Prediction of Vessel RAOs:

Applications of Deep Learning to Assist in Design

James Frizzell

Texas A&M University

Dr. Mirjam Furth

Texas A&M University

***Proceedings of the 27th Offshore Symposium, February 22nd, 2022, Houston, Texas***

***Texas Section of the Society of Naval Architects and Marine Engineers***

***Copyright 2022, The Society of Naval Architects and Marine Engineers***

**ABSTRACT**

In an age of high-paced design, a need arises for engineers to quickly estimate the feasibility of their ideas without spending weeks developing a computer model.  At the same time, the use of machine learning models, or neural networks, in the maritime industry has grown substantially over the past years.  By further extending the use of these predictive models in the design phase, marine engineers and naval architects can expedite their work.

This paper focuses on the creation of a neural network that can estimate the Response Amplitude Operators (RAOs) of a vessel given its characteristic properties such as length, beam, and draft.  A dataset was collected through a parametric design analysis of box barges using ANSYS AQWA, and the RAO was simulated for all 6 degrees of freedom. A critically damped spring equation was generated for each degree. A Keras Neural Network Model was trained on the three parameters and the wave heading angle, with the hidden layers and neuron count being adjusted to optimize the loss and maximize R-squared.

To validate the results, a series of box barges with dimensions that were not a part of the training dataset were simulated in ANSYS, while the virtual model with identical characteristics was simulated with the Neural Network.  The resulting RAOs were compared to validify the accuracy of the Neural Network.

With this predictive model, engineers can quickly determine a hullform’s RAOs, and compare the response with the common sea states along the intended route. Additionally, the model can assist in design iteration. As the hull shape gradually changes, the new RAOs can be estimated to ensure that the design is progressing in an appropriate direction.

**Keywords: SNAME, Offshore Symposium, Machine Learning, Deep Neural Network, RAO**

# Introduction

Finding the Response Amplitude Operators (RAOs) of a vessel is a numerically expensive process. Industry standard programs such as ANSYS AQWA use complex procedures to solve Equation 1 and return a precise solution.

(1)

This methodology has benefits and drawbacks. While it is a highly scalable process and can be applied to an object of any shape, the computational time is long. Additionally, the quality of results is directly coupled to the quality of the input geometric model. In an iterative design environment where an engineering is constantly adjusting a hullform to suit a client’s needs, the corresponding digital model must be updated as well in order to maintain the integrity of the numerical results. This method, although highly accurate when performed correctly, is time-consuming and by extension, costly.

To eliminate the need for complex mathematical equations, Neural Networks (NN) can be implemented to derive the correlation between inputs and outputs. In this case, a Deep Neural Network (DNN) has been created to predict the RAOs of a vessel with respect to frequency given only the characteristic dimensions of length, beam, draft, and the wave heading angle. With this approach, the calculation of approximate RAOs only requires a few seconds of computational time.

Simply inserting a neural network into a complex problem is never an ex-machina solution, though. To obtain accurate results, a comprehensive dataset must be collected during development to train the NN on. Hundreds or thousands of vessels must have their RAOs solved in the conventional method to provide learning material for the predictive model. Additionally, the model is only able to accurately predict vessel with similar body shapes as those given in the training dataset. To create a robust, accurate model would take a long time to develop.

Although the number of applications of NN in the marine industry are growing rapidly, the data science sector is still rather young . In 2020, a study was performed using neural networks to predict the roll RAO value and the wave frequencies at which they occurred (Jae and Hyo 2020). However, the scope of the previous work was limited in comparison to the objectives of this project. In (Gjeraker 2021), a neural network was created to estimate the RAOs of a drillship with differing wave spectrum inputs. The accuracy of this model was highly accurate, but the results are limited to Subsea 7’s vessel Cybership Inocean Cat I Drillship.

The purpose of this paper is to detail the process of creating a proof-of-concept neural network that can predict the RAOs of a simple box barge. Future work will be needed to expand the capability to complex hull structures.

# Methodology

## **Data Collection**

The first step in developing any neural network is to collect a dataset. The dataset must include:

1. The model inputs (independent variables)
2. The model outputs (dependent variables)

The inputs are easy to define and mark the bounds of viable prediction. Table 1 outlines the four parameters that were used in this study. The boundary of accuracy is the upper and lower limits of inputs, meaning that this model will work best for box barges with characteristic dimensions that fall within these ranges. The model will still work for barges outside of these values, but then the predicted RAOs are subject to higher inaccuracies. In total, 142 box barges were included in the dataset. The maximum size of the barges was limited by the mesh generation feature in ANSYS AQWA, which limits the number of nodes to 40,000. To streamline the data collection, a parametric study was performed in ANSYS, during which the mesh resolution was unable to be changed. Therefore, the largest model tested fell just within the 40,000-node limit, while the smallest barges maintained a fine enough mesh grid to ensure accurate results.

Table 1: Inputs to Neural Network Model

|  |  |  |
| --- | --- | --- |
| Input Type | Minimum Value | Maximum Value |
| Vessel Length (meters) | 2 | 25 |
| Vessel Beam (meters) | 1 | 16 |
| Vessel Draft (meters) | 0.15 | 1.23 |
| Wave Heading Angle (degrees) | -180 | 180 |

To collect the model outputs, numerically determined RAO values from ANSYS AQWA were used.

First, a validation study was performed to ensure that the analysis setup would provide accurate results (Lee, Kim and Goo 2012). A barge that had been previously analyzed for pitch RAOs with strip theory was evaluated in ANSYS, and the comparison is shown in Figure 1.

Figure 1: Training data for the NN was simulated in ANSYS AQWA. To ensure accuracy of the ANSYS AQWA model the results were compared with existing strip theory calculations. Overall, good accuracy is seen.

A parametric study was setup in ANSYS DesignModeler, in which a box barge with a length, *L*, beam, *B*, and draft, *T*, would be created and passed into AQWA. In AQWA, a point mass was inserted at the volume centroid to ensure that the weight of the barge would be appropriate to produce the given draft. This weight was determined by Equation 2, where the density of freshwater was selected.

(2)

The point mass was also given specified moments of inertia based on the dimensions of the box barge. It was assumed that the mass was evenly distributed throughout the vessel, so finding the mass moments of inertia were able to be found with Equations 3, 4 and 5.

(3)

(4)

(5)

The computation time averaged 5 minutes per vessel, for a total of about 17 hours on a 12-Core Dell Precision 5280 with 128 GB RAM.

ANSYS AQWA stores RAO data in a text table format enumerated by frequency, which was saved to be used in the pre-processing. Other information in the text table includes added mass, diffraction, Froude-Krylov Forces, and damping. These could all be used in future studies as alternative predictive models.

## **Data Pre-Processing**

Data preprocessing was conducted in steps. First, the frequency-dependent RAO value for a datapoint was collected. Then, a Python script fit the frequency-dependent RAO value of each degree of freedom to a curve of a particular form, based on the natural similarity of the response to certain equations. This process was done using the ‘curve\_fit’ function provided by scipy. Equation 6 was used for the surge and sway degrees of freedom, where the response on the barge closely mirrors the pattern of a critically damped spring-mass damper. Equation 7 was fit the heave response, where the response at very short frequencies trended to 1, and high frequency waves resulted in 0 response. This closely represents a transformation of the arctangent function, reflected around the x-axis. Finally, a Gaussian distribution shown in Equation 8 was used for the pitch, roll, and yaw responses, where a single peak appeared in mid-range frequencies and the response trended to 0 outside of that range. In Equations 6, 7, and 8, x represents the wave frequency in radians per second, and the result of the function is the RAO value. Other curves were considered, including 3rd, 6th and 9th order polynomials. These were eventually discarded because the curves generated by the neural network were poorly fit to the actual data in all degrees of freedom. Additionally, to produce well-fit polynomials required more coefficient parameters when compared to the selected equations, meaning that the neural network would need to predict more values. This tended to decrease model accuracy and thus was avoided for this project.

|  |  |  |
| --- | --- | --- |
| **Equation** | **Degrees of Freedom** |  |
|  | Surge, Sway | (6) |
|  | Heave | (7) |
|  | Roll, Pitch, Yaw | (8) |

The free parameters *A, B* and *C* were then collected and stored. These became the model outputs.

This process was repeated for each degree of freedom for each wave heading of each barge size. The result of the data processing was 1136 datapoints, each with 18 coefficients that described the shape of the RAO curve with respect to frequency – 3 for all 6 degrees of freedom. To check that the curve fitting was accurate, the R-Squared and Mean Average Error (MAE) between the true data and the fit curve was evaluated, and the summary is shown in Table 2. The highest discrepancy was found in sway. However, after comparing the median and average errors, it was found that evaluating the R-Squared metric for two series which are both very close to zero results in the R-Squared being equal to zero. This pulled the average down substantially for the surge and sway. Nonetheless, a median R-Squared of 0.955 is very high and suggests that the curve fitting generally went well.

Table 2: R-Squared Results of Curve Fit to Raw Data

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | X (Surge) | | Y (Sway) | | Z (Heave) | | RX (Roll) | | RY (Pitch) | | RZ (Yaw) | |
|  | R2 | MAE | R2 | MAE | R2 | MAE | R2 | MAE | R2 | MAE | R2 | MAE |
| Average | 0.85 | 0.038 | 0.89 | 0.038 | 0.94 | 0.063 | 0.92 | 3.734 | 0.93 | 5.681 | 0.96 | 0.540 |
| Median | 0.928 | 0.044 | 0.939 | 0.043 | 0.989 | 0.042 | 0.951 | 1.276 | 0.977 | 0.645 | 0.980 | 0.052 |

## **Neural Network Architecture**

The creation and optimization of neural networks has been made simple with modern packages. Keras and Tensorflow were the two packages used in this study. This paper will not detail the inner workings of machine learning algorithms, as there are better sources to reference for more information on that subject.

Simple pre-processing methods were employed to prevent any errors – namely dropping any rows with NaN values. These NaN values appeared due to certain barge sizes failing to converge within the default 800 iterations. The number of rows dropped was 178, about 15.7% of the total data. Of these, 163 occurred in the yaw category. It is possible that Equation 8 is a poor representation of the yaw response for certain wave conditions, although the high R-Squared speaks to the overall quality of the Gaussian equation. The input and output data were read into Python and split into a training and test dataset with an 80/20 split, supplying 766 datapoints for the model to train on and 192 for testing.

The neural network itself consists of an input layer, multiple hidden layers, and an output layer, each having a set number of neurons. Between each layer, a dropout layer randomly sets the input units to 0 at a rate of 20% (Keras Team n.d.). Between the input layer and first hidden layer, a batch normalization layer scales the input values. The input and output layers are dictated by the data that will be passed through the model, while the hidden layers are more variable and can be adjusted to optimize the model.

A parametric study was done to select the optimal architecture for the neural network and can be seen in Figure 2 and Figure 3.. The number of trainable parameters is defined by Equation 9, where *n* is the number of hidden layers and *An* is the number of neurons in each hidden layer. 18 is added to the product because of the 18 output parameters that are used in the model. The parametric study consisted of 648 model architectures, each trained for 75 epochs. For each trained model, the R-Squared Score and Akaike Information Criteria (AIC) were evaluated. Figure 2 and Figure 3 summarize the results of the parametric study. Figure 2 also shows that there is a maximum obtainable model accuracy given the data preprocessing, as the R-Squared plateaus at around 0.71.

|  |  |
| --- | --- |
|  | (9) |

Ultimately, a model with 3 hidden layers was selected. Table 3 lists the number of neurons in each layer in the model. This model had the highest R-Squared score and lowest AIC value, implying that it will produce the best results. (Tensorflow, 2022-a). A high neuron count additionally increases the ability of the model to fit the input to the outputs due to having more linear combinations, but also requires a larger training time.

The model chosen requires approximately 30 seconds to train for 250 epochs in Python 3.9, on a 12-Core 3.6 GHz system with 32 GB RAM. Training time can be adjusted to optimize the model. A shorter training time tends to avoid the risk of over-fitting, but if the training time is too short the model may not be fully fit at the end of the training. To determine whether the model has been properly fit, a visual inspection of the Loss-Epoch graph shown in Figure 4 was used. If the line is horizontal and mostly unchanging by the final epoch, the model is well-fit (Tensorflow, 2022-b). To check that the model is properly predicting the *A*, *B*, and *C* parameters, the true values can be plotted against the predicted values, as seen in Figure 5.

The specific hyperparameters that were used in the model are outlined in Table 4. Due to time constraints, an in-depth hyperparameter tuning analysis was unable to be performed.

|  |  |
| --- | --- |
| Figure : Results of Neural Network Architecture Parametric Study. Increasing the neuron count does not necessarily improve the model accuracy beyond about 105 trainable parameters. | Figure : Results of Neural Network Architecture Parametric Study. The Akaike Information Criteria (AIC) is used to compare the relative goodness of fit between different models. |

Table 3: Selected Neural Network Architecture. Each layer contains the specified number of neurons.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input Layer | Hidden Layer 1 | Hidden Layer 2 | Hidden Layer 3 | Output Layer |
| 4 | 128 | 256 | 256 | 18 |

Table : Hyperparameters of the Neural Network.

|  |  |
| --- | --- |
| Model Optimizer | Adam |
| Learning Rate | 0.001 |
| Activation | ReLu |
| Loss Type | Mean Absolute Error |
| Layer Dropout | 20% between each hidden layer |
| Normalization | 1 Batch Normalization layer before first hidden layer |
| Validation Split | 0.2 |
| Training Duration | 250 Epochs |

|  |  |
| --- | --- |
| Figure 4: Model Loss During Training Progression. A slope of zero at the final epoch suggests that the model has been appropriately fit. | Figure 5: True Values (ANSYS AQWA) Plotted Against Predicted Values (Present Model), R2 = 0.668. There is a large number of parameters that the model fail to train to be zero, shown by the vertical line of dots on the left side of the plot. |

# Results

With the model creation and training completed, it can be benchmarked against random datapoints. Figure 7 shows the similarity between the predicted values and true values. Two metrics were used to quantify performance; Relative percent difference (RPD) assigns a number between 0 and 200 and explains the difference between the predicted curve and actual curve, as defined in Equation 10. A simple error calculation cannot be used since the true value is often 0 and would result in a division by 0. In the cases where the true value is 0, the RPD is often very high, so the analysis has been supplemented with the MAE.. In Figure 7, sway has an RPD of 200.0 but a MAE of 0.029 m/m response. So, even though the percent difference is large, the response is only incorrect by about 3 centimeters for every meter of wave height – which is avery accurate for a barge 15 meters in length.

|  |  |
| --- | --- |
|  | (10) |

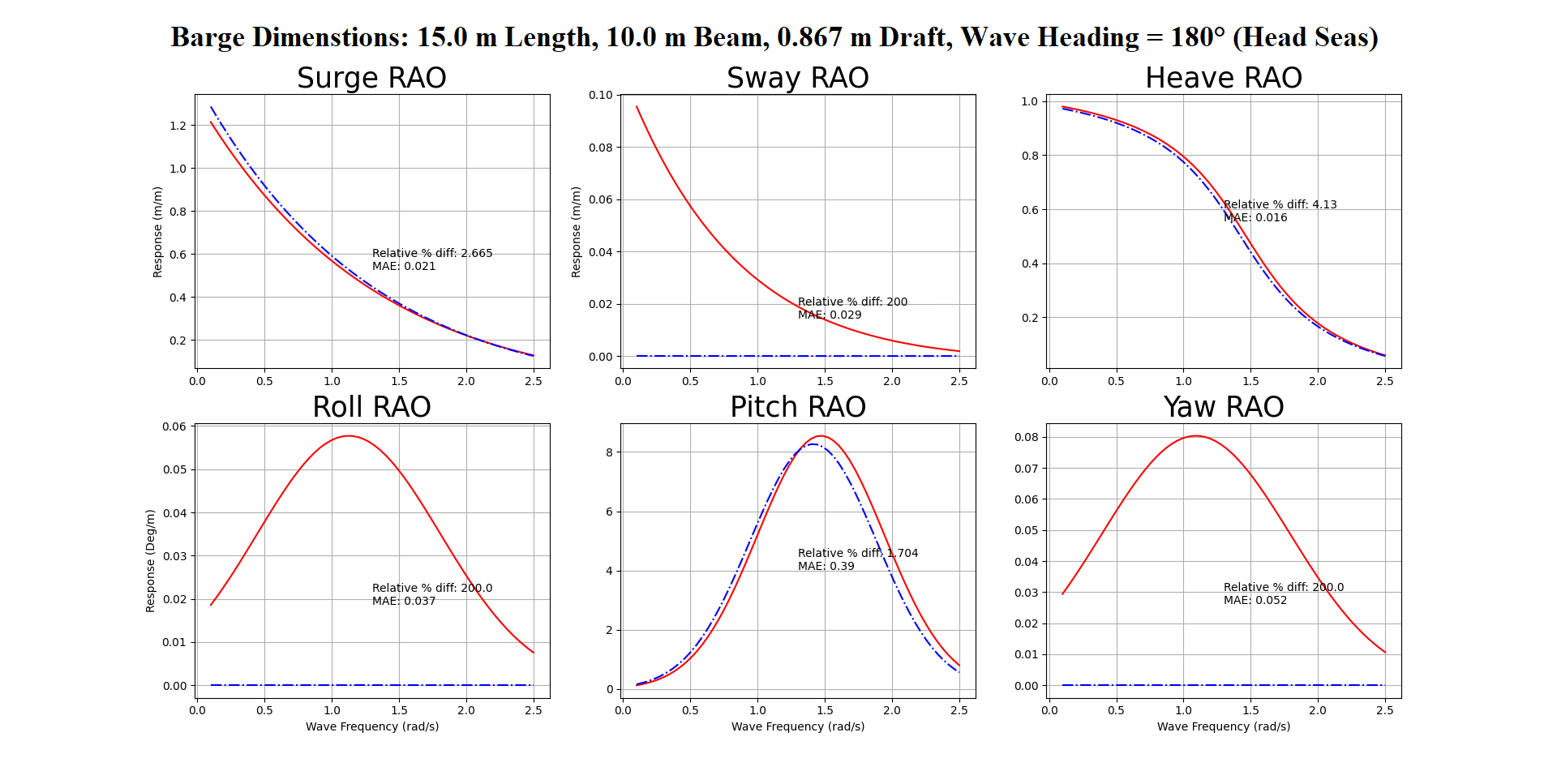


Figure 6: Predicted RAO Values. Surge, sway, and pitch look almost indistinguishable, while the other three degrees of freedom stray from the true values. This is because the true value for these degrees of freedom should be zero. Note the scaling on the y-axis is inflated for these degrees of freedom to better show this phenomenon

## **Error Analysis**

The robustness of this model is highly important, as the goal of this project was to predict RAOs for any sized vessel. To check this, 120 barge sizes were randomly sampled from the model predictions and compared to the true RAO values. The RPD and MAE for each datapoint was compared to study the effects of barge size, wave heading, and degree of freedom on the accuracy of the model.

Figure 7 shows the effect of vessel size on the model’s prediction accuracy. The average RPD error was calculated by averaging the RPD error at each evaluated wave frequency for each degree of freedom of the randomly sampled barges. Admittedly, this tends to smooth over abnormalities, but after seeing that the two variable have little correlation, it is assumed that there were few anomalies in the data.. To assist in the visualization, the weave heading was differentiated by marker type and color. By considering the wave direction as well, a weak trend suggests that the 45- and 135-degree waves tended to produce higher results with lower RPD. It is important to note that the error distribution for very small barges is more extreme than for larger barges. This promotes the idea that the model will give consistent predictions for larger vessels, although there may be some inaccuracy. Note that this does not mean the model will be more accurate for larger barges, but rather that the expected error in the prediction is more uniform. This means that a user can assign error tolerances to an output based on the wave heading input, seeing as certain headings have lower errors.

Figure 8 plots the statistical distribution of the RPD Error by wave direction, broken down into each degree of freedom. As shown by the box plots, for the 0-, 90-, and 180- degree wave headings, the model had trouble producing accurate parameters for 3 degrees of freedom. These three correspond with the 3 directions in which the actual vessel response should be 0. Because of the way the RPD is calculated, the near-0 in the denominator causes the RPD calculation to become large if the predicted value is not exactly 0 as well. Figure 9 and Figure 10 expand on this with the MAE distributions for the linear and rotational degrees of freedom and show that although the RPD Error may be high, the actual difference between the true and predicted values are small .

Figure 11, Figure 12, and Figure 13 all show the distribution of error for each degree of freedom. Figure 12 shows that the RPD error for sway, roll, and yaw had the highest relative error. This likely stems from the above-mentioned division issue that arises in the calculation of the RPD. Figure 12 and Figure 13explain the RPD error, by showing that in general, the difference between the predicted and actual RAOs is small.

Although there are some inaccuracies in the predictive power of this model, it is generally able to produce results that are within reason. There are a few ways to improve this. First, the model could be split into two models – one for the linear degrees of freedom and one for the rotational degrees of freedom. It is possible that the curve shape of the surge and sway RAOs does not fit cleanly into the exponential equation provided, which would also explain the errors seen in Table 2. If a better general equation was found for these degrees of freedom, a new model could be trained to find parameters and the results may be more accurate.

Additionally, a larger dataset could be collected. With more data, the neural network has more ability to learn the correlations between the input and output parameters, which increases the accuracy. This could possibly increase the R-Squared score from Figure 6.

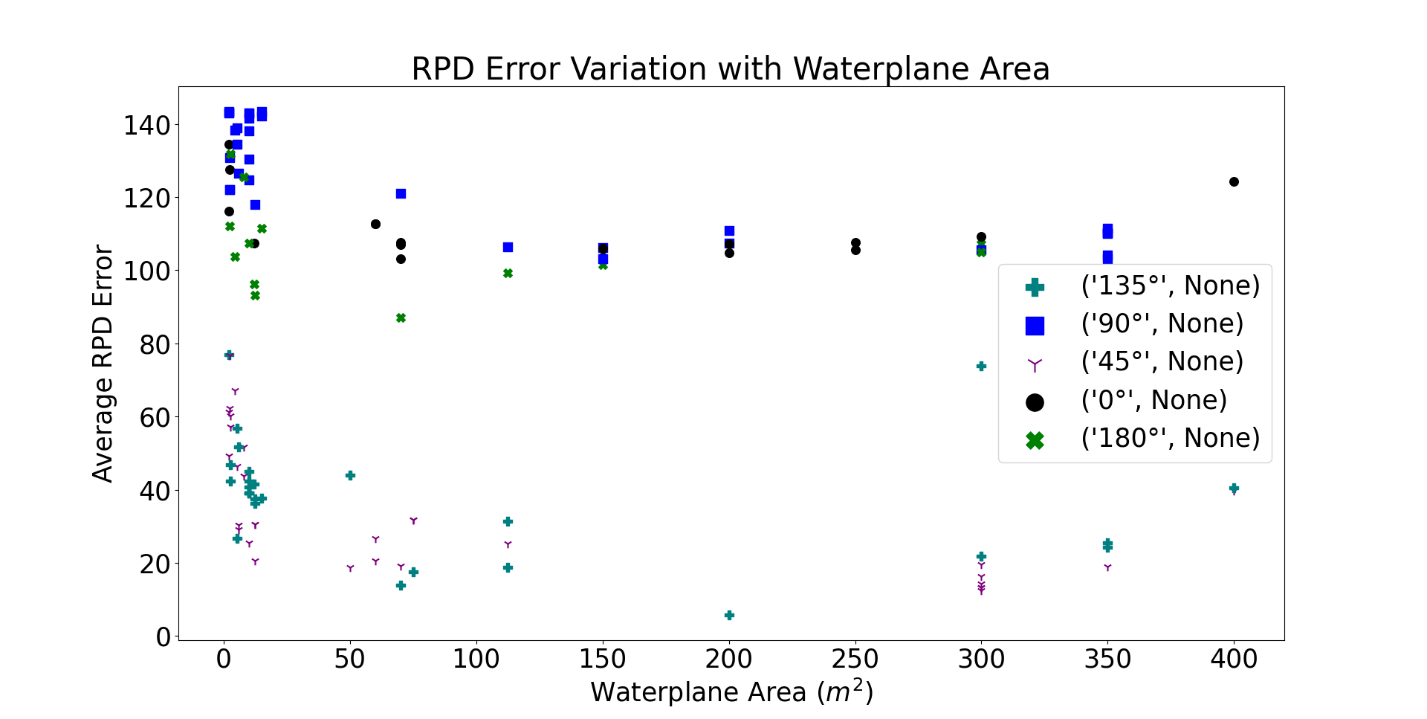


Figure 7: RPD Error Variation with Waterplane Area. There is no clear correlation between barge size and accuracy of the model. Wave heading is depicted as well, and shows that at 0, 90, and 180 degrees the model has a much higher RPD error.

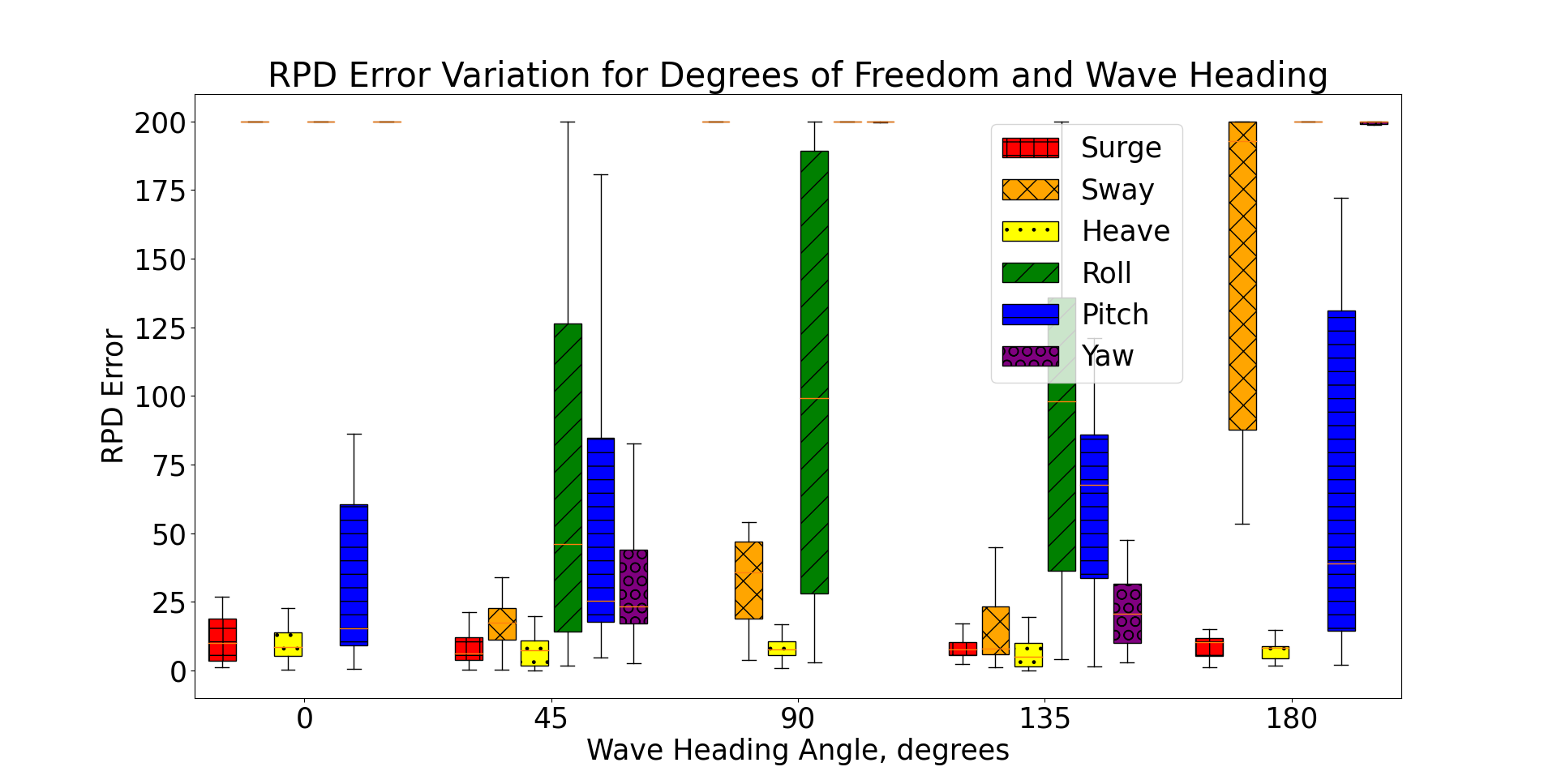


Figure 8: RPD Error Variation for Degrees of Freedom with Wave Heading. Again, at 0, 90, and 180 degrees three of the six degrees of freedom have high error. These three correspond to the directions where the actual response is zero.

|  |  |
| --- | --- |
| Figure 9: Raw Error Variation for Rotational Degrees of Freedom with Wave Heading. The degrees of freedom with large raw error generally have low RPD error at the same wave heading, as pictured in Figure 8. | Figure 10: Raw Error Variation for Linear Degrees of Freedom with Wave Heading. The degrees of freedom with large raw error have low RPD error at the same wave heading, as pictured in Figure 8. |

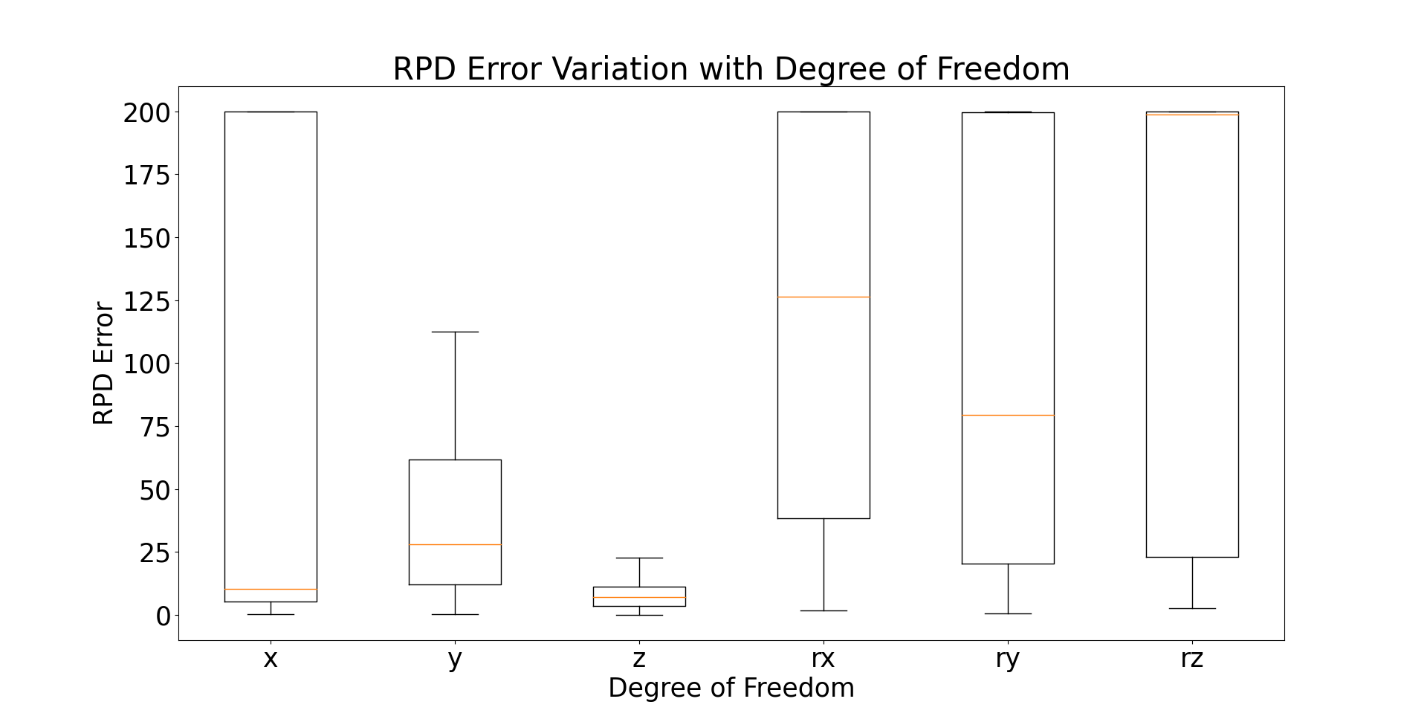


Figure 11: RPD Error Variation with Degree of Freedom. Surge, sway, roll, and yaw have the highest RPD error. Of these, yaw is the most concerning, seeing as the median RPD error is 200, while the lower quartile is around 25. This suggests that the model is failing to predict remotely accurate parameters for yaw approximately 50 percent of the time and misses the mark by a large margin in a further 25 percent of cases.

|  |  |
| --- | --- |
| Figure : Raw Error Variation with Rotational Degrees of Freedom. Like Figure 9 and Figure 10, the degrees of freedom with high MAE are the same degrees of freedom with low RPD error. | Figure : Raw Error Variation with Linear degrees of freedom. The sway degree of freedom has high MAE but low RPD error, while surge has low MAE and high RPD error, seen in Figure 11. |

## **Runtime Analysis**

Once the NN was trained and tuned, a software wrapper was created to easily interface with prediction inputs and outputs. Python’s built-in libraries ‘cProfile’ and ‘timeit’ were used to benchmark the speed of the code. cProfile reported that 437698 function calls were executed within 0.322 seconds. Timeit averaged an execution time of 0.094 seconds across 100 trials. This was performed on a Windows 11 System with 32 GB RAM and a i7-12000KF clocked at 3.6 GHz.

The time to specify barge dimensions and run the simulation using the NN averaged less than one minute during testing. Comparatively, creating a CAD model the same barge in ANSYS DesignModeler and simulating the results in AQWA averaged 10 minutes.

When considering the time spent on design modelling, the use of the NN model is much more efficient than the commercial alternatives due to its simplicity. Because the software is based in Python and uses a defined class-structure to process user inputs, it can easily be integrated in other systems in a plug-and-play fashion. However, this comes with downsides – namely a very restrictive modelling capability and non-negligible errors in the RAO predictions. Additionally, this NN cannot predict any hydrodynamic features such as added mass, Froude-Krylov forces, or damping – all of which can be evaluated in ANSYS AQWA in the same amount of time needed to solve for the RAOs. Such capabilities could be added in the future, but the dataset would need to be expanded.

# Conclusion

This paper outlines the process by which the predictive model was created and tuned. Bypassing the computationally expensive CAD models and providing another path for vessel design to grow into will assist in the development of digital twins in the modern era. The model creation starts with the collection of RAO data from known sources. In the case that this data is unavailable, RAO curves must be generated with a modelling software. Fitting the raw data to a curve of a general equation allows for the simple expression of the form shape of the responses. Simplifying the data wherever possible is important when setting up a neural network, as having a lower number of outputs improves accuracy when supplying the same amount of input data. A neural network is set up and adjusted to produce a reasonable estimate of the RAOs for the collected training data, and then implemented into a design studio where engineers can quickly input parameters and investigate the vessel’s response characteristics.

With the model functioning properly for box barges, the first phase of this research project is complete. Given the accuracy and robustness of the model so far, the proof-of-concept has shown the feasibility of using a neural network to supplement the traditional solution process of solving for the RAOs of a vessel. Of course, this model is far from perfect. A ‘silver-bullet’ model which completely replaces math is unlikely to ever be created. However, this project builds upon past research about the use of neural networks in the maritime industry and will hopefully be useful in the early stages of design where very little is known about a hullform.

Future steps to improve the model are already in motion. The model itself can continually be refined, through changes in the layer architecture or through implementation of different curve fitting functions. Expanding the dataset to include hullforms beyond simple box barges is another such improvement, which will make the predictive power more useful to real-world vessel shapes. Finally, experimental validation of the model predictions should be performed to ensure that the model can obtain the accuracy that is claimed in this project.

# References

Gjeraker, Anna Holm. 2021. "Response Amplitude Operator." Norwegian University of Science and Technology, June.

Jae, Hwan Lim, and Jae Jo Hyo. 2020. "Prediction of Barge Ship Roll Response Amplitude Operator Using Machine Learning Techniques." *Journal of Ocean Engineering and Technology* 167-179.

Keras Team. n.d. "Dropout layer." *Keras.* Accessed February 17, 2020. https://keras.io/api/layers/regularization\_layers/dropout/.

Lee, S., Y. B. Kim, and J. Goo. 2012. "Analysis of motion response of barge ships in regular waves." *12th International Conference on Control, Automation and Systems.* IEEE. 1920-1922.

Tensorflow. 2022. *Basic regression: Predict fuel efficiency.* 01 19. Accessed 01 21, 2022. https://www.tensorflow.org/tutorials/keras/regression#the\_auto\_mpg\_dataset.

—. 2022. *Overfit and underfit.* 01 19. Accessed 01 21, 2022. https://www.tensorflow.org/tutorials/keras/overfit\_and\_underfit.