2020

A Neural Network Approach to Estimate Buoy Mooring Line Sensor Deflection

Tom Price

A NEURAL NETWORK APPROACH TO ESTIMATE BUOY MOORING LINE SENSOR DEFLECTION

By

Thomas Price

A THESIS

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

In Mechanical Engineering

MICHIGAN TECHNOLOGICAL UNIVERSITY

2020

This thesis has been approved in partial fulfillment of the requirements for the Degree of MASTER OF SCIENCE in Mechanical Engineering.

Department of Mechanical Engineering - Engineering Mechanics

Thesis Advisor:	*Dr. Gordon Parker*
Committee Member:	*Dr. Guy Meadows*
Committee Member:	*Dr. Andrew Barnard*
Department Chair:	*Dr. William Predebon*

Contents

List of Figures . . . . **vii**

List of Tables . . . . **ix**

Abstract . . . . **xi**

1 Introduction **1**

1.1 Neural Networks 6

1.1.1 An Introduction to Neural Networks 6

1.1.2 Applications of Neural Networks 9

1.2 ProteusDS 10

1.2.1 Overview of ProteusDS 11

1.2.2 ProteusDS Simulations 12

2 Approach **17**

2.1 Creating a Model of the MSWB 18

2.2 Iterative Runs in MATLAB 21

3 Results and Discussion **27**

3.1 Buoy Behavior 27

3.2 Neural Network Configuration Results 30

3.3 Conclusions 33

References 35

List of Figures

1.1 TIDAS 900 Buoy 2

1.2 Location of MSWB 3

1.3 Illustration of Sensor Vertical Displacement 4

1.4 Neural Network Architecture 7

1.5 Inside the node of a neural network 9

1.6 Free body diagram for top node (left) and all other nodes (right) . . 13

1.7 Visual Representation of α 15

1.8 Calculation of Positions of Nodes 15

2.1 Flowchart of Deflection Estimation Method 17

2.2 MSWB Component Diagram 19

2.3 Vertical Deflection of Thermistors from Shear Current 21

2.4 Current Profiles Recorded May 20 - 25, 2019 23

2.5 Neural Network Input/Output Configurations 25

3.1 XY Position of Buoy 28

3.2 Vertical Deflection of Thermistors vs. Buoy Distance from Anchor . 29

3.3 Neural Network Test Dataset Estimation of Deflection for 3 Input Configurations 31
3.4 Neural Network Vertical Deflection Error 32

List of Tables

2.1 Properties of Rope Materials Used in MSWB Model 18

3.1 Neural Network Errors and Correlation 32

Abstract

Instrumented moorings are often used to measure characteristics, such as temperature and current, over the water column. However, the moorings deflect from the effects of currents and waves, which could lead to innacurate measurements. In this work, a computationally efficient method to compensate for mooring sensor position errors is developed. The two-step process first uses a hydrodynamic model of the buoy and mooring line system to create estimated mooring line deflections in a steady current. A neural network model is trained to approximate the hydrodynamic model's mooring line displacement given the spatial location of the buoy and current profile measurements. The method is illustrated using the Mackinac Straits West buoy that is part of the Upper Great Lakes Observing System (UGLOS). Its mooring line is instrumented with 10 thermistors, attached to the mooring line at varying intervals. Since the approach naturally provides interpolation, it allows researchers, with access to publicly available UGLOS data, to request temperatures at any depth. While the vertical deflection compensation method is illustrated here is for a particular mooring system, the process involved is applicable to a wide class of instrumented mooring systems. It was found that access to the current data of a mooring line increases the accuracy of the Neural Network, but knowing the position of the buoy in relation to the anchor can still give adequate results.

Chapter 1

Introduction

The Upper – Great Lakes Observing System is a cooperative agreement between private businesses, universities, and First Nations to observe the behaviors of the Great Lakes of Michigan [1]. It is a growing network of buoys located throughout Lake Superior, Lake Michigan, and Lake Ontario. These buoys measure several characteristics of the lake environment, such as wind speed, direction and gust, air temperature, humidity, wave height and period. The data is recorded every 10 minutes while the buoys are active and uploaded to an online database for public access. The buoy used in this thesis is the TIDAS 900, shown in Figure 1.1 [1]. The TIDAS 900 has a diameter of 1.12 meters, a draft of 1.97 meters, and a total weight of 97.52 kilograms [2].

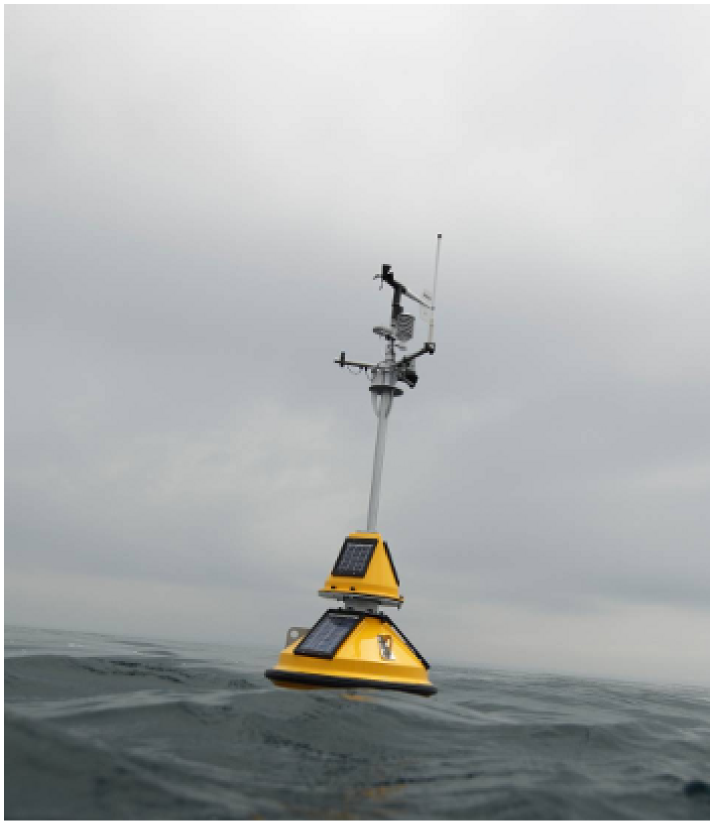


Figure 1.1: TIDAS 900 Buoy

The purpose of these buoys is to monitor the significant changes in the behavior of the lake near the coast. The Great Lakes are a drinking source, therefore accurate temperature measurements are required consistently.

The buoy selected for this study is Station 45175, the Mackinac Straits West Buoy (MSWB). It is located in the straits between Lake Huron and Lake Michigan, at coordinates 45 49.5156 N, 84 46.3302 W (Degrees, Decimal Minutes) [1]. Figure 1.2 shows the location of the buoy. This buoy is of particular interest because of the location. The current profile patterns in this region are unique; the lake will flow in the direction of Lake Michigan, and then reverse and flow in the direction of Lake Huron. The current direction changes every 1.5 days on average [3]. Also, the buoy is located next to an aging oil pipeline. If the pipeline leaks, the buoy data would be critical in containing the spill.

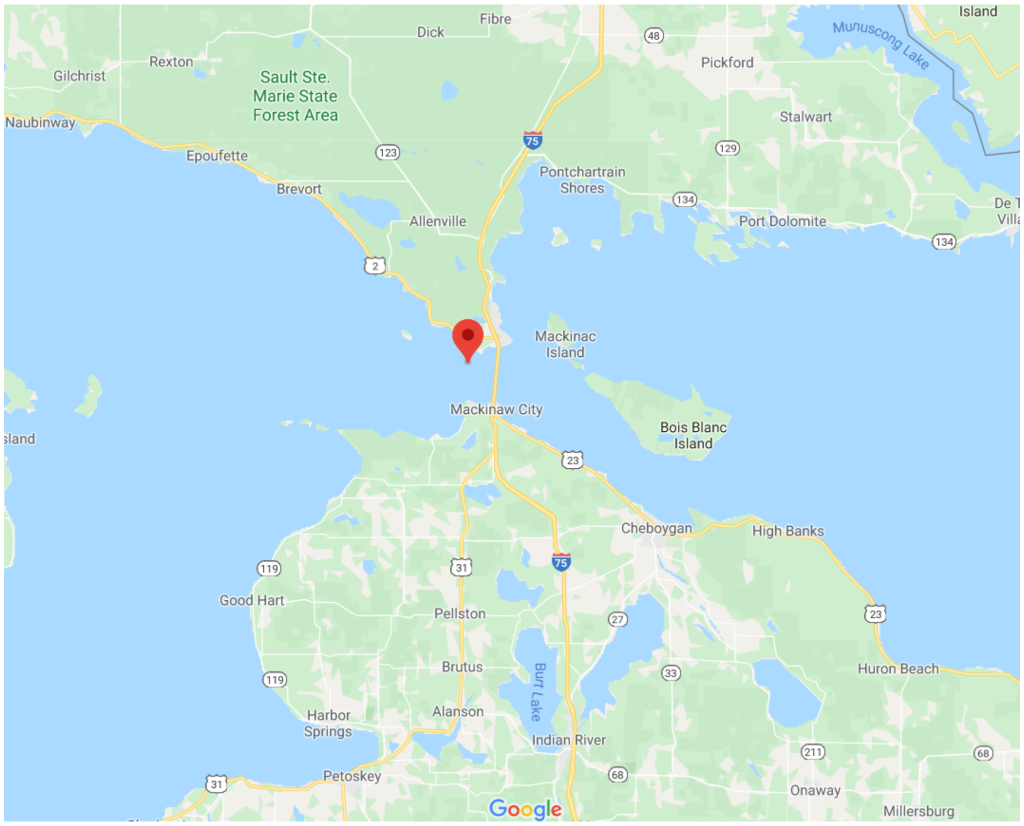


Figure 1.2: Location of MSWB

Since the lake exhibits a significant amount of activity with strong currents and large waves, there is concern that the currents acting on the MSWB cause it to drift, resulting in vertical deflection of the sensors attached to the mooring line, as shown in Figure 1.3. The dashed line represents the mooring line in its ideal position, and the solid line represents the mooring line under deflection due to the current. A weaker current will cause the buoy to drift away from the anchor but will not cause significant increase in the angle of the mooring line. This results in insignificant vertical deflection of the sensors, and the assigned values for depth will be adequate. However, a stronger current will cause the buoy to drift farther from the anchor. The force applied on the buoy is stronger than the force applied by the weight attached to keep the mooring line vertical, resulting in a vertical deflection in all components between the buoy and the anchor.

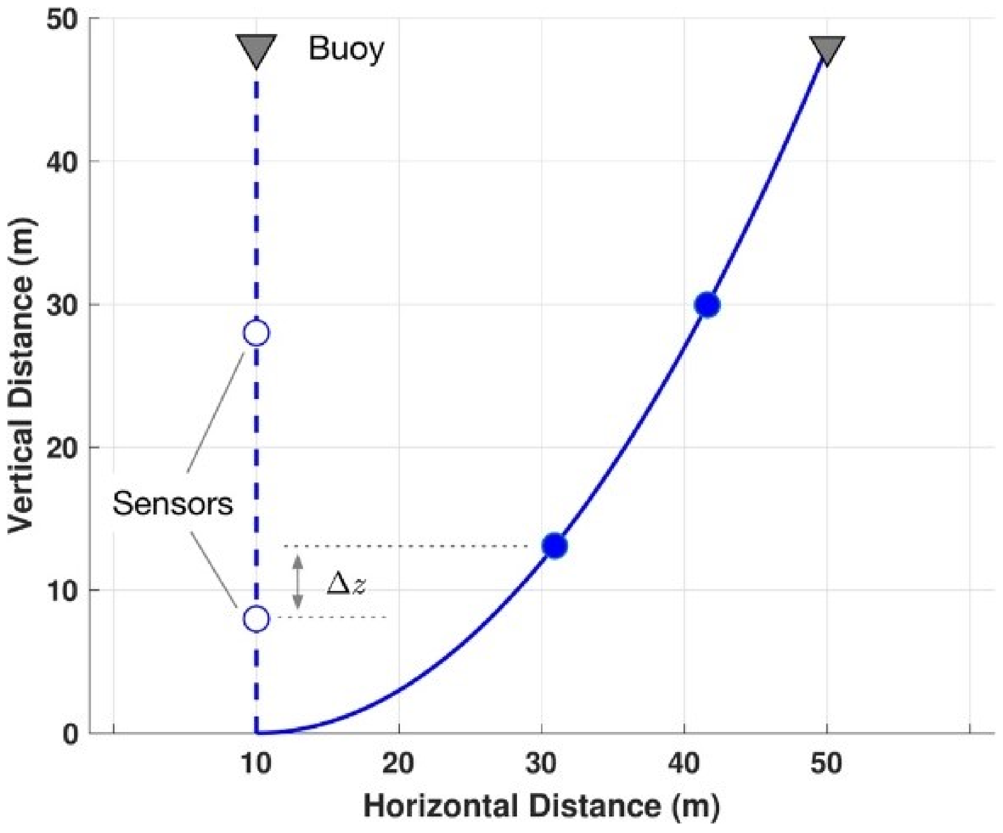


Figure 1.3: Illustration of Sensor Vertical Displacement

The buoy also detects the current in the North and East directions, using an Aquadopp Z-Cell 600 kHz Current Profiler attached to its bottom [4]. The Aquadopp emitts sound waves, and uses the Doppler effect to monitor how the sound reacts to the current profile [5]. This allows the sensor to detect the magnitude and direction of the current profile at user-defined depths. Unlike the temperature, recording the current profile does not depend on the vertical position of any sensors. This means the current data is accurate regardless of the horizontal displacement of the buoy.

The purpose of this project is to determine an efficient way to calculate the vertical deflection of the thermistors, as there is currently no way to account for this discrepency. For the scope of this project, only the steady state position of the mooring line is analyzed, not considering the dynamics of the system. Also, no wave profiles were implemented.

ProteusDS, developed by DSA, was used to estimate the vertical deflection of the mooring line. ProteusDS performs dynamic analysis of marine technologies, including mooring lines [6]. The software facilitates the creation of a finite element model, which was used to create an approximate model of the MSWB. The environment includes specifications of the current profile.

The process for computing the steady state position of the mooring line can be time consuming. Several methods have been developed; one such method involves calculation of the catenary in two dimensions [7]. However, the current behavior often exhibits three dimensional currents. Methods that calculate the global position are preferred. Additionally, many calculations of the quasistatic position of a mooring line are iterative [7]. ProteusDS uses an iterative method. In testing, one successful run of ProteusDS took approximately 50 seconds to compute the steady state position of the buoy model. This is useful for long term applications, but the real-time applications are limited.

1.1 Neural Networks

1.1.1 An Introduction to Neural Networks

A neural network is a method of mapping input data to output data without considering the physical relationship. The general setup of a Feed Forward Neural Network is shown in Figure 1.4. A neural network consists of a series of layers; the input layer, hidden layers, and the output layer. The input layer consists of inputs u_1 through u_j [8]. The inputs are mapped, or fed forward, to nodes inside the network, in a series of hidden layers. Each hidden layer consists of a series of nodes, which are connected in a way reminiscent of an animal neuron [9]. Each hidden layer performs a transformation of the data into the next layer. The number of hidden layers is determined by the user; more hidden layers will result in more calculations being performed in each iteration. After the hidden layers, the output x_o is generated in the output layer.

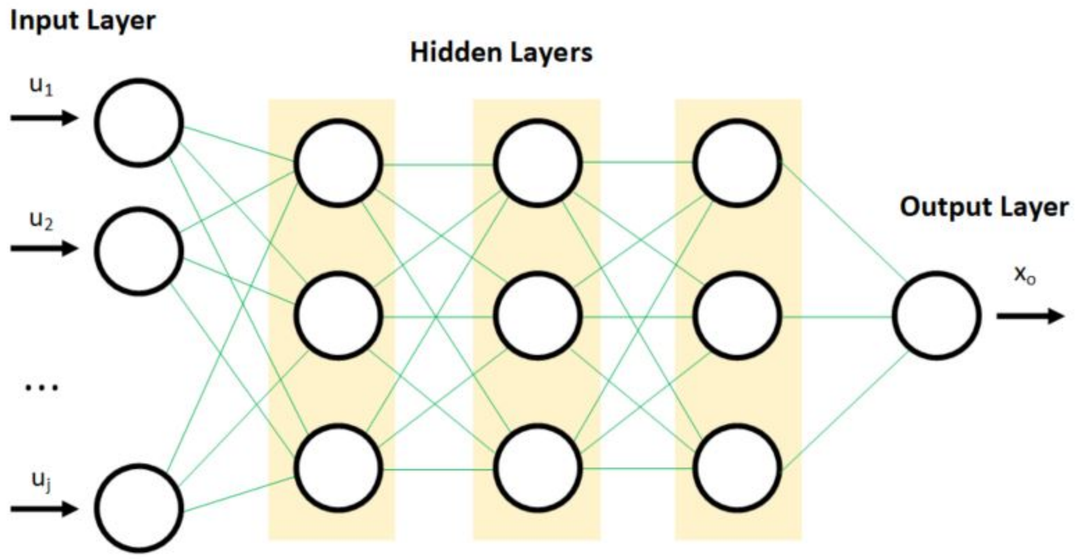


Figure 1.4: Neural Network Architecture

A neural network is trained by assigning a series of weights to the internal layers, and optimizing them using an objective function. The objective function E can be defined by Equation 1.1 [8]. The difference between the user defined outputs, $\hat{x}_o$, and the calculated outputs of the neural network, x_o is squared to ensure it is always positive.

$$E = \sum_o \frac{1}{2}(x_o - \hat{x}_o)^2 \tag{1.1}$$

The value of x_i, the output of neuron i, is determined by Equation 1.2 [8].

$$x_i = f(\xi_i) \tag{1.2}$$

The value of neuron i, ξ_i, is defined by Equation 1.3 [8]. It is a function of the weights of the other neurons and its own threshold. The weights between the previous node and current node is denoted ω_{ij}. The weighting of the node is represented by ϑ_i.

$$\xi_i = \vartheta_i + \sum_{j=1}^{N} \omega_{ij} x_j \tag{1.3}$$

Once ξ_i is calculated, the transfer function $f(\xi)$ can be calculated as shown in Equation 1.4 [8]. Each of these equations is implemented in each node, as shown in Figure 1.5.

$$f(\xi) = \frac{1}{1 + e^{-\xi}} \tag{1.4}$$

The series of transfer functions within the nodes generates an output x_o. The objective function E is optimized to minimize the difference between the outputs by iterating the variables ϑ_i and ω_{ij}.

When creating the input and output data for the neural network, its best to cover the full range of the possible output values. A neural network can effectively interpolate

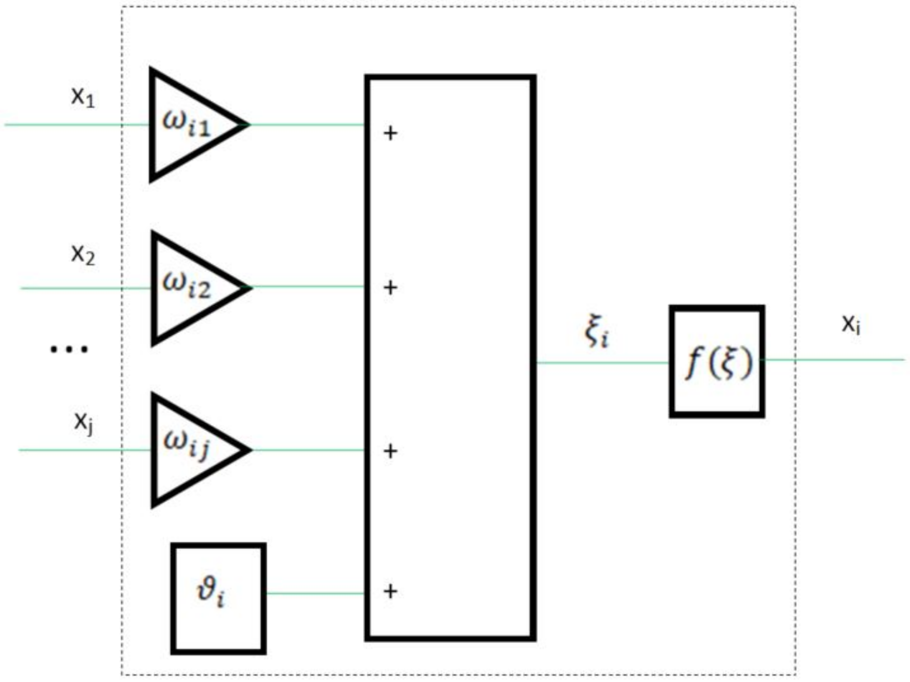


Figure 1.5: Inside the node of a neural network

between datapoints, but the accuracy decreases significantly when extrapolating.

1.1.2 Applications of Neural Networks

Neural networks have been used in a variety of oceanic or hydrodynamic applications. Many of the applications of neural networks is reconstructing missing data that is normally measured. In 2005, Agrawal and Deo utilized neural networks to estimate wave parameters off the coast of India [10]. For example, relationships between significant wave height and average zero-cross wave period, as well as between significant wave height and spectral wave period, were studied. The neural network proved to have a stronger correlation than an exponential model [10]. At that point in time, the applications of neural networks in ocean analysis were not fully realized. Several other applications for neural networks have been discovered since. One example is calculating the missing wave height out of a network of six buoys, using wave data from the other five buoys [11]. Londhe created six networks, one for each buoy using the other buoys as the input data, effectively being able to fill in any gaps in the data. Bhattacharya, Shrestha and Solomatine used wind data to reconstruct missing wave data in the North Sea off the Dutch coastline [12].

1.2 ProteusDS

As previously mentioned, ProteusDS was used to model the MSWB and calculate its position. ProteusDS is optimized to test prototypes in a variety of marine-related applications [6]. It performs its analysis in the time domain, which can be useful for consistently changing environments. The coordinate system used by ProteusDS is standard to marine applications. The positive X axis represents $0°$ North, the positive Y axis represents $90°$ East, and the positive Z axis represents down towards the seabed.

1.2.1 Overview of ProteusDS

ProteusDS uses a series of objects, called DOjects, connected with the user's choice of connection types. These include the base of cables, point masses, rigid bodies, nets, etc. The cable was the focus of this project, since all elements were connected along one line. A cable's 2 ends in ProteusDS are denoted node 0 and node N. Every other node in a cable is numbered 1 through $N-1$. The number of nodes in a cable are determined by the user. The distance between each node represents a straight section of the cable. A cable can have several different materials present throughout its length, with a node acting as an attachment point between the materials. A material must have the following parameters defined [13]:

† Bending Rigidity, EI (N m^2)

† Torsional Rigidity, GJ (N m^2)

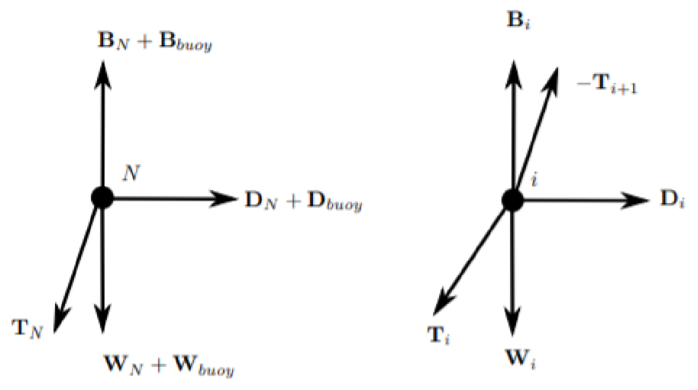
† Axial Rigidity, EA (N)

ProteusDS can create external masses in the form of spheres (ExtMass) and cylinders (ExtMassCylinder). External masses can be added along the line as well. The vertical position along the node is specified, and the center of the mass is rigidly attached to the specified vertical coordinate position.

1.2.2 ProteusDS Simulations

ProteusDS can calculate the dynamics of a surface mooring. However, in this case, only the steady state position of the buoy is desired. ProteusDS has a convenient method of calculating a steady state position given an environment and a buoy; the QuasiStaticCable (QSC). The only dynamics used with the QSC are is fluid acceleration loading [6]. In order to use a QSC, there are a few requirements. The first requirement is compressive elasticity. The second requirement is tension throughout the system. If the current profile is strong enough to pull the buoy away from the anchor point, that will cause the entire system to be in tension, which will allow for a result to be calculated. The third requirement is that it cannot be connected to other DObjects. The model was set up as a single cable from the anchor to the buoy, so this is not an issue.

The method of calculating the final position is adapted from a process created by Dewey [14]. The method is adapted from a MATLAB package that calculates the steady state position of a mooring system using an iterative method. First, the tensions of the system are calculated with the mooring line situated vertically in the body of water. The tension T for a given node is calculated from the buoyancy B, weight W, and drag D, based on the free body diagram of Figure 1.6. On the left is the topmost node, which is treated as the end node N. On the right is every other node in the system.



$\mathbf{B}_N + \mathbf{B}_{buoy}$
N
$\mathbf{D}_N + \mathbf{D}_{buoy}$
$\mathbf{T}_N$
$\mathbf{W}_N + \mathbf{W}_{buoy}$
$\mathbf{B}_i$
$-\mathbf{T}_{i+1}$
i
$\mathbf{D}_i$
$\mathbf{T}_i$
$\mathbf{W}_i$

Figure 1.6: Free body diagram for top node (left) and all other nodes (right)

The left section of Figure 1.6 shows that the tension of node N, T_N, is calculated using Equation 1.5. The total tension applied to the node is a sum of the buoyancy, drag, and weight of both the node itself and the buoy.

$$\vec{T}_N = \vec{B}_N + \vec{B}_{buoy} + \vec{D}_N + \vec{D}_{buoy} + \vec{W}_N + \vec{W}_{buoy} \tag{1.5}$$

The right section of Figure 1.6 shows the tension of all other nodes i, from 1 to N–1. It is calculated in a similar way, except for the tension of the next point must be implemented. The equation for the other nodes is shown in Equation 1.6.

$$\vec{T}_i = \vec{B}_i + \vec{D}_i + \vec{W}_i \tag{1.6}$$

This method for calculating the mooring line position can take as few as three iterations for a simple subsurface mooring line. However, more complex currents will increase the number of iterations required. Additionally, a surface mooring will require more iterations than a subsurface mooring.

The hydrodynamic drag per unit length, h, is calculated at each node as a function of the relative velocity at each node, as shown in Equation 1.7 [13]. The reference density ρ is defined in the environment profile. The normal drag coefficient C_{Dc}, tangential coefficient C_{Dt}, and fluid diameter d_f are defined on the cable itself. The rotation matrix R_{IH}^i transforms the load from the local reference frame to the global frame. The hydrodynamic load coefficients f_p and f_q are calculated in Equations 1.8 and 1.9 respectively, as a function of α.

$$h = -\frac{1}{2}\rho d_f R_{IH}^i \begin{bmatrix} C_{Dc} f_p v_x \sqrt{v_x^2 + v_y^2} \\ C_{Dc} f_p v_y \sqrt{v_x^2 + v_y^2} \\ C_{Dt} f_q v_z \|v_z\| \end{bmatrix} \tag{1.7}$$

$$f_p = 0.5 - 0.1\cos\alpha + 0.1\sin\alpha - 0.4\cos 2\alpha - 0.11\sin 2\alpha \tag{1.8}$$

$$f_q = 1.004 - 0.1929\alpha - 0.9579\alpha^2 - 2.0807\alpha^3 + 1.7532\alpha^4 - 0.6937\alpha^5 \tag{1.9}$$

The angle between the normal and tangential velocity components, α, is defined by Equation 1.10. For this study, the current velocity in the Z direction is negligible, causing f_p and f_q to have values of 1 and 0 respectively.

$$\alpha = \arcsin \frac{\sqrt{v_x^2 + v_y^2}}{\sqrt{v_x^2 + v_y^2 + v_z^2}} \tag{1.10}$$

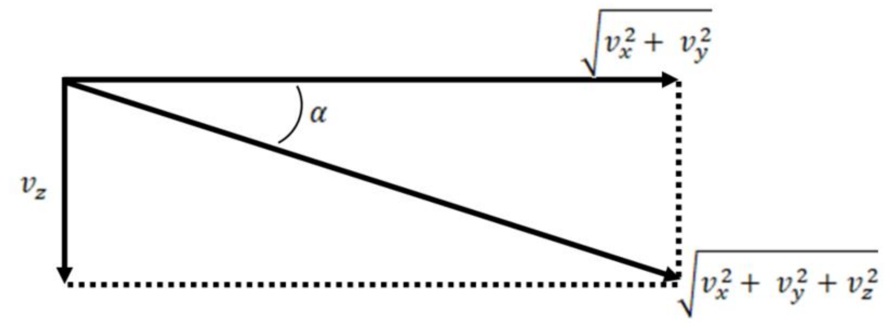


Figure 1.7: Visual Representation of α

Once all the tensions in the cable are calculated, the next step is to calculate the position of each node. The first node calculated is the node adjacent to the anchor, and the rest of the node positions are calculated sequentially through node N.

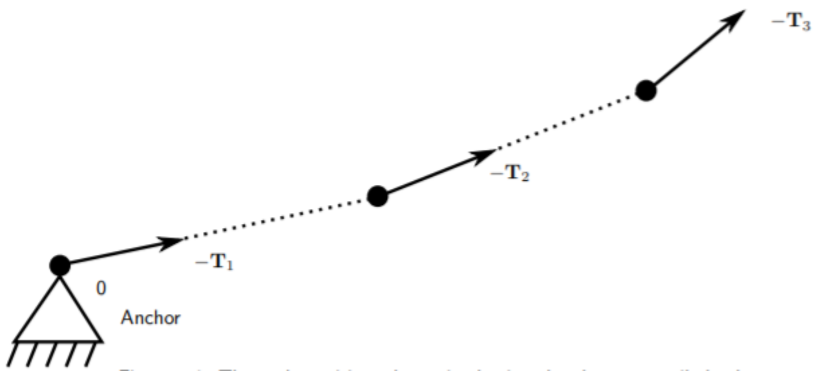


Figure 1.8: Calculation of Positions of Nodes

Any elongation of the element i, denoted dL_i, is calculated here using Equation 1.11, where L_i is the unstretched length, and EA_i is the axial rigidity.

$$dL_i \;=\; \frac{\|T_i\| L_i}{EA_i} \tag{1.11}$$

Once this elongation is calculated, the position of the current node is calculated using the position of the previous node and the total distance between the nodes in the direction of the tension vector. The position of node i, $\vec{p}_i$, is shown in Equation 1.12.

$$\vec{p}_i \;=\; \vec{p}_{i-1} - (L_i + dL_i)\frac{\|T_i\|}{T_i} \tag{1.12}$$

Chapter 2

Approach

Figure 2.1 shows the process of estimating vertical deflection of the MSWB using neural networks. This process can be translated to any buoy configuration.

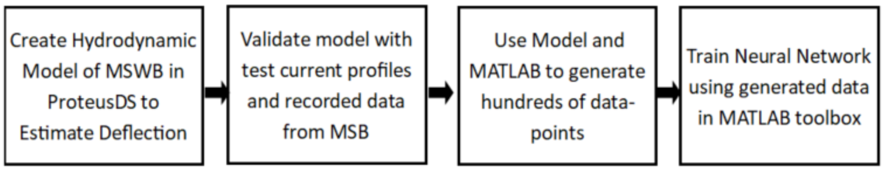

Create Hydrodynamic Model of MSWB in ProteusDS to Estimate Deflection
Validate model with test current profiles and recorded data from MSB
Use Model and MATLAB to generate hundreds of data-points
Train Neural Network using generated data in MATLAB toolbox

Figure 2.1: Flowchart of Deflection Estimation Method

2.1 Creating a Model of the MSWB

The configuration of the MSWB model is shown in Figure 2.2. The dynamically significant components of the MSWB are a buoy, buoy weight, wire rope, thermistor anchor, nylon mooring line, mooring chain, and mooring anchor. These are arranged as shown in Figure 2.2. There are some changes between the components of the MSWB and the model in ProteusDS; instead of a buoy matching the shape of the TIDAS 900, a spherical shape was used. The end of the mooring line was treated as connected to the lakebed instead of a large weight acting as an anchor.

The materials for the mooring line were imported from the ProteusDS library, and their attributes of the different rope materials, shown in Table 2.1, are Normal Drag Coefficient C_{Dc}, Tangential Drag Coefficient C_{Dt}, Axial Rigidity EA, Torsional Rigidity GJ, Density ρ, and Diameter D.

Table 2.1
Properties of Rope Materials Used in MSWB Model

Material	C_{Dc}	C_{Dt}	EA **(N)**	GJ **(N m^2)**	ρ **(kg/m^3)**	D **(m)**
Nylon	1.5	0.045	1.39E5	0	830	0.013
Chain	2.2	1.2	1.65E7	0	7800	0.025
Wire	1.5	0.045	8.87E6	89.4	5400	0.013

The buoy was modeled with an ExtMass. The diameter was set to 1.12 m. The weight in water was set to -408 kg to indicate buoyancy. It is located at the end

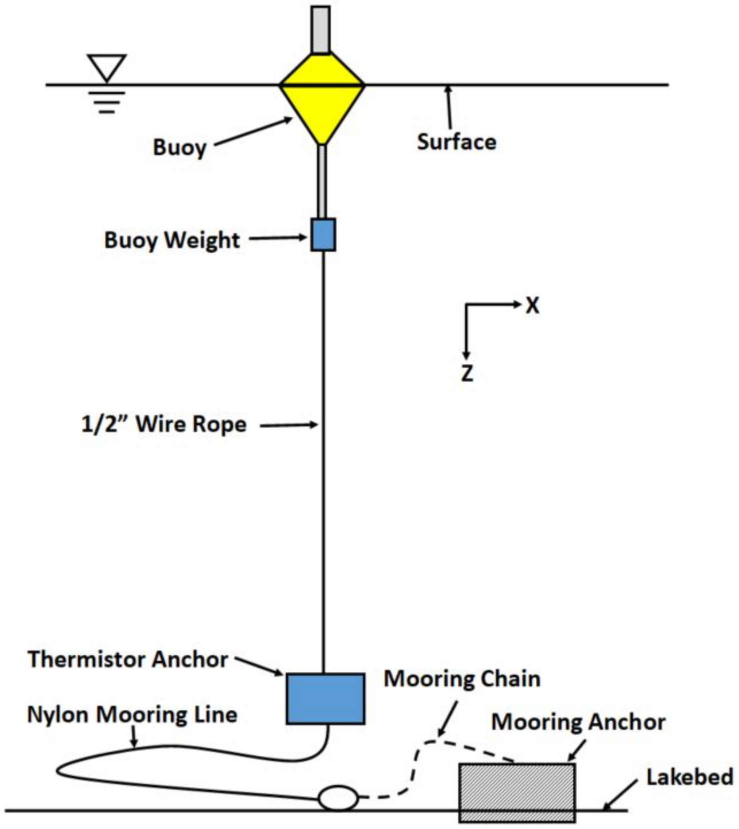


Figure 2.2: MSWB Component Diagram

of the wire rope. The thermistor anchor and buoy weight were both modeled as ExtMassCylinders. The thermistor anchor is made of concrete, which has a density of 2400 kg/m^3 abd a diameter of 0.302 m and a height of 0.35 m. The buoy weight was set to 1 kg in water. Along the line are 10 thermistors, which are small and do not impact the hydrodynamic response of the buoy. They are modeled with ExtMassCylinders. These cylinders add negligible mass and volume to the system; they just exist to record the Cartesian coordinates of each element.

Attributes of the lake environment were implemented in the ProteusDS Environment file. The environment was set with a depth of 30.48 meters and no wave profiles. ProteusDS can implement wave profiles and wind in simulations, but for this project, these were set to off. The properties of the air and water were set to default values. Water density was set to 1025 kilograms per cubic meter, and the kinematic viscosity was set to 1.8E-6 meters squared per second. Air density was set to 1.29 kilograms per cubic meter, and kinematic viscosity was set to 1.57E-5 meters squared per second. These were kept constant for all simulations.

Once all the components were created, the model was set to the setting of QuasiStaticCable (QSC). The QSC is used to detect the steady state of the mooring line with a given environment, in this case the given current profile. For the scope of this project, only the steady state positions are necessary, and the dynamic response is not needed. The components of the cable were changed to match the characteristics of the mooring line for the given length (wire and nylon rope, chain).

The model was checked using shear current profiles, which have an implemented maximum current magnitude at the surface. The magnitude decreases as the depth approaches the seabed, with a magnitude of 0 at the seabed. To test the general model behavior, shear currents with magnitude ranging from 0.1 to 1.6 meters per second were implemented, and the vertical deflection was calculated. Figure 2.3 shows the vertical deflection of each thermistor as a function of the shear current magnitude. For a uniform current, the lowest thermistor always has the largest deflection, and the deflection increases as the current magnitude increases.

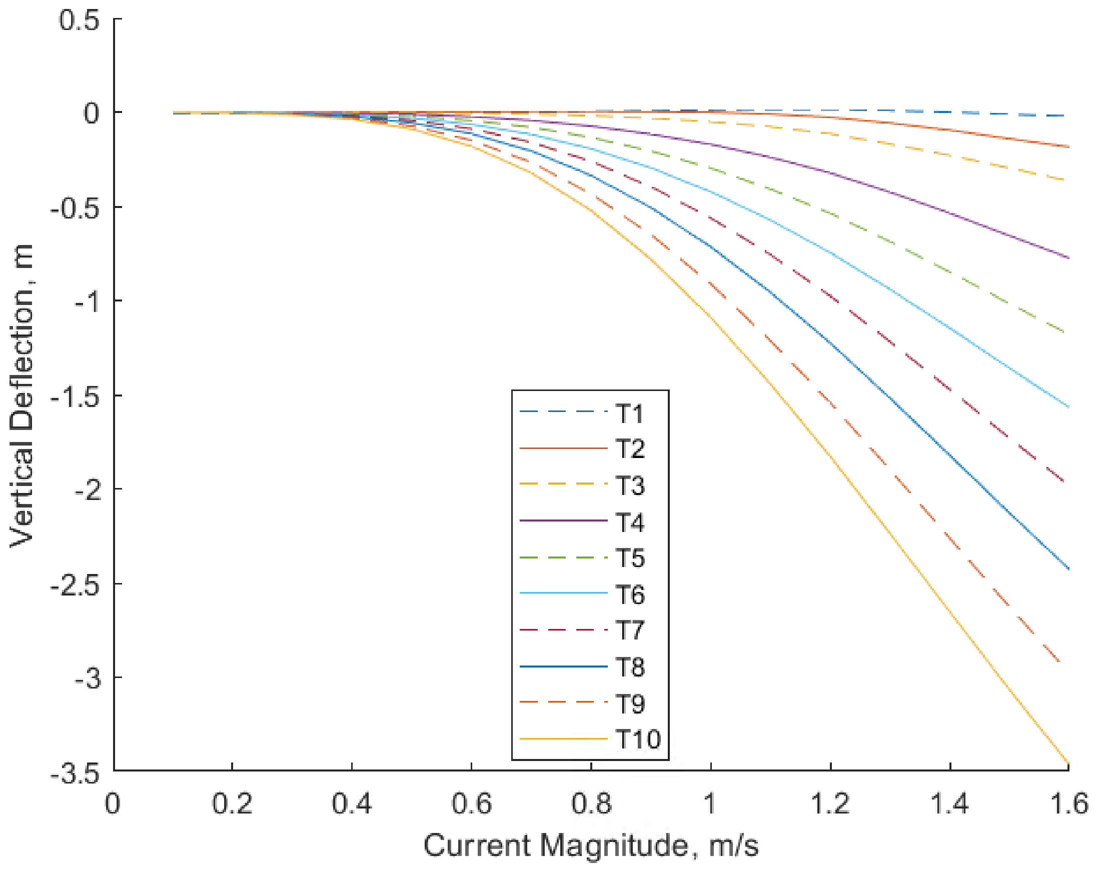


Figure 2.3: Vertical Deflection of Thermistors from Shear Current

2.2 Iterative Runs in MATLAB

In order to generate enough data to create a neural network, a MATLAB script was written to run ProteusDS iteratively. Proteus can be run in the command prompt window, and MALTAB is able to run a command line prompt. This combination allows a run of Proteus to be initiated from MATLAB code. The current profile was changed between each run, keeping everything else constant.

In 2019, the MSWB was actively recording data between May 3 and November 21. The current profile varies depending on the time of year. For example, in October, the current is close to uniform in magnitude and direction throughout the depth of the lake, with minimal turbulence in the bottom of the lakebed. In June, the current is turbulent in the bottom half of the lake. A sample current profile is shown in Figure 2.4. The data shown is from May 20 - 25, 2019. The current profiles are recorded at 1-meter intervals, starting from 2 meters below the surface through 29 meters below the surface. A strong current near the surface flows in a similar direction with a similar magnitude throughout the entire current profile, with some increased turbulence near the surface. A weak surface current can be accompanied by some moderate turbulence near the lakebed. A variety of current profiles are demonstrated by this body of water, and it is beneficial to evaluate as many as possible. It can also be seen that the current is strongest in opposite directions, between May 20 and May 22, and between May 22 and May 25.

To ensure a more accurate neural network model, current profiles from throughout the year were used in generating data. The Proteus model saves the environment data in a separate text-based file, which can be edited in MATLAB. Several MATLAB scripts were written to extract the current profile data from the log of the MSWB, overwrite the environment file changing only the current profile, and extract the data from

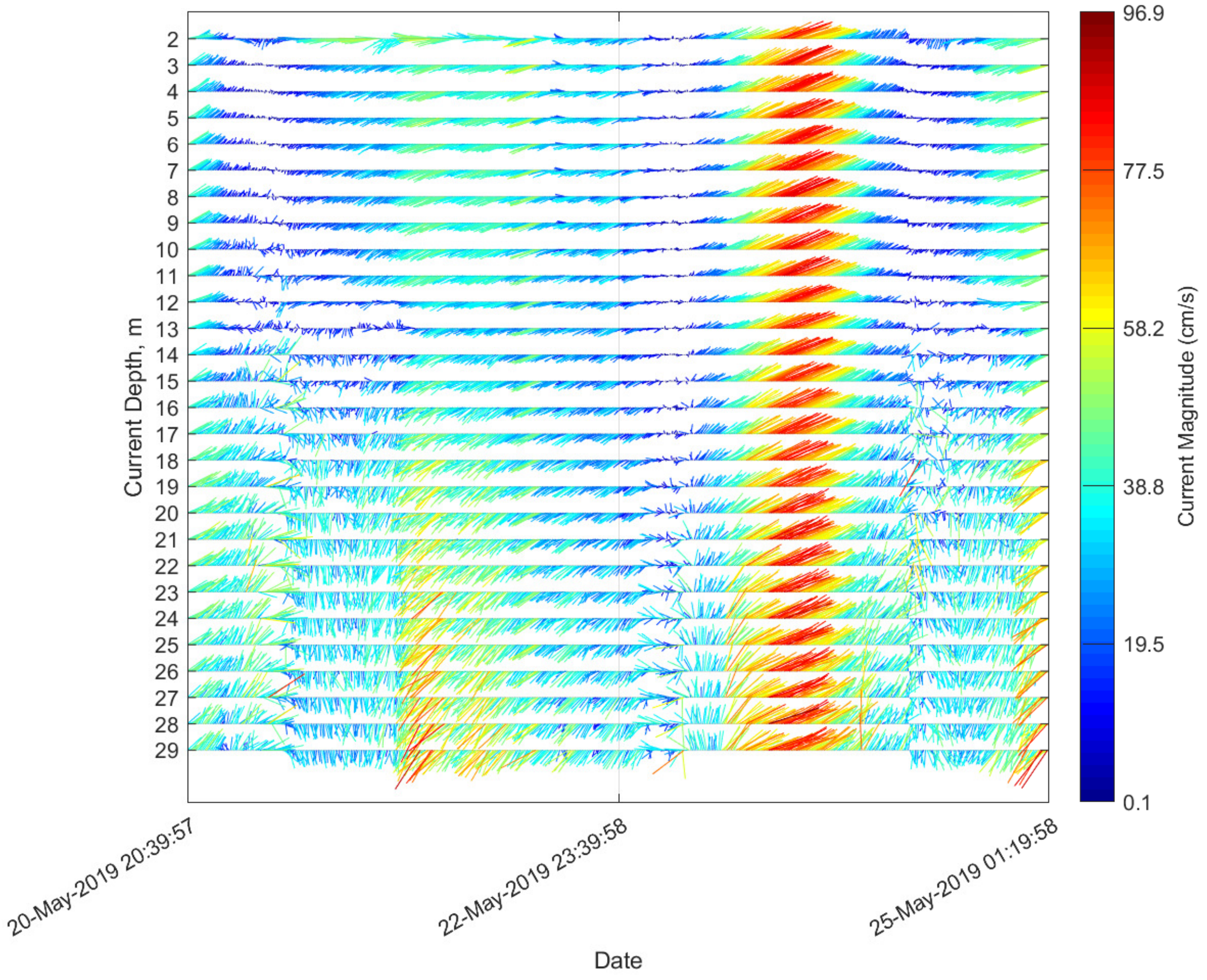


Figure 2.4: Current Profiles Recorded May 20 - 25, 2019

the Proteus simulation. The data saved was the change in X position, change in Y position, initial Z position, and final Z position of each thermistor. These scripts were run for several iterations. For each iteration, the script would load the current profile from the downloaded data, convert to Proteus usable form, overwrite the environment file, run ProteusDS, and save the data if the run was successful. A successful run was considered if ProteusDS converged on a steady state position. It takes ProteusDS approximately 50 seconds to reach a steady state position, and at least 400 seconds to determine if the simulation cannot converge. If the simulation took more than 100 seconds, it would be considered a failed run, and the data would not be used. This ensured that only successful runs were saved, to prevent any errors.

Once a large enough dataset was acquired, neural networks were created to set up a predictive model for the vertical deflection. Three different input sets were used to see the effect of the sensor displacement estimate as a function of the quantities used. A visual representation of these networks is shown in Figure 2.5. Network C network used the entire current profile, which consisted of all 28 current headings and magnitudes, for 56 inputs total. Network XY used the final X and Y positions, which was calculated by adding the starting X-Y coordinates to the changes in those positions, resulting in 2 inputs. Network CXY used all the current data as well as the XY data, resulting in 58 inputs.

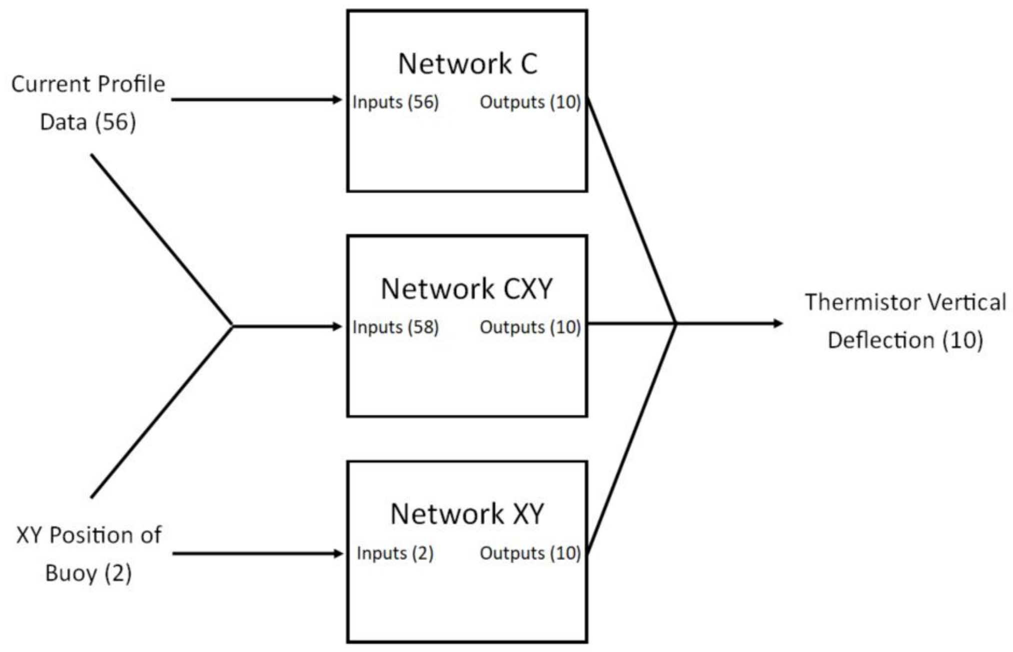


Figure 2.5: Neural Network Input/Output Configurations

Chapter 3

Results and Discussion

3.1 Buoy Behavior

Figure 3.1 shows the location of the buoy as it drifts away from the anchor, located at the origin. The points are color coded by training and testing data; the data for training the neural network (red) contained 2874 datapoints, and the data for testing the neural networks (blue) contained 266 datapoints. A majority of the points show the buoy located between 35 and 45 meters away from the anchor. A smattering of points fall within the 35 meter radius; these are due to weaker surface currents. There are likely more points within this area than shown; ProteusDS has difficulty calculating the steady state position if the current is weak enough. Any currents that are weak enough to not converge to a steady state likely do not cause significant vertical deflection in the thermistors, so these points can be safely ignored. The strong currents that cross through the straits at opposite directions are also visible in this plot.

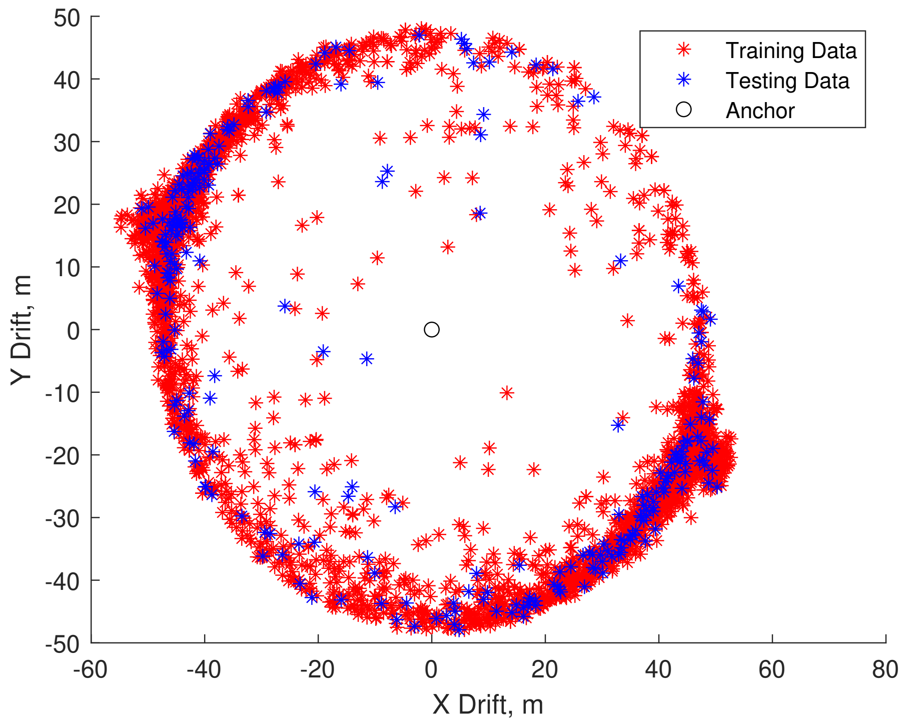


Figure 3.1: XY Position of Buoy

Figure 3.2 shows the vertical deflection of the thermistors as a function of the buoy drift from the anchor. The lines each represent a thermistor; T1 represents the thermistor closest to the surface, T10 represents the thermistor closest to the lakebed, and T2 through T9 represent the thermistors in between. The vertical deflection for all thermistors is negligible while the buoy is within 47 meters of the anchor. As the buoy moves beyond that threshold, the vertical deflection of the thermistors increases, particularly in the lower thermistors. To more clearly illustrate this deflection, the plot bounds were set to start at 48 meters.

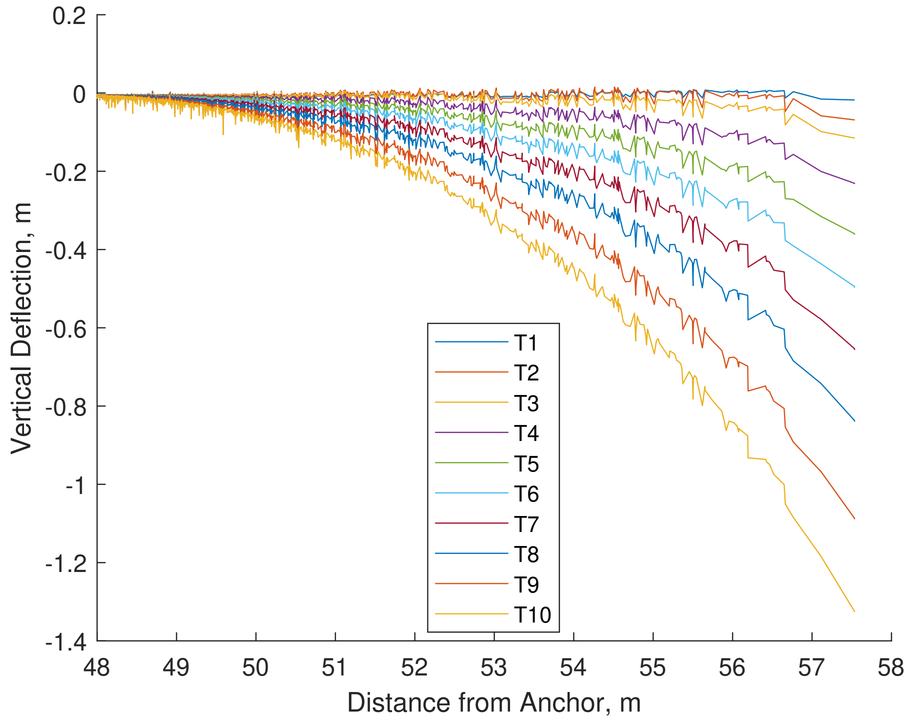


Figure 3.2: Vertical Deflection of Thermistors vs. Buoy Distance from Anchor

The maximum deflection of T1 is -0.0181 meters, with a mean of -0.0097 meters. Conversely, the maximum deflection of T10 is -1.32 meters, with a mean deflection of -0.0495 meters. The maximum deflection of T1 is 1.86 times greater than the mean deflection. The maximum deflection of T10 us 26.8 times greater than the mean deflection. The deflection of the thermistors remains in order from closest to the surface to closest to the lakebed; that is, T10 has a greater displacement than T9, which has a greater displacement than T8, and so on, as long as the deflection is greater than 0.02 meters. If the deflection of one or more thermistors is less than 0.02 meters, the order of the amount of deflection can vary in those smaller measurements.

3.2 Neural Network Configuration Results

Figure 3.3 shows the outputs of the three neural network configurations that were trained with the same input test dataset, as well as the reference output from ProteusDS (test output). Network has been abbreviated to NN in the legend.

When the deflection is small, the error between the neural networks and the test data is small. Network XY exhibits higher error in a few datapoints. As the distance from the anchor increases above 45 meters, the error of the neural networks increases. This is especially prevalent in Network XY, as it does not have the current data to calculate the lower thermistor positions accurately. The extent of the error of the neural networks can be seen in Figure 3.4.

Table 3.1 shows the error characteristics for each of the three Neural Network configurations. The maximum error for Network C was within 3 mm of the maximum error of Network CXY, and the difference between the mean error was within 5e-5 m. The maximum error and mean error for Network XY were higher than for the other

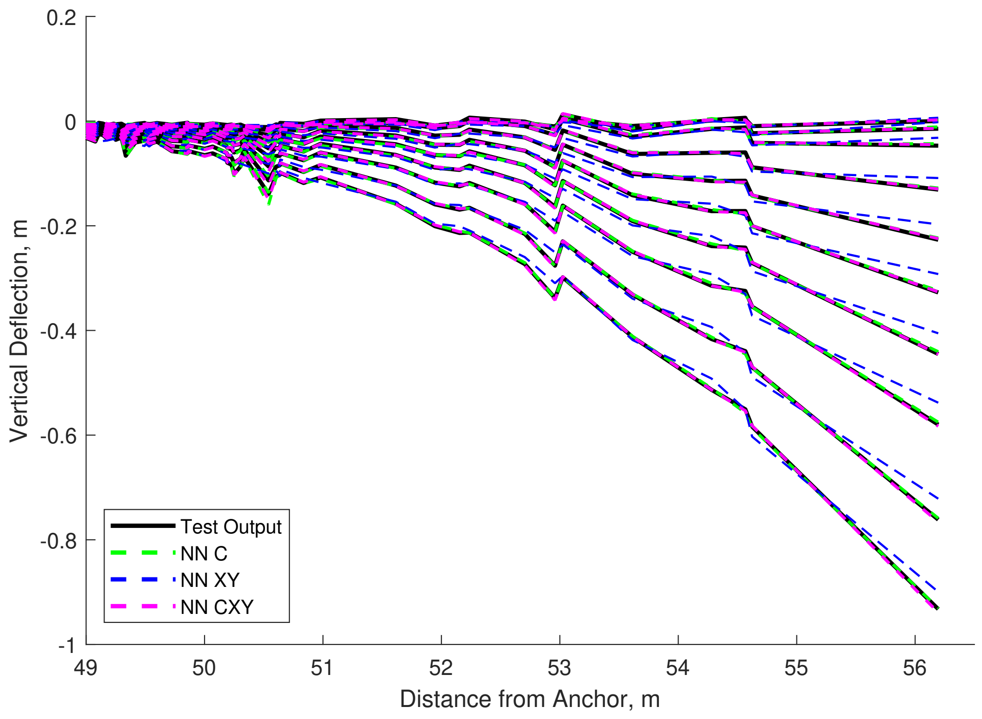


Figure 3.3: Neural Network Test Dataset Estimation of Deflection for 3 Input Configurations

networks. Figure 3.2 shows that as the distance from the anchor increases, the vertical displacement generally increases, but it is not a perfect correlation. This results in a higher error when the current data is not present. If current data is available, including the XY position of the buoy does not significantly improve the behavior of the Neural Network.

The maximum vertical deflection calculated in the observed runs was 1.32 meters, which is 4.3% of the depth of the lakebed. This is enough error that a form of calculating that deflection is necessary in this application of data collection. The

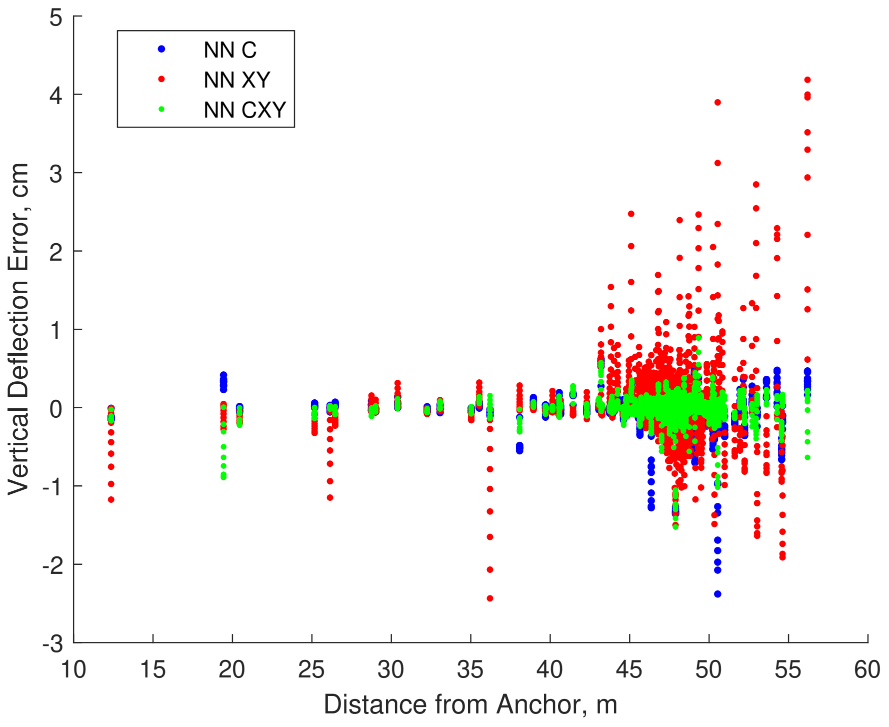


Figure 3.4: Neural Network Vertical Deflection Error

Table 3.1
Neural Network Errors and Correlation

Network	NN C	NN XY	NN CXY
Max Error (m)	0.023	0.042	0.015
Mean Error (m)	8.4E-4	2.2E-3	7.9E-4
R	0.9993	0.9962	0.9996

model developed in Proteus is adequate for estimating this vertical deflection. If possible, experimental data collection would help to validate the model and determine how well it estimates deflection, whether it be for the model of the MSWB or a smaller scale model.

Figure 3.3 shows that all three of the neural networks can estimate the vertical deflection of the thermistors. When comparing the errors of the Neural Networks, including the current profile data in the input decreases the error associated with the system. The difference between the R value for Network C and Network CXY is negligible, and the R value for Network XY is still very high.

3.3 Conclusions

If the current profile data acting on a buoy is available, that would be the best way to estimate sensor deflection. However, XY position of the buoy is adequate for improved real-time estimations of the vertical deflection throughout the mooring line.

A neural network can learn from experimental data and match it satisfactorily. The main benefit of the neural network is time; ProteusDS takes approximately 60 seconds to calculate the steady state position of the MSWB model. Data from the UGLOS website for the MSWB could be updated after this amount of time. For real time measurements, improving the neural network model by validating with real data would be a logical next step.

References

[1] "Great lakes nearshore buoy network." Online. Available: http://uglos.mtu.edu/.

[2] "Specifications." Online. Available: https://www.tidasbuoys.com/specifications.

[3] "Predicting currents in the straits of mackinac." Online. Available at https://www.glerl.noaa.gov/res/straits/.

[4] NORTEK, *Aquadopp Z-Cell 600 kHz Current Profiler*, 2018.

[5] "New to current measurement?." Online. Available at https://www.nortekgroup.com/knowledge-center/wiki/new-to-current-measurement.

[6] "ProteusDS." Online. Available: https://dsaocean.com/proteusds/overview/.

[7] L. O. Garza-Rios, M. M. Bernitsas, and K. Nishimoto, "Catenary mooring lines with nonlinear drag and touchdown," January 1997.

[8] D. Svozila, V. Kvasnicka, and J. Pospichab, "Introduction to multi-layer feed-forward neural networks," *Chemometrics and Intelligent Laboratory Systems*, vol. 39, pp. 43–62, 1997.

[9] K. Gurney, *An introduction to neural networks.* 11 New Fetter Lane, London, EC4p 4EE: CRC Press, 1997.

[10] J. D. Agrawal and M. C. Deo, "Wave parameter estimation using neural networks," *Marine Structures*, vol. 17, pp. 536 – 550, 2004.

[11] S. N. Londhe, "Soft computing approach for real-time estimation of missing wave heights," *Ocean Engineering*, vol. 220, p. 571, 2005.

[12] B. Bhattacharya, D. L. Shrestha, and D. P. Solomatine, "Neural networks in reconstructing missing wave data in sedimentation modelling," *Proceedings of the XXXth IAHR Congress*, vol. 500, pp. 770–778, 2003.

[13] DSA, *Proteus Manual*, 2018.

[14] D. R. K., "Mooring design dynamics–a matlab package for designing and analyzing oceanographic moorings," *Marine Model*, vol. 1, pp. 103–157, 1999.