LUCAS PEREIRA COTRIM

NEURAL NETWORK META-MODELS FOR FPSO MOTION ESTIMATION FROM ENVIRONMENTAL DATA

São Paulo
2023

LUCAS PEREIRA COTRIM

NEURAL NETWORK META-MODELS FOR FPSO MOTION ESTIMATION FROM ENVIRONMENTAL DATA

Dissertation submitted to the Escola Politécnica of the University of São Paulo to obtain the degree of Master of Science.

São Paulo
2023

LUCAS PEREIRA COTRIM

NEURAL NETWORK META-MODELS FOR FPSO MOTION ESTIMATION FROM ENVIRONMENTAL DATA

Revised Version

Dissertation submitted to the Escola Politécnica of the University of São Paulo to obtain the degree of Master of Science.

Concentration area:

Control and Mechanical Automation Engineering

Advisor:

Prof. Dr. Eduardo Aoun Tannuri

Co-Advisor:

Prof. Dra. Anna Helena Reali Costa

São Paulo
2023

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 22 de Maio de 2023

Assinatura do autor: Lucas Pereira Cotrim

Assinatura do orientador:

Catalogação-na-publicação

Pereira Cotrim, Lucas
Neural Network Meta-Models for FPSO Motion Estimation From Environmental Data / L. Pereira Cotrim -- versão corr. -- São Paulo, 2023.
68 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1.Aprendizado de Máquina 2.Redes Neurais 3.Inteligência Artificial 4.Engenharia Naval e Oceânica 5.Otimização Estocástica I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos II.t.

RESUMO

O processo atual de projeto de sistemas de amarração de unidades flutuantes de produção, armazenamento e transferência (em inglês *Floating Production, Storage, Offloading*, ou FPSOs) é altamente dependente de um modelo matemático hidrodinâmico da plataforma e da precisão de simulações dinâmicas, através das quais séries temporais de movimento são avaliadas de acordo com requisitos de projeto. Esse processo é demorado e pode apresentar resultados imprecisos devido às limitações do modelo matemático e à complexidade geral da dinâmica da plataforma. Neste trabalho é proposto um Simulador Neural, um conjunto de modelos alternativos baseados em dados, que recebem condições meteoceânicas como entrada e são especializados na previsão de diferentes estatísticas de movimento relevantes ao projeto do sistema de amarração de uma FPSO: Máximo ângulo de *roll*, Máximo *Offset* do centro de gravidade da plataforma e deslocamentos de seus *fairleads*. Os meta-modelos são treinados por dados de correntes, ventos e ondas fornecidos em períodos de 3 horas da Bacia de Campos de 2003 a 2010 e a resposta dinâmica associada de uma FPSO do tipo *spread-moored* obtida por simulação através do software *Dynasim*. Técnicas de otimização de hiperparâmetros são realizadas para obtenção de arquiteturas de Redes Neurais Artificiais (ANNs) otimizadas para previsão de cada variável de projeto e para diferentes calados da FPSO. Finalmente, mostra-se que os modelos propostos capturam corretamente a dinâmica da plataforma quando comparados com os resultados obtidos pelo Dynasim. Conclui-se que redes neurais podem ser treinadas em dados meteoceânicos reais para previsão adequada de variáveis de projeto em tempo computacional reduzido em comparação com métodos tradicionais baseados em simulação dinâmica. A arquitetura proposta pode ainda ser integrada em um *framework* de aprendizado automatizado por meio do treinamento contínuo dos modelos conforme novos dados são medidos.

Palavras-Chave – FPSOs, Redes Neurais Artificiais, Modelos Alternativos, Otimização de Hiperparâmetros, Busca de Arquiteturas de Redes Neurais.

ABSTRACT

The current design process of mooring systems for Floating Production, Storage and Offloading units (FPSOs) is highly dependent on the availability of the platform's mathematical model and accuracy of dynamic simulations, through which resulting time series motion is evaluated according to design constraints. This process can be time-consuming and present inaccurate results due to the mathematical model's limitations and overall complexity of the vessel's dynamics. We propose a Neural Simulator, a set of data-based surrogate models with environmental data as input, each specialized in the prediction of different motion statistics relevant to mooring system design: Maximum Roll, Platform Offset and Fairlead Displacements. The meta-models are trained by current, wind and wave data given in 3h periods at the Campos Basin (Brazil) from 2003 to 2010 and the associated dynamic response of a spread-moored FPSO obtained through time-domain simulations using the Dynasim software. Hyperparameter Optimization techniques are performed in order to obtain optimal Artificial Neural Network (ANN) models specialized in different platform drafts. Finally, the proposed models are shown to correctly capture platform dynamics, providing good results when compared to motion statistics obtained from Dynasim. We conclude that an ANN surrogate model can be trained directly on real metocean conditions and corresponding measured FPSO motion statistics to provide increased accuracy and reduced computational time over traditional methods based on dynamic simulation. Moreover, the proposed architecture can be integrated into an automated learning framework: The data-based surrogate models can be continuously fine-tuned and updated with newly measured data, resulting in improved accuracy over time.

Keywords – Floating Offshore Platforms, Artificial Neural Networks, Surrogate Models, Hyperparameter Optimization, Neural Architecture Search.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AR - Autoregressive models

ANN - Artificial Neural Network

AutoML - Automated Machine Learning

BO - Bayesian Optimization

CFD - Computational Fluid Dynamics

CG - Center of Gravity

DSM - Draft Sensitivity Matrix

DoF - Degrees of Freedom

EI - Expected Improvement

EX - Exogenous models

FD - Fairlead Displacements

FEM - Finite Element Method

FPSO - Floating Storage Production and Offloading

GPS - Global Positioning System

GS - Grid Search

HL - Hidden Layer

HO - Hyperparameter Optimization

IMU - Inertial Measurement Unit

MAE - Mean Absolute Error

ML - Machine Learning

MLP - Multi-Layer Perceptron

MSE - Mean Squared Error

NAS - Neural Architecture Search

NARX - Nonlinear Autoregressive models with Exogenous inputs

NED - North-East-Down reference frame

RAO - Response Amplitude Operator

ROV - Remotely Operated underwater Vehicle

RS - Random Search

RMSE - Root Mean Squared Error

SA - Simulated Annealing

TPE - Tree-structured Parzen Estimator

CONTENTS

1 INTRODUCTION

Current FPSO's Mooring System Design consists in the obtainment of local environmental conditions over a representative period of time and subsequent dynamic simulation of the FPSO model subject to combinations of the extreme winds, waves and currents expected in the next 10 to 100 years of operation, which are obtained from statistical projections of the environmental conditions. The maximum offsets and mooring line tensions are then obtained and verified to safely remain within project limits.

This process, however, relies on the numerical simulation of a dynamic model on softwares such as Dynasim [Nishimoto, Fucatu and Masetti (2002)], which multiplies the approximated wave energy spectrum and the FPSO's RAOs (Response Amplitude Operators) to obtain the expected vessel's movement. This process can be both time consuming and present slightly inaccurate responses when compared to the actual measured movement.

Recently, the increasing performance of data-based machine learning models in a variety of domains in conjunction with the high computational times of traditional models and the unprecedented availability of data have motivated the study and development of alternative models, denominated surrogate or meta-models. The main motivation behind the use of such models is to directly model complex, computationally costly dynamics through available data. Meta-models have been successfully implemented as alternatives for Finite Element (FEM) [Guarize et al. (2007), Pina et al. (2014), Sidarta et al. (2017), Gonzalez et al. (2020)] and Computational Fluid Dynamics (CFD) models [Damasceno et al. (2020)] for predicting mooring line tensions and submerged riser's vibration responses, respectively. Gumley, Henry and Potts (2016) successfully implemented a neural network capable of predicting the hourly mean offset of a turret-moored FPSO from environmental conditions, showing that differences in mooring configuration result in differences between predicted and measured offset, which can be used to monitor mooring system integrity.

1.1 Objective

The main objective of this research is to design and validate a set of data-based meta-models capable of predicting relevant statistics associated with an FPSO's dynamic response to generic environmental conditions. The models are trained and validated through data obtained from the simulation of a spread-moored platform subject to 6 years of currents, waves and winds obtained from hindcast data at the Campos Basin, Brazil.

In the proposed framework meta-models are trained to correctly predict the maximum roll, offset and fairlead displacements obtained through dynamic simulation, rather than real FPSO responses. This process allows for the validation of the proposed architecture's performance without the interference of sensor noise on measured platform responses. Results indicate that the proposed set of meta-models correctly captures the simulated platform's responses and suggest that a similarly structured neural simulator trained on real FPSO responses can be more accurate than traditional dynamic simulation methods.

As the proposed set of artificial neural network meta-models is trained directly on environmental conditions and associated FPSO motion, no meaningful physical interpretation of its underlying operations is available. However, this allows for the models to capture complex nonlinear dynamics, such as varying mooring line damping and second order wave drift, which are approximated in dynamic simulation methods. Overall, a data-based approach is expected to present three main advantages in comparison to traditional methods:

- **Increased accuracy when training on real FPSO data:** Training directly on real environmental conditions and the corresponding platform's motion responses avoids several approximations and simplifications of physical phenomena implemented on traditional simulation softwares. Moreover, the availability of a considerable volume of data (over 18 thousand 3h periods) improves the accuracy of trained data-based models.
- **Automated Learning:** The resulting system is designed to be integrated to other design tools and continuously updated with newly measured environmental and platform motion data. This continuously improves the meta-model's accuracy over time.
- **Reduced Computational Time:** After training, the computational time associated with the evaluation of a neural network prediction is significantly shorter

than that associated with time integration of the system's dynamic equations. As a result, given a set of different environmental conditions, a set of data-based meta-models is capable of obtaining relevant platform movement information faster than traditional methods.

1.2 Organization of the manuscript

The remainder of this manuscript is organized as follows: Chapter 2 provides an overview of the state of the art of surrogate models both in general applications and in the problem of FPSO motion prediction specifically. Chapter 3 introduces the general Theoretical Concepts applied in this research, which are necessary to understand the Proposed Methods described in Chapter 4. Chapters 5 and 6 describe the conducted experiments and results respectively, presenting a brief discussion of results afterwards. Finally, Chapter 7 offers preliminary conclusions and indicates the next steps to be taken in this research.

2 LITERATURE REVIEW

Over the past few decades, data-driven Machine Learning (ML) models have presented increasing performance in a wide variety of applications, from image recognition and product recommendation to medical diagnosis and language translation. In areas such as Offshore Engineering, where complex non-linear physical phenomena are involved and there's a high availability of data due to existing sensor measurements required to ensure safety of operations, this approach has been particularly successful when compared to analytical hydrodynamic models.

Machine Learning applications in Offshore Engineering can be divided in two main categories: Mooring Failure Detection and Response-based Seakeeping Analysis. The former can be seen as a classification problem, in which models are trained to correctly identify mooring line breakage given the vessel's motion, allowing for the appropriate repair of compromised lines while saving expenses associated ROV inspection. The latter is considered a regression task, in which data-driven meta-models are trained to accurately predict FPSO motion given incident metocean conditions, in order to verify that vessel's responses such as roll, offset and line tensions remain safely within design criteria.

Initial efforts in FPSO motion prediction through ML models have focused on the use of simple Artificial Neural Networks (ANNs) to predict a small number of motion statistics. Mazaheri (2006) has trained MLPs with 1 Hidden Layer to predict an FPSOs surge, sway and total excursion over a 3h period given the following 6 environmental variables: Wave height, wave direction, current speed, current direction, wind speed and wind direction. The ANN models were trained on data obtained by a Hydrodynamic mathematical model in order to reduce the computational time associated with time-series simulations in response-based FPSO design.

In the following decade, with the introduction and rise in popularity of Digital Twins, ML models have been applied to time-series motion prediction of FPSO, focusing on real-time prediction of individual variables: such as heave [Masi et al. (2011)], roll [Yin, Perakis and Wang (2018)] and coupled heave-pitch motion [Huang, Jiang and Zou (2021)]. ML-

based time-series prediction has also been successfully applied to the prediction of mooring line tensions [Pina et al. (2014)], with the initial segments of the tension time-series obtained by FEM and the remaining by Wavelet Neural Network meta-models, which resulted in significant reduction of computational time. De Pina et al. (2013) presented a comparison of different types of surrogate models in the prediction of mooring line tension, classified according to the time-series input: Purely Autoregressive (AR) which utilizes exclusively the top tension series itself, Exogenous models (EX) that use surge, sway and heave time-series and Nonlinear Autoregressive models with Exogenous inputs (NARX) which combine both present and past values of exogenous series as well as past values of the desired series itself.

Gumley, Henry and Potts (2016) have successfully used ANNs to predict a turret-moored FPSO's mean offset given input variables such as significant wave height, current speed and vessel draft. By changing mooring line configuration, a significant difference between ANN predicted and actually measured motion statistics is observed, which can be used to monitor the integrity of an FPSOs mooring system. They also determined the optimal input variable combinations to be used in the prediction of each motion statistics.

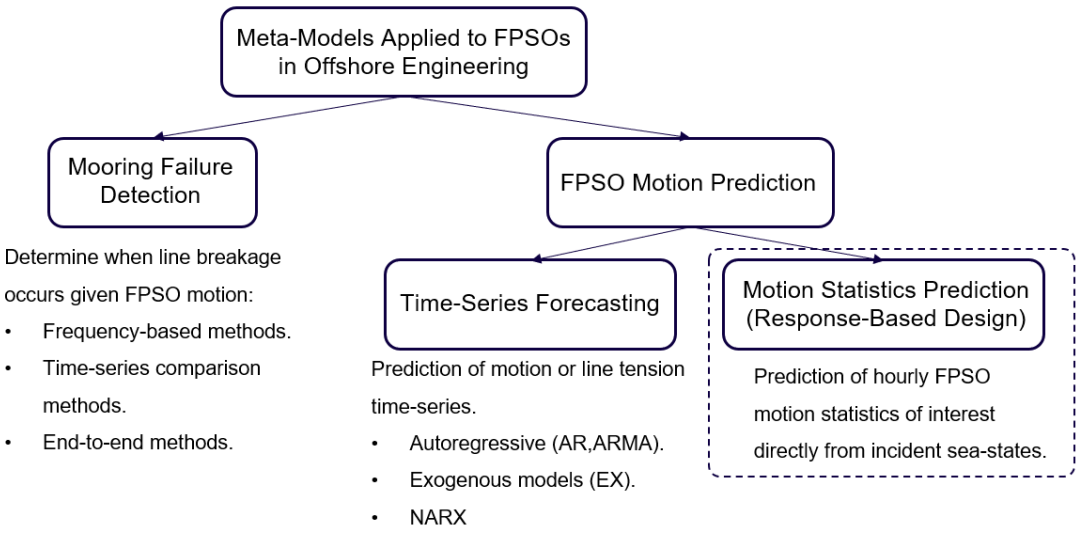


Figure 1: Diagram representing the main applications of FPSO meta-models in Offshore Engineering.

It is important to notice that, unlike traditional methods based on hydrodynamic equations, there's no universal standardization of data-driven meta-models. Their structure varies according to application and is often determined through trial and error or hyperparameter optimization (section 5.3), which is exemplified by different authors presenting different choices of input variables to be considered in each model as shown in the papers presented above. This is a significant challenge to ML applications overall,

with considerable research efforts currently being directed towards the development of more general methods, reducing the necessity of domain-specific knowledge and human intervention. With this in mind, the framework described in 4.1 attempts to be as general as possible, making use of all available variables with no assumption of their relevance to different output variables. Figure 1 illustrates a simple diagram of the main applications of FPSO meta-models in offshore engineering, encapsulating the areas of papers cited in this section.

3 THEORETICAL CONCEPTS

This chapter aims to provide an overview of the theoretical foundations required to understand the problem definition and proposed methods presented in the next chapter. Section 3.1 describes Dynasim's Hydrodynamic model of an FPSO and how it is used to generate simulations of platform motion when subject to arbitrary environmental conditions. Section 3.2 illustrates the concept of ANNs, particularly Multi-Layer Perceptrons (MLPs), focusing on the process of learning from data. Sections 3.3 and 3.4 refer to Hyperparameter Optimization and the validation of trained ML models respectively.

3.1 Platform Dynamic Model

Mooring systems are used to provide station-keeping characteristics of FPSOs subject to incident currents, wind and waves by anchoring the platform through the use of mooring lines attached to the sea bed. Such systems are not only a key component to ensure the safety of offshore operations, but also pose several engineering challenges, from project and conception to operation and maintenance. Dynamic simulation is a crucial step in the design and subsequent monitoring of mooring systems, allowing for the prediction of FPSO motion given incident metocean conditions and platform-specific variables such as draft and the mooring configuration itself.

Dynamic simulation requires a hydrodynamic model of the FPSO and parameterized wind and wave energy spectrums: The former is obtained by calculating hydrostatic properties of the vessel given its geometry, such as its Mass Matrix and Center of Gravity (CG) position, which are used in conjunction with the vessel's mesh to determine hydrodynamic coefficients of added mass, radiation damping, first order and mean drift wave forces for each platform draft. A hydrodynamic model then calculates the vessel's Response Amplitude Operators (RAOs), which are essentially transfer functions used along with wave energy spectrums to determine the amplitude of FPSO motion. The latter is obtained by data which comes from a hindcast model of the area, calibrated by a large amount

of measurements by wave radars, anemometers, wave buoys and current meters, that are either installed in the platforms or moored to the seabed. The complete dynamic model of a spread-moored FPSO is outside the scope of this work and is presented in detail by Nishimoto, Fucatu and Masetti (2002).

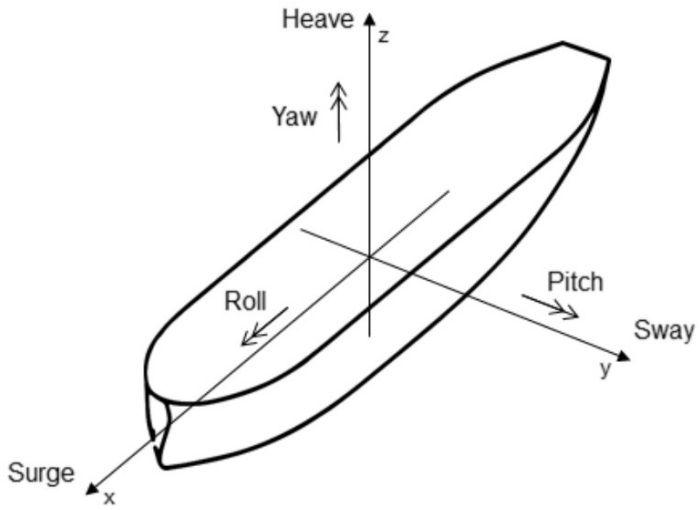


Figure 2: Illustration of the 6 DoF of a marine vessel.

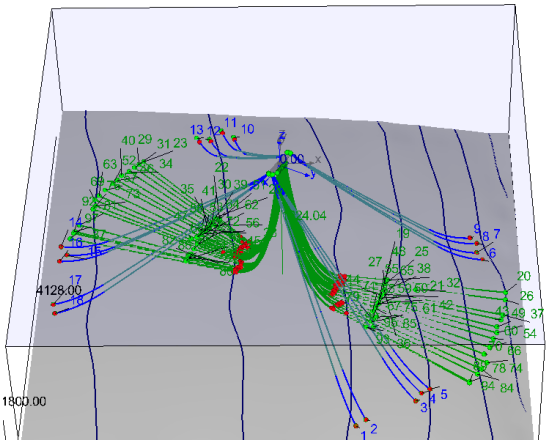
Simulation results consist in the 6 Degrees of Freedom (DoF) time-series motion of the platform, which are measured at a North-East-Down (NED) geographic reference frame and, given the platform's heading, can be converted to the following 6 DoF (Fig. 2):

1. Surge: Longitudinal motion.
2. Sway: Sideways motion
3. Yaw: Rotation about the vertical axis.
4. Roll: Rotation about the longitudinal axis.
5. Pitch: Rotation about the transverse axis.
6. Heave: Vertical motion.

Simulations are performed on a model of a spread-moored FPSO with 18 mooring lines located at the Campos Basin (Fig. 3a), the platform's main variables are given by table 1. Figure 3b illustrates the simulated platform as viewed in the Dynasim software interface, where mooring lines are presented in blue and risers in green.

Fig. 5. FPSO platform at campos basin

(a) FPSO platform at Campos Basin.



(b) FPSO platform in Dynasim interface.

Figure 3: Illustration of Simulated FPSO at Campos Basin and Dynasim Interface.

Variable	Value
Vessel Type	FPSO
Mooring	Spread-Moored
Length (L_{pp})	320m
Breath	54.5m
Mean Heading	206.16°
Draft Ballasted Condition	8m
Draft Full-Load Condition	21m
Displacement Ballasted Condition	102931 ton
Displacement Full-Load Condition	302918 ton

Table 1: Main Variables of Simulated FPSO

3.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) are mathematical models inspired by biological neural networks in which neurons are interconnected in a layered structure and their output signals are obtained by applying a nonlinear activation function to the weighted sum of their input signals. Figure 4 illustrates the model of a single artificial neuron.

Let $\mathbf{x} \in \mathbb{R}^N$ be an input vector and $f_{\mathbf{W}}(\mathbf{x})$ be a single-hidden-layer ANN parameterized by weights $\mathbf{W} = \{\mathbf{W^{(1)}}, \mathbf{W^{(2)}}\}$, where $\mathbf{W^{(1)}} \in \mathbb{R}^{MxN}$ are the weights associated with connections from input layer to hidden layer and $\mathbf{W^{(2)}} \in \mathbb{R}^{KxM}$ are the weights associated with connections from hidden layer to output layer. For a regression MLP with a linear activation function in the output layer, the network's k-th output is given by:

$$y_k = f_k(\mathbf{x}|\mathbf{W}) = \sum_{j=1}^{M} w_{kj}^{(2)} h\Big(\sum_{i=1}^{N} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \Big) + w_{k0}^{(2)}. \tag{3.1}$$

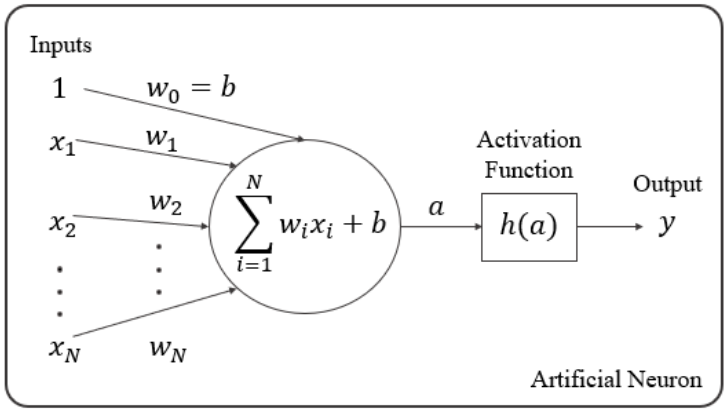


Figure 4: Diagram of an Artificial Neuron.

Given a dataset $\mathbb{D} = \{\mathbf{x^{(i)}}, \mathbf{y^{(i)}}\}_{i=1}^{N_D}$ with known inputs $\mathbf{x^{(i)}} \in \mathbb{R}^N$ and outputs $\mathbf{y^{(i)}} \in \mathbb{R}^K$, the process of training an ANN consists in finding the weights $\mathbf{W}$ that minimize a loss function, typically the Mean Squared Error (MSE) given by:

$$L(\mathbf{W}) = \frac{1}{N_D} \sum_{i=1}^{N_D} \Big[\frac{1}{K} \sum_{k=1}^{K} \big(y_k^{(i)} - f_k(\mathbf{x^{(i)}}|\mathbf{W}) \big)^2 \Big]. \tag{3.2}$$

This can be done by performing gradient descent on the loss function and updating the weights in the direction that minimize it: $\mathbf{W} \leftarrow \mathbf{W} - \alpha \nabla L(\mathbf{W})$, where α is the learning rate. The gradient of the loss function with respect to the weights $\nabla L(\mathbf{W})$ is found by back-propagation of the error through the network.

3.3 Hyperparameter Optimization

Hyperparameters are parameters used to control the training process itself. Instead of being learned as the network weights are, they are fixed during training and define the model's architecture or learning algorithm itself. The appropriate choice of Hyperparameters is problem-specific and of fundamental importance in the development of robust models with high generalization capability. Examples of ANN hyperparameters are:

- **Network architecture**: Number of hidden layers and number of neurons in each layer.
- **Activation Function**: The activation function $h : \mathbb{R} \to \mathbb{R}$ used in hidden-layer neurons.
- **Number of Epochs**: The number of times the entire training dataset is passed to the network during training.

- **Batch size**: The number of samples in each training batch used to approximate the gradient of the loss function.
- **Optimizer**: The gradient estimation algorithm implemented during training.

Hyperparameter Optimization (HO) consists in the general process of determining a model's optimal hyperparameters for a given task. Finding the optimal hyperparameters for an ML model has always been an important, yet demanding and time-consuming task. Recently, new methods have been developed to automate these tasks, greatly reducing the human effort required to optimize models [Feurer and Hutter (2019)] and thus creating the field of Automated Machine Learning (AutoML). The positive results obtained by NNs and DL in recent years motivated the development of Neural Architecture Search (NAS), a sub-field of AutoML dedicated to the optimization of ANN architectures specifically.

NAS methods can be defined by three dimensions [Elsken, Metzen and Hutter (2019)]: a *search space* $\mathcal{S}_\mathcal{S}$, which bounds the possible architectures evaluated during NAS; a *performance estimation strategy*, which defines an *objective function* C used to evaluate model performance during the search process; and a *search strategy*, which determines how the algorithm explores the search space. The search space and performance estimation strategy are manually defined to obtain the best optimization performance for the shortest execution times and to ensure that the models generated by this process are able to generalize to unseen data.

3.3.1 Grid Search

Grid Search (GS) is a simple hyperparameter optimization algorithm in which, rather than defining a $\mathcal{S}_\mathcal{S}$ from which to sample architectures, all architectures are initially predefined, usually distributed in a "grid" pattern. The objective function C is then evaluated for each of them and the best performing architecture is chosen. This is ineffective, as very few architectures can be evaluated and information from previous trials is not used in subsequent ones.

Algorithm 1: Grid Search

Input: Architectures $\mathcal{M}_i$, objective function C

Output: search history $\mathcal{H}$

$\mathcal{H} \leftarrow \emptyset$;

for $i = 1, 2, \ldots, k$ **do**

  $\mathcal{M} \leftarrow \mathcal{M}_i$;

  $\mathcal{H} \leftarrow \mathcal{H} \cup \{\{\mathcal{M}, C(\mathcal{M})\}\}$;

end

3.3.2 Random Search

Random Search (RS) is implemented by sampling architectures randomly from $\mathcal{S}_\mathcal{S}$, aiming to find the architecture which most closely reaches the goal of the optimization process, as described in Algorithm 2. RS is most commonly used as baseline for comparison with other methods, yet has been shown to achieve performance similar to that of state-of-the-art NAS algorithms on some specific problems [Sciuto et al. (2019)].

Algorithm 2: Random Search

Input: search space $\mathcal{S}_\mathcal{S}$, objective function C

Output: search history $\mathcal{H}$

$\mathcal{H} \leftarrow \emptyset$;

for $i = 1, 2, \ldots, k$ **do**

  $\mathcal{M} \leftarrow$ `random(`$\mathcal{S}_\mathcal{S}$`)`;

  $\mathcal{H} \leftarrow \mathcal{H} \cup \{\{\mathcal{M}, C(\mathcal{M})\}\}$;

end

3.3.3 Simulated Annealing

Simulated Annealing (SA) is a local optimization algorithm analogous to the metallurgical process annealing. SA relies on the definition of a *neighborhood* $n(\mathcal{M})$ of a model architecture $\mathcal{M}$, which must be defined during the algorithm implementation so that neighbor models are deemed to possess similar architectures. In each trial, a random neighbor architecture $\mathcal{M}' \in n(\mathcal{M})$ is chosen, evaluated and compared to the current architecture $\mathcal{M}$. If $\mathcal{M}'$ is better than $\mathcal{M}$ (i.e. $C(\mathcal{M}') < C(\mathcal{M})$), it is chosen as the next architecture; if it is not (i.e. if $C(\mathcal{M}') \geq C(\mathcal{M})$), its acceptance probability is given by

$$p(\mathcal{M}', \mathcal{M}, T) = \exp\left(-\frac{C(\mathcal{M}') - C(\mathcal{M})}{T}\right).$$

The *temperature* T is a parameter that regulates the probability of accepting worst models and decreases after each trial, commonly by exponential decay, in which the current temperature is multiplied by a constant decay rate α, $0 < \alpha < 1$. Both α and the initial temperature T_0 must be tuned to each specific problem. The general procedure is shown in Algorithm 3.

Due to the temperature decay mechanism, SA can explore less promising model architectures in early trials, yet still converge to a local minimum, since the probability of accepting a worse model decreases with the temperature. It is also relatively simple to implement and use, as the only parameters that need to be tuned are T_0 and α. However, as a local search algorithm, it can only consider architectures similar to the current one, according to the definition of its neighborhood n. Thus, it has limited capacity to explore the search space in a limited number of trials and is heavily dependent on $\mathcal{M}_0$. A more thorough discussion on Simulated Annealing can be found in [Laarhoven and Aarts (1987)].

Algorithm 3: Simulated Annealing

Input: search space $\mathcal{S}_\mathcal{S}$, objective function C, neighborhood n, initial architecture $\mathcal{M}_0$ and temperature T_0, temperature decay rate α

Output: search history $\mathcal{H}$

$\mathcal{M} \leftarrow \mathcal{M}_0$;
$T \leftarrow T_0$;
$\mathcal{H} \leftarrow \{\{\mathcal{M}_0, C(\mathcal{M}_0)\}\}$;
for $i = 1, 2, \ldots, k-1$ **do**
  $\mathcal{M}' \leftarrow \texttt{random}(\{\mathcal{M}' \in n(\mathcal{M})\})$;
  if $(C(\mathcal{M}') < C(\mathcal{M}))$ *or* $(p(\mathcal{M}', \mathcal{M}, T) > \texttt{random}([0,1]))$ **then**
    $\mathcal{M} \leftarrow \mathcal{M}'$;
  end
  $\mathcal{H} \leftarrow \mathcal{H} \cup \{\{\mathcal{M}, C(\mathcal{M})\}\}$;
  $T \leftarrow \alpha \cdot T$;
end

3.3.4 Bayesian Optimization

Bayesian Optimization (BO) consists of two key components: a probabilistic surrogate model S of the objective function C; and a policy P, denoted as *acquisition function*, for selecting new parameters based on the surrogate model. In each trial, an evaluation of

C updates the surrogate model S, allowing P to select a new architecture $\mathcal{M}$ most likely to achieve the objective of the optimization for the next trial. The general procedure is shown in Algorithm 4.

Algorithm 4: Bayesian Optimization

Input: search space $\mathcal{S}_\mathcal{S}$, objective function C, initial architecture $\mathcal{M}_0$, surrogate model S, acquisition function P

Output: search history $\mathcal{H}$

$\mathcal{M} \leftarrow \mathcal{M}_0$;

$\mathcal{H} \leftarrow \{\{\mathcal{M}_0, C(\mathcal{M}_0)\}\}$;

Initialize the surrogate model S;

for $i = 1, 2, \ldots, k-1$ **do**

  $\mathcal{M} \leftarrow \arg\max_{\mathcal{M}' \in \mathcal{S}_\mathcal{S}} P(\mathcal{M}', S)$;

  $\mathcal{H} \leftarrow \mathcal{H} \cup \{\{\mathcal{M}, C(\mathcal{M})\}\{$;

  $S \leftarrow$ `updateSurrogate(`S, $\{\mathcal{M}, C(\mathcal{M})\}$`)`;

end

The surrogate model is used as an estimate of the objective function C, and can be generated in a number of ways. Often Gaussian Process regression is used [Frazier (2018)]. However, there has been a rise in the use of Tree-structured Parzen Estimator (TPE), as it is more flexible to non-numerical search spaces, is able to scale to bigger search spaces with a smaller computational cost and has been shown to achieve better performance on some optimization problems [Feurer and Hutter (2019), Bergstra et al. (2011)]. Various choices are also available for the acquisition function, the most common being Expected Improvement (EI). This method is based on selecting the model with the best estimated performance at each turn. EI is calculated using the surrogate model, and the architecture with the largest EI is selected for evaluation in the next trial. A more thorough discussion on Bayesian Optimization can be found in [Frazier (2018), Bergstra et al. (2011)].

3.4 Validation of ML Models

In Machine Learning, model validation is defined as the process in which a trained model is evaluated with a separate validation or testing dataset, comprised of samples unseen in the training process. This is performed in order to assess the model's generalization capability and is used in NAS methods in order to find an optimal set of model hyperparameters [Wang and Zheng (2013)].

Typically, available data is subdivided into three different sets [James (2013)]:

- **Training Dataset**: Set of examples used during the learning process to fit the model's parameters, usually contains most of the available data. Due to the high number of trainable parameters, ML models tend to exploit correlations in the training data that do not hold in general, which compromises the model's capability to generalize to unseen data. This process is commonly known as overfitting.
- **Validation Dataset**: A set of examples not present on the training dataset, but which follow the same probability distribution. The Validation Dataset is used to tune hyperparameters of the model, functionally working as a hybrid between the low-level Training Dataset and high-level final Test Dataset.
- **Test Dataset**: The set of data examples used exclusively to asses the performance of the final trained model. Significantly poor performance on the Test Dataset when compared to the Training Dataset indicates overfitting.

4 PROPOSED METHODS

This chapter aims to describe the methodology adopted in this research, indicating how the theory discussed in chapter 3 is applied to the problem of FPSO motion prediction through data-driven meta-models. The first section illustrates the overall architecture of the proposed framework, denominated NeuroSim, explaining its functionality both in response-based design and in seakeeping applications as well as detailing inputs and outputs of the system. Section 4.2 describes the research workflow adopted and its main stages, from data acquisition to testing the final models. The Workflow decomposition is highly correlated to the implemented code architecture, with each stage associated with its corresponding Kedro pipeline module. Finally, sections 4.3 and 4.4 describe the implemented HO algorithm and the error metrics used to validate trained models, which are used to discuss results in Chapter 6.

4.1 NeuroSim Architecture

NeuroSim is an intelligent, data-driven FPSO Dynamics Simulator based on Artificial Neural Networks with two main applications in offshore engineering:

- Mooring System Design Tool: The proposed Neural Simulator can be trained on a number of highly accurate simulations of centenary critical metocean conditions which are normally time-consuming to obtain through traditional time-series dynamic simulation. After training, NeuroSim can provide design variables such as maximum FPSO roll or center of gravity offset associated with unseen conditions instantaneously, without the need to compute the entire motion and extract these values. As a result NeuroSim is a valuable auxiliary design tool as it allows for a higher number of metocean conditions to be evaluated.

- Monitoring of an Operational FPSO: On operational platforms, NeuroSim can be used to convert a batch of future expected metocean conditions into expected motion

statistics. This applicability provides an extra layer of safety as critical short term motion is predicted and appropriate measures can be taken. In this case, NeuroSim's meta-models are continuously trained through FPSO motion already measured by IMU and GPS systems.

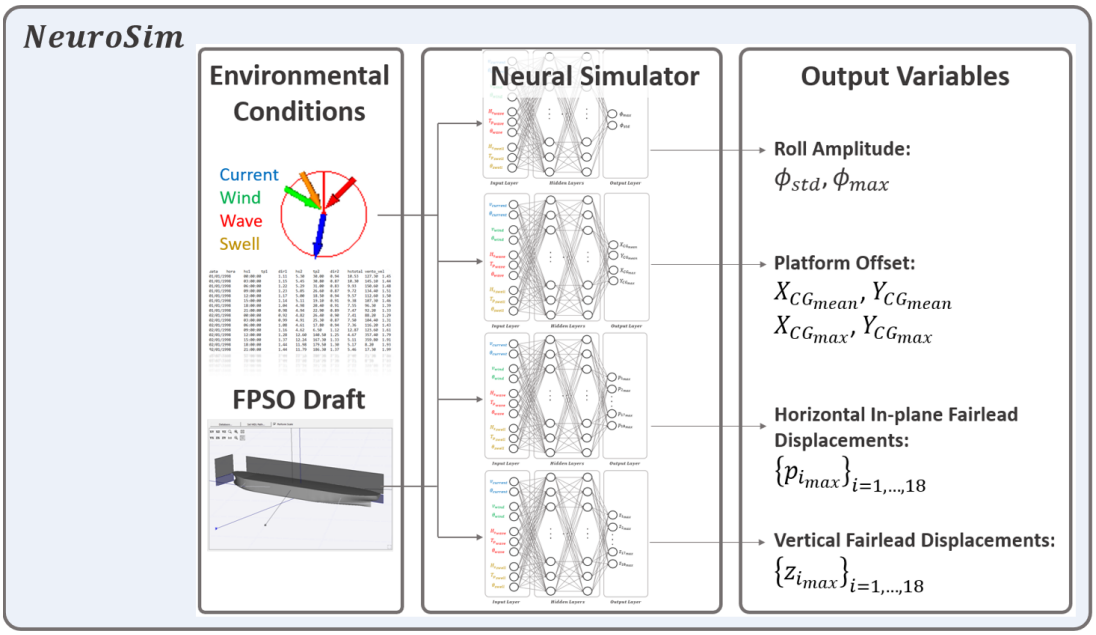


Figure 5: NeuroSim Architecture Diagram.

4.1.1 Metocean Conditions

Index	v_c (m/s)	θ_c (°)	v_w (m/s)	θ_w (°)	Hs_1 (m)	Tp_1 (s)	θ_1 (°)	Hs_2 (m)	Tp_2 (s)	θ_2 (°)
1	0.11	118.33	5.47	161.1	1.73	7.10	132.5	0.61	3.74	180.5
2	0.13	133.33	7.46	186.9	1.68	7.61	152.7	1.08	5.62	198.2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
18006	0.55	178.10	11.41	1.4	2.56	7.46	9.6	0.85	8.07	55.3
18007	0.54	178.88	10.21	3.9	2.54	7.35	9.1	0.00	0.00	0.0

Table 2: Samples of metocean conditions.

The environmental data is provided by Petrobras Oceanography Group in 3h periods from November 2003 to December 2009 at the Campos Basin. This specific duration was chosen because sea states typically last for about 3h. The data comes from a hindcast model of the area, calibrated by a large amount of measurements by wave radars, anemometers, wave buoys and current meters, that are either installed in the platforms or moored to the seabed. Metocean conditions can be described by the following variables:

1. **Current velocity**: Mean current velocity v_c (m/s).

2. **Current direction**: Current propagation angle θ_c (°).
3. **Wind velocity**: Mean wind velocity v_w (m/s).
4. **Wind direction**: Wind incidence angle θ_w (°).
5. **First Wave component height**: Significant wave height $H_{s1}(m)$ corresponding to highest energy wave.
6. **First Wave component period**: Peak Period $T_{p1}(s)$ corresponding to highest energy wave.
7. **First Wave component direction**: Incidence angle θ_1 (°) corresponding to highest energy wave.
8. **Second Wave component height**: Significant wave height $H_{s2}(m)$ corresponding to second highest energy wave.
9. **Second Wave component period**: Peak Period $T_{p2}(s)$ corresponding to second highest energy wave.
10. **Second Wave component direction**: Incidence angle θ_2 (°) corresponding to second highest energy wave.

Table 2 shows samples of the metocean data and the corresponding values of each input variable. In this work, two wave components were chosen as input variables since seastates in the Campos Basin are typically bimodal. The application of the same framework for FPSOs in different locations may be simplified to include a single wave component without significant performance loss.

4.1.2 Platform Draft

The motion response of an FPSO subject to incident metocean conditions is highly dependant on variables such as its natural frequency and moments of inertia, which themselves depend on the overall platform load. Therefore, it is crucial for data-driven models to take this information into account when estimating motion statistics. A simple approach is to include a single scalar variable in each of the models that represents overall platform load: Draft.

Platform draft is defined as the vertical distance between a the waterline and the bottom of a vessel's hull (keel) 6. By adding platform draft as the 11th input variable

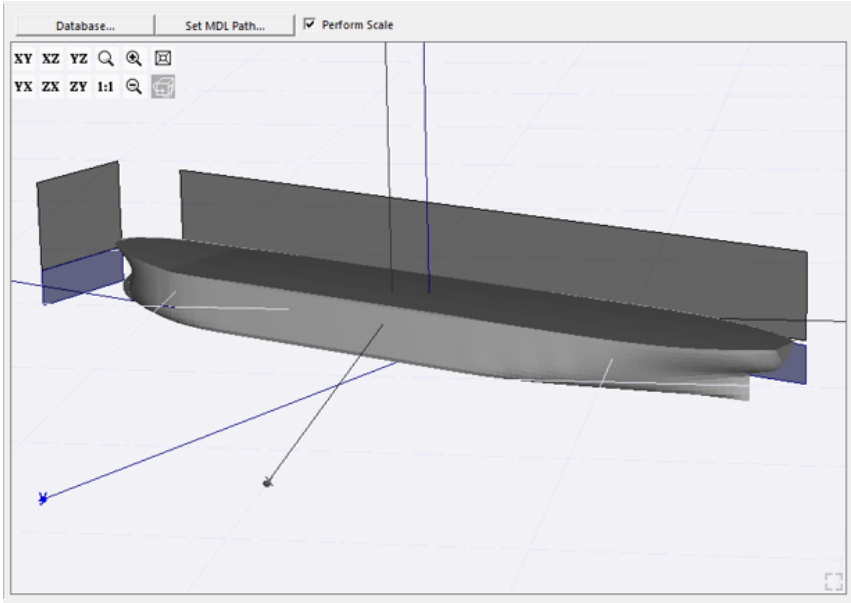


Figure 6: Representation of platform draft on simulation software Dynasim.

for each meta-model we allow the training process to generate high-level abstractions of its influence on overall FPSO motion. Alternative approaches may calculate physical characteristics such as the natural frequency and use those as inputs.

4.1.3 Proposed NeuroSim Framework

During loading and offloading operations, FPSO tank levels go through significant variations. This process result in variations to platform load, which can be measured by the draft, the vertical distance between the waterline and the bottom of the vessel's hull, also known as the keel. An FPSO's dynamic response to incident metocean conditions is highly dependent on its draft, as it's related to fundamental properties of the vessel, such as its mass, natural period and the position of its Center of Gravity (CG). As a result, data-driven predictive models need to account for the draft's influence on FPSO motion. NeuroSim is an extension of the ML framework proposed in [Cotrim et al. (2021)], which was designed to predict motion statistics of an FPSO subject to a single draft of 16m.

Figure 5 illustrates a diagram of the proposed NeuroSim's architecture: The system is comprised of a set of 4 individual neural network Meta-Models, each specialized in the prediction of different motion statistics of interest. Each model takes as input a batch of environmental conditions representative of 3h time periods and the corresponding platform drafts, these conditions are described by 10 variables (specified in section 4.1.1) and platform draft is the 11th input variable. Each specialized model then predicts the set of output variables corresponding to the relevant FPSO motion statistics: Roll, Center of

Gravity Offset and Maximum Horizontal In-Plane and Vertical Fairlead Displacements.

This modular architecture is highly scalable, as it allows for new meta-models to be added in the future in order to predict other relevant variables without compromising previously trained ones. It also provides the system with more flexibility, as different models have different hyperparameters which define their structure and help them specialize in the accurate prediction of FPSO motion for each motion statistic.

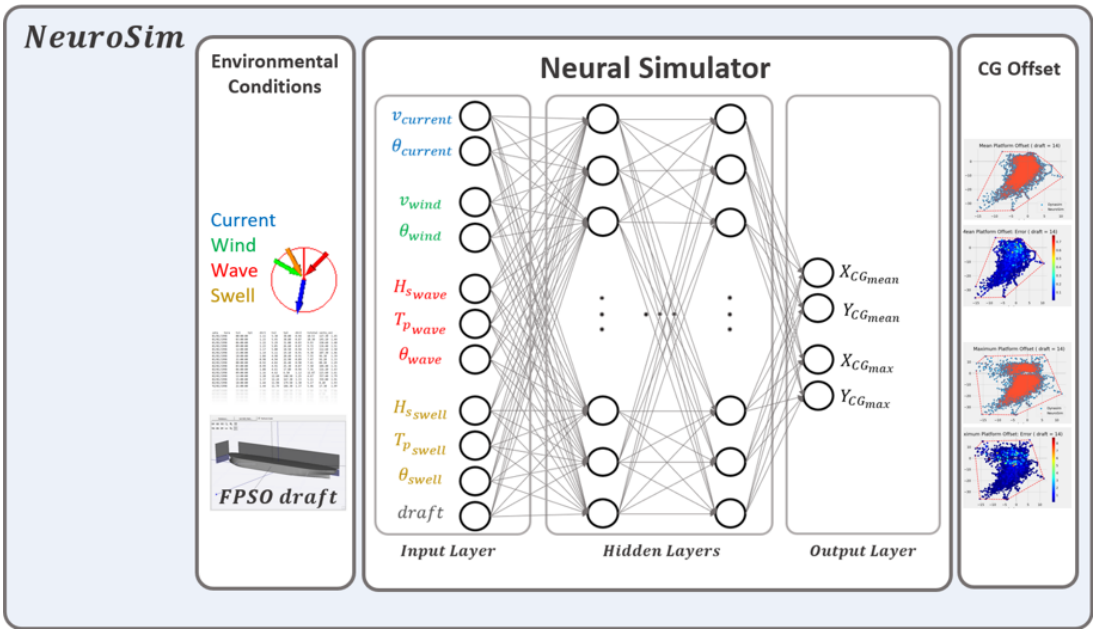


Figure 7: Diagram of NeuroSim's CG Offset meta-model.

Figure 7 details the structure of an individual meta-model, namely the one specialized in the prediction of CG Offset motion statistics given incident metocean conditions. This Multi Layer Perceptron model takes as input the same 11 variables and predicts both the mean and the maximum Center of Gravity Offset, described by the 4 output variables: $X_{CG_{mean}}$, $Y_{CG_{mean}}$, $X_{CG_{max}}$, $Y_{CG_{max}}$. The Neural Network's structure is determined through Bayesian Optimization and the synaptic weights are obtained during training. Each layer provides an increasingly higher level abstraction of the information contained in the previous one, with the fourth and final one resulting in the desired motion statistics.

It is important to notice that, while some input variables are more relevant for the prediction of specific output variables, the same environmental conditions are applied identically to all four meta-models. For instance, Maximum Roll Amplitude is more affected by incident wave height, period and direction than by wind. However, the same 10 input variables described in the Data Preparation pipeline in section 4.2 are used in all models. This is done because, during training, the Neural Network meta-models automatically learn an abstract representation of inputs that is related to this physical

behavior and providing less information to specific models because of expert domain knowledge may lead to biased or inaccurate models.

4.2 Research Workflow

This section aims to provide an overview of the methodology adopted in this work, briefly describing the different steps followed from the obtention of environmental conditions to the preparation of the models' training dataset and validation of results. Figure 8 depicts the research workflow in its six sequential pipelines:

1. Post Processing.
2. Data Preparation.
3. Hyperparameter Optimization.
4. Proposed Model Training.
5. Proposed Model Test.
6. Fine-Tuning

The NeuroSim Research workflow is subdivided into modular sequential pipelines not only as a way to better segment code in terms of functionality, resulting in easier debugging and less intrusive updates, but also to allow the execution of individual desired pipelines through the shared Kedro framework. This allows for the simultaneous execution of multiple pipelines in different experiments.

P01: Post-processor. After Dynasim simulation of the 18k metocean conditions, which took approximately 170h to complete, the Post-processor performs time-series analysis and stores the desired motion statistics.

Figure 9 illustrates 20 minutes of the roll angle time-series obtained from Dynasim simulation of environmental condition 2680. During the first seconds the resulting motion is highly dependent on initial configuration, while subsequent dynamics are governed by the incident environmental conditions. In order to isolate the effects of environmental conditions, a cutoff time t_{cutoff} of 3600s was implemented and this pipeline analyzes the remaining 6 DoF time-series to extract the following output variables:

Figure 8: Research Workflow Diagram illustrating six pipelines: Post-processing, Data Preparation, Hyperparameter Optimization, Proposed Models Training, Proposed Models Test, Fine-Tuning.

- **Roll** ϕ_{max}: Roll Standard Deviation and Maximum absolute roll angle given by:

$$\phi_{std} = \sqrt{\frac{1}{N-1}\sum(\phi_t - \overline{\phi})^2}, t > t_{cutoff}. \tag{4.1}$$

$$\phi_{max} = \max_{t>t_{cutoff}} |\phi(t) - \phi_{eq}| \tag{4.2}$$

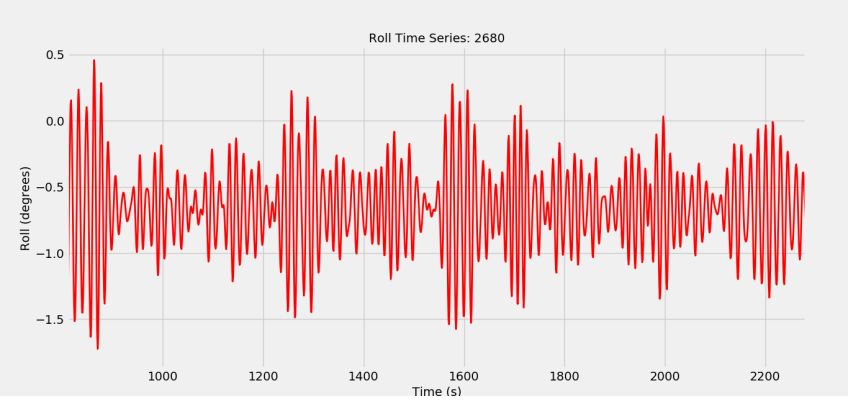


Figure 9: Roll angle time-series corresponding to simulation 2680 obtained from dynasim.

- **Platform Offset** X_{CG}, Y_{CG}: Global X and Y positions of the platform's center of gravity associated with mean and maximum observed offsets:

$$\begin{cases} X_{CG} & = X(t*) \\ Y_{CG} & = Y(t*) \end{cases} \text{where } t^* = \underset{t>t_{cutoff}}{\arg\max} \left\| (X(t), Y(t)) - (X_{eq}, Y_{eq}) \right\|_2 . \tag{4.3}$$

$$\begin{cases} X_{CG_{max}} & = \frac{1}{N} \sum X(t) \\ Y_{CG_{max}} & = \frac{1}{N} \sum Y(t) \end{cases} \text{where } t > t_{cutoff}. \tag{4.4}$$

- **Maximum Horizontal In-plane Fairlead Displacements** $p_{i_{max}}$: Maximum displacements of each fairlead in the direction parallel to the XY-projection of the corresponding mooring line during equilibrium, as illustrated in 11.

$$p_{i_{max}} = \max_{t>t_{cutoff}} p_i(t) \tag{4.5}$$

- **Maximum Vertical Fairlead Displacements** $z_{i_{max}}$: Maximum vertical displacements of each fairlead, as illustrated in 11.

$$z_{i_{max}} = \max_{t>t_{cutoff}} z_i(t). \tag{4.6}$$

The product of pipeline P01: Post-processing is a csv file for each platform draft from 8m to 21m in which environmental conditions and the corresponding four variables above that represent the resulting FPSO motion are stored. Figure 10 illustrates a section of this file for the 14m draft.

Figure 11 shows a Horizontal In-Plane, Horizontal Off-Plane, Vertical (p_i, w_i, z_i) reference frame in which Maximum Fairlead Displacements are determined. Fairlead motion in this reference frame is obtained by applying a time dependent transformation of coordinate frames to the 6 DoF $\{X(t), Y(t), Z(t), \phi(t), \theta(t), \psi(t)\}$ time-series generated by

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Draft	Combinat	Current V	Current D	Wind Velc	Wind Dire	Wave Heig	Wave Peri	Wave Dire	Swell Heig	Swell Peri	Swell Dire	Roll_std	MaxRollA	x_cg_mea	y_cg_mea	x_cg	y_cg	p1_min	p1_avg	p1_max
2	8	2	0.11	118.33	5.47	161.1	1.73	7.1	132.5	0.61	3.74	180.5	0.056395	0.234929	-1.46871	1.258309	-2.98235	4.226227	-1.219	1.390282	3.758641
3	8	3	0.13	133.33	7.46	186.9	1.68	7.61	152.7	1.08	5.62	198.2	0.039621	0.158445	-0.96895	0.906072	-1.80163	3.001655	-0.94247	1.055423	2.856228
4	8	4	0.14	155.33	9.51	200.7	1.71	7.62	204.9	1.22	7.94	76.6	0.045874	0.164393	-0.39501	0.406069	-0.06673	2.060398	-0.79444	0.475518	1.717604
5	8	5	0.15	180.6	10.55	194.7	2.2	8.23	202.9	1.23	9.44	87.1	0.14034	0.503874	-0.35697	0.789328	-0.17018	3.25818	-0.89038	0.84562	2.561089
6	8	6	0.16	201.21	11.63	189.9	2.67	8.7	199.7	1.08	10.22	83.2	0.168223	0.646219	-0.21176	1.458246	0.253699	4.067223	-0.57539	1.408204	3.379595
7	8	7	0.17	214.92	11.56	192.7	2.84	8.75	196.6	0.97	10.31	96.9	0.188728	0.749991	0.113301	1.457071	0.361104	3.966847	-0.7375	1.328178	3.335501
8	8	8	0.19	223.75	11.49	195.5	2.87	8.84	195.4	0.89	10.19	111.4	0.135407	0.55131	0.429596	1.357982	0.849219	4.124047	-0.88133	1.173873	3.349594
9	8	9	0.2	229.65	11.07	195	2.96	8.68	194.4	0	0	0	0.059456	0.221316	0.636574	1.272978	1.121156	3.931708	-1.26431	1.060858	3.033212
10	8	10	0.21	233.79	10.67	194.4	2.88	8.41	193.7	0	0	0	0.052515	0.195704	0.718083	1.14616	1.164027	3.27876	-0.85281	0.938356	2.830114
11	8	11	0.21	235.68	9.44	196.3	2.78	7.19	194.2	0	0	0	0.026635	0.119666	0.733021	0.792104	1.140625	2.965683	-1.09904	0.612085	2.349701
12	8	12	0.2	237.63	8.22	198.8	2.68	9.47	190.8	0	0	0	0.07769	0.286211	0.68001	0.69626	1.313298	3.662485	-1.38737	0.503788	2.48237
13	8	13	0.2	239.66	8.62	204.3	2.63	9.53	188.9	0	0	0	0.080087	0.296093	0.754423	0.717265	1.268206	3.291295	-1.62383	0.479369	2.483138
14	8	14	0.2	241.76	9.03	209.3	2.6	6.79	201.7	0	0	0	0.016005	0.079362	1.018762	0.484902	1.439198	1.870542	-0.91242	0.191738	1.179764
15	8	15	0.19	243.93	9.4	197.9	2.6	7.45	197.2	0	0	0	0.026705	0.110395	0.741394	0.903585	1.144204	2.934879	-0.90963	0.664732	2.194392
16	8	16	0.19	246.17	9.78	187.4	2.67	7.82	192.2	0	0	0	0.038115	0.141021	0.399872	1.42766	0.827574	3.675868	-0.67041	1.212621	3.098429
17	8	17	0.19	248.47	8.47	178.1	2.77	11.01	194	0	0	0	0.147543	0.554389	0.271399	1.453204	0.832114	4.206635	-0.68156	1.233798	3.280099
18	8	18	0.18	250.82	7.17	165.5	2.83	11.51	192.4	0	0	0	0.172056	0.624993	0.029665	1.406892	0.234045	4.28632	-0.52817	1.219977	3.377897
19	8	19	0.19	246.04	6.68	171.4	2.87	11.84	189.5	0	0	0	0.185543	0.665994	0.224168	1.107939	0.534858	4.379568	-1.10681	0.94318	3.202525
20	8	20	0.2	241.47	6.21	178.3	2.92	12.18	186.8	0	0	0	0.203327	0.737039	0.399326	0.796386	0.47675	4.226338	-1.36871	0.663493	3.156535
21	8	21	0.21	237.16	6.77	180.8	2.94	12.41	185.7	0	0	0	0.214965	0.775545	0.431057	0.740205	0.820503	3.855506	-1.66366	0.632134	3.067386
22	8	22	0.21	233.13	7.35	183	2.92	12.49	184.9	0	0	0	0.217921	0.785594	0.386623	0.743786	0.238739	3.328306	-1.18781	0.659164	2.659804

Figure 10: Part of the Dataset.csv file which stores metocean conditions and resulting motion statistics after Pipeline P01 has been executed. Column B contains platform draft, C-L store incident metocean conditions and the remaining columns store the corresponding motion statistics.

Dynasim simulation at each time step.

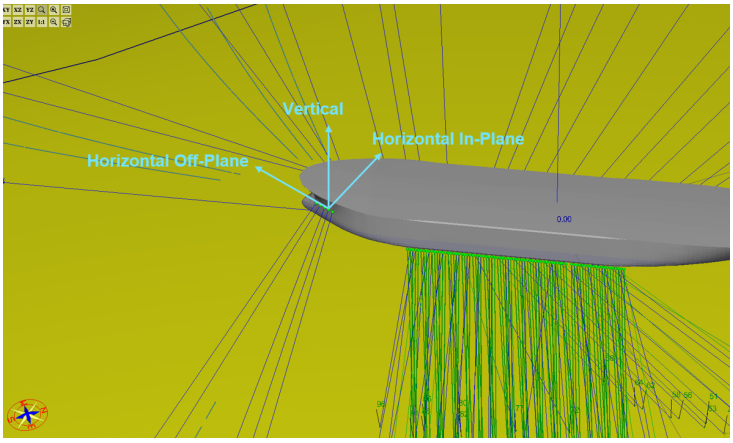


Figure 11: Visualization of a local $\{p_i, w_i, z_i\}$ fairlead reference frame.

P02: Data Preparation. In order to improve the numerical convergence of learning algorithms, several data preparation techniques are applied to the data prior to training. Let $\mathbf{e} = (v_c, \theta_c, v_w, \theta_w, H_{s1}, T_{p1}, \theta_1, H_{s2}, T_{p2}, \theta_2)$ denote a metocean condition as described in section 4.1.1. As angular variables are defined in $[0°, 360°]$, their periodic property implies that values such as $0.1°$ and $359.9°$ are functionally close despite being numerically distant. This can cause slow convergence of the ANN as similar seastates may be far apart in the network input space. As a result, the projections of current velocity, wind velocity and wave height in the N-S and E-W directions were used, rather than their magnitude and

incidence angle, so that the same seastate can be represented as:

$$\begin{aligned}\mathbf{e^{proj}} = (&v_c \sin(\theta_c), v_c \cos(\theta_c),\\ &v_w \sin(\theta_w), v_w \cos(\theta_w),\\ &H_{s1} \sin(\theta_1), H_{s1} \cos(\theta_1), T_{p1},\\ &H_{s2} \sin(\theta_2), H_{s2} \cos(\theta_2), T_{p2}).\end{aligned}$$

This process, known as projection of environmental conditions, is illustrated in fig. 12. Since the values of different input variables have different orders of magnitude e.g., local wind velocity can be as high as 20 m/s while current velocity is lower than 1 m/s, a Gaussian Standardization method was applied to transform the projected environmental conditions into the actual model's input data:

$$x_i = \frac{e_i^{proj} - \mu_i}{\sigma_i}, \ i = 1, ..., 10,$$

where μ_i and σ_i are the mean and standard deviation of the i-th variable on the complete dataset. This scales input variables and improves numerical convergence.

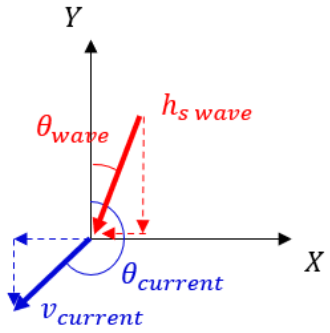


Figure 12: Visualization of Environmental Condition Projection for incident current (blue) and wave (red).

P03: Hyperparameter Optimization. The third pipeline is responsible for determining the optimal Neural Network Architectures for each meta-model and each FPSO draft through a Hyperparameter Optimization algorithm known as Bayesian Optimization (BO), described in section 4.3. In this research, BO was applied to determine the optimal number of neurons in each of the three NN hidden layers, while remaining hyperparameters such as activation function, optimizer, batch size and number of training epochs were kept fixed at appropriate values found through trial and error and expert knowledge [Cotrim et al. (2021)].

P04: Proposed Models Training.

Once the optimal model architectures, which are defined by the proposed set of hyperparameters, have been determined an extensive training with 90% of the entire dataset is performed and the trained models are saved for testing in the following pipeline. During this training, the Train+Validation dataset was used and proposed neural networks described in table 5 were trained for 5000 epochs as opposed to the 200 epochs during Hyperparameter Optimization.

P05: Proposed Models Test.

Finally, the proposed models for each desired output variable and each FPSO draft are tested in 10% of the original dataset, which is comprised exclusively of unseen metocean conditions. This ensures the resulting metrics are representative of the proposed models' generalization performance when applied to realistic scenarios where obtained environmental conditions were not used during training and validation.

The main objective of this pipeline is to generate analytic results in order to verify NeuroSim's performance, comparing error metrics with those of traditional simulation methods and validating the proposed approach for a data-driven FPSO dynamics simulator. These results visualized both numerically and graphically, as discussed in Chapter 6.

P06: Fine-Tuning.

After the proposed models were trained and tested on Dynasim simulated data, they were tested on real, measured FPSO motion and metocean conditions from the period of 2007-2014. While the models' predictions followed dynasim motion statistics closely, there was a discrepancy when compared to the real motion. In order to improve NeuroSim's performance on real data for real-time FPSO monitoring applications, each meta-model was fine-tuned using real measured motion data. The fine-tuning process consisted in further training the models on the new data with a reduced learning rate, in order to keep the information gained during the previous training process while improving error metrics with real data.

4.3 Neural Architecture Search

This section aims to detail the Hyperparameter Optimization algorithm implemented in pipeline P03. Initially, a preliminary study indicated appropriate values of hyperparameters such as activation function, batch size and number of training epochs [Cotrim et

al. (2021)], showing that an optimization of the ANN architecture (NAS) presented the highest potential for model performance improvement. As a result, subsequent research focused on NAS methods in order to determine the optimal number of neurons in each hidden layer of the final ANN models [Suller et al. (2021)], where the following methods were compared for the CG Offset meta-model:

- Random Search (RS)
- Simulated Annealing (SA)
- Bayesian Optimization (BO)

Figure 13 illustrates a comparison of the three investigated NAS methods in [Suller et al. (2021)], showing BO and RS performed a more comprehensive search of the proposed search space, while SA's initial architecture tends to impact evaluated architectures over its optimization trials. Overall, BO yielded the best models and was ultimately chosen as the NAS method for this work.

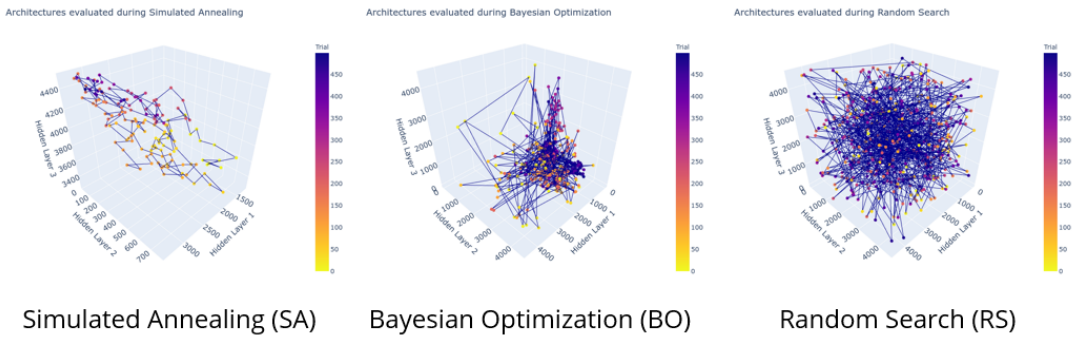


Figure 13: Visualization of evaluated ANN architectures over optimization trials (colorbar) for three different NAS methods: SA (left), BO (center) and RS (right). The 3 axis correspond to the number of neurons in each ANN Hidden Layer.

4.4 Error Metrics

The definition of precise error metrics allows for parameter tuning during training, the numerical comparison of different ML models during validation and indicates expected error ranges for the final models during testing. As presented in section 4.3, error metrics are also used to define the Objective Function used in BO. As a result, despite the insights provided by visualization tools, error metrics remain of paramount importance and must be precisely defined. In this work, meta-model performance is evaluated by the following error metrics:

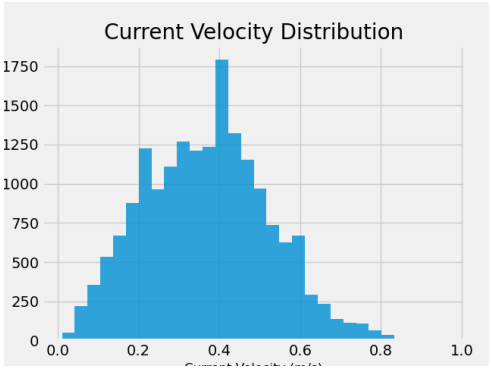
- **MSE**: The Mean Squared Error is obtained by averaging the squared differences between model predictions ($\hat{y}_i$) and true values (y_i): $MSE = \frac{1}{N_D} \sum_{i=1}^{N_D} (y_i - \hat{y}_i)^2$
- **RMSE**: The Root Mean Squared Deviation is analogous to the standard deviation and is obtained by taking the square root of the Mean Squared Error. It has the same unit as the output variable: $RMSE = \sqrt{MSE}$
- **MAE**: The Mean Absolute Error is the average of the absolute differences between true and predicted values and has the same unit as the output variable: $MAE = \frac{1}{N_D} \sum_{i=1}^{N_D} |y_i - \hat{y}_i|$
- **Max Error**: Overall maximum error observed over test set, highly sensitive to outliers:
  $MaxError = \max_{i=1,\dots,N_D} |y_i - \hat{y}_i|$.

5 EXPERIMENTS

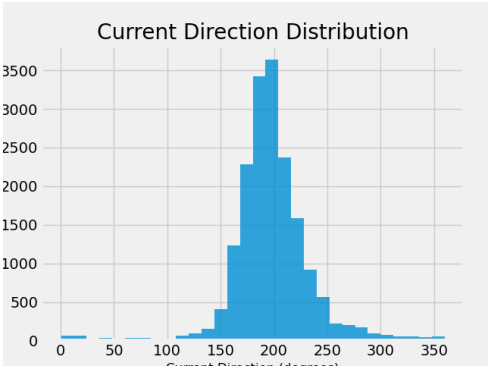
The main objective of this Chapter is to describe the experiments performed according to the Methodology illustrated in section 4.2. Firstly, an exploratory analysis of the environmental data is conducted (section 5.1) in order to verify cohesion of metocean variables in relation to expected typical values. This analysis is crucial in response-based methodologies, as the distributions of metocean conditions depends on local typical sea states and divergences may lead to unbalanced datasets, which could produce models not fit for prediction of FPSO motion in these regions. Then, section 5.2 details the dataset generation process performed in pipelines P01 and P02, specifying the implemented train-validation-test split. Section 5.3 illustrates the results of pipeline P03, namely the optimal ANN architectures for each of the 4 meta-models, section 5.4 presents the results of Pipeline P04 graphically. Finally, section 5.5 contains a comparison of results obtained by two different overall architectures of NeuroSim, which was used in order to determine which architecture would be implemented moving forward.

5.1 Exploratory Analysis of Environmental Data

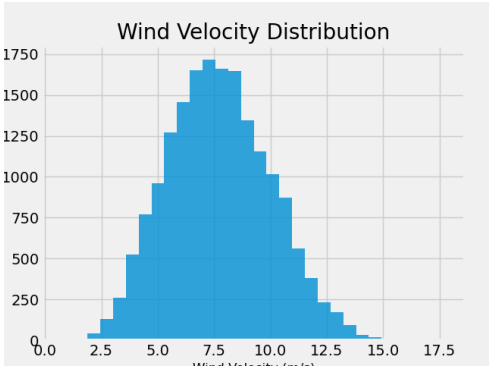
As described in 4.1.1, environmental conditions were obtained by hindcast models in 3h periods from November 2003 to December 2009 at the Campos Basin, which results in 18007 conditions. With the objective of verifying the coherence of obtained data prior to costly simulations or model training, the frequency distributions of the main metocean variables were plotted in histograms (Fig. 14). It is possible to notice that, across all environmental conditions, the range of values observed for each variable is within expectation, with higher occurrences being compatible with typical sea states at the Campos Basin. Characteristics such as the concentration of the current direction histogram or the small number of incident waves in the $200° - 350°$ range are coherent with the FPSO's geographical location, as this range corresponds to waves coming from the coast.



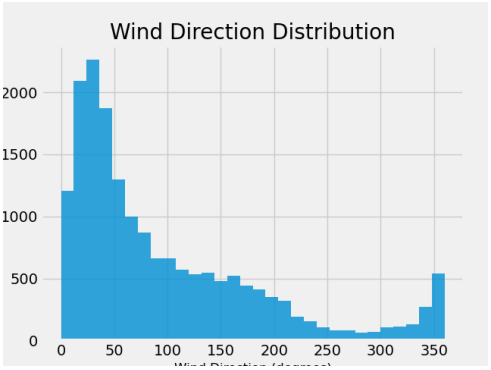
(a) Current Velocity (m/s)



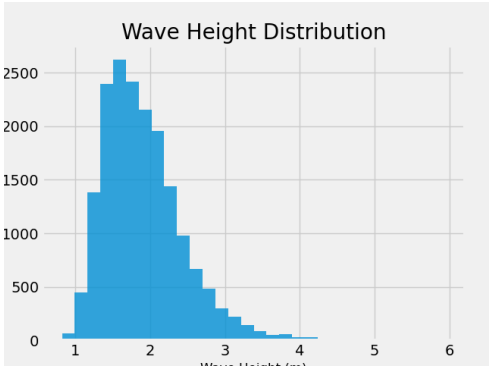
(b) Current Direction (°)



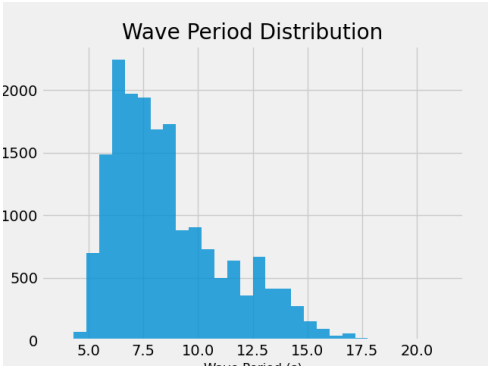
(c) Wind Velocity (m/s)



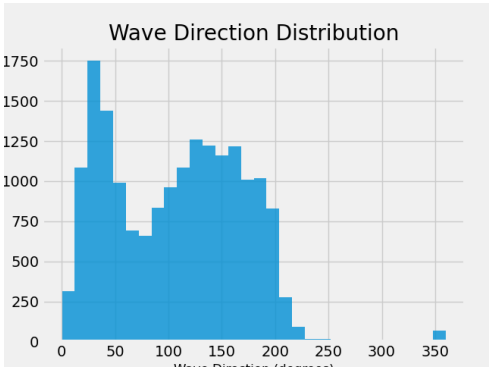
(d) Wind Direction (°)



(e) Wave Height (m)



(f) Wave Period (s)



(g) Wave Direction (°)

Figure 14: Histograms illustrating the distribution of current velocity (a), current direction (b), wind velocity (c), wind direction (d), significant wave height (e), peak wave period (f), wave direction (g).

5.2 Dataset Generation

Following the exploratory analysis of environmental data, the motion of a spread-moored FPSO with 18 mooring lines was simulated in Dynasim subject to each metocean condition and for the 14 draft values between $8m$ and $21m$. The generated time-series were then processed according to the process described in section 4.2 for pipeline P01 and for each metocean condition the corresponding 4 output variables of interest were extracted: Maximum Roll Angle and Standard Deviation of Roll motion, Mean and Maximum CG Offset and Maximum Horizontal In-Plane and Vertical Fairlead Displacements.

Prior to constituting the meta-models Dataset, the output variables were plotted as a function of the most relevant environmental variables in order to check for potential errors in the simulation process and verify cohesion in terms of the underlying physical phenomena. Figures 15 and 16 illustrate, for all metocean conditions, the maximum observed roll angle and CG offset respectively. It is possible to notice that Roll Amplitude tends to increase as Wave height increases and as its period approaches the FPSO natural period, while the two peaks in the direction plot correspond to perpendicularly incident waves relative to the platform's heading of 210°. In the CG Offset plot we notice higher offset values are observed in the third quadrant (South-East), which is compatible with typical metocean conditions (section 5.1).

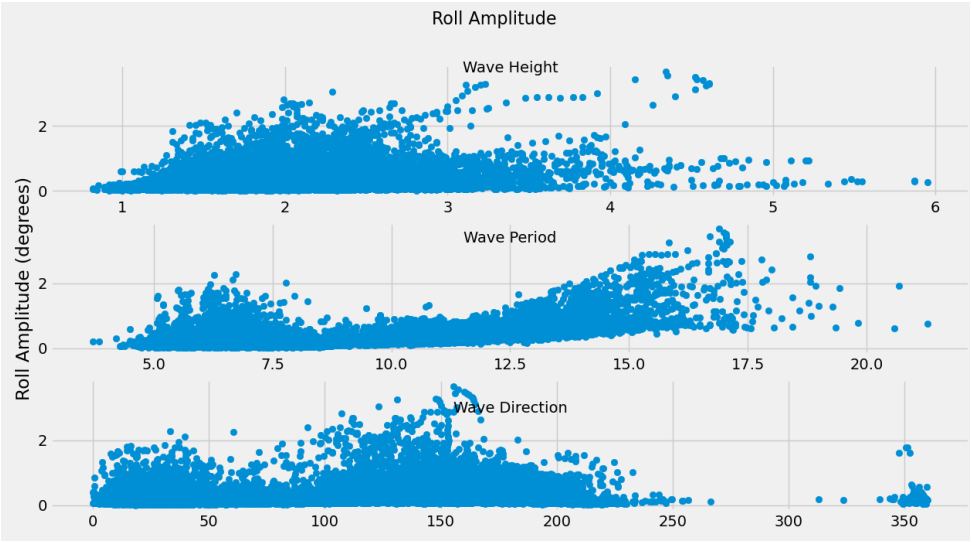


Figure 15: Visualization of Maximum Roll values as a function of Wave Height (top), Wave Period (middle) and Wave Direction (bottom).

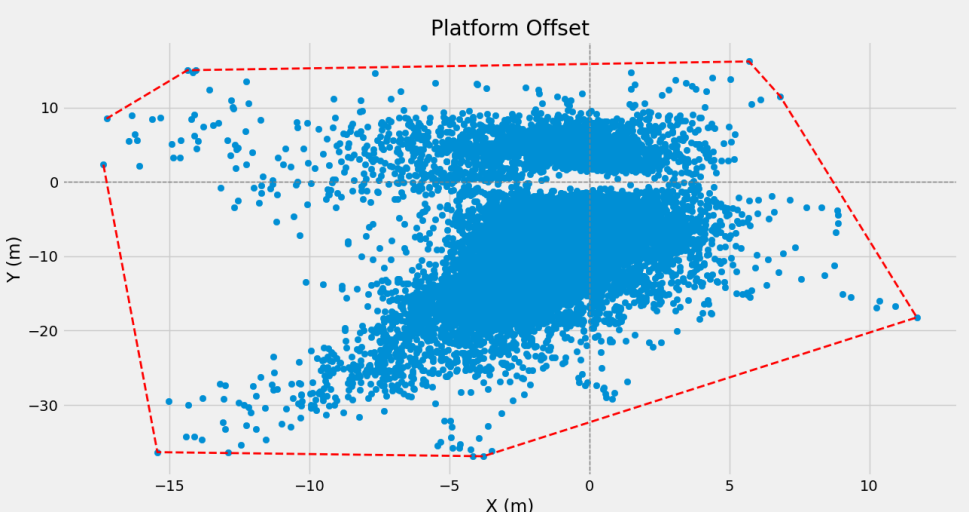


Figure 16: Visualization of Maximum CG Offset values for all 18k environmental conditions in global X (East-West) and Y (North-South) axis.

5.2.1 Cross-Validation Split

After the extraction of output variables and preliminary analysis of their values, metocean conditions are projected and normalized according to the process described in pipeline P02 (section 4.2), this results in a dataset with over 252k samples, associated with 18007 metocean conditions for different integer draft values from 8m to 21m and the corresponding motion statistics of interest obtained through dynamic simulation. In order to appropriately evaluate the performance of trained models, this dataset was split into a Training+Validation set (used for 5-fold Cross-Validation) and a separate Test set used exclusively to assess the performance of the final models (section 5.4). Figure 17 illustrates the implemented dataset split, indicating the number of samples in each set.

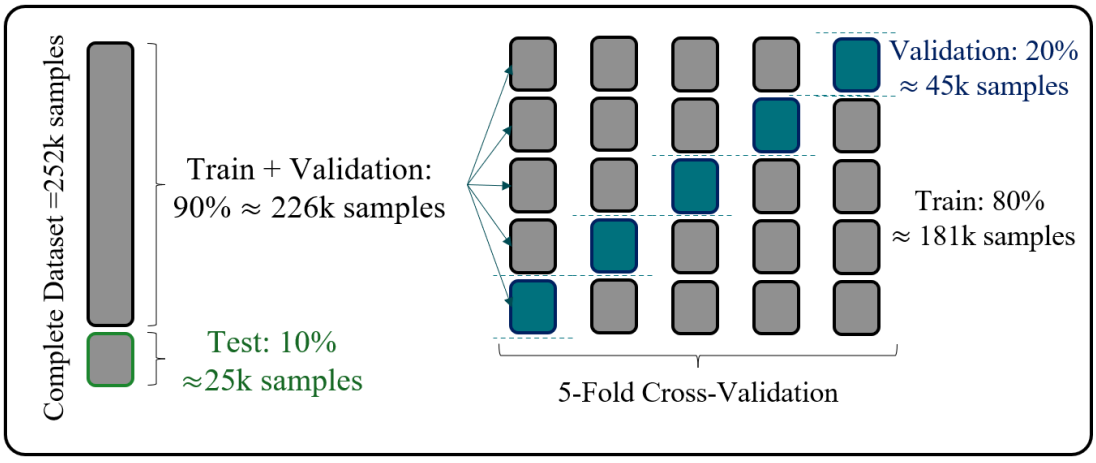


Figure 17: Diagram of Dataset split for 5-fold Cross-Validation.

5.3 Bayesian Optimization

After processing the inputs through projection and standardization and splitting the Dataset into Training, Validation and Test sets, NAS methods were used to determine the optimal meta-model architectures for each output variable. As a result, the four ANN architectures associated with the four different meta-models were optimized in Pipeline P03.

Initially, a simple Grid Search (GS) algorithm was used, in which a reduced number of candidate architectures are trained and the model with minimum Cross-Validation MSE was chosen [Cotrim et al. (2021)]. This approach, however, is extremely time-consuming and evaluates only a small number of architectures. Then, Simulated Annealing (SA) was implemented in order to encourage exploration of new architectures during the initial trials and convergence to optimal regions of the search space. SA presented better results, exploring better quality ANN architectures overall, but with highly correlated to the initial random architecture (Fig. 13).

Finally, Bayesian Optimization (BO) was executed through the python Optuna library and solved the issue of exploring only ANN architectures close to the initial random model, as the first 10 trials are randomly generated and subsequent architectures explored are not necessarily close to each other, but correspond to regions of the search space in which the Expected Improvement (EI) is maximized, given the current approximation of the objective function (section 4.3). It is possible to notice that ANN architectures vary considerably with meta-model output variable.

Since BO showed better results, yielding ANN architectures that presented smaller Cross-Validation MSEs in reduced computational time [Suller et al. (2021)], it was chosen as the main NAS method for pipeline P03 in this work. Table 3 summarizes parameter values used for Bayesian Optimization. An initial analysis of the model's training curves indicated that 200 training epochs during HO is an appropriate value, providing representative measurements of a model's performance without leading to unfeasible training times. Hyperparameters such as batch size, optimizer and activation functions were found in [Cotrim et al. (2021)]. After 500 BO trials, optimal ANN architectures were found for each model, as illustrated in table 5. It is important to note that BO is a general hyperparameter optimization algorithm and can be used to optimize not only the numbers of neurons in each layer (ANN architecture), but also the activation functions, batch size, optimizer, learning rate as well as any other hyperparameter. In this work only the number of neurons in each hidden layer was subject to optimization due to time

Hyperparameter	Value
Number of Optimization Trials	500
Objective Function	Average Cross-Validation MSE
Number of Hidden Layers	3
Number of Training Epochs	200
Batch Size	800
Optimizer	Adam
Learning Rate	0.001
Activation Function	ReLU*
HL 1	Optimized through BO
HL 2	Optimized through BO
HL 3	Optimized through BO

Table 3: Summary of Hyperparameters used in Bayesian Optimization. HL 1-3 denote the number of neurons in Hidden Layers 1-3
. *: Horizontal In-Plane FD meta-model exceptionally uses a sigmoid Activation Function instead of ReLU due to better observed results [Cotrim et al. (2021), Cotrim et al. (2021)].

constraints.

The results from Pipeline P03 are the summaries of the 500 BO trials executed for each meta-model, with the studied architectures and the corresponding objective function values (Table. 4). The optimal architecture of each meta-model is determined by the set of hyperparameters associated with the model with best performance across all trials, in other words, the model with smallest Cross Validation MSE.

trial number	Avg. MSE	HL 1	HL 2	HL 3
47	0.690073712	2540	3819	1235
98	0.695419056	2371	3764	1611
73	0.698096976	1924	3102	3787
70	0.701736770	1641	3529	3781
82	0.704541140	1950	2976	1310
78	0.707276533	1898	3546	1511
99	0.707439355	1235	3805	1930
26	0.708902534	2640	2941	2094
13	0.710539064	3272	3243	2287
33	0.712738303	2688	3141	1578

Table 4: Samples of the 10 best Bayesian Optimization Trials for the CG Offset meta-model. HL 1 - HL 3 denote the number of neurons on Hidden Layers 1 to 3 respectively and the Avg. MSE is the objective functions minimized during Bayesian Optimization.

Table 5: Optimal ANN architectures for each meta-model obtained in Pipeline P03 through Bayesian Optimization

Meta-Model	HL 1	HL 2	HL 3	N. Parameters	Optimization Time
Roll	1786	333	1099	986 thousand	106h
CG Offset	2540	3819	1235	14.4 million	171h
Horizontal In-Plane FD	1986	4430	1605	15.9 million	204h
Vertical FD	649	907	3608	3.9 million	153h

5.4 Proposed Models Test

Once the optimal ANN architectures were found, the corresponding models are trained with the Train+Validation dataset (Fig. 17) for 5000 training epochs and their performance is evaluated using the proposed metrics (section 4.4) applied to the Test dataset, comprised of unseen metocean conditions.

5.4.1 Roll

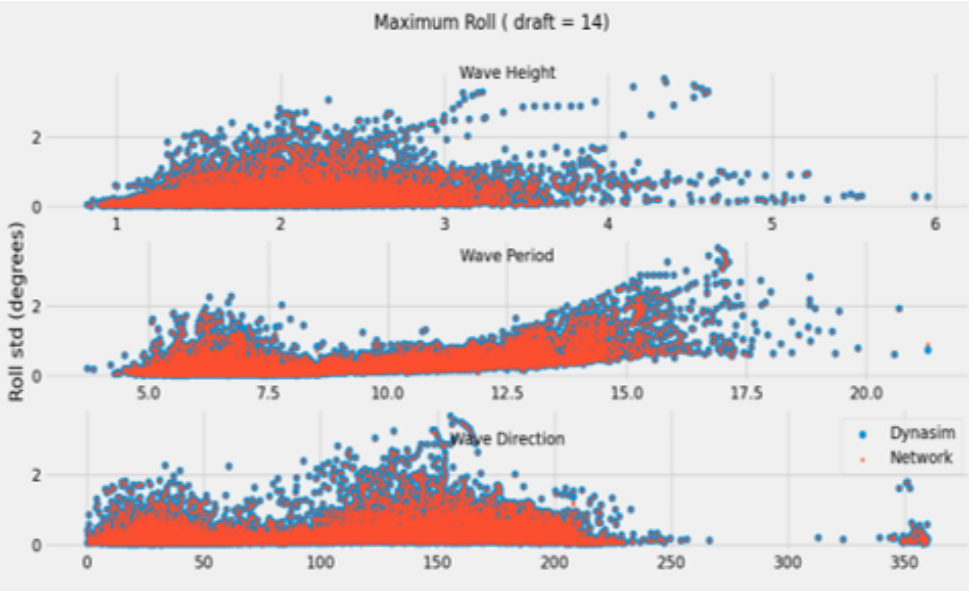
The Roll meta-model's performance is illustrated in figure 18a, which presents three scatter plots of maximum Roll as a function of wave height, period and direction respectively. In each plot, individual points represent metocean conditions, values predicted by Dynasim are shown in blue while NeuroSim predictions are shown in orange.

In order to better visualize error regions as a function of the same set of input variables, figure 18b illustrates a colormap of the associated absolute error between Dynasim and NeuroSim. In this plot, red indicates higher errors (higher than 0.20°) while blue indicates prediction errors in the range of 0° to 0.1°. An analysis of NeuroSim's error plots for the 14m draft Roll Amplitude meta-model leads to two immediate conclusions:

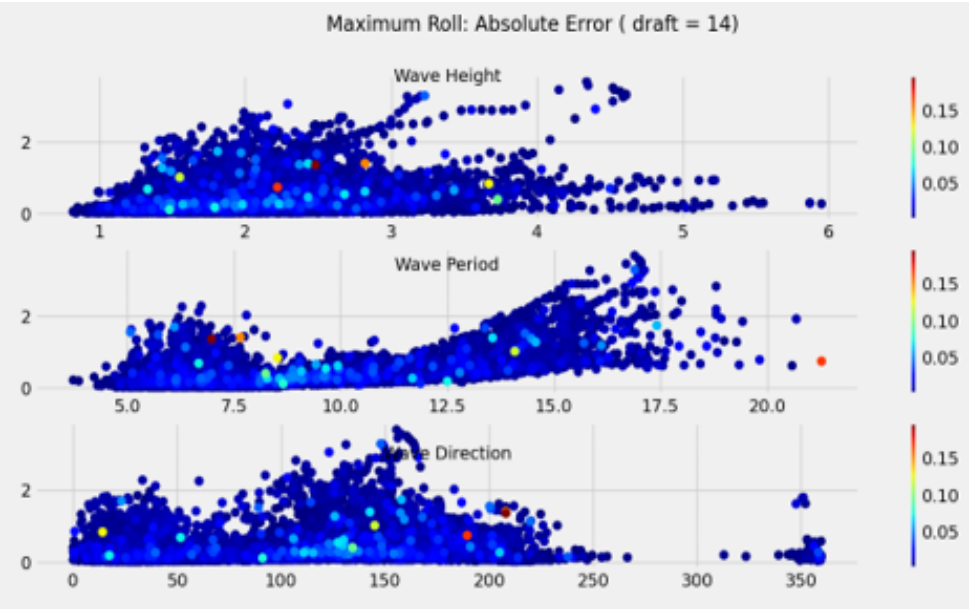
- Good performance on critical cases: All metocean conditions which resulted in maximum platform roll of over 2.0° were accurately predicted with NeuroSim. This is a positive result, as extremely accurate predictions of roll motion for calm conditions are not as relevant in mooring system design.

- Localized error spikes: NeuroSim's Roll Amplitude meta-model's highest errors were observed on an extremely limited number of incident metocean conditions. In other words, there is a large gap between the first and second highest errors. The condi-

tion which presented highest error corresponds to a 2.5m high frequency wave with incidence direction parallel to the platform's main axis. In this case, NeuroSim's predicted roll was higher than the actual maximum roll measured by Dynasim. This result is preferable to the alternative of having multiple conditions with poor prediction performance, as higher errors tend to be extremely uncommon.



(a) Comparison between NeuroSim meta-model (orange) and Dynasim values (blue) for maximum FPSO roll at 14m draft.

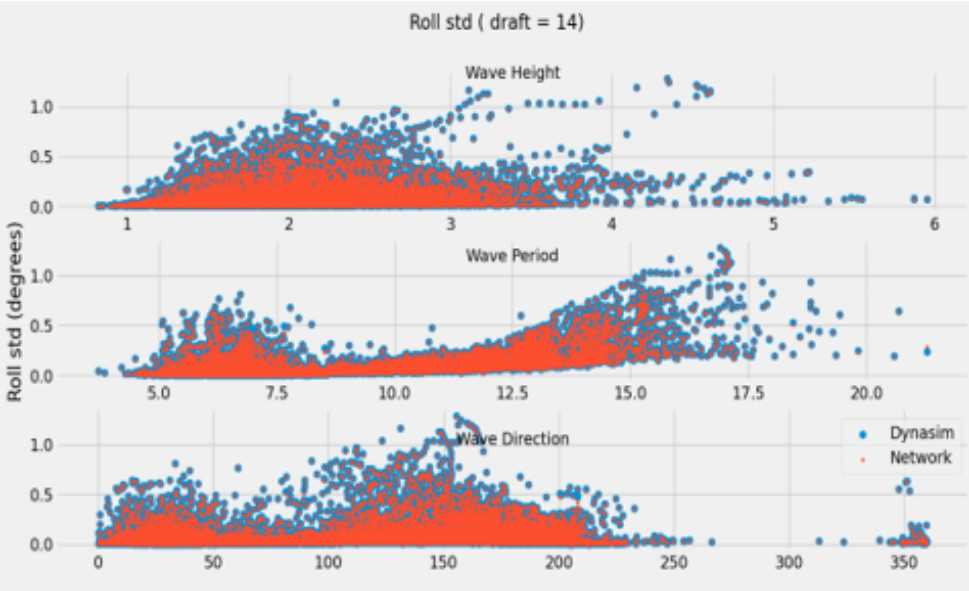


(b) Roll std error plot.

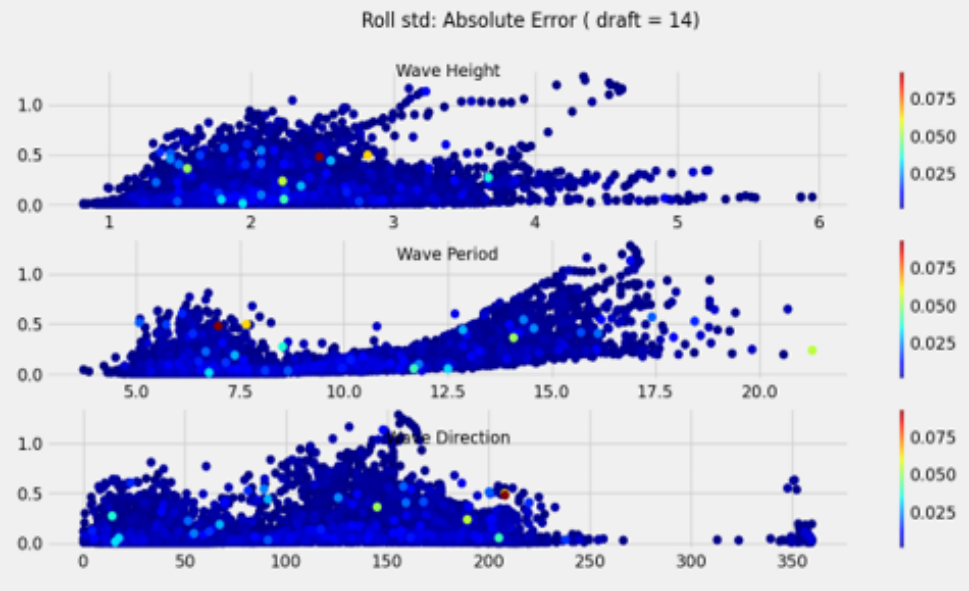
Figure 18: Roll std results as a function of Wave Height (top), Wave Period (middle) and Wave Direction (bottom) for 14m FPSO draft.

Analogously, figure 19 depicts the same results for the standard deviation of the roll motion, given by the same neural network meta-model. This variable's overall behaviour

given incident metocean conditions is extremely similar to the maximum absolute roll angle, scaled roughly by a factor of 4.



(a) Comparison between NeuroSim meta-model (orange) and Dynasim values (blue) for maximum FPSO roll at 14m draft.



(b) Max roll error plot.

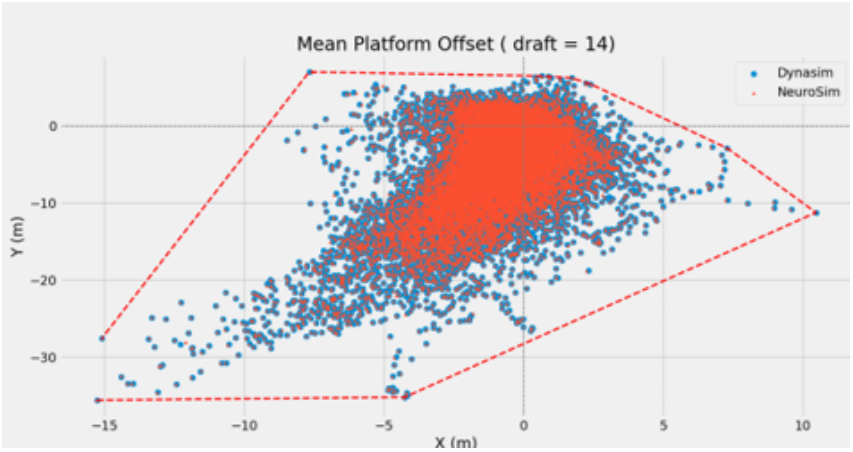
Figure 19: Max roll results as a function of Wave Height (top), Wave Period (middle) and Wave Direction (bottom) for 14m FPSO draft.

5.4.2 Platform Offset

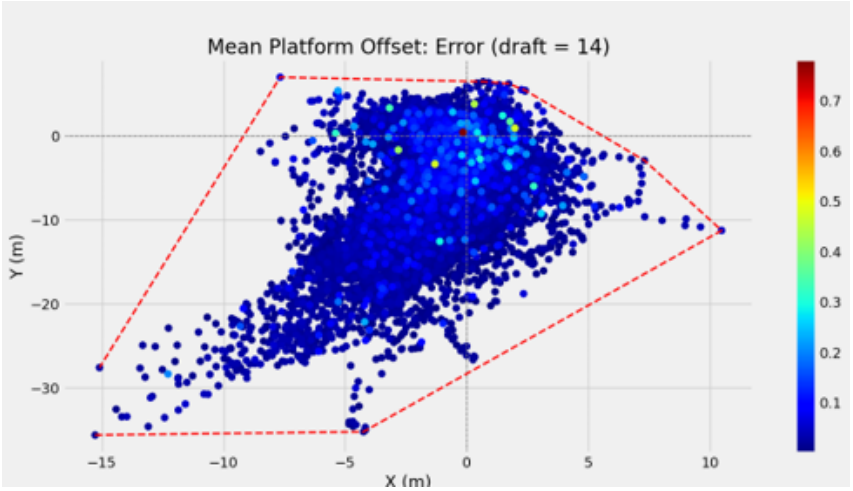
Similarly to the first meta-model, NeuroSim's CG Offset prediction results for an FPSO draft of 14m are plotted in figures 20 and 21. Figures 20a and 21a illustrate Dynasim and NeuroSim predicted values for the X and Y position of the mean and

maximum center of gravity offset while figures 20b and 21b display a colormap of the prediction error. The convex hull that encapsulates all offsets across the entire dataset is represented in both figures as the dotted red line.

NeuroSim's CG Offset meta-model presented relatively low absolute errors in cases where the offset was larger, which tend to occur when incident currents present higher velocity. This is a positive result similar to the Roll Amplitude meta-model, where accurate predictions occur on critical cases. One possible explanation for this behavior is that the neural network models find it more challenging to correctly predict the desired variables in calm sea states as similar environmental conditions may lead to more varied dynamic responses, while critical conditions may have a narrower range of responses. This hypothesis is supported by Fig.21b where points closer to the origin tend to display larger absolute errors ($> 3m$).

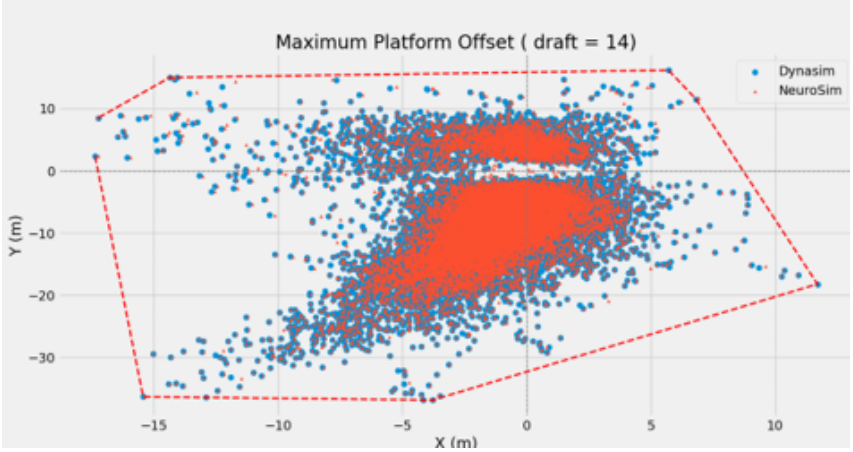


(a) Comparison between NeuroSim meta-model (orange) and Dynasim values (blue) for mean CG Offset at 14m draft.

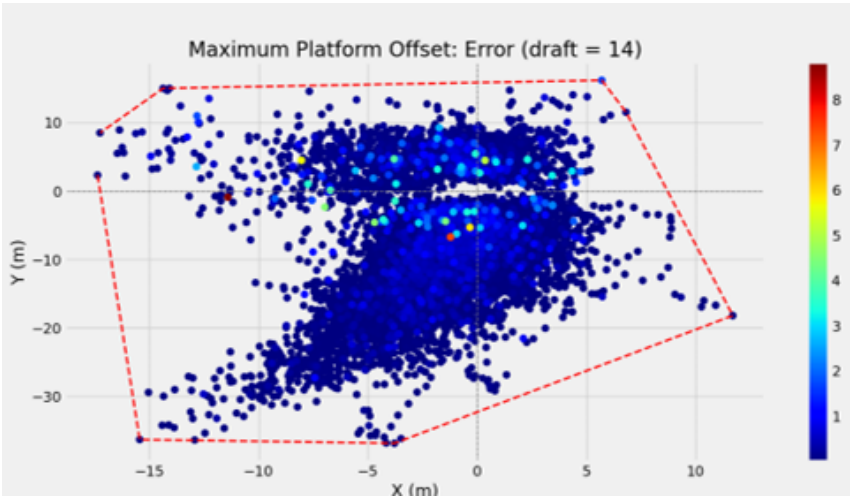


(b) NeuroSim CG Offset meta-model error plot.

Figure 20: Mean offset meta-model results at global XY reference frame for 14m FPSO draft.



(a) Comparison between NeuroSim meta-model (orange) and Dynasim values (blue) for maximum CG Offset at 14m draft.



(b) NeuroSim CG Offset meta-model error plot.

Figure 21: Max offset meta-model results at global XY reference frame for 14m FPSO draft.

Overall, NeuroSim's CG Offset meta-model's error is within acceptable range for almost all of the metocean conditions tested. Error spikes of 8m are observed for only two conditions in less densely populated areas of the plot, which might indicate the model's error is due to insufficient similar training samples and resulting generalization issues in these regions.

5.4.3 Maximum Horizontal In-Plane Fairlead Displacements

The last two meta-models, Maximum Horizontal In-Plane and Vertical Fairlead Displacements, have output variables of higher dimension than the previous two. Both outputs are 18-dimensional, as they contain numerical values for the maximum displacements of each of the 18 fairleads in the corresponding local reference frames. This leads to vi-

sualization issues when trying to plot NeuroSim's performance in a similar fashion to previous meta-models. As a result, the Horizontal In-Plane Fairlead Displacement results are plotted in terms of the average value among mooring lines in a given group. The 18 mooring lines can be divided into four different groups (Fig. 22):

- Group 1: Lines 1-5 (bow-port)
- Group 2: Lines 6-9 (stern-port)
- Group 3: lines 10-13 (stern-starboard)
- Group 4: Lines 14-18 (bow-starboard)

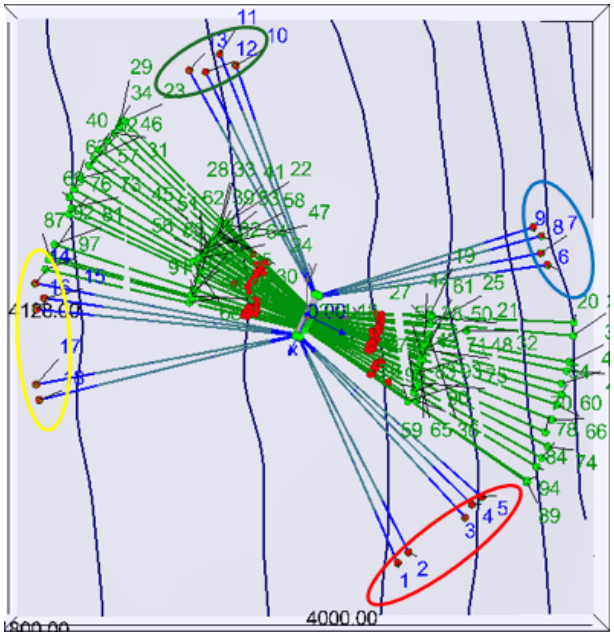
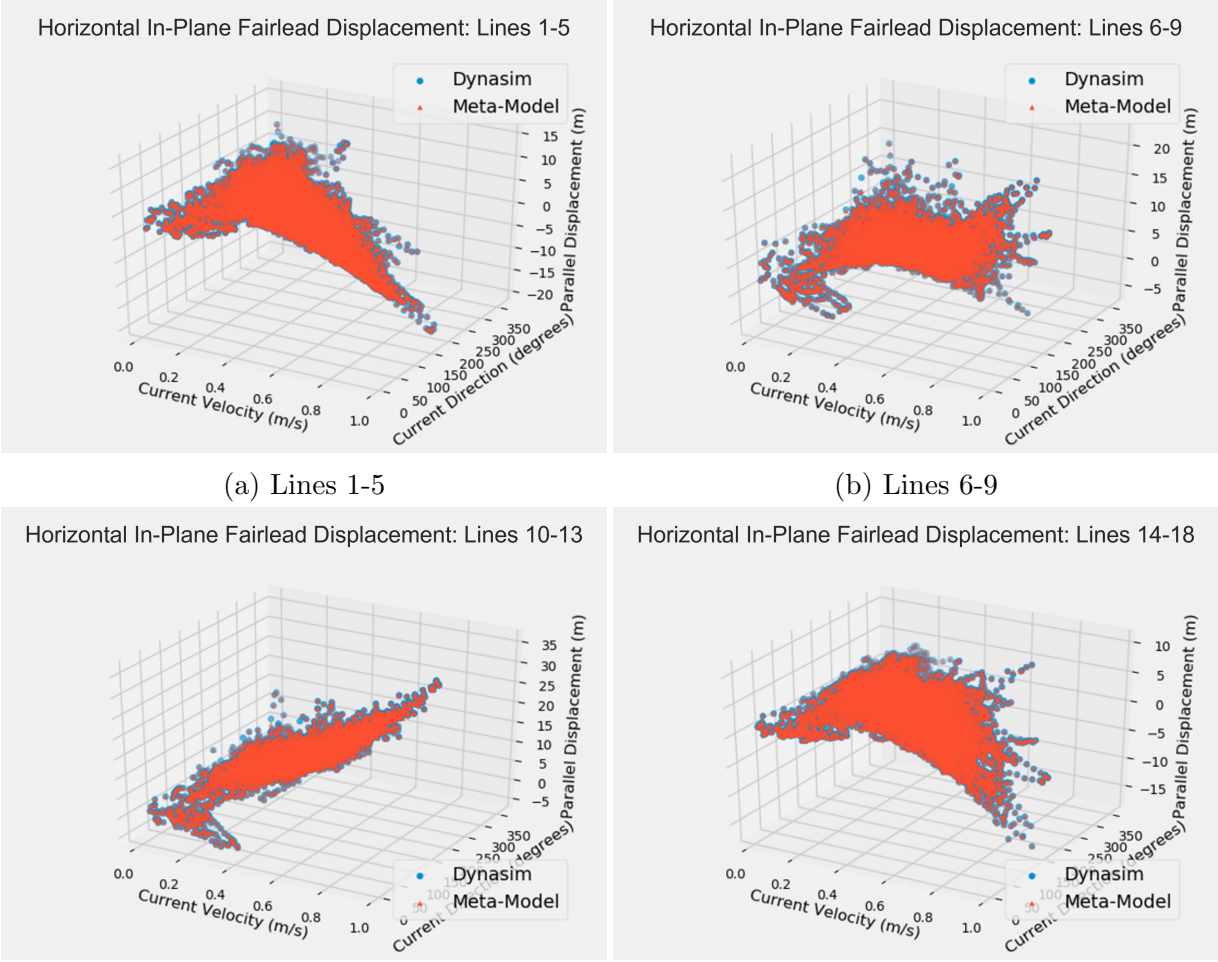


Figure 22: P50 Line groups illustrated in Dynasim interface. Group 1 (red), group 2 (blue), group 3 (green) and group (yellow).

Figure 23 contains four 3D scatter plots of the group-average maximum Horizontal In-Plane Fairlead Displacements obtained through Dynasim simulation (blue) and predicted by NeuroSim (orange). The output variable is plotted as a function of the most relevant inputs, current velocity and direction, for easier physical interpretation of results. It is worth noting that taking the average maximum displacements among line groups is a valid approach for visualization as the output variable presents small variations inside each group.

As expected, the typical sea states in the Campos Basin and the resulting FPSO dynamic motion is reflected in the overall behavior of the Horizontal In-Plane Fairlead



(a) Lines 1-5

(b) Lines 6-9

(c) Lines 10-13

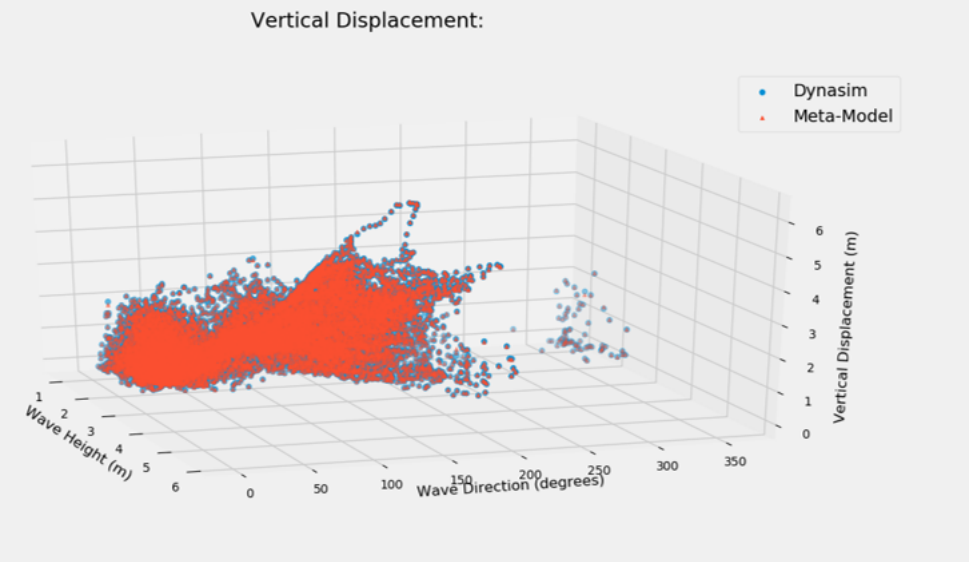
(d) Lines 14-18

Figure 23: Maximum Horizontal In-Plane Fairlead Displacement meta-model results for each line group as a function of Current Velocity and Direction for 14m FPSO draft.

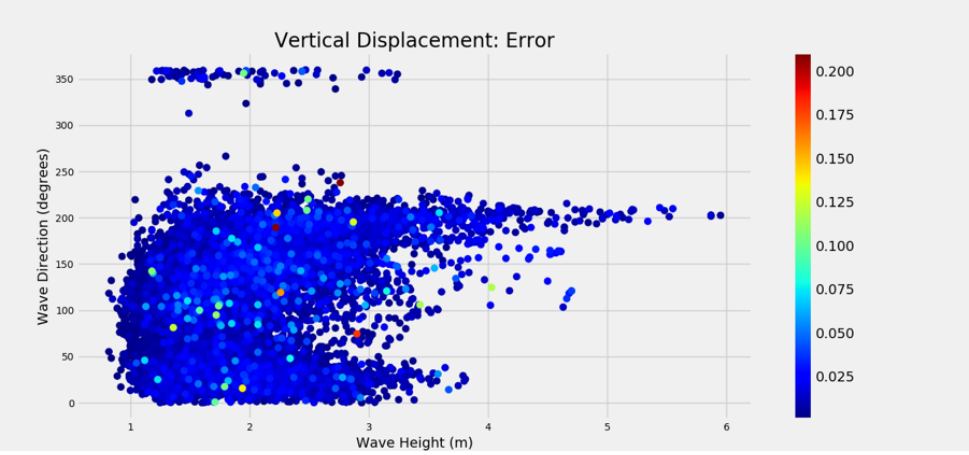
Displacement (FD) meta-model for each line group. Groups 2 and 3 (stern) tend to display mostly positive values of maximum Horizontal In-Plane FD, while groups 1 and 4 show negative values of the output variable. This behavior is appropriate given typical local sea states and the FPSO heading. Additionally, NeuroSim presented predictions with small errors when compared to Dynasim, these results are numerically analyzed in Chapter 6.

5.4.4 Maximum Vertical Fairlead Displacements

Finally, the performance of the final meta-model is evaluated as the previous one. However, since no group-specific behavior was observed, the maximum vertical fairlead displacements (FD) are plotted as an average across all 18 lines, as a function of the environmental variables it is most sensitive to, incident wave height and direction (Fig. 24).



(a) Comparison between NeuroSim meta-model (orange) and Dynasim values (blue) for Maximum Vertical Fairlead Displacements at 14m draft.



(b) NeuroSim Maximum Vertical Fairlead Displacements meta-model error plot.

Figure 24: Maximum Vertical Fairlead Displacements meta-model results as a function of Wave Height and Direction.

Similarly to CG Offset prediction, error spikes were observed on a small number of metocean conditions while most conditions were predicted with high accuracy. Unlike the first meta-models however, NeuroSim's prediction of maximum vertical FD presented no regions of error concentration in the input space, even in plots of different combinations of input variables. A positive result is that even the maximum observed test error (approx. $0.2m$) is sufficiently small to allow for accurate line tension estimation in the future.

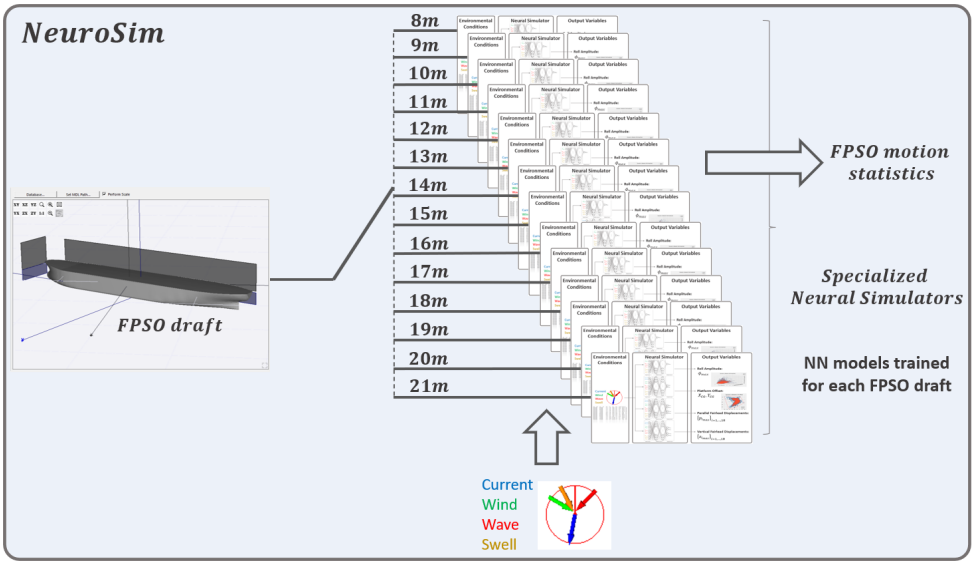
5.5 NeuroSim Architecture Comparison

The way different models and components are organized and interact with each other can lead to considerable differences in the number of necessary models, computational time, model storage space, overall complexity, scalability, among other factors. In order to identify the most appropriate architecture for NeuroSim, several solutions were implemented and evaluated. The two most promising ones were further investigated and their performance on the test dataset was numerically compared. Figure 25 illustrates the two architectures:

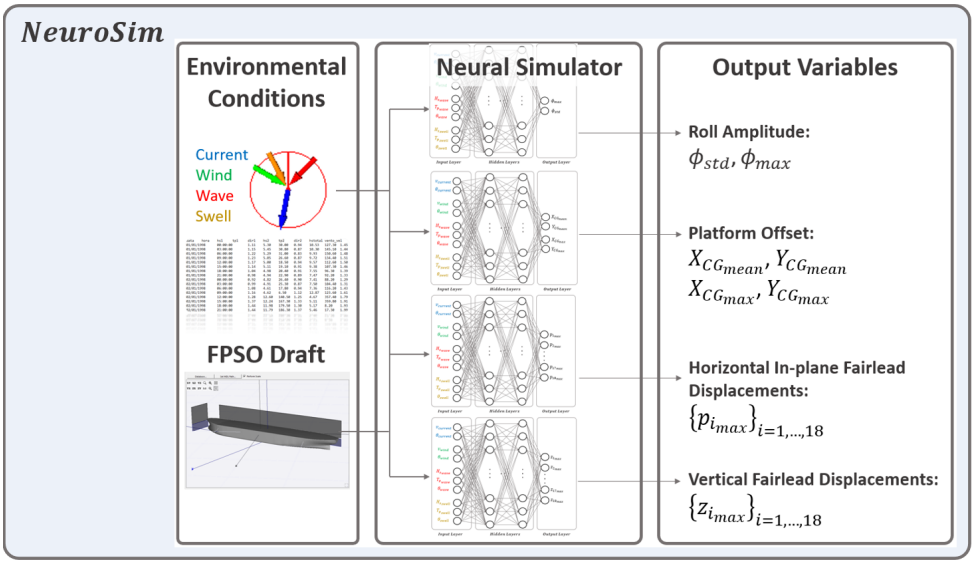
- **Draft Switching Architecture**: 14 different simulation modules are trained separately, each specialized in the platform dynamics when subject to a different draft (from 8m to 21m). Each module is comprised of 4 ANN models designed to predict the desired motion statistics (Roll, Offset, Horizontal In-Plane and Vertical Fairlead Displacements), totaling 56 different ANN models. In order to generate a set of estimations, a switching mechanism which takes the current platform draft decides which module to use.
- **Draft as Input Architecture**: Only 4 ANN models are trained, each takes as input the 10 variables associated with incident metocean conditions as well as the platform draft. Each neural network is specialized in the prediction of different motion statistics and is capable of abstracting the effect of platform draft. As a result, there's no need for a switching mechanism.

Before analyzing the results obtained by both architectures, it is important to note that they both present positive and negative points. The first architecture, characterized by switching between a set of 14 draft-specific modules, allows for each model to be highly specialized in the platform's dynamics for a specific load, which leads to less complex models with a smaller average number of trainable parameters and facilitates the identification of potential issues in a model without compromising the rest. However, the second architecture allows for the prediction of resulting motion statistics in cases of non-integer draft values without the need for manual interpolation. It also allows the four ANN models to abstract the influence of draft and learn how it affects the platform's dynamics. It's also simpler and has less restrictive memory requirements for model storage.

With the objective of quantitatively comparing both architectures, the five first pipelines were executed and the obtained error metrics were analyzed. Figure 26 illustrates



(a) Draft Switching Architecture: 14 modules are trained, each specialized on a single platform draft and comprised of 4 ANN models. A switching mechanism determines the module used to generate predictions according to the current draft.

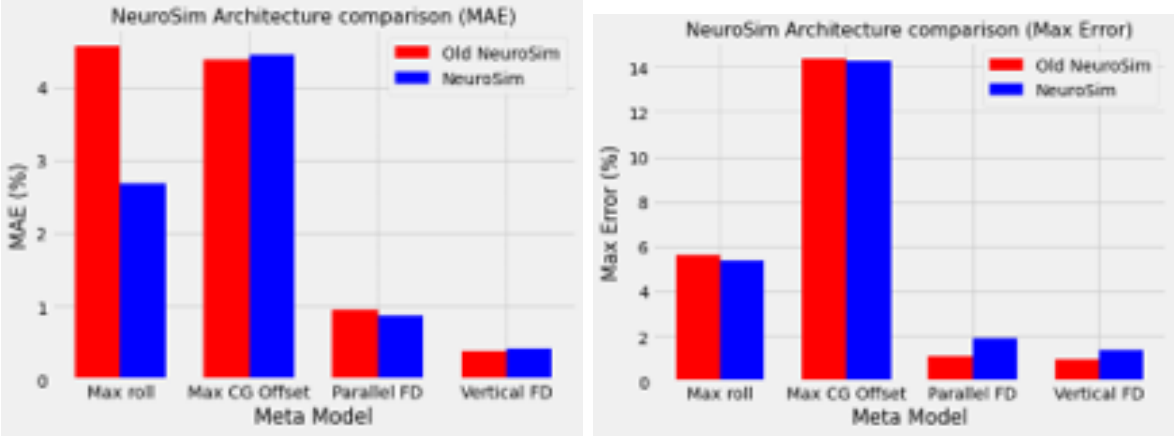


(b) Draft as Input Architecture: Only 4 ANN models are trained, each specialized in the prediction of a different motion statistic, which take platform draft as input.

Figure 25: NeuroSim Architecture Diagrams.

the average errors of each of the 4 meta-models for both architectures, calculated as a percentage of the average output variable value to allow their comparison with each other.

It is possible to verify that, while no significant changes were observed in the Maximum Errors of each meta-model for both architectures, the new architecture presented a lower MAE in comparison to the old one (Fig. 26a). Considering the second architecture also contains a smaller total number of ANN models to be optimized, trained and saved, as



(a) Mean Absolute Error (MAE) of both NeuroSim architectures for each meta-model.

(b) Maximum Error of the two NeuroSim architectures for each meta-model.

Figure 26: Comparison of the old architecture of draft switching (red) and the new architecture of models that receive platform draft as input (blue) for each of the four meta-models.

well as the fact that the remaining test metrics presented no major variations between architectures, it was decided to adopt the Draft as Input Architecture (Fig. 25b) for NeuroSim.

5.6 Sensitivity Analysis

The results presented thus far consider estimations calculated by NeuroSim assuming complete knowledge of incident metocean conditions. The input variables which represent these conditions, however, are obtained by analytical models and represent complex physical phenomena. This means there's an uncertainty associated with incident environmental conditions and it is of paramount importance to analyze NeuroSim's performance considering such uncertainties, in order to better understand its performance in real applications. In this experiment, several disturbances of different magnitudes were applied to the input variables of each trained model to verify their robustness to uncertainties in different environmental variables. The main objective is to evaluate the increase in prediction errors as the disturbances in input variables increase. The sensitivity analysis was performed with the following methodology:

- 1000 random samples were chosen from the test dataset.
- For each of them, 100 new samples were generated from a Gaussian noise applied to one of the ten environmental input variables.

- The factor $K = \frac{MAE_{\text{noise}}}{MAE}$ was calculated given the MAE measured with and without noise.
- This process was repeated for each of the 10 environmental input variables (current velocity and direction, wind velocity and direction, significant height, peak period and direction of wave components with highest energy) and for 10 different standard deviations of the applied noise.

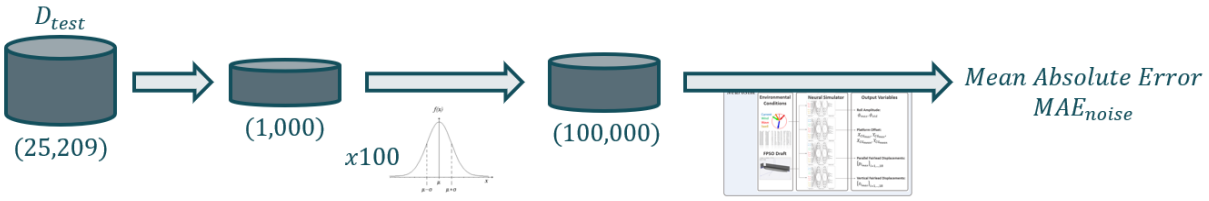


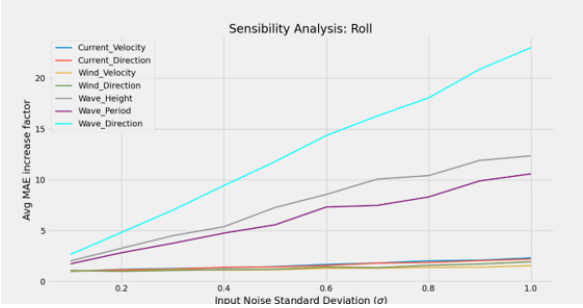
Figure 27: Diagram of the adopted methodology for Sensitivity Analysis of the four meta-models.

Figure 27 illustrates the meta-model sensitivity analysis process adopted for each different input variable. The factor K represents how much NeuroSim's error increases when noise is applied to the environmental conditions relative to the original error. We then studied how this factor increases as we increase the noise intensity, represented by the standard deviation σ of the Gaussian Distribution used to create the noises. Figure 28 illustrates the value of K as a function of σ for each of the four meta-models and for each environmental input variable.

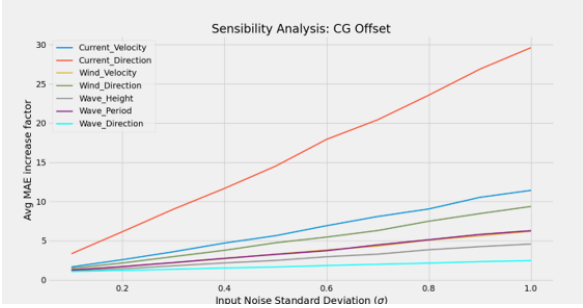
Sensitivity Analysis results of the four meta-models indicate the factor K tends to increase approximately linearly as the uncertainty over incident metocean conditions increases. We confirm that each meta-model is more sensitive to the environmental variables which are more relevant for that specific motion statistic estimated by the model. For example, the roll meta-model is more sensitive to wave height and direction, while the offset meta-model is more sensitive to current direction and velocity. These results are consistent with the physical behaviour expected from both dynamics and allow us to estimate the range of noises over input variables which lead to satisfactory errors on output variables: For noises with standard deviation $\sigma <= 0.2$ over the direction of incident waves we have $K <= 5$, which leads to average absolute errors smaller than $MAE_{\text{noise}} \approx 0.05°$.

5.7 Fine-Tuning

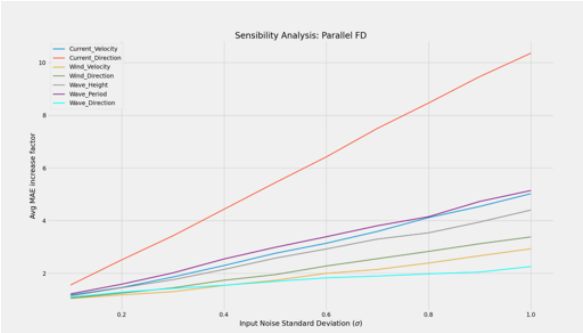
After evaluating the results obtained by NeuroSim in the test dataset (Pipeline P05), we noticed that the trained models performed better when compared to motions statistics



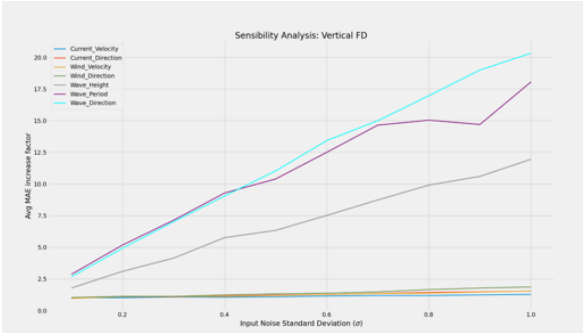
(a) Roll meta-model



(b) Offset meta-model



(c) Horizontal In-Plane FD meta-model



(d) Vertical FD meta-model

Figure 28: Sensitivity Analysis: Factor K as a function of standard deviation σ of the noise applied to incident environmental variables for each of the four meta-models.

obtained by Dynasim than real measured motion, which is expected given the training dataset was generated by Dynasim simulations. When comparing the results to real platform motion, it was possible to see that NeuroSim presented higher errors, despite following the overall tendency of the signal. This behavior is illustrated in figure 29, where the maximum roll measured (in blue) is compared to the maximum roll obtained by NeuroSim's first meta-model (in orange).

Upon analyzing these results, the following question arises: How to improve estimations generated by NeuroSim when it is applied to real data without disregarding the previous training with simulated data? A solution is to apply methods from Transfer Learning, which consist in the storage of knowledge utilized in the solution of a problem and application of it on a different but related problem. In this case, the new problem is the estimation of real motion statistics given incident generic metocean conditions. A known technique of Transfer Learning is denominated Fine-Tuning, where a trained model is further tuned through re-training on a new dataset, but with a reduced learning rate, in order to make fine adjustments to the model.

By making use of real measured platform motion data in the time period of 2007-2014 and the respective environmental conditions and platform drafts in the same period, the post-processing and data preparation pipelines were executed in order to generate a

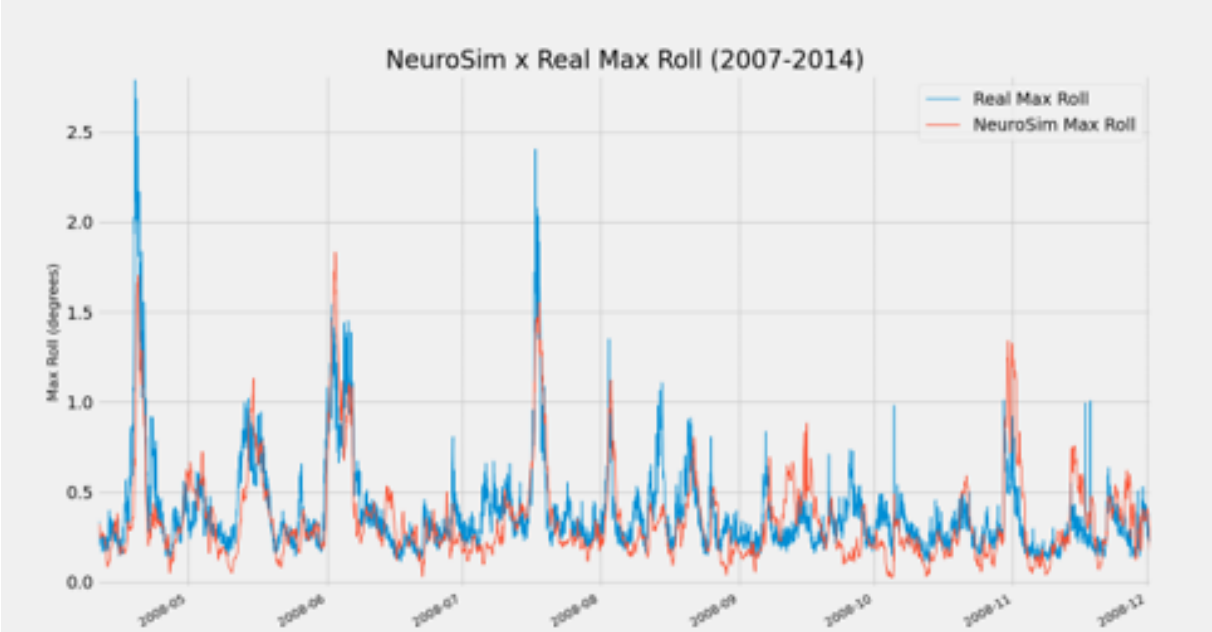


Figure 29: Comparison between real measured maximum roll and maximum roll obtained by NeuroSim after training with Dynasim simulated data.

fine-tuning dataset. Table 6 presents the number of hourly samples of this dataset for different platform drafts. It is important to notice that the distribution of real data is not balanced over drafts: During loading and off-loading operations, the intermediate range of drafts from $13m$ to $17m$ is considerably more common than extreme draft values.

Table 6: Fine-Tuning Dataset

Draft (m)	10	11	12	13	14	15	16
Number of Samples	131	993	3195	6219	8532	8681	7412
Draft (m)	17	18	19	20	21		
Number of Samples	5191	2893	1312	260	40		

The meta-models were re-trained with 90% of the measured data and the remaining 10% were utilized for testing, in order to evaluate the fine-tuning process and check if there was an increase in performance on new, unseen real data. The fine-tuning process was performed with different learning rates ($1e-3$ to $1e-8$) in logarithmic scale in order to determine the optimal learning rate to learn real platform motion without disregarding the previous training process.

Figure 30 illustrates a section of the real maximum roll signal compared to estimations obtained by NeuroSim after fine-tuning with different learning rates, from $1e-3$ to $1e-8$. We notice that, with lower learning rates ($\approx 1e-8$) the model replicates the predictions obtained when trained with simulated data. For higher learning rates ($\approx 1e-3$), however,

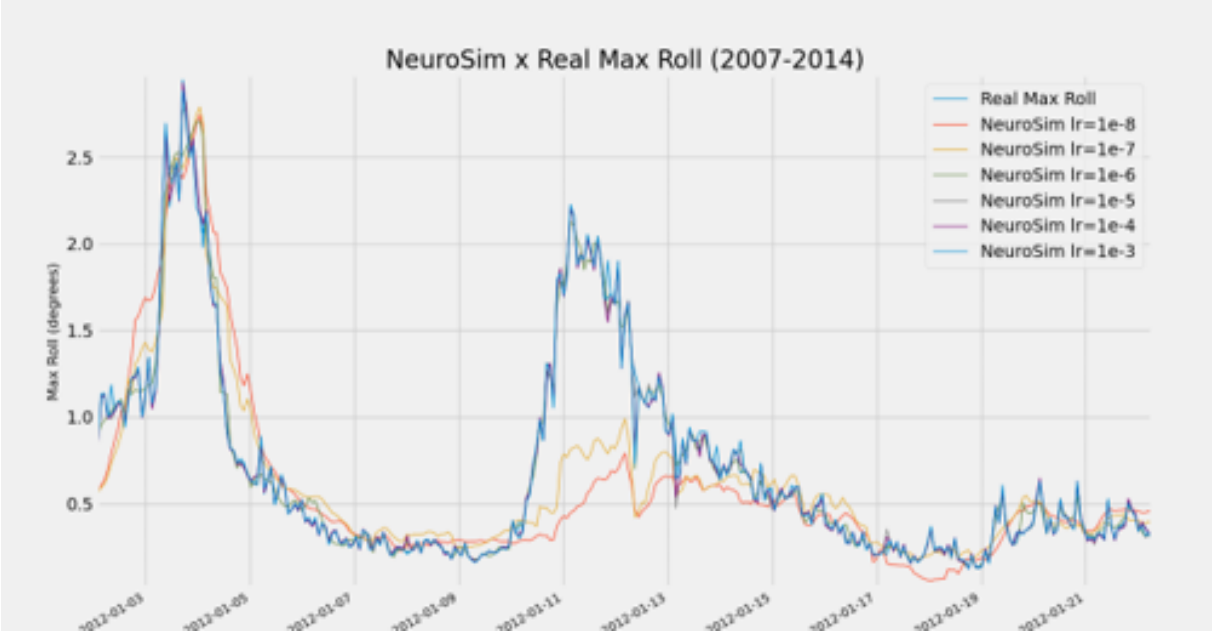
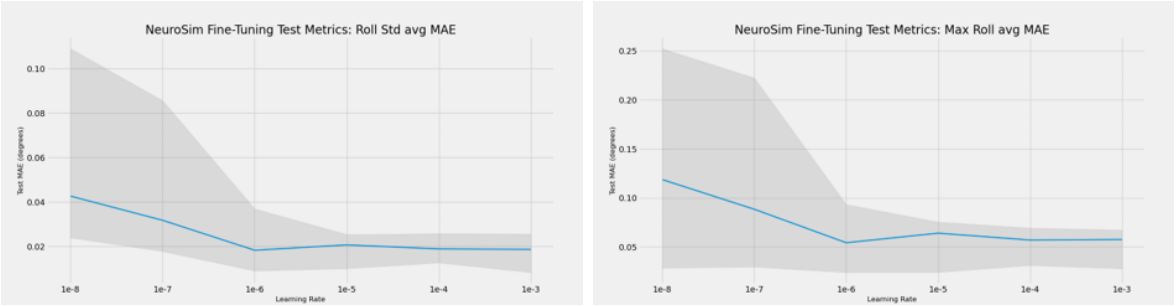


Figure 30: Comparison between NeuroSim after fine-tuning with different learning rates on real maximum roll angle data.



(a) Roll std. (b) Max roll.

Figure 31: Roll meta-model test MAE as a function of different learning rates used during the fine-tuning process.

the model trains with real data as if training from scratch and overfits on the new dataset. The test MAE of the meta-model after fine-tuning varies according to the learning rate used (figure 31). For learning rates higher than $lr = 1e - 6$ it is not possible to notice significant performance increase for the meta-model, both for maximum roll and for its standard deviation. As a result, we opted to utilize a learning rate of $lr = 1e - 6$ during the fine-tuning process.

6 RESULTS AND DISCUSSION

Previous chapters presented the theoretical concepts behind this research, detailed the adopted methodology and illustrated the experiments performed from the obtention of data to the analysis of trained and validated models. The objective of this chapter is to expose a summary of the results obtained by NeuroSim, provide a numerical overview of the relevant error metrics and finally a discussion of results.

Data-driven approaches to FPSO motion prediction and response-based design are relatively new and currently in the stage of validation, working as additional tools to traditional dynamic simulation methods. As a result, the research performed thus far constitutes a proof of concept of the proposed framework as an auxiliary tool to response-based design and, in the future, an addition to FPSO digital twins in seakeeping applications.

As described in section 4.2, 18k metocean conditions measured over a period of 6 years at the Campos Basin were analyzed and simulated in a dynamic simulation software know as Dynasim, which generated the time-series motion of a spread-moored FPSO subject to these conditions for each integer draft value between 8m and 21m. Relevant motion statistics were then extracted from the 6 DoF time-series and associated with the corresponding metocean conditions, which were processed in order to generate Training, Validation and Test datasets. The datasets were then used to perform a Hyperparameter Optimization technique know as Bayesian Optimization, which provided the optimal ANN architectures for each of the meta-models. Finally, the proposed models were trained and tested, some of the results obtained are visually presented in section 5.4. A sensitivity analysis was also conducted in order to verify NeuroSim's performance in cases where there's uncertainty involved in the measurement of incident metocean conditions. The roll meta-model was then fine-tuned using real measured FPSO motion data from 2007-2014, on a dataset of over 44k samples. Table 7 summarizes the test errors of each meta-model.

Overall, the optimized models obtained through BO and trained on simulated motion data presented good test results when compared to the average accuracy of traditional

Table 7: NeuroSim Error Metrics on Test Dataset

Meta-model	**Error Metrics**			
	MSE	RMSE	MAE	MaxError
Roll std.	1.96e-5	0.0044°	0.0020°	0.092°
Roll max.	2.74e-4	0.0163°	0.0098°	0.239°
Mean Offset	0.008	0.092m	0.061m	0.726m
Max Offset	0.552	0.739m	0.397m	6.342m
Maximum Horizontal In-Plane FD	0.075	0.273m	0.184m	1.928m
Maximum Vertical FD	9.1e-4	0.030m	0.019m	0.378m

dynamic simulation methods. The estimates for the roll motion standard deviation presented mean absolute errors in the order of magnitude of about 1/4 to 1/5 of the error associated with the maximum observed roll angle. This behavior is consistent with the overall magnitude of both variables. It is also possible to verify that the Maximum Horizontal In-Plane FD meta-model performed with an MAE of $0.184m$, closer to the error of the CG Offset meta-model than to its vertical counterpart, which is explained by both output variables being closely related as statistics of motion parallel to the water's surface.

Across all four meta-models, the Maximum Error metric displayed the highest variance, as it depends on a small number of specific, problematic metocean conditions, as illustrated in figures 19 to 24. However, this metric is useful in order to determine the absolute highest prediction errors given by NeuroSim across the entire Test set, providing useful insights into high confidence, conservative upper boundaries of errors for unseen metocean conditions.

In addition to the training, validation and test of the four meta-model ANNs, several other experiments were conducted. From the sensitivity analysis results we conclude that the neural networks were able to correctly capture the complex underlying platform dynamics, as not only different output variables were more sensitive to changes in different environmental variables, but this variation of the error increase rate as a function of the noise was more prominent in the variables which are physically known to be more relevant for each type of motion. For example: The first meta-model correctly captured the roll dynamics and displayed a higher sensitivity to changes in wave direction and height than to current direction, which was more relevant to the second meta-model. The final experiment, dedicated to the fine-tuning of obtained ANN models with real measured FPSO motion data, also showed positive results: The test MAE on real data was reduced by approximately 50% after fine-tuning the model with a learning rate of $1e-6$.

7 CONCLUSION AND NEXT STEPS

In this work we proposed a framework for the development, training and validation of a data-driven, neural network based FPSO dynamics simulator called NeuroSim. The adopted methodology is comprehensively described in chapter 4 section 4.2.

Firstly, a dataset of over 252k samples (18k environmental conditions for 14 different platform draft values) was used to determine the optimal ANN architectures for each meta-model through a state of the art algorithm for hyperparameter optimization. The proposed models are then trained and tested on unseen environmental conditions. The results obtained are extremely close to the metrics obtained by the Dynasim simulation software, but still differ from the actual measured motion of the platform. Finally, the roll meta-model is fine-tuned on real data with a variety of different learning rate values and is shown to perform better when compared to real platform motion. Several experiments were also conducted, such as the comparison of NeuroSim architectures, the sensitivity analysis of models to changes in input variables and an initial exploratory analysis of typical environmental conditions.

One of the main contributions of this research is the fine-tuning of ANN models using real measured FPSO motion data and the corresponding incident metocean conditions. This work presents a way of improving the accuracy of meta-models through the use of data that is already measured, allowing for the abstraction of complex, unmodeled dynamics and increase in accuracy on monitoring applications.

Over the course of this research, the following papers have been written and published:

- **Neural Network Meta-Models for FPSO Motion Prediction From Environmental Data**: The main objective of this work is to provide a proof of concept that a system of data-driven ANN models can be successfully utilized to estimate the motion of an FPSO subject to generic environmental conditions. In this work a simple Grid Search algorithm was used to optimize ANN hyperparameters and only simulated data from a single platform draft of 14m was used to train the models

[Cotrim et al. (2021)].

- **Neural Network Meta-Model for FPSO Roll Motion Prediction from Environmental Data**: This research focused on the development of a meta-model designed specifically for the estimation of roll motion statistics given different platform loads and incident environmental conditions. Single model approaches were compared to results obtained from ensembles of different sizes [Cotrim et al. (2021)].

- **Evaluation of Neural Architecture Search Approaches for Offshore Platform Offset Prediction**: The main objective of this paper is to provide a comparative study of different Hyperparameter Optimization algorithms applied to the estimation of an FPSO's CG Offset through Artificial Neural Networks. An in-depth analysis of the performance of three different algorithms is conducted: Random Search (baseline), Simulated Annealing (SA) and Bayesian Optimization (BO) [Suller et al. (2021)].

- **Neural Network Meta-Models for FPSO Motion Prediction From Environmental Data With Different Platform Loads**: The main objective of this work is to design a system of data-driven meta-models capable of estimating different FPSO motion statistics given platform load and incident metocean conditions. A switching mechanism is implemented in order to obtain output values from models specialized in different platform loads. Additional experiments are presented, such as ensemble regression and a sensitivity analysis [Cotrim et al. (2022)].

- **Combining Model-Based and Data-Driven Methods to Estimate the Roll Motion of a Spread-Moored FPSO**: The objective of this paper is to design and validate a hybrid Physics-ML approach for correcting the estimation of roll motion prediction from traditional dynamic simulation softwares. A Neural Network model is trained to calculate the residue between real roll motion and estimated roll motion obtained by Dynasim. The resulting hybrid system is shown to provide better results when compared to real FPSO motion data.

The results obtained (chapter 6) indicate NeuroSim is capable of capturing an FPSO's dynamic response to arbitrary incident metocean conditions, as mean errors are within acceptable range. These results solidify the proposed framework as a successful proof of concept for the future integration of NeuroSim with other monitoring tools as part of a complete digital twin of an operating FPSO. Future work includes:

- Fine-tuning the remaining three meta-models on real measured FPSO motion and testing NeuroSim in seakeeping applications as an auxiliary tool for real-time motion statistics prediction. Results will then be compared to values obtained through traditional simulation methods, indicating whether the meta-models are able to capture unmodeled dynamics.

- Study additional fine-tuning techniques, such as adding a layer to each ANN model freezing the weights of previous layers during the new training process.

- Investigate Ensemble Regression techniques, such as Boosting or weighted average prediction of ANN models specialized in different regions of the input space, rather than using a single ANN model for each of the four types of output variables of interest.

- Increment the proposed framework to include additional relevant motion statistics to be estimated, such as line tensions.

- Consider the approach of hybrid physics-ML models, which combine consolidated analytical equations of motion with data-driven approaches in order to improve explainability of models through the use of domain knowledge rather than rely exclusively on data.

REFERENCES

BERGSTRA, J.; BARDENET, R.; BENGIO, Y.; KéGL, B. Algorithms for hyper-parameter optimization. In: *Advances in Neural Information Processing Systems*. [s.n.], 2011. p. 1–9. Disponível em: ⟨https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf⟩.

COTRIM, L.; OLIVEIRA, H.; FILHO, A.; TANNURI, E.; COSTA, A. H.; GOMI, E.; SANTOS, I.; BARREIRA, R. Neural network meta-model for fpso roll motion prediction from environmental data. In: *Proceedings of the Ibero-Latin-American Congress on Computational Methods in Engineering (CILAMCE)*. [S.l.: s.n.], 2021.

COTRIM, L. P.; BARREIRA, R. A.; SANTOS, I. H.; GOMI, E. S.; COSTA, A. H. R.; TANNURI, E. A. Neural network meta-models for FPSO motion prediction from environmental data with different platform loads. *IEEE Access*, IEEE, v. 10, p. 86558–86577, 2022.

COTRIM, L. P.; OLIVEIRA, H. B.; FILHO, A. N. Q.; SANTOS, I. H.; BARREIRA, R. A.; TANNURI, E. A.; COSTA, A. H. R.; GOMI, E. S. Neural network meta-models for FPSO motion prediction from environmental data. In: ASME. *International Conference on Offshore Mechanics and Arctic Engineering*. [S.l.], 2021. v. 85116, p. V001T01A003.

DAMASCENO, M.; NETO, H. R.; COSTA, T.; JúNIOR, A. C.; AGUIAR, L.; MARTINS, M. Using Kriging Surrogate Models to Predict the Vibration Responses of a Submerged Riser. In: . [S.l.: s.n.], 2020. (OMAE2020-18915).

De Pina, A. C.; De Pina, A. A.; ALBRECHT, C. H.; Leite Pires De Lima, B. S.; JACOB, B. P. Ann-based surrogate models for the analysis of mooring lines and risers. *Applied Ocean Research*, v. 41, p. 76–86, 2013. ISSN 0141-1187. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0141118713000199⟩.

ELSKEN, T.; METZEN, J. H.; HUTTER, F. Neural architecture search: A survey. *Journal of Machine Learning Research*, v. 20, n. 55, p. 1–21, 2019. Disponível em: ⟨http://jmlr.org/papers/v20/18-598.html⟩.

FEURER, M.; HUTTER, F. Hyperparameter optimization. In: _____. *Automated Machine Learning: Methods, Systems, Challenges*. [S.l.]: Springer International Publishing, 2019. p. 3–33. ISBN 978-3-030-05318-5.

FRAZIER, P. I. *A Tutorial on Bayesian Optimization*. 2018. ArXiv:1807.02811.

GONZALEZ, G. M.; SIQUEIRA, M. Q.; SIMãO, M. L.; VIDEIRO, P. M.; SAGRILO, L. V. S. On the Use of Artificial Neural Networks for Estimating the Long-Term Mooring Lines Response Considering Wind Sea and Swell. In: . [S.l.: s.n.], 2020. (OMAE2020-18868).

GUARIZE, R.; MATOS, N.; SAGRILO, L.; LIMA, E. Neural networks in the dynamic response analysis of slender marine structures. *Applied Ocean Research*, v. 29, n. 4, p. 191–198, 2007. ISSN 0141-1187. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0141118708000060⟩.

GUMLEY, J. M.; HENRY, M. J.; POTTS, A. E. A Novel Method for Predicting the Motion of Moored Floating Bodies. In: . [S.l.: s.n.], 2016. (OMAE2016-54674).

HUANG, B.; JIANG, J.; ZOU, Z. Online prediction of ship coupled heave-pitch motions in irregular waves based on a coarse-and-fine tuning fixed-grid wavelet network. *Journal of Marine Science and Engineering*, v. 9, p. 989, 09 2021.

JAMES, G. *An introduction to statistical learning : with applications in R*. New York, NY: Springer, 2013. ISBN 978-1-4614-7138-7.

LAARHOVEN, P. J. M. van; AARTS, E. H. L. *Simulated Annealing: Theory and Applications*. Springer Netherlands, 1987. Disponível em: ⟨http://dx.doi.org/10.1007/978-94-015-7744-1⟩.

MASI, G. D.; GAGGIOTTI, F.; BRUSCHI, R.; VENTURI, M. Ship motion prediction by radial basis neural networks. In: *2011 IEEE Workshop On Hybrid Intelligent Models And Applications*. [S.l.: s.n.], 2011. p. 28–32.

MAZAHERI, S. The usage of artificial neural networks in hydrodynamic analysis of floating offshore platforms. *The International Journal of Maritime Engineering*, v. 3, p. 48–60, 09 2006.

NISHIMOTO, K.; FUCATU, C. H.; MASETTI, I. Q. Dynasim—A Time Domain Simulator of Anchored FPSO . *Journal of Offshore Mechanics and Arctic Engineering*, v. 124, n. 4, p. 203–211, 10 2002. ISSN 0892-7219. Disponível em: ⟨https://doi.org/10.1115/1.1513176⟩.

PINA, A. C. D.; ALBRECHT, C. H.; LIMA, B. S. D.; JACOB, B. P. Wavelet network meta-models for the analysis of slender offshore structures. *Engineering Structures*, v. 68, p. 71–84, 2014.

SCIUTO, C.; YU, K.; JAGGI, M.; MUSAT, C.; SALZMANN, M. Evaluating the search phase of neural architecture search. *CoRR*, abs/1902.08142, 2019. Disponível em: ⟨http://arxiv.org/abs/1902.08142⟩.

SIDARTA, D.; KYOUNG, J.; O'SULLIVAN, J.; LAMBRAKOS, K. Prediction of offshore platform mooring line tensions using artificial neural network. In: . [S.l.: s.n.], 2017. (OMAE2017-61942).

SULLER, T.; GOMES, E.; OLIVEIRA, H.; COTRIM, L.; SA'AD, A.; SANTOS, I.; BARREIRA, R.; TANNURI, E.; GOMI, E.; COSTA, A. Evaluation of neural architecture search approaches for offshore platform offset prediction. In: *Anais do XVIII Encontro Nacional de Inteligência Artificial e Computacional*. Porto Alegre, RS, Brasil: SBC, 2021. p. 326–337. ISSN 0000-0000. Disponível em: ⟨https://sol.sbc.org.br/index.php/eniac/article/view/18264⟩.

WANG, H.; ZHENG, H. Model validation, machine learning. In: _____. *Encyclopedia of Systems Biology*. Springer New York, 2013. p. 1406–1407. ISBN 978-1-4419-9863-7. Disponível em: ⟨https://doi.org/10.1007/978-1-4419-9863-7_233⟩.

YIN, J.; PERAKIS, A.; WANG, N. A real-time ship roll motion prediction using wavelet transform and variable rbf network. *Ocean Engineering*, v. 160, p. 10–19, 07 2018.