formats this data so that they can be used as forcing inputs on the open boundaries of the model. The 2dm2fvcom script uses a configuration file to set parameters such as which data to retrieve, for which time period to retrieve it for, and what transformations to apply to the data. This configuration file also specifies how to create the initial temperature and salinity gradient. Appendix C shows a sample configuration file for 2dm2fvcom.

To facilitate and streamline the process of grid conversion, formatting input data, and uploading all input files to the server for running, a script called “bas” (build and sync) was created (see Appendix D). This script is intended to be run from within a directory containing the .2dm grid file output by SMS and a 2dm2fvcom.conf file. The directory should be named such that it identifies the run. The script calls sms2fvcom and getdata to build the FVCOM input files and then syncs them to a run folder on the destination host. It also takes, as an optional input parameter, the node to use for river input.

Running the Model

In order to achieve shorter run times, the FVCOM model was run in parallel mode on a multiprocessor computer. In this mode, the grid is divided in to sections for each processor to compute. This results in significantly faster run times than running the model in serial mode. Available multiprocessor machines included a sixteen-core server owned by CBI (Lighthouse), a 10 node cluster on TAMU-CC campus (Thresher) and the clusters at the Texas Advanced Computing Center (TACC) in Austin (Ranger and Lonestar). Previous run times for a 9 day case modeling of Corpus Christi Bay on a PC required 16 hours of runtime in serial mode (Nevel, 2010). Tests of the same runs in parallel mode on Lighthouse showed a reduction in run-time to approximately two hours.

Testing and model setup were performed on Lighthouse and Thresher. These machines were used because they did not have a limited allocation (as the TACC machines do) and the model could be run in the foreground immediately rather than having to be submitted to a queue and run in the background. This facilitated the setup phase where the model would regularly become unstable and need to be stopped, adjustments made, and then re-run.

Once the model was set up and appeared stable, it was run on Ranger at the TACC. Ranger is a Sun Constellation Linux Cluster with 3,936 nodes, 62,976 total processing cores, and 123 terabytes total memory. This system is capable of a peak performance of 579.4 teraflops (trillion floating point operations per second). Although only 5% of this system is allocated for use by Texas higher education institutions, this was sufficient for running the model with reasonable run times. With the optimal processor core count (discussed later), run times for one year were approximately 50 hours.

Patching FVCOM

During the testing of FVCOM, it was discovered that FVCOM was not properly initializing temperature and salinity from the input file. This made the use of an initial temperature and salinity gradient impossible. A small patch was developed and applied to the FVCOM code to correct the initialization error and allow the use of an initial temperature and salinity gradient. This patch, shown in Appendix E, was applied to the initial_uvel.F FVCOM source code file and FVCOM was recompiled.

Adjusting External and Internal Time Steps

For computational efficiency, FVCOM employs a mode-split model with an external 2D mode and an internal 3D mode. The modes can operate at different time steps with the relationship between the external time step (DTE) and the internal time step (DTI) represented by ISPLIT.

$$ISPLIT=\frac{DTI}{DTE}$$

Initially, an ISPLIT of 1 was used, which causes FVCOM to use equivalent internal and external time steps. A higher ISPLIT (and thus, a longer internal time step) resulted in shorter run times, but raising it too high would cause the model to become unstable. Through experimentation, it was determined that the optimal settings for DTE and ISPLIT for this case were determined to be 1 second and 10, respectively. Therefore the external time step used was 1 second, while the internal time step was 10 seconds. Longer time steps would result in shorter run times, but the model would become numerically unstable.

Efficiency and Number of Cores

When running software such as FVCOM in a parallel computing environment, the relationship between number of processing cores and computational time is not linear and quickly becomes asymptotic. This is due to the time it takes for communication among the cores. At some point, the cost of adding additional cores outweighs the benefit of reduced computational time. In fact, testing to determine the optimal number of cores for running FVCOM showed that computational time actually increased after a certain point. Figure 21 shows the relationship between processor cores and computational time for FVCOM. As can be seen, the optimal core count based on computational efficiency was found to be at least 256. Partial model runs using 1024 cores had a longer per-iteration time than 512 cores and therefore would have a longer run time for the full model run. In parallel computing, there is generally an optimum number of cores after which adding more will not increase and may actually decrease performance due to increased communication time between the processor cores.

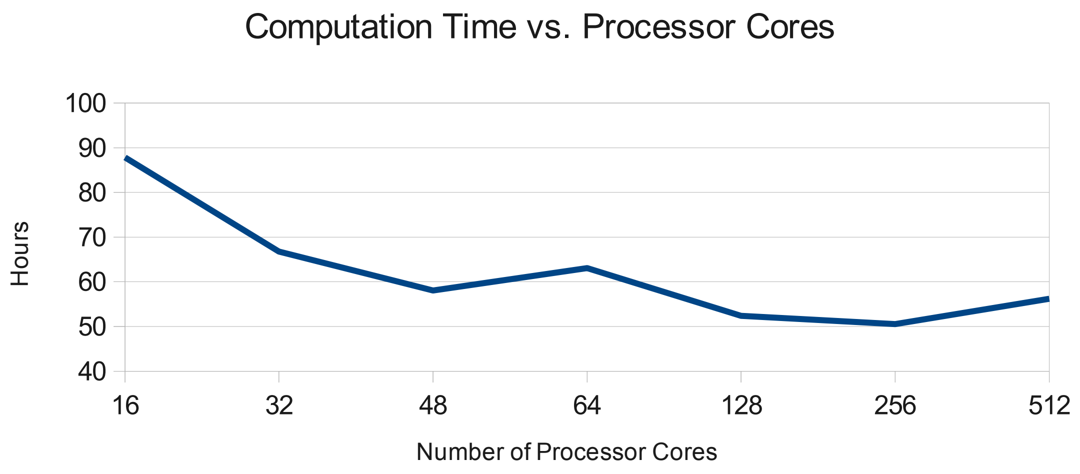

Figure 21: Computation Time vs. Processor Cores

The number of processing cores was further selected based on other considerations. During the process of optimizing the model it was found that increasing the core count to 1024 prevented the model from encountering numerical instability. This showed that the model could be run with this increased core count and remain stable, but at the increased cost of four times as many cores and increased computational time. Fortunately, FVCOM has the ability to dump "restart" files at a regular interval and then be "hot started" using a restart file to set initial conditions. This ability led to the following procedure:

1. start the model normally using 256 cores
2. allow the model to run until it encounters numerical instability and crashes
3. stop the model
4. restart the model using the last restart file dumped before the crash and 1024 cores
5. allow the model to run until it dumps a restart file
6. stop the model
7. restart the model using the restart file and 256 cores

This procedure allowed the use of 1024 cores to get past these "sticking points" but avoid using 1024 cores during the entire model run. Increasing the core count breaks the overall model grid into smaller sub-grids. It is not entirely clear why smaller sub-grids are more computationally stable, but it is possibly due to smaller matrices to invert and/or decreased occurrence of rounding errors leading to division by zero or very small numbers.

Visualization and Analysis of Model Output

While FVCOM outputs files that are viewable in SMS, it produces a separate file for each dump. This allows one to visualize the output at a single point in time, but makes it difficult to see change over time. A script called "fvcom2sms" (see Appendix F) was created in order to convert the FVCOM output files to a single file for each time series containing all time steps. These files could then be loaded into SMS, allowing the ability to step through each time step and visualize change over time.

Comparison of gridded model output data to collected data required determining which grid node is closest to the data collection location and extraction of a time series for that node. This was facilitated by the creation of two scripts, "findnode" (see Appendix G) and "extractnodes" (see Appendix H). The former finds the nearest grid node to a given location (provided in latitude and longitude coordinates). The latter extracts the time series for nodes specified in a configuration file (see Appendix I for a sample configuration file). The nodes can be specified by node number (as determined by findnode) or by latitude and longitude (in which case it will find the nearest node). The extractnodes scripts outputs the time series in comma separated values (csv) format, suitable for importing into various spreadsheet and data analysis software packages.

Model set-up and Verification

Model set-up was performed by comparing its output to historical salinity data collected at monitoring stations in Nueces Bay as well as water level stations in Corpus Christi and Nueces Bays. The period of historical measurements selected for model set-up was January - December 2008. This period contains a complete data set for all relevant stations. It contains periods of high and low salinity, water level and river flow (see Figures 15, 16, & 17).

During model set-up, it became clear that the model was not performing satisfactorily. An analysis of water levels (covered in the next section) pointed to the source of some of the error. Unfortunately, even after calibrating the model so that it predicted water levels fairly well, it still did not perform well for salinity. The model set-up results as well as a hypothesis of the reason for the large error in salinity are covered in the Results and Discussion section.

Had the model been able to successfully predict salinity during set-up, verification would have been performed using a separate time period. The time period of August through October 2010 was chosen due to the relative completeness of data (see Tables 3 & 4) and the inclusion of a large river flow event resulting in very low salinity in Nueces bay and two high water events (see Figures 18, 19, & 20). This verification step would have shown whether the model was able to model events outside of the set-up range, such as short-term high river flow and high water level.

Analyzing Water Levels

In an effort to determine why the model was predicting very little variation in salinity in Nueces Bay, an analysis of water levels was conducted to determine if the model was predicting them accurately. The findnode script was used to locate the nearest grid node to the Texas State Aquarium (008) TCOON water level station and extractnodes was used to extract the predicted water levels at that node. The extracted water levels were then compared to the measured data for the same time period. It was found that the water levels predicted by the model were significantly dampened when compared to the measured data. It was hypothesized that this was due to removing the Gulf portion of the model and forcing water level at Port Aransas instead of at a boundary in the Gulf. This was likely due to the much shorter boundary in the channel at Port Aransas (when compared to the long boundary in the Gulf). The shorter boundary probably resulted in much lower tidal velocities as water was raised and lowered at that boundary (when compared to the Gulf boundary). If this was the case, the dampened water levels within Corpus Christi Bay were due to these lower tidal velocities causing a much smaller volume of water to flow in and out of the bay.

This analysis seemed to indicate that it was not possible to remove the Gulf portion of the grid. The Gulf portion was restored and the water level analysis re-run. This time, water levels were analyzed at not only the Texas State Aquarium station, but also Port Aransas (009), Ingleside (006), and White Point (011). The analysis showed a much better prediction of water levels at Texas State Aquarium and reasonably accurate water levels at Port Aransas and Ingleside. The water levels at White Point, in Nueces Bay; however, showed very little fluctuation and did not correlate at all with the measured values. This lead to a visual analysis of water levels using SMS. The visual analysis showed almost no change in water level throughout Nueces Bay over the course of the model run. There appeared to be a hard boundary between Nueces and Corpus Christi Bays. An analysis of the bathymetry in this area indicated that indeed the entire boundary appeared to the model as dry land (>0m above MHW). This is not an accurate representation of the bathymetry in this area and would need to be corrected for water to flow between the bays.

The errors in bathymetry at the boundary of Nueces and Corpus Christi Bays are most likely due to LIDAR reflecting off the causeway that runs along this boundary. To correct these errors in bathymetry aerial imagery was overlaid to determine what section of the causeway was built over water. Bathymetry was then calculated for nodes underneath the causeway by interpolating from nodes on either side of the causeway. While not perfect, this provided a much better representation of bathymetry along the boundary of the bays. Correcting bathymetry allowed for water to flow between Corpus Christi and Nueces Bays, which was apparent in the water levels predicted at White Point. This exchange of water also had a significant effect on salinity in Nueces Bay.

The next issue was that, when initializing salinity at all nodes to 30 PSU, it would take a long time for the system to stabilize. In order to decrease this time, salinity was instead initialized with a gradient from the back of Nueces Bay to the boundary between Corpus Christi Bay and the Gulf at Port Aransas using measured data for the start time of the model run. Gulf salinity was initialized to that of the Port Aransas salinity station. This resulted in a much closer approximation of initial salinity.

<u>Sea Level Rise Scenarios</u>

Although the model did not perform well enough in calibrating it for salinity to make sound predictions, two sea-level rise scenarios were still tested to see if any significant change in salinity could be found. These scenarios included a “conservative” scenario and a “accelerating sea level rise” scenario. In the “conservative” scenario, the local average rate of sea level rise for 1957-2006 was assumed to remain constant from the model set-up year (2008) through the prediction target year (2100). For the “accelerating sea level rise” scenario, the upper bound of sea level rise for 2100, as predicted by the IPCC, was added to the local difference from historical global sea level rise.

The tide station at Rockport, TX is the nearest station to the study area with long-term historical sea level data. It is approximately 50 km from Bob Hall Pier, the station from which water levels were used as model input. A local sea level rise of 4.6 mm/year was observed at the Rockport tide station over a period of 52 years from 1948 through 1999 (Zervas, 2001). More recent data (see Figure 22) show an average rate of sea level rise of 5.16 mm/year from 1957 through 2006. If it assumed that the average rate of sea level rise from 2008 to 2100 will be the same as from 1957 to 2006, local sea level at Rockport can be predicted to be 0.47 m higher in 2100 than 2008.

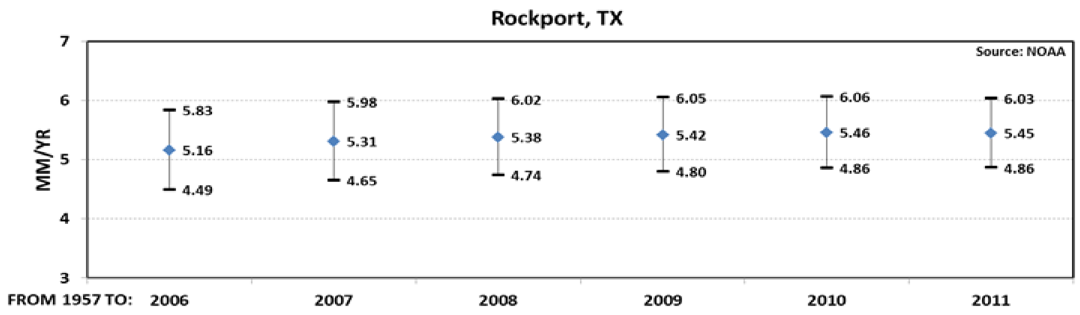

Figure 22: Sea Level Rise at Rockport, TX

The International Panel on Climate Change (IPCC) has reported that global sea level rose at an average rate of 1.8 mm/year from 1961 to 2003 (Bindoff et. al., 2007). This rate is lower than the rate of 5.16 mm/year observed at Rockport for approximately the same time period (1957-2006). This is due to a majority of the local apparent sea level rise being due to subsidence in the area. The approximate rate of subsidence can be calculated by subtracting the global rate (1.8 mm/year) from the local rate (5.16 mm/year). This results in a local subsidence rate of approximately 3.36 mm/year.

The IPCC has predicted that global sea level could rise by as much as 59 cm above 1999 levels by the end of 2099. This works out to an average rate of 5.9 mm/year. If the subsidence rate for 2008-2100 is assumed to be the same as for 1957-2006 (3.36 mm/year), a local apparent sea level rise prediction can be calculated as simply the sum of these rates (9.26 mm/year). Using this rate, apparent local sea level can be predicted to rise by as much as 0.85 m from 2008 to 2100.

These two predictions for sea level rise at Rockport were used to create two test cases. In each case, input water levels on the open Gulf boundary were calculated by adding the predicted rise in sea level from 2008 to 2100 to the water level data for 2008. The model was then run with each of these adjusted water level inputs. For all other forcings, including meteorological data and Gulf salinity, 2008 data was used unchanged as model input. While these forcings will likely be somewhat different in 2100, this study focused only on how higher sea level would affect salinity.

Results and Discussion

Model Set-up Results

The model was first set-up for prediction of water levels. After corrections were made to bathymetry, the model predicted water level fairly well, as can be seen in Figures 23-26.

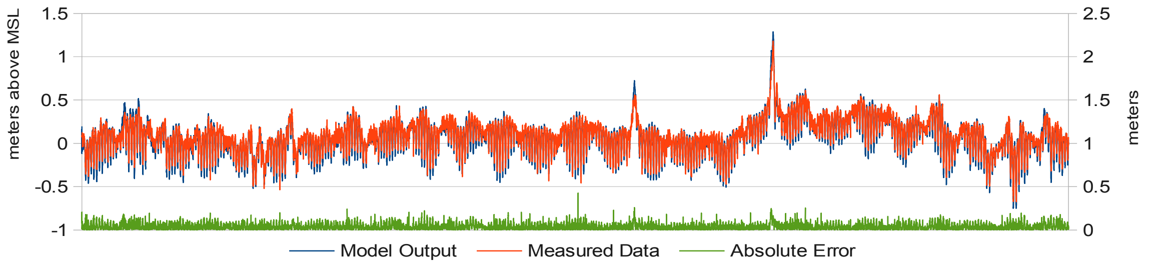

Figure 23: Model Output and Measured Water Level Data for Port Aransas

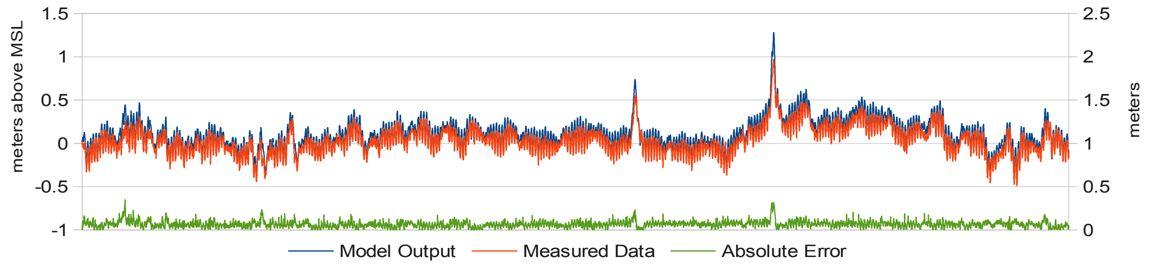

Figure 24: Model Output and Measured Water Level Data for Ingleside

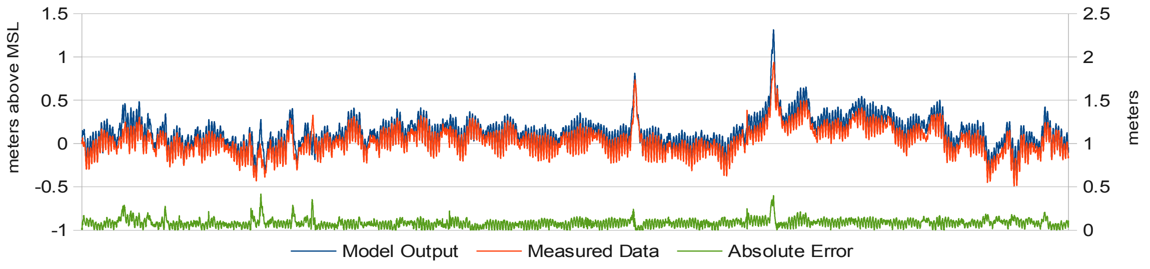

Figure 25: Model Output and Measured Water Level Data for Texas State Aquarium

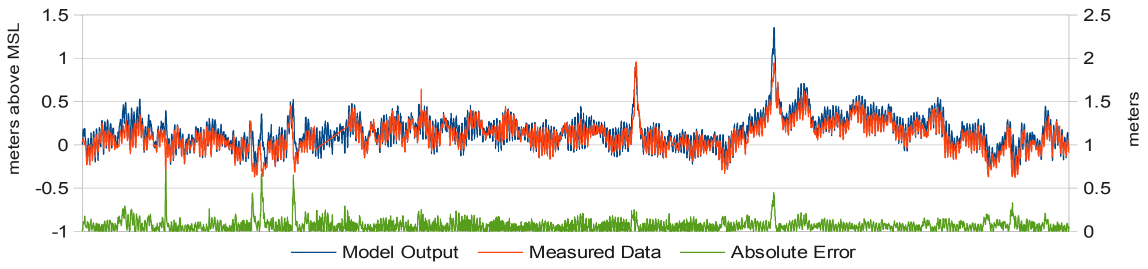

Figure 26: Model Output and Measured Water Level Data for White Point

As shown in Table 5, the mean absolute error for water levels for all stations was no greater than 8.3 cm and averaged approximately 6.5 cm.

	Mean Absolute Error	Maximum Error	Measured Variance	Predicted Variance
Port Aransas (009)	0.040 m	0.425 m	0.036 m^2	0.045 m^2
Ingleside (006)	0.066 m	0.349 m	0.025 m^2	0.030 m^2
Texas State Aquarium (008)	0.083 m	0.418 m	0.026 m^2	0.032 m^2
White Point (011)	0.074 m	0.702 m	0.028 m^2	0.032 m^2

Table 5: Error Between Model Output Water Levels and Measured Water Levels

Figures 27-29 show a comparison of model output data to measured data for each of the salinity stations in Nueces Bay. It is immediately clear from these graphs that the model did not capture the salinity variation.

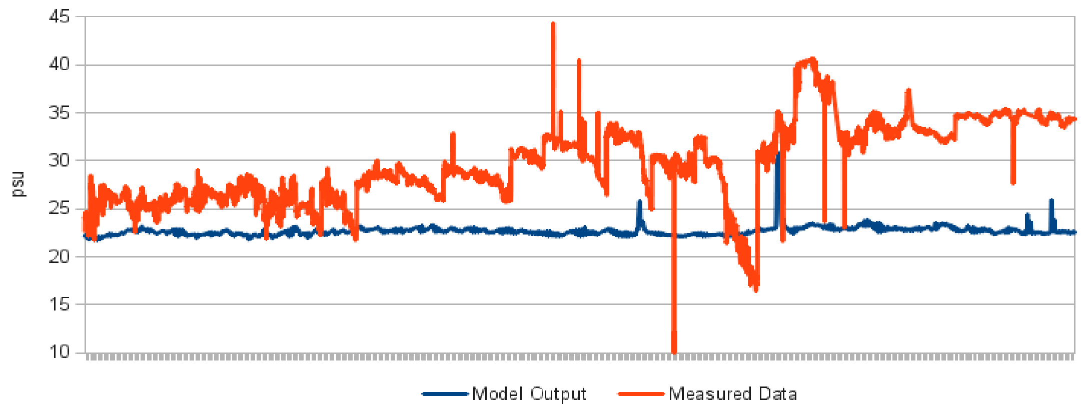

Figure 27: Model Output and Measured Salinity Data for SALT01

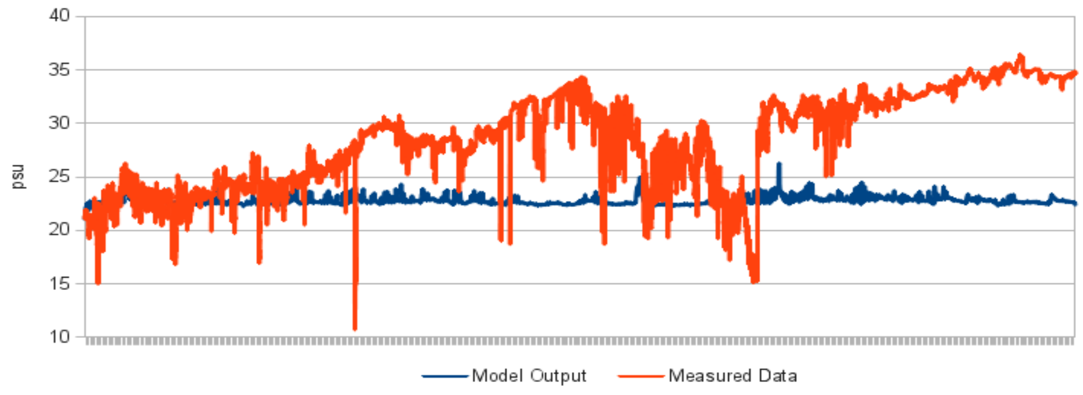

Figure 28: Model Output and Measured Salinity Data for SALT03

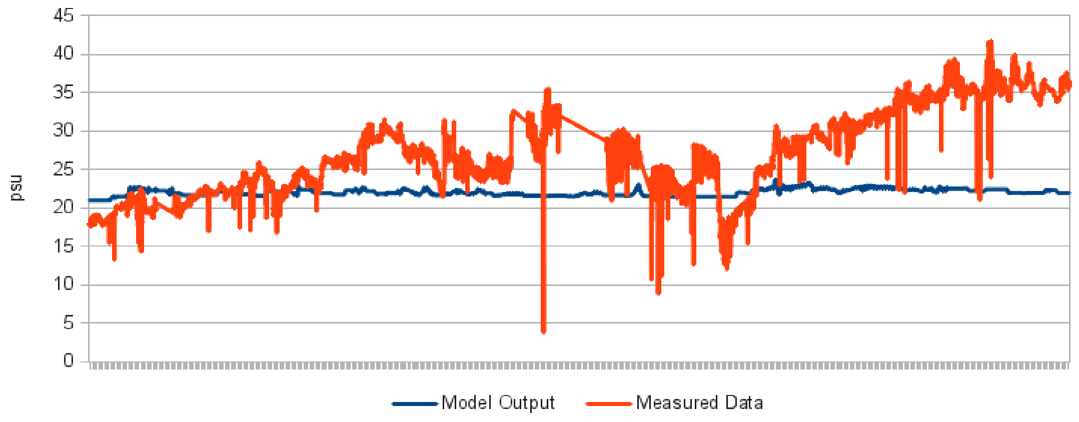

Figure 29: Model Output and Measured Salinity Data for SALT08

The mean absolute error and maximum error for the salinity stations in Nueces Bay are shown in Table 6. This mathematical analysis also makes it clear that model performed quite poorly for salinity at these locations with a mean absolute error greater than 6 psu and maximum absolute error of greater than 13 psu.

	Mean Absolute Error	Maximum Error
SALT01 (072)	7.2 psu	22.2 psu
SALT03 (074)	6.0 psu	13.8 psu
SALT08 (079)	6.1 psu	19.4 psu

Table 6: Error Between Model Output Salinity and Measured Salinity

Table 7 lists the variances of the model output and measured data at each salinity station. This even more dramatically shows how poorly the model performed with very little variance for the model output (~0.2 psu) compared to variances of 17-29 psu for measured data.

	Model Output Variance	Measured Variance
SALT01 (072)	0.3 psu	17.0 psu
SALT03 (074)	0.2 psu	19.2 psu
SALT08 (079)	0.2 psu	29.0 psu

Table 7: Model Output Salinity Variance vs. Measured Salinity Variance

<u>Model Set-up Results in Corpus Christi Bay</u>

In order to determine if poor salinity performance was only isolated to Nueces Bay or included the entire study area, salinity data for a station in the Corpus Christi Ship Channel near Ingleside was obtained from Solomon Negusse of the Texas Water Development Board in March 2013. Comparison of this data to model output at this location showed much better results than in Nueces Bay. The mean absolute error was 1.7 psu with a maximum absolute error of 7.6 psu. This indicates that the model does work for salinity in locations where salinity is primarily influenced by tidal events, just not for Nueces Bay where riverine input is the primary influence for short-term fluctuations. Other variables influencing salinity include tidal fluctuations, evaporation, and direct precipitation and freshwater input from runoff outside of the Nueces River. Figure 30 shows the forcing salinity input at the open boundary in the Gulf of Mexico and the model output and measured data at Ingleside. It is clear from this graph that salinity is being model much more accurately here than in Nueces Bay.

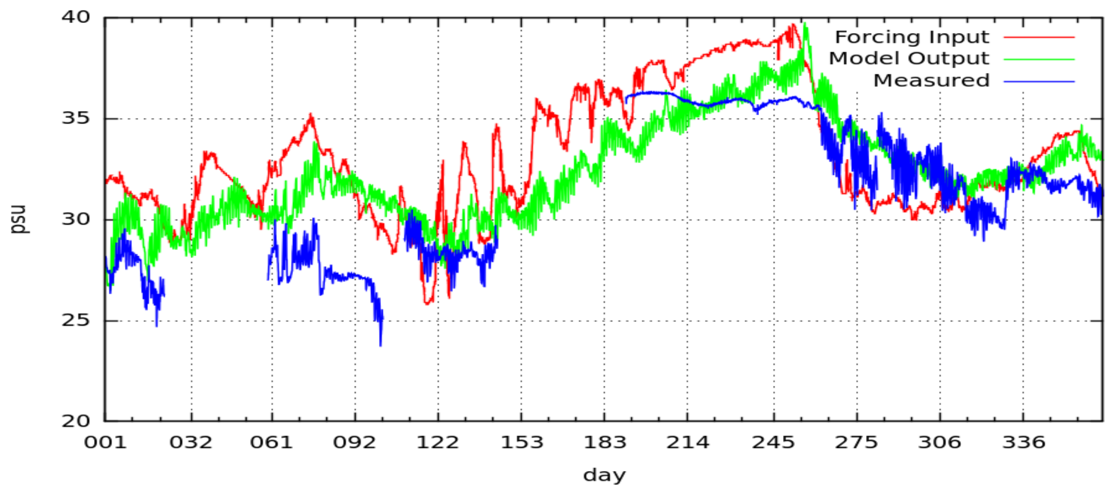

Figure 30: Salinity Forcing Input in the Gulf of Mexico and Model Output vs. Measured Salinity at Ingleside

Investigation of Poor Model Performance for Nueces Bay Salinity

The model set-up results for the Ingleside station in Corpus Christi Bay indicate that salinity works decently when the primary influence is not freshwater riverine input, so the focus of the investigation of poor model performance for Nueces Bay salinity was on the input from the Nueces River. A visual analysis of salinity over time near the freshwater input point on the Nueces River showed fresh water entering the system and influencing the nearby tidal flats during strong southeasterly winds (Figure 31), but quickly turning saline as it moved down the river. This seemed to indicate that the model was possibly not accounting for the correct amount of freshwater entering the system. This was tested by analyzing the model outputs for average current just down river from the input point during times of high river flow (see Figure 32). The expected current was calculated by dividing the flow rate (in cms) by the cross sectional area of the river. The model current output seemed to match the expected current, indicating that the correct amount of freshwater was indeed entering the system. This result shows that while the model is correctly representing the volume of water moving down the river and entering Nueces Bay further transport and mixing of the freshwater within the river channel was however not modeled properly. In particular, Figure 32 shows the freshwater plume following the large riverine outflow of 12 cms. During this event a salinity front makes its way back towards the river input at the same time as strong modeled riverine currents should be pushing freshwater towards Nueces Bay. The cause of this problem could not be identified and future work could involve a more in depth review of the model internal calculations for salinity and is beyond the scope of this thesis.

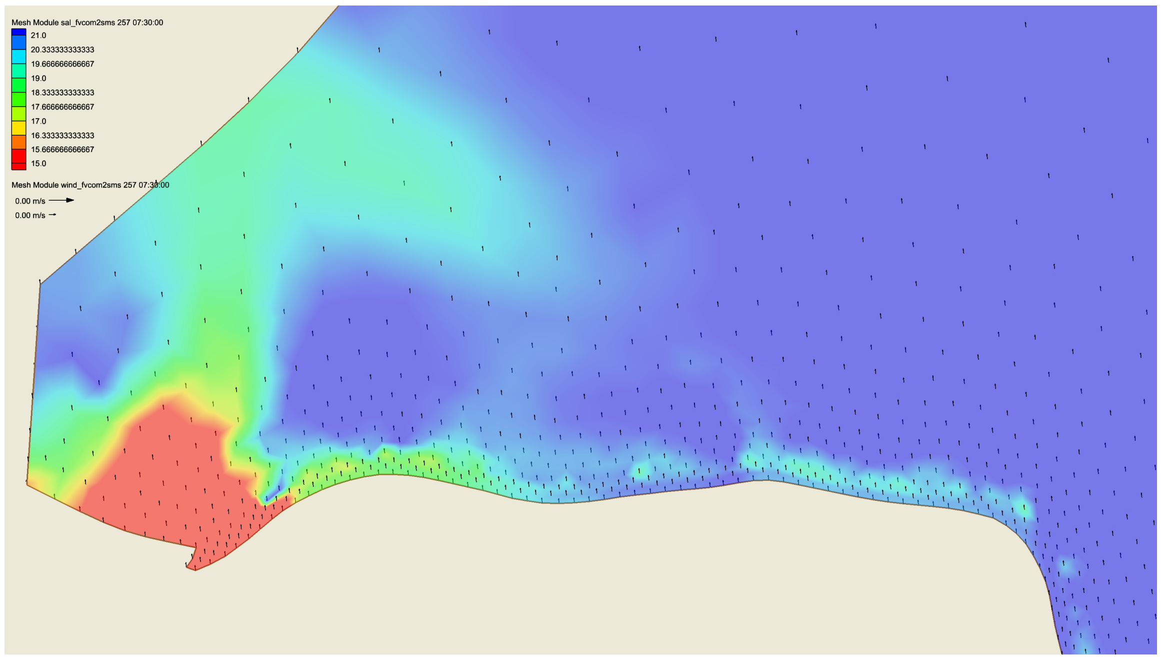

Figure 31: Fresh Water Influence on Tidal Flats (Salinity and Wind)

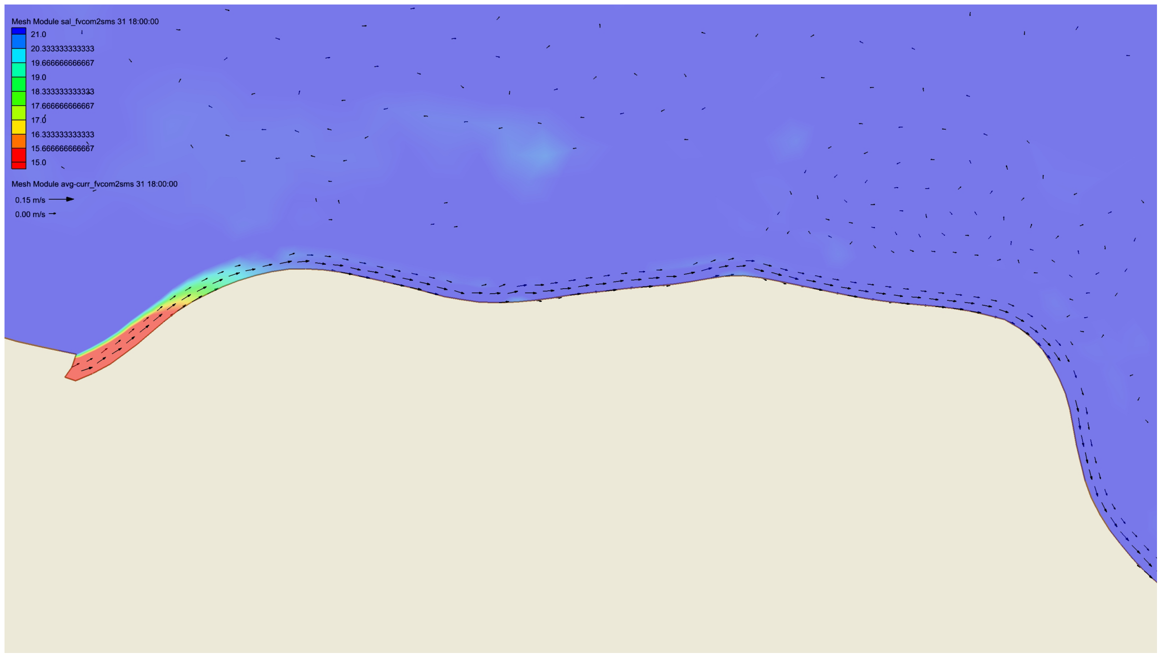

Figure 32: High River Flow (Salinity and Average Current)

<u>Sea Level Rise Scenarios</u>

Even though the model could not be set-up successfully for salinity, the sea level rise scenarios were still tested. These results show that the mechanics of the method worked, but the actual results for salinity can unfortunately not be analyzed carefully due to the inability to accurately calibrate the model for salinity.

For the "conservative" scenario, where the local average rate of sea level rise for 1957-2006 was assumed to remain constant from the model set-up year (2008) through the prediction target year (2100), salinity increased by an average of 0.5 psu across all three of the Nueces Bay stations. Table 8 shows the average salinity difference at each station for this scenario.

	Average Salinity Difference
SALT01 (072)	0.8 psu
SALT03 (074)	0.4 psu
SALT08 (079)	0.4 psu

Table 8: Salinity Difference for the "conservative" Sea Level Rise Scenario

For the "accelerating sea level rise" scenario, where the upper bound of sea level rise for 2100, as predicted by the IPCC, was added to the local difference from historical global sea level rise, salinity increased by an average of 0.6 psu across all three of the Nueces Bay stations. Table 9 shows the average salinity difference at each station for this scenario.

	Average Salinity Difference
SALT01 (072)	1.0 psu
SALT03 (074)	0.3 psu
SALT08 (079)	0.4 psu

Table 9: Salinity Difference for the "accelerating sea level rise" Scenario

Future Work

Future work could involve review of the model internal calculations for salinity and adding precipitation and evaporation inputs for greater accuracy. Reviewing the internal salinity calculations would require a detailed analysis of the model code which relates to mixing and salinity diffusion. Adding precipitation inputs would involve analysis of Doppler radar to determine the amount of rainfall over the bays and the immediate area around the bays. Evaporation could be estimated using air temperature, solar radiation, humidity, and wind speed. To use precipitation and evaporation for prediction would require the use of an atmospheric model to predict the needed weather inputs.

Summary

In summary, this study set-up an instance of the Finite Volume Coastal Ocean Model (FVCOM) for Nueces Bay and Corpus Christi Bays and attempted to use this model to predict how salinity could be affected by sea level rise. An unstructured triangular mesh was created and optimized for the study area and the model was forced with inputs including water level, water temperature, salinity, wind speed and direction, and the flow rate of the Nueces River. Several software utilities were created to facilitate the set-up and assessment of the model. To assess model performance, model output predictions were compared with measured water levels and salinity at several locations in the study area for 2008. The model was successfully calibrated for water levels and for salinity in at least some areas of Corpus Christi Bay, but performed poorly for salinity in Nueces Bay. While the initial goal of this study was to investigate the impact of sea level rise on salinity levels, the study focused instead on model performance for salinity predictions in Nueces Bay. The investigation revealed that while freshwater from the Nueces River was entering the system at the correct rate, the model was not accurately reflecting salinity response in cells further down river and ultimately in Nueces Bay.

Literature Cited

Bek, M.A., Lowndes, I.S., Hargreaves, D.M. , 2010. The application of a validated hydrodynamic model to improve the water management of an Egyptian shallow water coastal lake. Proceedings of the 2010 International Congress on Environmental Modelling and Software Modelling for Environment's Sake, Fifth Biennial Meeting.

Bindoff, N.L., Willebrand, J., Artale, V., Cazenave, A., Gregory, J., Gulev, S., Hanawa, K., Le Quéré, C., Levitus, S., Nojiri, Y., Shum, C.K., Talley, L.D., Unnikrishnan, A., 2007. Observations: Oceanic Climate Change and Sea Level. In: Climate Change 2007: The Physical Science Basis. Contribution of Working Group I to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change [Solomon, S., Qin, D., Manning, M., Chen, Z., Marquis, M., Averyt, K.B., Tignor, M., Miller, H.L. (eds.)]. Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA.

Blackburn, J., 2004. The Book of Texas Bays, 205-212.

Chen, C., Liu H., Beardsley, R.C., 2003. An unstructured grid, finite-volume, three-dimensional, primitive equations ocean model: application to coastal ocean and estuaries. Journal of Atmospheric and Oceanic Technology 20, 159-186.

Chen, C., Beardsley, R.C., Cowles, G., 2006. An Unstructured Grid, Finite-Volume Coastal Ocean Model; FVCOM User Manual.

Chen, C., Qi, J., Li, C., Beardsley, R.C., Lin, H, Walker, R., Gates, K., 2008. Complexity of the flooding/drying process in an estuarine tidal-creek salt-marsh system: an application of FVCOM. Journal of Geophysical Research 113, C07052.

Galperin, B., Kantha, L.H., Hassid, S., Rosati, A., 1988. A quasi-equilibrium turbulent energy model for geophysical flows. Journal of the Atmospheric Sciences 45 (1), 55-62.

Hardisty, J, 2007. Estuaries: Monitoring and Modeling the Physical System.

Luettich, R.A., Jr., 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, Dredging Research Program Technical Report DRP-92-6, U.S. Army Engineers Waterways Experiment Station, Vicksburg, MS, 137p

Mellor, G.L., Yamada, T., 1982. Development of a turbulence closure model for geophysical fluid problems. Reviews of Geophysics and Space Physics 20 (4), 851-875.

Nevel, Y., 2010. Data assimilation for a hydrodynamic model. Master's Thesis. Texas A&M University-Corpus Christi.

Parris, A., Bromirski, P., Burkett, V., Cayan, D., Culver, M., Hall, J., Horton, R., Knuuti, K., Moss, R., Obeysekera, J., Sallenger, A., Weiss, J., 2012. Global Sea Level Rise Scenarios for the US National Climate Assessment. NOAA Tech Memo OAR CPO-1. 37 pp.

Rahmstorf, S., 2007. A Semi-empirical approach to projecting future sea-level rise. Science, 315, 368-370.

Savenije, H.H.G., 2005. Salinity and Tides in Alluvial Estuaries.

Shchepetkin, A.F., McWilliams, J.C., 2005, The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. Ocean Modelling, 9, 347-404

Signell, Rich, 2010. National Oceanic and Atmospheric Administration National Geophysical Data Center Coastline Extractor. <http://rimmer.ngdc.noaa.gov>.

Smagorinsky, J., 1963. General circulation experiments with the primitive equations, I. The basic experiment. Monthly Weather Review, 91 (3), 99-164.

Taylor, L.A., Eakins, B.W., Carignan, K.S., Warnken, R.R., Sazonova, T., Schoolcraft, D.C., 2008 Digital elevation model of Corpus Christi, TX: Procedures, data sources and analysis. NOAA Technical Memorandum NESDIS NGDC-11, National Geophysical Data Center, Marine Geology and Geophysics Division.

TerraServer, 2010. Aerial Imagery retrieved for Nueces Bay from TerraServer via the Surface-water Modeling System.

Uddameri, V., Parvathinathan, G., 2007. Climate Change Impacts on Water Resources in South Texas. In: The Changing Climate of South Texas 1900-2100: Problems and Prospects, Impacts and Implications, Chapter 6, pp.109-125.

Yang, G., Zhu, J., 1993. A study of impacts of global sea level rise on salt water intrusion into the Changjiang Estuary,. Science in China, Series B, 36 (11): 1391.

Yang, Z., Khangaonkar, T., 2008. Modeling tidal circulation and stratification in Skagit River estuary using an unstructured grid ocean model. Ocean Modelling 28, 34-39.

Zervas, C., 2001. Sea Level Variations of the United States 1854-1999 (PDF). NOAA technical report NOS CO-OPS 36. NOAA National Ocean Service, Silver Spring, MD, 186 pp. Most recent data available from: <http://tidesandcurrents.noaa.gov/sltrends/sltrends_station.shtml?tnid=8774770>.

Zhang, Y. and Baptista, A.M., 2008. SELFE: A semi-implicit Eulerian-Lagrangian finite- element model for cross-scale ocean circulation. Ocean Modelling, 21(3-4), 71-96.

Appendices

Appendix A: Source Code: getdata

```
#!/usr/bin/perl

use WWW::Mechanize;

my %a = (sslist => '014-pwl,005-pwl,014-wsd,014-wdr,009-wsd,009-wdr',
           when => '2/1/2010,+6d',
           unit => 'metric',
           elev => 'msl',
         interp => 'lin',
         extrap => 'n');

for my $arg (@ARGV) {
    my ($key,$val) = $arg =~ /^([^=]*)=([^=]*)$/;
    $a{$key} = $val;
}

$a{when} =~ s/\+/%2b/g;

my $agent = WWW::Mechanize->new();
$agent->get("http://lighthouse.tamucc.edu/pd?
sslist=$a{sslist}&when=$a{when}&unit=$a{unit}&elev=$a{elev}&-
action=csv&interp=$a{interp}&extrap=$a{extrap}&survey_skip=y&interval=3
60");

open(OUT,'>','data.csv');
print OUT $agent->{'content'};
close(OUT);
```

Appendix B: Source Code: 2dm2fvcom

```
#!/usr/bin/perl

use FindBin;
use Config::General;
use Geo::Coordinates::UTM;

local $| = 1;

unless ( -f "2dm2fvcom.conf" ) {
     print STDERR "Error: missing config file!\nPlease place a config
file named 2dm2fvcom.conf in the current directory.\n";
    print STDERR "See: $FindBin::Bin/2dm2fvcom.conf.sample for a sample
config file\n";
    exit;
}

$conf = new Config::General("2dm2fvcom.conf");
my %config = $conf->getall;

my $rev = 1;
$rev = -1 if $conf{'reversebathsign'} =~ /^y/i;

unless ($ARGV[0] && -f $ARGV[0]) {
    print STDERR "Error: must specify 2dm file\n";
    exit;
}

use Math::Trig qw(deg2rad rad2deg acos);

my @wlstns = split ',', $config{'wlstns'};
my @wtpstns = split ',', $config{'wtpstns'};
my @salstns = split ',', $config{'salstns'};
my @windstns = split ',', $config{'windstns'};

if ($config{'getdata'} =~ /^y/i) {

    my %seen;
    my @sslist;
    for my $wlstn (@wlstns) {
        push @sslist, "$wlstn-pwl" unless $seen{"$wlstn-pwl"};
        $seen{"$wlstn-pwl"}++;
    }
    for my $wtpstn (@wtpstns) {
        push @sslist, "$wtpstn-wtp" unless $seen{"$wtpstn-wtp"};
        $seen{"$wtpstn-wtp"}++;
    }
    for my $salstn (@salstns) {
        push @sslist, "$salstn-sal" unless $seen{"$salstn-sal"};
        $seen{"$salstn-sal"}++;
    }
    for my $windstn (@windstns) {
        push @sslist, "$windstn-wsd" unless $seen{"$windstn-wsd"};
        $seen{"$windstn-wsd"}++;
        push @sslist, "$windstn-wdr" unless $seen{"$windstn-wdr"};
        $seen{"$windstn-wdr"}++;
    }
    if (exists $config{rivnode} or exists $config{rivedge}) {
              push  @sslist,  "$config{rivflow}-flow"  unless  not  exists
$config{rivflow} or $config{rivflow} =~ /^constant:/;
               push  @sslist,  "$config{rivwtp}-wtp"  unless  not  exists
$config{rivwtp} or $config{rivwtp} =~ /^constant:/;
               push  @sslist,  "$config{rivsal}-sal"  unless  not  exists
$config{rivsal} or $config{rivsal} =~ /^constant:/;
    }
    my $sslist = join ',', @sslist;

    my $interp = 'n';
    $interp = 'lin' if $config{'interp'} =~ /^y/i;
    my $extrap = 'n';
    $extrap = 'y' if $config{'extrap'} =~ /^y/i;

    print "Getting data...";
       system("$FindBin::Bin/getdata  sslist=$sslist  when=$config{when}
unit=metric elev=$config{datum} interp=$interp extrap=$extrap") and die
"could not retrieve data: $!";
    print "Done!\n";
}

unless (-f 'data.csv') {
      print  STDERR  "Error:  no  data  file  found.  You  may  need  to  set
'getdata = y' in the config file.\n";
    exit;
}

unless (system('grep NA data.csv >/dev/null')) {
      print  STDERR  "Error:  missing  values  found  in  the  data.\nYou  may
need to set ";
    print STDERR "interp = y" unless $interp eq 'lin';
    print STDERR " and/or " if $interp eq 'n' and $extrap eq 'n';
    print STDERR "extrap = y" unless $extrap eq 'y';
       print  STDERR  "  in  the  config  file  or  use  a  different  date
range.\nSee: data.csv\n";
    exit;
}

my $inpdir = 'INPDIR_' . uc($config{'name'});

my @E3T;
my @ND;
my @NS;

print "Reading 2dm file...";
while (<>) {
    if (/^E3T\s/) {
        my @line = split;
        push @E3T, \@line;
    }
    if (/^ND\s/) {
        my @line = split;
        push @ND, \@line;
    }
    if (/^NS\s/) {
        my @line = split;
        push @NS, \@line;
    }
}
print "Done!\n";

my @boundary_nodes;
my @station_nodes;

my $node = 1;
for my $r (@NS) {
    my $end_node_seen = 0;
    shift @{$r};
    for my $c (@{$r}) {
        next if $end_node_seen;
        push @boundary_nodes,abs($c);
        if ($c<0) {
            push @station_nodes,$node if $c<0;
            $end_node_seen = 1;
        }
        $node++;
    }
}

my %data;
my @headers;
print "Reading data file...";
open(DATA, '<', 'data.csv') or die "could not open data.csv: $!\n";
while (<DATA>) {
    chomp;
    if (/^"?#\s*date\+time/) {
        s/["#]//g;
        @headers = split /[, ]+/;
        next;
    }
    next if /^"?#/;
    s/"//g;
    my @row = split /[, ]+/;
    my $col = 0;
    for my $dv (@row) {
        push @{$data{$headers[$col]}}, $dv;
        $col++;
    }
}
close(DATA);
print "Done!\n";

mkdir($inpdir) unless -d $inpdir;
```

```
print "Writing $inpdir/$config{name}".'_bfw.dat...';
open(OUT, '>', "$inpdir/$config{name}".'_bfw.dat') or die "could not
open $inpdir/$config{name}"."_bfw.dat: $!\n";
print OUT "Number of bottom fresh water input points\n   0\n";
close(OUT);
print "Done!\n";

my $ellipsoid = 'WGS-84';
for my $point (@{$config{tsinitgrad}->{point}}) {
    if (exists $point->{lat} and exists $point->{lon}) {
                ($point->{zone},$point->{easting},$point->{northing}) =
latlon_to_utm($ellipsoid,$point->{lat},$point->{lon});
    }
}

print "Writing $inpdir/$config{name}".'_its.dat...';
open(OUT, '>', "$inpdir/$config{name}".'_its.dat') or die "could not
open $inpdir/$config{name}"."_its.dat: $!\n";

print OUT "Input temperature and salinity\n";

my $constant = 0;

if ($constant) {
    print OUT "constant\n20. 30.\n";
}
else {
    print OUT "observed\n";
       open(SAL, '>', "$inpdir/$config{name}".'_sal_init.dat') or die
"could not open $inpdir/$config{name}"."_sal_init.dat: $!\n";
       open(TMP, '>', "$inpdir/$config{name}".'_tmp_init.dat') or die
"could not open $inpdir/$config{name}"."_tmp_init.dat: $!\n";
    my $numnodes = @ND;
    my $numcells = @E3T;
    print SAL qq(DATASET
OBJTYPE "mesh2d"
BEGSCL
ND   $numnodes
NC  $numcells
NAME "sal_init"
TIMEUNITS seconds
TS 0 0
);
    print TMP qq(DATASET
OBJTYPE "mesh2d"
BEGSCL
ND   $numnodes
NC  $numcells
NAME "tmp_init"
TIMEUNITS seconds
TS 0 0
);
    for my $n (@ND) {
        my ($node,$easting,$northing) = @{$n}[1..3];

        my @points = @{$config{tsinitgrad}->{point}};
        my @dist;
        my $dsum = 0;

        for my $point (@points) {
            if (exists $point->{usefornodes}) {
                  if ($point->{usefornodes} eq 'eastof' and $easting >
$point->{easting}) {
                    @points = ($point);
                    last;
                }
                  if ($point->{usefornodes} eq 'westof' and $easting <
$point->{easting}) {
                    @points = ($point);
                    last;
                }
                 if ($point->{usefornodes} eq 'northof' and $northing >
$point->{northing}) {
                    @points = ($point);
                    last;
                }
                 if ($point->{usefornodes} eq 'southof' and $northing <
$point->{northing}) {
                    @points = ($point);
                    last;
                }
            }
                if (exists $point->{usefornodeslt} and $node < $point-
>{usefornodeslt}) {
                @points = ($point);
                last;
            }
            if (exists $point->{circle}) {
                  my $d = sqrt( ($point->{circle}->{x} - $easting)**2 +
($point->{circle}->{y} - $northing)**2 );
                if ($d < $point->{circle}->{r}) {
                    @points = ($point);
                    last;
                }
            }
             my $d = sqrt( ($point->{easting} - $easting)**2 + ($point-
>{northing} - $northing)**2 );
            $dsum += $d;
            push @dist, $d;
        }

        my ($t,$s) = (0,0);

        my $m = 0;

        my $numpoints = @points;
        if ($numpoints < 2) {
            $t = $points[0]->{t};
            $s = $points[0]->{s};
        }
        else {
            for ($i=0;$i<$numpoints;$i++) {
                my $weight = 1 - $dist[$i]/$dsum*($numpoints - 1);
                $t += $points[$i]->{t} * $weight;
                $s += $points[$i]->{s} * $weight;
                $m += $weight;
            }
        }

        my (@trow,@srow);
        for (my $c = 0; $c < $config{'layers'}+1; $c++) {
            push @trow, sprintf("%.1f",$t);;
            push @srow, sprintf("%.1f",$s);;
        }
        print OUT join("\t", @trow),"\n";
        print OUT join("\t", @srow),"\n";
        print SAL " $s\n";
        print TMP " $t\n";
    }

    print SAL 'ENDDS';
    print TMP 'ENDDS';
    close(SAL);
    close(TMP);
}
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_spg.dat...';
open(OUT,  '>',  "$inpdir/$config{name}".'_spg.dat')  or  die  "could  not
open $inpdir/$config{name}"."_spg.dat: $!\n";
print OUT "0\n51   100000.   0.01\n12    80000.   0.002\n";
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_cor.dat...';
open(OUT,  '>',  "$inpdir/$config{name}".'_cor.dat')  or  die  "could  not
open $inpdir/$config{name}"."_cor.dat: $!\n";
for my $r (@ND) {
    print OUT uc(sprintf("%.8e\t%.8e\t%.2f\r\n",@{$r}[2],@{$r}[3],27));
}
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_dep.dat...';
open(OUT,  '>',  "$inpdir/$config{name}".'_dep.dat')  or  die  "could  not
open $inpdir/$config{name}"."_dep.dat: $!\n";
for my $r (@ND) {
        print  OUT  uc(sprintf("%.4f\t%.4f\t%.4f\r\n",@{$r}[2],@{$r}[3],
(@{$r}[4]+$config{'shiftbath'}*-$rev)*$rev));
```

```
}
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_el_ini.dat...';
my $initwl = 0;
$initwl = $config{initwl} if exists $config{initwl};
open(OUT, '>', "$inpdir/$config{name}".'_el_ini.dat') or die "could not
open $inpdir/$config{name}"."_el_ini.dat: $!\n";
for (@ND) {
    print OUT sprintf('%.1f  ',$initwl + $config{'shiftbath'})."\r\n";
}
close(OUT);
print "Done!\n";

my @wave = split /\s+/, $config{'wave'};

print "Writing $inpdir/$config{name}".'_elj_obc.dat...';
open(OUT,  '>',  "$inpdir/$config{name}".'_elj_obc.dat')  or  die  "could
not open $inpdir/$config{name}"."_elj_obc.dat: $!\n";
for (my $i=0; $i<@{$data{'date+time'}}; $i++) {
    my $col = 1;
    my $stn = 0;
    for my $node (@boundary_nodes) {
        $stn++ if $col > $station_nodes[$stn];
        my $header = "$wlstns[$stn]-pwl";
        if ($i < $wave[$col-1]) {
            print OUT sprintf('%.3f  ',$config{'shiftbath'});
        }
        else {
             print OUT sprintf('%.3f  ',$data{$header}->[$i-$wave[$col-
1]] + $config{'shiftbath'});
        }
        $col++;
    }
    print OUT "\r\n";
}
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_grd.dat...';
open(OUT,  '>',  "$inpdir/$config{name}".'_grd.dat')  or  die  "could  not
open $inpdir/$config{name}"."_grd.dat: $!\n";
for my $r (@E3T) {
         print  OUT  uc(sprintf("%d\t%d\t%d\t%d\t%d\r\n",@{$r}[1],@{$r}
[2],@{$r}[3],@{$r}[4],@{$r}[5]));
}
for my $r (@ND) {
      print  OUT  uc(sprintf("%d\t%.8e\t%.8e\t%.8e\t\r\n",@{$r}[1],@{$r}
[2],@{$r}[3],(@{$r}[4]+$config{'shiftbath'}*-$rev)*$rev));
}
close(OUT);
print "Done!\n";
print "Writing $inpdir/$config{name}".'_jmpobc.dat...';
open(OUT, '>', "$inpdir/$config{name}".'_jmpobc.dat') or die "could not
open $inpdir/$config{name}"."_jmpobc.dat: $!\n";
    print OUT scalar(@boundary_nodes),"\r\n";
    for my $node (@boundary_nodes) {
        print OUT "$node\r\n";
    }
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_obc.dat...';
open(OUT, '>', "$inpdir/$config{name}".'_obc.dat') or die "could not
open $inpdir/$config{name}"."_obc.dat: $!\n";
    print OUT uc($config{'name'}), " BOUNDARY POINTS\r\n";
    my $i = 1;
    for my $node (@boundary_nodes) {
        print OUT "$i\t$node\t1\r\n";
        $i++;
    }
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_uv_ini.dat...';
open(OUT, '>', "$inpdir/$config{name}".'_uv_ini.dat') or die "could not
open $inpdir/$config{name}"."_uv_ini.dat: $!\n";
    for my $r (@E3T) {
        my @row;
        for (my $c = 0; $c < $config{'layers'} * 2; $c++) {
            push @row, "0.0";
        }
        print OUT join("\t", @row),"\r\n";
    }
close(OUT);
print "Done!\n";

print "Writing $inpdir/$config{name}".'_mc.dat...';
open(OUT, '>', "$inpdir/$config{name}".'_mc.dat') or die "could not
open $inpdir/$config{name}"."_mc.dat: $!\n";
print OUT "Time variable meterological data\r\n";
my $hour = 0;
for (my $i=0; $i<@{$data{'date+time'}}; $i++) {
    next unless $data{'date+time'}->[$i] =~ /00$/;
    print OUT "\t$hour.0\r\n";
    $hour++;
    my $avgwsd;
    my $avgwdr;
    if (scalar(@windstns) == 1) {
        $avgwsd = $data{"$windstns[0]-wsd"}->[$i];
        $avgwdr = $data{"$windstns[0]-wdr"}->[$i];
    }
    else {
        my $sumx = 0;
        my $sumy = 0;
        my $sumwsd;
        for my $windstn (@windstns) {
            my $wsd = $data{"$windstn-wsd"}->[$i];
            my $wdr = $data{"$windstn-wdr"}->[$i];
            $sumx += $wsd * cos(deg2rad($wdr));
            $sumy += $wsd * sin(deg2rad($wdr));
        }
        my $avgx = $sumx / scalar(@windstns);
        my $avgy = $sumy / scalar(@windstns);
        $avgwsd = sqrt($avgx**2 + $avgy**2);
        $avgwdr = rad2deg(acos($avgx/$avgwsd));
    }
    if ($avgwdr < 180) { $avgwdr += 180; }
    else { $avgwdr -= 180; }
         print   OUT   sprintf("  0.00000\t0.00000  %.5f  %.5f  0.00000
0.00000\r\n",$avgwsd,$avgwdr);
}
close(OUT);
print "Done!\n";

if ($config{'tsobc'} =~ /^constant/i) {
    my @wtps = split ',', $config{'wtps'};
    if (scalar(@wtps) != scalar(@station_nodes)) {
        print STDERR "Error: number of wtps != number of nodestrings\n";
        exit;
    }
    my @sals = split ',', $config{'sals'};
    if (scalar(@sals) != scalar(@station_nodes)) {
        print STDERR "Error: number of sals != number of nodestrings\n";
        exit;
    }
    print "Writing $inpdir/$config{name}".'_tsobc.dat...';
     open(OUT, '>', "$inpdir/$config{name}".'_tsobc.dat') or die "could
not open $inpdir/$config{name}"."_tsobc.dat: $!\n";
    my @days = (0,999);
    for my $day (@days) {
        print OUT sprintf("%11.4f\r\n",$day);
        print OUT "NO/Lay";
        for (my $l = 1; $l <= $config{'layers'}; $l++) {
            print OUT sprintf('%7d',$l);
        }
        print OUT "\r\n";

        # temp
        my $row = 1;
        my $stn = 0;
        for my $node (@boundary_nodes) {
            $stn++ if $row > $station_nodes[$stn];
            print OUT sprintf('%6d',$row);
            for (my $l = 1; $l <= $config{'layers'}; $l++) {
                print OUT sprintf('%7.3f',$wtps[$stn]);
            }
            print OUT "\r\n";
            $row++;
        }
        # salinity
        my $row = 1;
        my $stn = 0;
        for my $node (@boundary_nodes) {
            $stn++ if $row > $station_nodes[$stn];
            print OUT sprintf('%6d',$row);
            for (my $l = 1; $l <= $config{'layers'}; $l++) {
                print OUT sprintf('%7.3f',$sals[$stn]);
            }
            print OUT "\r\n";
            $row++;
        }

    }
    close(OUT);
    print "Done!\n";
}

if ($config{'tsobc'} =~ /^data/i) {

    my $data = \%data;

    if (exists $config{'tsobcfile'}) {
        my %tsobc_data;
        my @tsobc_headers;
        print "Reading tsobc data file...";
          open(DATA, '<', $config{'tsobcfile'}) or die "could not open
$config{tsobcfile}: $!\n";
        while (<DATA>) {
            chomp;
            if (/^"?#\s*date\+time/) {
                s/["#]//g;
                @tsobc_headers = split /[, ]+/;
                next;
            }
            next if /^"?#/;
            s/"//g;
            my @row = split /[, ]+/;
            my $col = 0;
            for my $dv (@row) {
                push @{$tsobc_data{$tsobc_headers[$col]}}, $dv;
                $col++;
            }
        }
        close(DATA);
        print "Done!\n";
        $data = \%tsobc_data;
    }

    my $tsobc_interval = 360;
            $tsobc_interval   =   $config{'tsobc_interval'}   if   exists
$config{'tsobc_interval'};
```

```
    print "Writing $inpdir/$config{name}".'_tsobc.dat...';
     open(OUT, '>', "$inpdir/$config{name}".'_tsobc.dat') or die "could
not open $inpdir/$config{name}"."_tsobc.dat: $!\n";


    for (my $i=0; $i<@{$data->{'date+time'}}; $i++) {

        my $day = $i/(3600/$tsobc_interval)/24;
        print OUT sprintf("%11.4f\r\n",$day);
        print OUT "NO/Lay";
        for (my $l = 1; $l <= $config{'layers'}; $l++) {
            print OUT sprintf('%7d',$l);
        }
        print OUT "\r\n";

        # temp
        my $row = 1;
        my $stn = 0;
        for my $node (@boundary_nodes) {
            $stn++ if $row > $station_nodes[$stn];
            my $header = "$wtpstns[$stn]-wtp";
            print OUT sprintf('%6d',$row);
            for (my $l = 1; $l <= $config{'layers'}; $l++) {
                print OUT sprintf('%7.3f',$data->{$header}->[$i]);
            }
            print OUT "\r\n";
            $row++;
        }

        # salinity
        my $row = 1;
        my $stn = 0;
        for my $node (@boundary_nodes) {
            $stn++ if $row > $station_nodes[$stn];
            my $header = "$salstns[$stn]-sal";
            print OUT sprintf('%6d',$row);
            for (my $l = 1; $l <= $config{'layers'}; $l++) {
                print OUT sprintf('%7.3f',$data->{$header}->[$i]);
            }
            print OUT "\r\n";
            $row++;
        }

    }

    close(OUT);
    print "Done!\n";
}

print "Writing $inpdir/$config{name}".'_riv.dat...';
open(OUT,  '>',  "$inpdir/$config{name}".'_riv.dat')  or  die  "could  not
open $inpdir/$config{name}"."_riv.dat: $!\n";

if (exists $config{rivnode} or exists $config{rivedge}) {
    my ($flowconst) = $config{rivflow} =~ /^constant:(.*)/;
    my ($wtpconst) = $config{rivwtp} =~ /^constant:(.*)/;
    my ($salconst) = $config{rivsal} =~ /^constant:(.*)/;

     print OUT "node  specified\n1\n$config{rivnode}\n$config{rivnode}"
if exists $config{rivnode};
     print OUT "edge  specified\n1\n$config{rivedge}\n$config{rivedge}"
if exists $config{rivedge};
      print OUT sprintf(' %.1f',1/$config{layers}) x $config{layers},
"\n";
    my $flowcount = scalar @{$data{"$config{rivflow}-flow"}};
        print STDERR "Warning: expected flow data from station
$config{rivflow} not found\n" unless $flowcount or defined $flowconst;
    my $wtpcount = scalar @{$data{"$config{rivwtp}-wtp"}};
        print STDERR "Warning: expected wtp data from station
$config{rivwtp} not found\n" unless $wtpcount or defined $wtpconst;
    my $salcount = scalar @{$data{"$config{rivsal}-sal"}};
        print STDERR "Warning: expected sal data from station
$config{rivsal} not found\n" unless $salcount or defined $salconst;
    my $count;
    if ($flowcount) { $count = $flowcount; }
    elsif ($wtpcount) { $count = $wtpcount; }
    elsif ($salcount) { $count = $salcount; }
    print OUT sprintf('%d',$count/5), "\n";
    for (my $i=0; $i<=$count; $i+=5) {
        print OUT sprintf('%.1f',$i/10), "\n"; #hour
         if (defined $flowconst) { print OUT sprintf('%.1f',$flowconst),
"\n"; }
        else { print OUT $data{"$config{rivflow}-flow"}->[$i], "\n"; }
          if (defined $wtpconst) { print OUT sprintf('%.1f',$wtpconst),
"\n"; }
        else { print OUT $data{"$config{rivwtp}-wtp"}->[$i], "\n"; }
          if (defined $salconst) { print OUT sprintf('%.1f',$salconst),
"\n"; }
        else { print OUT $data{"$config{rivsal}-sal"}->[$i], "\n"; }
    }

}
else {
    print OUT "node  specified\n0";
}

close(OUT);
print "Done!\n";
```

Appendix C: Sample Configuration File for 2dm2fvcom (2dm2fvcom.conf)

```
# casename for the run
name = ccbay3

# number of layers
layers = 10

# whether or not to retrieve data from lighthouse (you can set this to
'n' if you have already retrieved data)
getdata = y

# time range for the run
when = 10/30/2002-11/7/2002

# stations to use for water level forcing
wlstns = 014,005

# stations to average for wind forcing
windstns = 014,009

# stations to use for temperature forcing
wtpstns = 149,149

# stations to use for salinity forcing
salstns = 149,149

# interpolate missing values?
interp = y

# extrapolate missing values at ends of time range?
extrap = n

# delay "wave" to be applied to water level inputs (length must equal
sum of nodestring lengths)
wave = "3 3 3 2 2 2 1 1 0 0 0 0 0 0 0 0 1 1 1 2 2 2 3 3 3 0 0 0 0"

# shift bathymetry data by this amount (in meters) so that it is all
negative
# this will also shift input water levels by this amount
# this is useful when you have rivers whose bottoms are above the zero
point of your datum
shiftbath = -2

# datum to retrieve water level in (should match bathymetry data)
datum = mhw

# initial water level for all nodes in grid
initwl = -0.1

# whether or not to reverse sign of bathymetic data
reversebathsign = n

# use temperature and salinity time series nudging at open boundry
```

```
nodes
tsobc = data
#tsobc = constant
#
#wtps = 25,25
#
#sals = 35,35

# river input node
rivnode = 10058

# station to use for river flow
rivflow = 401

# station to use for river water temperature
rivwtp = 076

# station or constant to use for river water salinity
rivsal = constant:0

# use a gradient for initial temperature and salinity
# build gradient using inverse distance weighted average of all points
<tsinitgrad>

    # points defined with lat/lon will be converted to UTM
    <point>
        name = 079:SALT08
        lat = 27.87078
        lon = -97.51770
        t = 16.6
        s = 17.9
         # use for all nodes west of this point (other options: eastof,
northof, southof)
        usefornodes = westof
    </point>

    <point>
        name = 149:MANER5
        lat = 27.83826
        lon = -97.05029
        t = 16.2
        s = 32.0
         # all points within the circle defined by center x,y and radius
r will use these t and s values
        # x,y = easting,northing
        <circle>
            x = 872648.24
            y = 2961467.57
            r = 217300
        </circle>
    </point>

    # can also define points in UTM
    <point>
```

```
        name = somepoint
        zone = 14
        easting = 600000
        northing = 2900000
        t = 15
        s = 30
    </point>

</tsinitgrad>
```

Appendix D: Source Code: bas

```
#!/bin/bash

RUNDIR=`basename \`pwd\``

if [ -z $1 ] || [ -z $2 ]
then
    echo "usage: bas desthost:rundir grid.2dm {rivnode}"
    exit
fi

if [ ! -f $2 ]
then
    echo "error: grid file \"$2\" not found"
    exit
fi

echo
echo "building fvcom input files with $2 and rivnode = $3"
echo

cp $2 .

if [ ! -f 2dm2fvcom.conf ]
then
    echo "error: 2dm2fvcom.conf missing"
    exit
fi

if [ ! -z $3 ]
then
         sed  "s/^rivnode  =  .*$/rivnode  =  $3/"  2dm2fvcom.conf  >
2dm2fvcom.conf.tmp
    mv 2dm2fvcom.conf.tmp 2dm2fvcom.conf
fi

2dm2fvcom $2

rsync -avz -e ssh $2 INPDIR_* "$1/$RUNDIR/"
```

Appendix E: Patch for initial_uvel.F

```
65a66,70
> !-------- INITIALIZE ELF  ----------------------------------------------------!
>
> ELF = EL
> ELF1 = EL1
>
```

Appendix F: Source Code: fvcom2sms

```
#!/usr/bin/perl

if (@ARGV < 1) {
    print "Usage: fvcom2sms 2dmfile {dti} {smsdir}\n";
    exit;
}

my $meshfile = $ARGV[0];

my $dti;

if (@ARGV > 1) {
    $dti = $ARGV[1];
    unless ($dti =~ /^\d+\.?\d*$/) {
        print "Error: non-numeric dti specified\n";
        exit;
    }
}
else {
    my $runfile;
    opendir(PWD,'.');
    my @files = readdir(PWD);
    close(PWD);

    for (@files) {
        if (/_run.dat$/) {
            $runfile = $_;
            last;
        }
    }
    unless (defined $runfile) {
        print "Error: could not find runfile to determine dti\n";
        exit;
    }
    my ($dte,$isplit);
    open(RUNFILE,'<',$runfile);
    while (<RUNFILE>) {
        $dte = $1 if /^\s*DTE\s*=\s*(\d+\.?\d*)/;
        $isplit = $1 if /^\s*ISPLIT\s*=\s*(\d+\.?\d*)/;
    }
    close(RUNFILE);
    unless (defined $dte) {
        print "Error: DTE not found in $runfile\n";
        exit;
    }
    unless (defined $isplit) {
        print "Error: ISPLIT not found in $runfile\n";
        exit;
    }
    $dti = $dte * $isplit;
}
```

```
my $outdir;
my $smsdir;
if (@ARGV > 2) {
    $smsdir = $ARGV[2];
    $outdir = $smsdir;
    $outdir =~ s/sms\/?$//;
}
else {
    opendir(PWD,'.');
    my @files = readdir(PWD);
    close(PWD);

    for (@files) {
        if (/^OUTDIR_/) {
            $outdir = $_;
            last;
        }
    }

    unless (defined $outdir) {
        print "Error: OUTDIR_* not found\n";
        exit;
    }
    $smsdir = "$outdir/sms";
}

$outdir =~ s/\/$//;
$outdir = '.' if $outdir eq '';

if (!-d $smsdir) {
    print "Error: $smsdir is not found or not a directory\n";
    exit;
}

open(MESHFILE,'<',$meshfile)  or  die  "Error:  could  not  open  $meshfile
for reading: $!";

my $numnodes = 0;
my $numcells = 0;
my @nodes;

while (<MESHFILE>) {
    $numnodes++ if /^ND\s+/;
    next unless /^E3T\s+/;
    $numcells++;
    my @line = split;
    push @{$nodes[$line[2]]}, $line[1];
    push @{$nodes[$line[3]]}, $line[1];
    push @{$nodes[$line[4]]}, $line[1];
}

close(MESHFILE);

opendir(SMSDIR,  $smsdir)  or  die  "Error:  could  not  open  sms  directory:
```

```
$!";
my @xyfiles =  readdir(SMSDIR);
close(SMSDIR);

@xyfiles = sort(@xyfiles);

my $IINT;
my %steps;
my $currelevstep, $currtempstep, $currsalstep;
my @sucells, @svcells, @uacells, @vacells, @uwindcells, $vwindcells;

for my $xyfile (@xyfiles) {
    my $type = 'none';
    $type = 'elts' if $xyfile =~ /_elts.xy$/;
    $type = 'vect' if $xyfile =~ /_uvi_uva.xy$/;
    next unless $type ne 'none';
       open(XYF,'<',"$smsdir/$xyfile")  or  die  "Error:  coud  not  open
$smsdir/$xyfile: $!";
    my $count = 1;
    while (<XYF>) {
        if (/^ IINT/) {
            ($IINT) = /^ IINT==\s+(\d+)/;
            if ($type eq 'vect') {
                my $currsstep = '';
                my $currastep = '';
                my $currwindstep = '';
                for (my $n=1;$n<@nodes;$n++) {
                    my $susum = 0;
                    my $svsum = 0;
                    my $uasum = 0;
                    my $vasum = 0;
                    my $uwindsum = 0;
                    my $vwindsum = 0;
                    for my $c (@{$nodes[$n]}) {
                        $susum += $sucells[$c];
                        $svsum += $svcells[$c];
                        $uasum += $uacells[$c];
                        $vasum += $vacells[$c];
                        $uwindsum += $uwindcells[$c];
                        $vwindsum += $vwindcells[$c];
                    }
                                         $currsstep  .=  sprintf('%.8f',
$susum/scalar(@{$nodes[$n]}));
                    $currsstep .= "  ";
                                         $currsstep  .=  sprintf('%.8f',
$svsum/scalar(@{$nodes[$n]}));
                    $currsstep .= "\n";
                                         $currastep  .=  sprintf('%.8f',
$uasum/scalar(@{$nodes[$n]}));
                    $currastep .= "  ";
                                         $currastep  .=  sprintf('%.8f',
$vasum/scalar(@{$nodes[$n]}));
                    $currastep .= "\n";
                                      $currwindstep  .=  sprintf('%.8f',
$uwindsum/scalar(@{$nodes[$n]}));
                    $currwindstep .= "  ";
                                  $currwindstep  .=  sprintf('%.8f',
$vwindsum/scalar(@{$nodes[$n]}));
                    $currwindstep .= "\n";
                }
                           push  @{$steps{'sur-curr'}},  {ts=>$IINT*$dti,
data=>$currsstep};
                           push  @{$steps{'avg-curr'}},  {ts=>$IINT*$dti,
data=>$currastep};
                              push  @{$steps{'wind'}},  {ts=>$IINT*$dti,
data=>$currwindstep};
                next;
            }
                             push   @{$steps{'elev'}},   {ts=>$IINT*$dti,
data=>$currelevstep};
          $currelevstep = '';
                             push   @{$steps{'temp'}},   {ts=>$IINT*$dti,
data=>$currtempstep};
          $currtempstep = '';
            push @{$steps{'sal'}}, {ts=>$IINT*$dti, data=>$currsalstep};
          $currsalstep = '';
            next;
        }
        next unless /^\s+\d/;
        my @line = split;
        if ($type eq 'elts') {
            my ($n,$m) = $line[2] =~ /([\-\d\.]+)E([\+\-\d]+)/;
            $v = $n * 10 ** $m;
            $currelevstep .= " $v\n";
            my ($n,$m) = $line[3] =~ /([\-\d\.]+)E([\+\-\d]+)/;
            $v = $n * 10 ** $m;
            $currtempstep .= " $v\n";
            my ($n,$m) = $line[4] =~ /([\-\d\.]+)E([\+\-\d]+)/;
            $v = $n * 10 ** $m;
            $currsalstep .= " $v\n";
            next;
        }
        my ($n1,$m1) = $line[2] =~ /([\-\d\.]+)E([\+\-\d]+)/;
        my ($n2,$m2) = $line[3] =~ /([\-\d\.]+)E([\+\-\d]+)/;
        $sucells[$count] = $n1 * 10 ** $m1;
        $svcells[$count] = $n2 * 10 ** $m2;
        my ($n1,$m1) = $line[4] =~ /([\-\d\.]+)E([\+\-\d]+)/;
        my ($n2,$m2) = $line[5] =~ /([\-\d\.]+)E([\+\-\d]+)/;
        $uacells[$count] = $n1 * 10 ** $m1;
        $vacells[$count] = $n2 * 10 ** $m2;
        my ($n1,$m1) = $line[6] =~ /([\-\d\.]+)E([\+\-\d]+)/;
        my ($n2,$m2) = $line[7] =~ /([\-\d\.]+)E([\+\-\d]+)/;
        $uwindcells[$count] = $n1 * 10 ** $m1;
        $vwindcells[$count] = $n2 * 10 ** $m2;
        $count++;
    }
}
```

```
my %sers = (elev => 'SCL', temp => 'SCL', sal => 'SCL', 'sur-curr' =>
'VEC', 'avg-curr' => 'VEC', wind => 'VEC');

for my $ser (keys %sers) {
      open(OUT,'>',"$outdir/$ser.dat") or die "Error: could not open
$outdir/$ser.dat for writing: $!";
    print OUT "DATASET
OBJTYPE \"mesh2d\"
BEG$sers{$ser}
ND   $numnodes
NC  $numcells
NAME \"$ser"."_fvcom2sms\"
TIMEUNITS seconds
";
    for my $s (@{$steps{$ser}}) {
        print OUT "TS 0 $s->{ts}\n";
        print OUT $s->{'data'};
    }
    print OUT "ENDDS\n";
    close(OUT);
}

print "Done! Files are: ", join(', ', map { "$_.dat" } keys %sers),
"\n";
```

Appendix G: Source Code: findnode

```
#!/usr/bin/perl

use FindBin;
use Config::General;

use Geo::Coordinates::UTM;

local $| = 1;

unless ($ARGV[0] && -f $ARGV[0]) {
    print STDERR "Error: must specify 2dm file\n";
    exit;
}

my $ellipsoid = 'WGS-84';

my ($latitude,$longitude) = (27.8143000, -97.3984000);

my   ($zone,$easting,$northing)   =   latlon_to_utm($ellipsoid,$latitude,
$longitude);

print "$zone $easting $northing\n";

my $d_min = undef;
my $closest_node = undef;

while (<>) {
    if (/^ND\s/) {
        my ($node,$e,$n) = (split)[1..3];
        my $d = sqrt( ($easting - $e)**2 + ($northing - $n)**2 );
        if (!defined($d_min) or $d < $d_min) {
            $d_min = $d;
            $closest_node = $node;
        }
    }
}

print "$closest_node\n";
```

Appendix H: Source Code: extractnodes

```
#!/usr/bin/perl

use FindBin;
use Config::General;
use Geo::Coordinates::UTM;
use Getopt::Std;
use POSIX;

getopt('msd');

local $| = 1;

my $ellipsoid = 'WGS-84';

unless ( -f "extractnodes.conf" ) {
     print STDERR "Error: missing config file!\nPlease place a config
file named extractnodes.conf in the current directory.\n";
      print STDERR "See: $FindBin::Bin/extractnodes.conf.sample for a
sample config file\n";
    exit;
}

$conf = new Config::General("extractnodes.conf");
my %config = $conf->getall;

unless ($ARGV[0] && (-f $ARGV[0] || -d $ARGV[0])) {
    print STDERR "Error: must specify dat file or sms directory\n";
    exit;
}

my $ser;
my $dti;

if (-f $ARGV[0]) {
    $ser = $ARGV[0];
    $ser =~ s/\.dat$//;
    $ser =~ s/^.*\///;
}
elsif (-d $ARGV[0]) {
    unless ($opt_s) {
             print STDERR "Error: must specify series with -s when
extracting from an sms directory\n";
        exit;
    }
    $ser = $opt_s;

    if ($opt_d) {
        $dti = $opt_d;
        unless ($dti =~ /^\d+\.?\d*$/) {
            print "Error: non-numeric dti specified\n";
            exit;
        }
    }
    else {
        my $runfile;
        opendir(PWD,'.');
        my @files = readdir(PWD);
        close(PWD);

        for (@files) {
            if (/_run.dat$/) {
                $runfile = $_;
                last;
            }
        }
        unless (defined $runfile) {
            print "Error: could not find runfile to determine dti\n";
            exit;
        }
        my ($dte,$isplit);
        open(RUNFILE,'<',$runfile);
        while (<RUNFILE>) {
            $dte = $1 if /^\s*DTE\s*=\s*(\d+\.?\d*)/;
            $isplit = $1 if /^\s*ISPLIT\s*=\s*(\d+\.?\d*)/;
        }
        close(RUNFILE);
        unless (defined $dte) {
            print "Error: DTE not found in $runfile\n";
            exit;
        }
        unless (defined $isplit) {
            print "Error: ISPLIT not found in $runfile\n";
            exit;
        }
        $dti = $dte * $isplit;
    }

}

my @nodeconf;
if (ref $config{node} eq "ARRAY") {
    @nodeconf = @{$config{node}};
}
elsif (ref $config{node} eq "HASH") {
    push @nodeconf, $config{node};
}

die "Error: no nodes found in conf file\n" unless @nodeconf;

my @extract_nodes;

for my $node (@nodeconf) {
    if (exists $node->{extract}) {
        my $foundser = 0;
        my @extract = split /[, ]+/, $node->{extract};
        for my $extract_ser (@extract) {
                $foundser = 1 if $extract_ser eq $ser or $extract_ser eq
'*';
            }
            next unless $foundser;
        }
        push @extract_nodes, $node;
}

die "Error: no nodes found in conf file for $ser\n" unless
@extract_nodes;

my @find_nodes;

for my $node (@extract_nodes) {
     if (!exists $node->{id} and exists $node->{lat} and exists $node-
>{lon}) {
                    ($node->{zone},$node->{easting},$node->{northing}) =
latlon_to_utm($ellipsoid,$node->{lat},$node->{lon});
            push @find_nodes, $node;
        }
}

if (@find_nodes) {

        unless ($opt_m && -f $opt_m) {
            print STDERR "Error: must specify 2dm file with -m to find
nodes specified with lat/lon\n";
            exit;
        }

     open(GRIDFILE,'<',$opt_m) or die "Error: could not open $opt_m for
reading: $!";

        while (<GRIDFILE>) {
            if (/^ND\s/) {
                my ($id,$easting,$northing,$depth) = (split)[1..4];
                next if $depth < 0; # skip nodes above 0 mhw
                for my $node (@find_nodes) {
                        my $d = sqrt( ($node->{easting} - $easting)**2 +
($node->{northing} - $northing)**2 );
                    if (!exists($node->{d_min}) or $d < $node->{d_min}) {
                        $node->{d_min} = $d;
                        $node->{id} = $id;
                    }
                }
            }
        }

        close(GRIDFILE);

        for my $node (@find_nodes) {
                print "Nearest node found for $node->{label}: $node->{id}
(distance: " . sprintf('%.2f',$node->{d_min}) . " m)\n";
        }
```

```
}

my %nodes;
my @nodes;

for my $node (@extract_nodes) {
     die "Error: node $node->{label} is missing id or lat/lon\n" unless
exists $node->{id};
    die "Error: invalid node id in config file: '$node->{id}'\n" unless
$node->{id} =~ /^\d+$/;
    push @nodes, $node->{id};
    $nodes{$node->{id}}++;
}

my $outfile = "extractednodes-$ser.csv";

print "Extracting nodes (" . join(', ',@nodes) . ") from $ARGV[0] to
$outfile...";

open(OUT, '>', $outfile) or die "could not open $outfile: $!\n";

print OUT "\"date+time\"";
for my $node (@extract_nodes) {
    if (exists $node->{label}) {
        print OUT ",\"$node->{label} ($ser)\"";
    }
    else {
        print OUT ",\"node $node->{id} ($ser)\"";
    }
}
print OUT "\n";

if (-f $ARGV[0]) {

     open(DATFILE,'<',$ARGV[0]) or die "Error: could not open $ARGV[0]
for reading: $!";

    my $currnode = 0;
    my $ts;
    my %vals;
    my @row;

    while (<DATFILE>) {
        if (/^TS \d+ (\d+)/ or /^ENDDS$/) {
            $ts = $1;
            $currnode = 0;
            for my $node (@extract_nodes) {
                   push @row, $vals{$node->{id}} if exists $vals{$node-
>{id}};
            }
            print OUT join(',',@row) . "\n" if @row;
            @row = ();
            push @row, '"' . ts2daytime($ts) . '"';
        }
        next if /^[A-Z]/;
        $currnode++;
        if ($nodes{$currnode}) {
            my $val = $_;
            $val =~ s/\s//g;
            $vals{$currnode} = $val;
        }
    }

    close(DATFILE);

}
elsif (-d $ARGV[0]) {

    # array of file name part and column numbers (add 1 because x and y
are columns 0 and 1)
    %ser_to_file_map = ( elev => ['elts',1], temp => ['elts',2], sal =>
['elts',3] );
    # TODO
    # Currently only supports extracting scalar series from elts. Need
to add vector support.
    # file map could look something like:
    # %ser_to_file_map = ( elev => ['elts',1], temp => ['elts',2], sal
=> ['elts',3], wind => ['uvi_uva',5,6], 'surr-curr' => ['uvi_uva',1,2],
'avg-curr' => ['uvi_uva',3,4] );

    opendir(SMSDIR, $ARGV[0]) || die "Error: could not read $ARGV[0]:
$!";
    my @files = readdir(SMSDIR);
    close(SMSDIR);

    my $numfiles = @files / 2;

    print "\n|--------25--------50--------75--------|\n";

    my $count = 0;
    my $progcount = 0;

    for my $file (sort(@files)) {
        next unless $file =~ /$ser_to_file_map{$ser}->[0]\.xy$/;

        my $ts;
        my %vals;
        my @row;

        open(XYFILE,'<',"$ARGV[0]/$file");

        my $currnode = 1;
        while (<XYFILE>) {
            next if /^(scat2d|xyd)/;

            if (/^\s*IINT==\s*(\d+)/) {
                $ts = $1 * $dti;
                last;
            }

            if ($nodes{$currnode}) {
                my @line = split;
                  my ($n,$m) = $line[$ser_to_file_map{$ser}->[1]+1] =~ /
([\-\d\.]+)E([\+\-\d]+)/;
                $v = $n * 10 ** $m;
                $vals{$currnode} = $v
            }

            $currnode++;
        }
        close(XYFILE);

        push @row, '"' . ts2daytime($ts) . '"';
        for my $node (@extract_nodes) {
            push @row, $vals{$node->{id}} if exists $vals{$node->{id}};
        }
        print OUT join(',',@row) . "\n" if @row;

        if ($count % ceil($numfiles/40) == 0) {
            print '#';
            $progcount++;
        }
        $count++;
    }
    for (my $i=$progcount; $i < 40; $i++) {
        print '#';
    }
    print "\n";

}

close(OUT);
print "Done!\n";

sub ts2daytime {
    my $ts = shift;
    my $day = floor($ts / 86400);
    my $hour = floor(($ts - $day * 86400) / 3600);
    my $min = floor(($ts - $day * 86400 - $hour * 3600) / 60);
    $day+= $config{firstday};
    return sprintf('%d%03d+%02d%02d',$config{year},$day,$hour,$min);
}
```

Appendix I: Sample Configuration File for extractnodes (extractnodes.conf)

```
# id: node id in SMS
# label: label to use in output csv (omit to use node id)

<node>
    id = 1
    label = one
</node>

<node>
    id = 2
</node>

<node>
    id = 3
    label = three
</node>

<node>
    lat = 27.8143000
    lon = -97.3984000
    label = AQUARI
</node>

year = 2008

firstday = 1
```