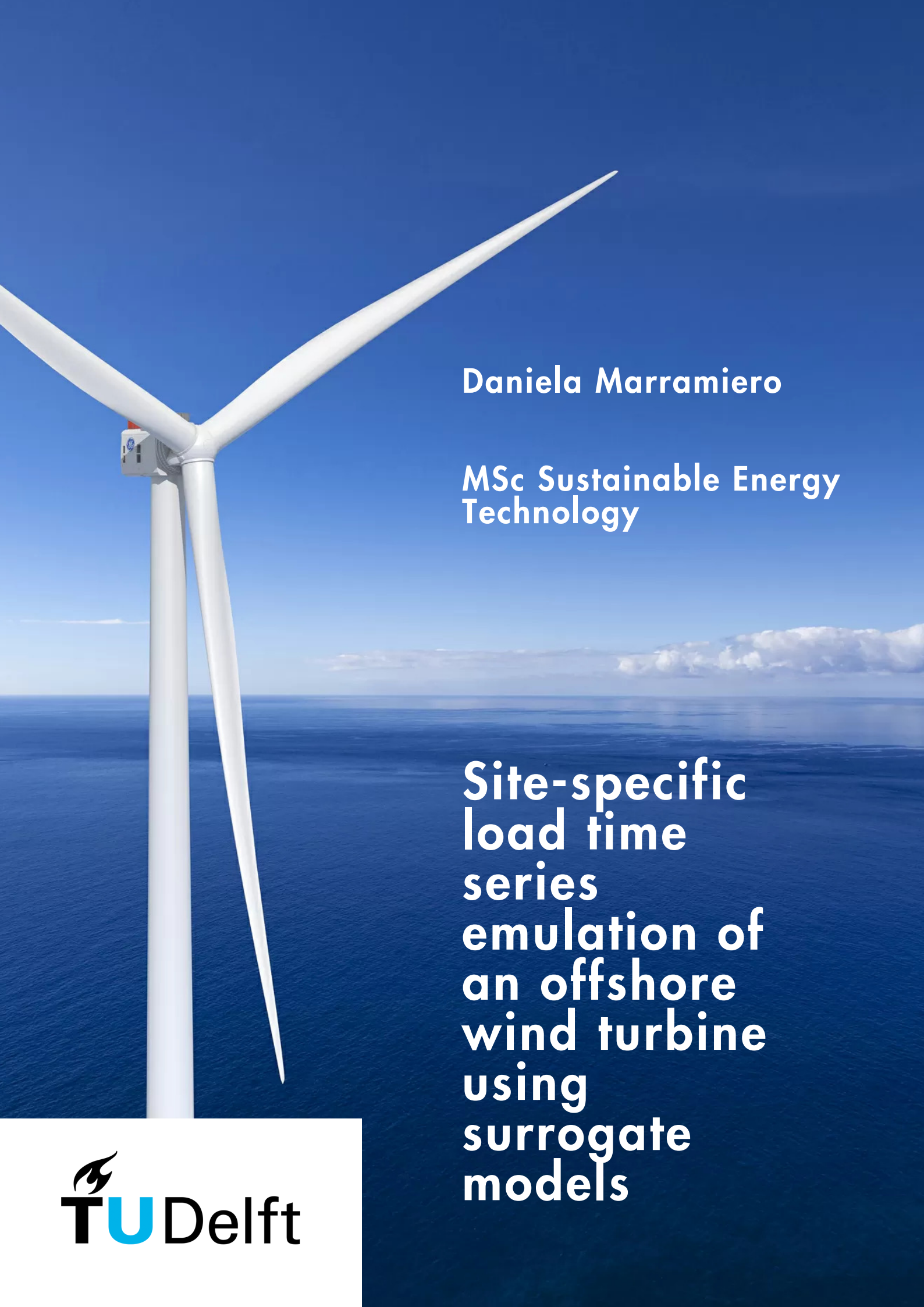


Daniela Marramiero

MSc Sustainable Energy Technology

Site-specific load time series emulation of an offshore wind turbine using surrogate models

TU Delft

Master of Science Thesis

Site-specific load time series emulation of an offshore wind turbine using surrogate models

Daniela Marramiero

August 2023

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of Master of Science in Sustainable Energy Technology

The work in this thesis was carried out in the:

Delft University of Technology

Supervisors: Dr. ir. Axelle Viré
PhD Deepali Singh

Abstract

Considering the goals set by the international community, the implementation of new energy sources has to increase considerably in the next seven years. In this thesis, the focus is on the acceleration and improvement of the application of offshore wind turbines. The power produced using this technology should become 3.6 times more before the end of this decade to comply with the set goals.

To achieve this target, new solutions have to be developed. To this aim, the implementation of model predictive control for wind turbines in the last years has been investigated. This would allow to optimize different parameters at the same time, such as maximization of energy production while minimizing the perceived loads. For this application, it is necessary to have forecasts of different features of the turbine, especially loads, with a higher frequency. As a result, the main research topic is defined as 'Can data-driven surrogate models be used for forecasting load time series on offshore wind turbines?'.

To answer this question, first environmental conditions are sampled within limits deducted from real data through Halton sequencing, and next simulations are run through OpenFAST to determine the resulting loads acting on the turbine. Within all the features resulting from the simulation, only five inputs and five target outputs are selected. This is the result of various considerations. Given the desire to develop a realistic methodology, the input variables are first filtered by assessing their availability from measurement devices. Next, the relationships between the variables are analyzed through cross-correlation to determine the degree of influence of each input on the output.

Using this data, a training database is created. It is used to train two different types of surrogate models, one linear and one non-linear, respectively ARIMAX and LSTM. These are implemented to generate a 30-second forecast of the moments acting at the root of the blade. To do so, the algorithms are trained using different variables as exogenous inputs to assess the models' performance in different cases. Given the wide range of target features, for LSTM two different behaviors are identified and the blade edgewise and flapwise moments are taken as examples. The hyper-parameters are tuned on the blade edgewise moment and lead to overfitting when applied to the blade flapwise and out-of-plane moment.

The obtained results show that the RMSE in ARIMAX is up to seven times larger than the one obtained from the application of LSTM. Within the non-linear models, the one resulting in the lowest percentage error for the blade edgewise, pitching, and in-plane moment considers the wind reference speed, the wind speed time series, and the corresponding tip deflection as exogenous inputs. Very low RMSE errors are obtained for all variables. Furthermore, it is concluded that while it is possible to implement LSTM in real-life, this is not achievable for ARIMAX.

Acknowledgements

Writing this thesis has been a constant learning process, challenging and rewarding at the same time. It wouldn't have been possible without the help and support of some people so this is for them.

Thank you to Prof. Axelle Vire for your supervision and advice. It was thanks to your input and feedback if I were able to go back on the right path and ultimately identify what I had to do.

Thank you to PhD. Deepali Singh for your mentoring, and patience. You were able to spot my mistakes from miles away and never failed to explain how to improve. It was really challenging, I learned a lot and I am really grateful for all the time you spent helping me.

Thank you also to Dr. Ir. Delphine De Tavernier, Prof. Jan Willem van Wingerdenjan, Dr. Ir Sebastiaan Mulders, and Prof. Richard Dwight for our meetings. Even in a short time, you helped a lot by providing insightful knowledge.

Of course, all this would have been less enjoyable if it weren't for the amazing people I met here. These two years of MSc have been probably two of the best of my life thanks to you. You have been a constant source of motivation and happiness. Thank you for bearing with me, and for being the most supportive and welcoming group of people. Without you, this experience wouldn't have been the same, and I feel extremely grateful for having had the opportunity of sharing these two years with you.

Thank you also to my friends in Italy, from Pescara and Milan, who have accompanied me through all the ups and downs. You managed to be there for me when I needed you even from far away.

Last but not least thank you to my whole family. You convinced me to come here and have been super supportive throughout the whole process. Thank you for believing in me even and most importantly when I didn't believe in myself.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

ADF Augmented Dickey Fuller . . . 42
AdaGrad Adaptive Gradient Algorithm . . . 58
AIC Akaike Information Criterion . . . 12
ARIMA AutoRegressive Integrated Moving Average . . . 52
ARIMAX AutoRegressive Integrated Moving Average with eXogenous input . . . xi
ARX AutoRegressive with eXogenous input . . . 10
BEM Blade Element Momentum . . . xi
CFD Computational Fluid Dynamics . . . 33
DoF degree of freedom . . . 23
DTU Technical University of Denmark . . . 25
FOWT Floating Offshore Wind Turbine . . . 7
HPC High-Performing Computer . . . 7
IRENA International Renewable Energy Agency . . . 1
LCOE Levelized Cost of Energy . . . 1
LBM Leishman-Beddoes Model . . . 28
LSTM Long Short Term Memory . . . xi
LVDT Linear Variable Differential Transformers . . . 39
MAPE Mean Absolute Percentage Error . . . 11
MLE Maximum Likelihood Estimation . . . 11
MPC Model Predictive Control . . . 1
NN Neural Network . . . xi
NREL National Renewable Energy Laboratory . . . 7
OLAF cOnvecting LAgrangian Filaments . . . xi
PI Proportional Integral . . . 2
PCE Polynomial Chaos Expansion . . . 10
ReLU Rectified Linear Unit . . . 14
RMSE Root Mean Square Error . . . xii
RMSProp Root Mean Square Propagation . . . 58
RNN Recurrent Neural Network . . . xi
ROSCO Reference OpenSource Controller . . . 27
SCADA Supervisory Control and Data Acquisition . . . 4
SGD Stochastic Gradient Descent . . . 58
SM Surrogate Models . . . xii
TSR Tip Speed Ratio . . . 27

List of Symbols

Symbol	Definition	Unit
a	Axial interference factor	[-]
a'	Azimuthal interference factor	[-]
a_j	Actiavtion of the j-th unit of a neural network	[-]
A	Area at the beginning of control volume	[m]
A_1	Area at the end of the control volume	[m]
A_R	Area at the rotor	[m]
B	Number of blades	[-]
b	Bias used in neural networks	[-]
C	Cost function	[-]
c	Chord length	[m]
C_d	Drag force coefficient	[-]
C_l	Lift force coefficient	[-]
C_n	Axial force coefficient	[-]
C_P	Power coefficient	[-]
C_t	Tangential force coefficient	[-]
C_T	Thrust coefficient	[-]
d	Differencing order in ARIMAX model	[-]
F_n	Axial force acting on the blade	[N]
F_t	Tangential force acting on the blade	[N]
$g(l)$	Activation function of the l-th layer in neural networks	[-]
h	Height	[m]
h_{ref}	Reference height	[m]
H_s	Significant wave height	[m]
k	Number of estimated parameters by AIC	[-]
$\mathcal{L}$	Likelihood function	[-]
ℓ	Number of lags	[-]
$\dot{m}$	Mass flow	[$\mathrm{kg\,s^{-1}}$]
n	Number of considered samples	[-]
N	Number of data points	[-]
p	Auto-regressive order in ARIMAX model	[-]
p	Pressure	[$\mathrm{kg^2\,m^{-1}}$]
P	Power	[W]
q	Moving average order in ARIMAX model	[-]
r	Radial position	[m]

Symbol	Definition	Unit
$r_{Kendall}$	Kendall correlation coefficient	[-]
$r_{Pearson}$	Pearson correlation coefficient	[-]
$r_{Spearman}$	Spearman's rank correlation coefficient	[-]
r_{xx}	Auto-correlation coefficient of time series x	[-]
r_{xy}	Cross correlation coefficient	[-]
r_{yy}	Auto-correlation coefficient of time series y	[-]
s_x	Standard deviation of time series x	[-]
s_y	Standard deviation of time series y	[-]
t	Output of neural network	[-]
T	Thrust force	$[N]$
TI	Turbulence intensity	[-]
T_p	Wave period	$[s]$
u	Mean wind speed	$[\mathrm{m\,s^{-1}}]$
u_0	Wind speed at the beginning of control volume	$[\mathrm{m\,s^{-1}}]$
u_1	Wind speed at the end of the control volume	$[\mathrm{m\,s^{-1}}]$
u_R	Wind speed at the rotor	$[\mathrm{m\,s^{-1}}]$
V_{rel}	Relative wind speed	$[\mathrm{m\,s^{-1}}]$
w	Weight used in neural networks	[-]
w_t	White noise at time-step t	[-]
x	time series	[-]
$\boldsymbol{x}$	OpenFAST inputs	[-]
x_t	Value of time series at time-step t	[-]
$\hat{x}$	Predicted time series	[-]
$\hat{x}_t$	Predicted time series value at time-step t	[-]
y	time series	[-]
y_t	Value of time series at time-step t	[-]
$\hat{y}$	Predicted time series	[-]
$\hat{y}_t$	Predicted time series value at time-step t	[-]
z_0	Roughness of the surface	[m]
$Z^{(l)}$	Intermediate status computed in the corresponding hidden layers l	[-]
α	Power law exponent or learning rate in LSTM	[-]
Γ	Coefficient matrix of the exogenous vector	[-]
θ_i	Coefficient of the moving average model	[-]
ρ	Air density	$[\mathrm{kg\,m^{-3}}]$
ρ_{xy}	Normalized cross correlation coefficient	[-]
σ	Solidity of the rotor	[-]
ϕ	Inflow wind angle	[rad]
ϕ_i	Coefficient of the auto-regressive model	[-]
Ω	Angular velocity of the rotor	$[\mathrm{rad\,s^{-1}}]$

Chapter 1

Introduction

In this chapter, a general overview of the topic analyzed in this thesis is given. In Section 1.1 the reason behind the choice of the theme is described, followed by Section 1.2 which is an in-depth section focusing on the integration of a surrogate model (SM) in the context of Model Predictive Control (MPC). Next, Section 1.3 presents a summary of the literature review and the chosen research questions are highlighted in Section 1.4. In Section 1.5 the method used to analyze the topic is described and finally, in Section 1.6, the general structure of the thesis is presented.

1.1. Motivation

In recent years, the global attention surrounding sustainability and energy transitions has been exponentially increasing, with nations worldwide trying to implement more renewable energy sources. This trend was concertized by the United Nations, which has established to provide clean and affordable energy access to all by 2030 [11]. This urgency and necessity have been further underscored by the consequences of the recent conflict between Russia and Ukraine, prompting many European countries to reassess their energy portfolios and pursue greater energy independence [12].

Moreover, the Paris Agreement, an international accord signed in 2016 by 196 countries, serves as an additional confirmation of this global trend [13]. This agreement sets a binding objective of limiting global warming to 1.5 °C, yet current national targets, as depicted in Figure 1.1, fail to achieve this aim. Consequently, the International Renewable Energy Agency (IRENA) asserts that more ambitious objectives are necessary. Specifically, IRENA advocates for the installation of 213 GW of offshore wind energy by 2030 to comply with the set goals [1].

In 2022, the global operational offshore wind capacity was 58.3 GW [14]. This means that in the next seven years, the installed power needs to become at least 3.6 times more. To achieve this goal, the wind industry has to develop more technologies that accelerate the design process, improve the performance of the turbines and lower the Levelized Cost of Energy (LCOE), so that they can still compete with other solutions which are currently less expensive.

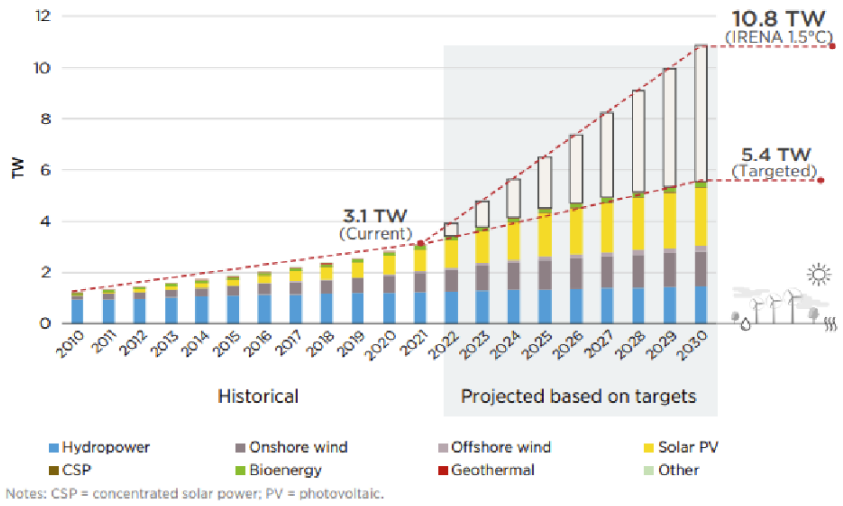


Figure 1.1.: Global cumulative renewable power, installed capacity, historical trends and future projections to 2030 [1].

As it can be inferred by the example reported by ORE Catapult, the operational and capital cost for the construction of an offshore wind turbine have an equal impact on the expenditures if a lifetime of twenty-seven years is considered [15]. Both aspects must be lowered to reduce the LCOE. A way to do so is to increment the energy output while prolonging the life of the turbine. One of the approaches to achieve this goal is to use model predictive controllers (MPC) instead of the traditional Proportional Integral (PI) controllers. However, the implementation of MPC is challenging because it requires forecasting of certain quantities into the future, based on the current state of the system.

Data-driven surrogate models (SM) may represent a solution to this problem. They can be used to reconstruct and predict the loads' history through a process of emulation of the functioning of the wind turbine. These can then be implemented into an MPC algorithm characterized by the ability to optimize the wind turbine's power output while minimizing the perceived loads. To reach this aim, the SM has to be trained with measurement data already available so that it can 'learn' how to forecast the loads given other features such as atmospheric conditions. Only the spots most sensible to fatigue will be considered in this analysis. Overall, no additional cost linked to technology will have to be taken into consideration to complete this evaluation.

When conducting a similar study, various aspects of the specific geographical locations are essential to determine the behavior of the turbine. Simply identifying the turbine model is insufficient. Factors such as the distance from shore, sea depth, and the range of wind variations play significant roles and require different operational decisions to be made accordingly. Thus, the achieved results cannot be automatically generalized to all offshore wind turbines.

Therefore, the goal of this thesis is to present a methodology to create a site-specific surrogate model that provides a time series forecast of the fatigue loads during normal operating conditions at different fatigue-sensible spots along the tower and blades of a 15 MW bottom-fixed offshore turbine located 30 km off the west coast of Ireland. The reason behind the

choice of this particular area is that it is the one in Europe with more approved and planned offshore wind farm installations that have not yet been realized [16]. Thus, it will certainly be of interest in the coming years.

The forecasted loads should then be used in a predictive control algorithm to optimize the turbine output while minimizing the perceived loads. The obtained results will then be confronted with the ones of aero-hydro-servo-elastic simulations in the time domain to assess the performance of the SM.

1.2. Model predictive control

The name 'model predictive control' is used to address a set of advanced control mechanisms that use a model to forecast the behavior of the analyzed system [17]. MPC implementation of wind turbines can be justified by numerous reasons [18]. First, only one controller can be implemented instead of implementing two different PI controllers with an overlapping region corresponding to the rated wind speed. This is important because a significant part of the energy is produced at these wind speeds and contributes significantly to the fatigue of the major structural components. Another reason to implement MPC in wind turbines is that it is capable of optimizing the functioning of the wind turbine taking into account different parameters, such as the optimal energy production considering the fatigue loads perceived by the structure. Next, especially for future developments, more sensors and actuators have already been proposed but their integration with the current controllers is rather difficult. MPC may be a solution to this problem. Recently, attention has been given to fault tolerant control and MPC has been proven to successfully handle many of the typical faults that occur in wind turbines. Finally, it would certainly improve the performance of offshore floating wind turbines, which are characterized by two timescales, one related to wind and one to waves, that current PI controllers are not able to manage optimally.

The MPC can be designed to maximize the energy production, minimize the loads perceived by the structure, or reduce the actuator error. MPC relies on an accurate representation of the dynamic system, which is generally based on a linearized physics-based model. Another approach to modeling the dynamical system is to use data-driven surrogates that rely on high frequency sensors to make estimates of the quantity of interest on the go.

1.3. Literature review

In the last decade, there has been significant research in both, surrogate modeling and model predictive control [19]. MPC has already been successfully implemented in wind turbines in numerous studies reported by Li et al. [20]. Hure et al. describe step-by-step the process of applying a model predictive control to an onshore wind turbine. In this study, loads acting on the turbine were modeled with a simplified linear model of the turbine dynamics that assumes rigid blades and uniform wind speed over the rotor, while tower fore-aft deflections were approximated using first mode [21]. Guadayol Roig also applied MPC to wind turbines using different techniques: linear and non-linear MPC, were implemented, along with LIDAR-assisted MPC [18]. In this case, the design was possible thanks to BLADED simulations but the results, including the loads acting on the structure, were eventually expressed as ten-minute averaged values.

There are different types of SM: white, grey and black box models. White models, also referred to as glass box models or interpretable machine learning models, are the SM that are easily understandable by humans [22]. This is because they are usually based on tools that are also used in our reasoning process: physics laws and regressions. On the other hand, black box SM infer underlying functions purely from input-output data pairs. They are especially useful when the system to be modeled is very complex or the closed-form solution to the mathematical relationship between variables is intractable [22]. The relationship between the input and the output is obscure to the users. In between these two alternatives, there are grey models. These are hybrid models that apply physical constraints to the solution space obtained from data [23].

When the interaction with MPC is considered, normally physics-based models are used to provide a linearized representation of the more complex system that is being studied. However, this solution is not always viable since linear representation may require to oversimplify the model, thus, the correct behavior is not captured. If this is the case, more complex alternatives may be necessary. Black box solutions are considered a promising option for multiple reasons [23]. First, developing black box SM requires minimal domain knowledge, even less than implementing grey box models. Next, it is easier to transfer the same framework to different wind turbines; thus, a more generalized approach might also be developed. Finally, the process of training data-driven surrogate models is well established. All these aspects together should contribute to achieving a faster implementation. This is a parameter to consider since it would help the industry lower the LCOE.

Nevertheless, it is not a straightforward process. Data quantity and quality that are available for training the SM may not be optimal for a successful implementation. This is because data-driven models use historical data to capture the system's underlying dynamics, which may not be completely accurate at all times due to inaccuracies of the measuring devices or missing measurements [23]. Furthermore, the computational complexity required from the machine used to perform the training of the SM may reach very high levels since the applied model is non-linear and non-convex and the performance of the algorithm is not assured [23]. Finally, it must be noted that the conditions (geometrical or environmental) for which the model was trained may vary in time. If that is the case, updating the SM by re-calibrating the model would also be necessary.

However, in previous studies, data-driven SM have already been applied to estimate the load statistics on turbines, both onshore and offshore. More in detail, this type of information can be of interest for different purposes: in some cases, it is to monitor the fatigue load of existing projects, while in other studies it is used in initial calculations to perform a design load assessment. Even if the output in both cases is the same, the input data is different. In the former one, most likely real data from Supervisory Control and Data Acquisition (SCADA) measurements will be used, while in the latter, only atmospheric conditions such as wind and wave data are implemented to simulate environmental conditions. A summary of the literature on the subject can be found in Table 1.1.

Dimitrov et al. in 2018 [35] present a procedure for estimating site-specific load statistics using various data-driven surrogates. The Gaussian process regression model is shown to perform the best in terms of the prediction accuracy, but with a time penalty. To train all SMs, a database with aeroelastic load simulations performed on the DTU 10 MW reference turbine was used. In [26], Dimitrov and Gorcmen focused on using surrogate models to capture the change in loads of an onshore turbine in the wake of another one expressed as a time series. To train the SM in this case simulations results from HAWC2 were generated by randomly sampling average ambient wind speed at hub height, ambient turbulence intensity, wind

Table 1.1.: Summary of studies using black box methods

Surrogate Model	Reference	Location	Data source	Purpose	Output's time-step
Neural Network	Cosack, 2010 [24]	Onshore	OpenFAST	FLM	10 min
	de Nolasco Santos, 2021 [25]	Offshore	Real (SCADA)	FLM	10 min
	Dimitrov, 2022 [26]	Onshore	HAWC2	FLM	0.04 – 0.1 sec
	Muller, 2018 [27]	Offshore	OpenFAST	DLA	
	Lee, 2019 [28]	Offshore	Real	FLM	10 min
	Obdam, 2010 [29]	Offshore	Real (SCADA) and simulated	FLM	10 min
	Schroder, 2018 [30]	Onshore	Simulated	DLA	
	Schroder, 2020 [31]	Onshore	HAWC2	DLA	10 min
	Smolka, 2014 [32]	Offshore	Real	FLM	10 min
	Souliotis, 2013 [33]	Offshore	BHawC	FLM	10 min
	Venu, 2020 [34]	Offshore	Real (SCADA)	FLM	10 min
Polynomial Chaos Expansion	Dimitrov, 2018 [35]	Onshore	Simulated	DLA	10 min
	Murcia, 2018 [36]	Onshore	HAWC2	DLA	10 min
	Schroder, 2020 [31]	Onshore	HAWC2	DLA	10 min
	Slot, 2020 [37]	Offshore	OpenFAST	DLA	10 min
Kringing	Dimitrov, 2018 [35]	Onshore	Simulated	DLA	10 min
	Okpokparoro, 2020 [38]	Offshore	Simulated	DLA	
	Slot, 2020 [37]	Offshore	OpenFAST	DLA	10 min
Bayesian Quadrature and Gaussian Processes	Clark, 2022 [39]	Onshore	Simulated with OpenFAST	DLA	
Heteroscedastic Gaussian Process Regression	Singh, 2022 [40]	Offshore	OpenFAST	DLA	10 min

shear exponent, distance to the upwind turbine in the rotor diameters and yaw misalignment of the downwind turbine. In both cases, the results were compared to actual data using normalized root mean square error [26]. In particular, the parameters taken into account were tower base fore-aft moment, tower base side-to-side moment, tower top yaw moment, main shaft torsion, blade root flapwise moment and blade root edgewise moment. Given the similarity of the input data, these studies are mainly considered as references for the considered parameters and errors evaluation.

The majority of the analyzed studies post-processes the results of the simulations to obtain 10-minute statistics. This is because typically wind data considered for the studies in the field is processed over 10 minutes. When analyzing the application to MPC, the sampling frequency needs to typically be much higher [41].

1.4. Research questions

Given the reported literature review, a gap in this research field has been identified in the time series forecasting of loads for offshore wind turbines. As a result, the main research question can then be formulated as follows:

Can data-driven surrogate models be used for forecasting load time series on offshore wind turbines?

To answer this question, multiple sub-questions were identified. First, the environmental conditions, for both wind and waves, have to be modeled. These have to be representative of all the operating conditions that can occur in the selected location, 30 km offshore the coast of Ireland in the Atlantic Ocean. The most representative parameters have to be identified and sampled. Thus, one sub-question is:

What variables should be chosen to represent the environmental conditions and how should they be sampled to generate the best training database?

Once the environment is modeled, the turbine's parameters that are taken into consideration when creating the data for training have to be chosen. It is important in this case to assess what features have the biggest impacts on loads and how they interact with each other. To reach this aim multiple options are viable. In this case cross-correlation is applied. Therefore, the second sub-question can be formulated as:

Which of the computed features influence the target loads the most?

Next, once the training data set has been created, different black box SM will be tested to achieve the best approximation. In order to assess if they are working or not, some specific parameters have to be identified. To do so, it is essential to understand the most sensible points on the turbine's blades. Thus, the third sub-question is:

What are the most load-sensible points on the blades?

Once these aspects have been addressed, different characteristics regarding the training of the model should be assessed. In particular, features such as the way to achieve the best compromise between results accuracy and computational time. Another example could be the definition of the parameters of the implemented models as well as which are the variables that actually influence the output the most. To this aim, another sub-question can be formulated as follows.

What features should be chosen for the implemented SM and which exogenous variables should be considered as inputs for each target parameter?

Finally, between the different SM considered, comparisons and recommendations on the best solutions have to be made. To do so, the results of the model will have to be compared to the ones of the aero-hydro-servo-elastic simulations to determine relative errors and deviations. Once this is finalized, it will be straightforward to determine the model that worked best and the reasons why that happened. As a result, the final sub-question can be formulated as:

What time series forecast model works best to accurately predict the loads and why?

Once these questions are answered, it should be possible to answer the main question. This would allow to reach the main goal stated in Section 1.1.

1.5. Research steps

In this thesis, the training data is obtained through hydro-aeroelastic simulations conducted via OpenFAST and not from real measurements. This is because firstly, when information is achieved through software, it usually samples the operating conditions of the turbine more uniformly, so it is more useful for the training of the algorithm. Secondly, retrieving enough real data from the chosen location was impossible. This area is particularly interesting because it shows numerous designs that still have to be realized, and the water depth is approximately sixty-five meters. This is convenient since it allows to start working with a monopile design, given that the level is still relatively shallow, but at the same time, it is deep enough for the floating installation. Thus the same study can be of interest to further studies investigating the implementation of Floating Offshore Wind Turbine (FOWT). The reference turbine is the National Renewable Energy Laboratory (NREL) 15 MW offshore monopile turbine, which at the moment represents state of the art in its category [14].

A MATLAB framework is created to sample from the joint distribution of environmental conditions and auto-generate input files for OpenFAST. Next, these files are uploaded to the TU Delft Aerospace Department's High-Performing Computer (HPC), where they are run with the open-source software OpenFAST v3.0.0. Both BEM and OLAF aerodynamic models are applied.

Once the results of the simulations are available, data is processed first to determine the differences in the results when the two different modeling methods are applied and later to identify the relationship between variables. To do so, Python's packages 'pyFAST' and 'statsmodels', which allow respectively to import OpenFAST results in Python and to perform time series analysis and modeling, are used. Next, 500 simulations are run in series through the definition of a pipeline on HPC to create the required data set to train the SM. Throughout these simulations, BEM is used to compute the aerodynamic forces.

When the training data set is complete, Python codes are created to implement the two chosen algorithms: Autoregressive Integrated Moving Average Exogenous Variable Models (ARIMAX) and Long Short-Term Memory (LSTM) network. To this end, packages 'pmdarima' and 'keras' are used. Finally, to evaluate the errors, RMSE from the 'sklearn' library, along with data visualization from 'matplotlib', is used for both SM.

1.6. Framework

The thesis structure for the remaining chapters is presented here following the path traced by the defined sub-questions. Chapter 2 presents the theoretical framework of data-driven models and some considerations on the applications that can be found in the literature for load forecasting. The main focus is on the two chosen methods for this analysis, ARIMAX and LSTM, which are explained and compared.

Next, in Chapter 3, the procedure used to create a training model setup is presented. In particular, first, the description of how environmental conditions are modeled, together with considerations between different sampling methods is presented. Later, the offshore turbine model implemented in OpenFAST with the considered degrees of freedom is introduced, followed by a comparison between BEM and OLAF.

Later, the correlation between different variables is investigated in Chapter 4. Cross-correlation is applied to this aim. First, the theoretical framework is described, followed by the application details. Finally, the results of this analysis are presented; thus, the inputs and target outputs implemented in the surrogate models are identified.

In Chapter 5, the implementation of ARIMAX and LSTM is analyzed. This includes the explanation of the creation of the data set followed by the description of the used algorithms. Furhtermore, the analysis performed to select LSTM hyper-parameters is shown.

The results are later displayed and expressed in terms of RMSE in Chapter 6. This leads to an evaluation of the performed analysis and a comparison between the two implemented models.

Finally, in Chapter 7, an overview of the conducted study is provided, along with a summary of the performed analysis. The report is then concluded with recommendations for future work.

Chapter 2

Data-driven surrogate models

Data-driven surrogate models aim to model systems based on input-output data sequences provided when training the algorithm. They are widely used to emulate the behavior of complex systems that are either expensive to compute or governed by intractable or unknown equations.

The surrogate model acts as a low-cost approximation of the actual process, allowing efficient and fast evaluations in tasks such as optimization and sensitivity analysis. In this project, the sensor measurements and the environmental conditions are used as inputs to obtain the moment acting at the blade's root as output. The required data has to be representative of the operating condition in which the system that has to be emulated works as well as the system's response. As a general rule, usually 80% of the available environmental data is used to train the algorithm and 20% to test its performance. The general logic behind their working mechanism can be visualized in Figure 2.1.

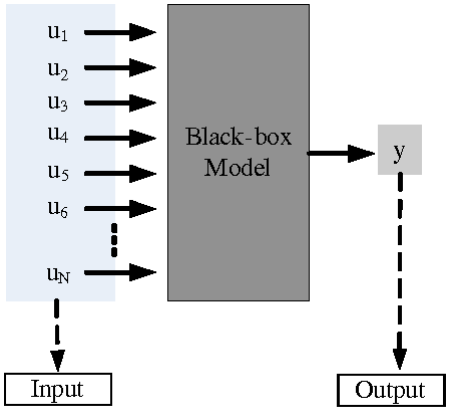


Figure 2.1.: Schematization of black box surrogate models [2]

Black box surrogate models are applied within many different fields, from finance to physics. In this study, they are used for forecasting load signals that can be fed into the MPC, which would then use this information to make control decisions to minimize the loads.

This chapter presents some considerations and previous studies on their application for load forecasting in Section 2.1. This is followed by the introduction of the data used in this study in Section 2.2. Next, a more in-depth explanation of the chosen methods, ARIMAX and deep learning techniques with a particular focus on LSTM, is reported respectively in Section 2.3 and Section 2.4. Finally, the chapter is concluded with a comparison of the two previously analyzed methods in Section 2.5.

2.1. Literature review

As it can be deduced by the table presented in Section 1.3, various data-driven models have been applied to estimate the loads acting on wind turbines. All the methods mentioned in the table, namely Polynomial Chaos Expansion (PCE), neural networks (NN), and Kringing method, apply non-linear relationships to forecast the loads acting on turbines.

However, we are interested in high-frequency loads forecast, which has not been extensively studied in the context of MPC. In the first step, the performance of ARIMAX model is evaluated. A closely related algorithm, AutoRegressive with eXogenous input (ARX) method has been previously used to forecast wind turbine power [42]. ARX can be interpreted as an ARIMAX algorithm without integrated moving average. This linear model incorporates past values of both the variable that has to be predicted and other input variables, referred to as exogenous inputs. It is commonly implemented when the target parameters show some auto-correlation and correlation with the variables considered as exogenous inputs. In the case analyzed by Duran et al. [43], the time-step considered for the gathered data is one hour, while forecasts are done for a time horizon of six, twelve, and twenty-four hours. In the study conducted by Aalborg Nielsen, the power output data is gathered with a time-step equal to five minutes but it is then averaged over thirty minutes [42]. This study evaluates prediction horizons from thirty minutes to three hours. It is concluded that ARX performs better for horizons up to one and a half hours while adaptive predictors, such as NN, return more accurate forecasts for longer time horizons.

Within the studies reported in previous Table 1.1, two apply LSTM for the predictions of loads acting on wind turbines. These are the ones by Dimitrov [26] and Lee [28]. In the former study, LSTM is applied to detect the wind turbine center location, to forecast blade root bending moment and the blade tip-tower clearance. To do so, high-frequency SCADA data is used as input and the results are presented as a time series characterized by a different frequency depending on the target variable. Lee, instead, uses LSTM to generate damage equivalent loads estimations for different locations along the tower while applying 10-minute statistics [28]. Given the output of his study, it is concluded that LSTM outperforms feedforward neural networks for all the target locations, even when trained with less than 2000 data points.

Two other studies were also retrieved that investigate other areas using this method. The first one regards the forecast of wind speed and solar irradiation [44]. It is relevant to this study since also in this case the stochastic nature of the wind speed heavily influences the loads perceived by the turbine; thus it is essential to get a good estimate of its behavior. The second area where LSTM is applied regards the forecast of electrical loads for a different range of future windows. The study of Muzaffar and Afhsari also compares traditional methods, such as ARIMAX to more advanced techniques and concludes that implementing more sophisticated algorithms significantly improves the prediction [45]. As a matter of fact,

both the RMSE and the Mean Absolute Percentage Error (MAPE) calculated for the application of LSTM are significantly lower compared to those obtained via ARIMAX. This is the case for all the considered forecast horizons spanning 24 hours to 30 days.

2.2. Data description

The data used in this study is introduced to better explain the following data-driven models. This is composed exclusively of numbers arranged in a 4-dimensional matrix. The first dimension is used to separate data coming from different simulations. In this analysis the database is generated through the results of 200 different output files; thus this is the first dimension. The second dimension represents the number of sections per simulation in which the data is divided. Each of these is made of 40 time steps, and the total length of the simulation is 800 time steps, thus the second dimension is equal to 20. The third dimension is the length of the section which is equal to 40 time steps as previously mentioned. Finally, the last dimension represents the number of variables considered. This varies between 3 and 5 depending on the case that is being analyzed.

In the following paragraphs, the operations described can be considered as happening in each section; thus, two-dimensional operations will be described. Furthermore, $\boldsymbol{y}$ will be used to denote the target variable vector and $\boldsymbol{x}$ to indicate the exogenous variable vector.

2.3. Auto-regressive integrated moving average models with exogenous variables

ARIMAX are data-driven surrogate models commonly used for time series analysis and forecasting. They integrate three different mathematical concepts within the same algorithm: auto-regression, differentiation, and moving average. The variables taken into account are the past values of the analyzed parameter and exogenous variables. First differencing is performed, next autoregression, and finally the moving average operation.

Auto-regression assumes that the output $\boldsymbol{y}_t$ at a certain time-step t depends on the values it assumed in previous time-steps and on the value of other external variables [46]. In particular, it assumes that $\boldsymbol{y}_t$ can be expressed as a linear combination of the previously mentioned features. An auto-regressive model of a certain order p can be defined as expressed in the following Equation 2.1. Here $\boldsymbol{y}_t$ and $\boldsymbol{x}_t$ are stationary, $\boldsymbol{\phi}_1$, $\boldsymbol{\phi}_2$, ... $\boldsymbol{\phi}_p$ are vectors of constants and $\boldsymbol{w}_t$ represents white noise [47]. The linear equation coefficients are typically solved using least squares, Maximum Likelihood Estimation (MLE), or recursive estimation.

$$\boldsymbol{y}_t = \boldsymbol{\phi}_1 \boldsymbol{y}_{t-1} + \boldsymbol{\phi}_2 \boldsymbol{y}_{t-2} + ... + \boldsymbol{\phi}_p \boldsymbol{y}_{t-p} + \boldsymbol{\phi}_{p+1} \boldsymbol{x}_{t-1} + \boldsymbol{\phi}_{p+2} \boldsymbol{x}_{t-2} + ... + \boldsymbol{\phi}_{p+p} \boldsymbol{x}_{t-p} + \boldsymbol{w}_t \quad (2.1)$$

The moving average model represents the current value of a time series as a linear combination of past residuals [47]. It assumes that the current value of the time series depends on the average of the recent error terms, denoted using $\boldsymbol{e}$, which represents the fluctuation in the data. It can be defined as in Equation 2.2. Here q represents the number of lags in

the moving average and θ_1, ... θ_t are vectors of constant coefficients. It is usually implemented to smooth out the noise or random fluctuations present in a time series, revealing the underlying trends or patterns.

$$y_t = e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + ... + \theta_q e_{t-1} \tag{2.2}$$

Differentiation is implemented to transform a non-stationary time series into a stationary one [47]. This is very important to achieve an accurate forecast since it helps in removing trends and seasonality from the data, allowing to capture better the relationship between exogenous and endogenous variables and a better understanding of the underlying dynamics of the series. To achieve stationarity, the data is expressed as the combination of two time series: one stationary and one non-stationary. Once it is differentiated, only the stationary part remains. It is assumed that $y_t = \mu_t + f_t$ where $\mu_t = \beta_0 + \beta_1 t$ and f_t is stationary. The differentiation of this variable would lead to Equation 2.3.

$$\nabla y_t = y_t - y_{t-1} = \beta_1 + f_t - f_{t-1} = \beta_1 + \nabla f_t \tag{2.3}$$

Designing the appropriate sampling space is an important part of developing a data-driven model, thus also for ARIMAX. The collection of environmental conditions - model response data pairs has to be sufficiently well distributed over the sampling area to cover the whole range of operating conditions and a sufficient time span to capture the system's dynamics.

The following step involves determining the appropriate order of the model. This is essential for the accuracy of the forecast since if it is inadequate, it may lead to under- or overfitting. Moreover, the higher the orders are, the more heavy does the model become from a computational point of view. Finally, these orders heavily influence the estimation of the coefficients, which are the means through which the model is interpreted.

For each of the three mathematical concepts involved, a number that best expresses the relation for each case is identified. Furthermore, the order attributed to each exogenous variable is determined based on its impact on the desired output. In this study, to determine the optimal order for an ARIMAX model, the Hyndman-Khandakar algorithm is implemented. This step-wise approach initially searches for the best values for the orders based on the Akaike Information Criterion (AIC) score. Next, starting from the parameters determined in the previous step, applies again AIC to determine the new orders. The features of this criterion are explained in more detail in the next subsection 'Akaike Information Criterion'.

The estimation of the parameters involves minimizing the negative log likelihood (MLE), essentially equivalent to maximizing the log likelihood. MLE measures the probability of observing the given data under the assumed model.

Finally, the calibrated ARIMAX model is used for making forecasts on data unknown to the SM (such as real output data that is not used to train the model) or using cross-validation techniques. This allows the user to estimate the accuracy and reliability of the model.

This algorithm is rather simple compared to other data-driven models, and this allows the users to have a good understanding of the functioning of the modeled system thanks to the interpretation of the determined parameters. However, it is not convenient to apply ARIMAX

for highly non-linear systems and it can be challenging to determine the correct model's orders.

Akaike Information Criterion

AIC is a statistical measure used to select and compare different models fit [48]. It is designed to trade off complexity and accuracy, providing a quantitative measure. AIC is defined as follows in Equation 2.4.

$$AIC = -2log(\mathcal{L}) + 2k \tag{2.4}$$

Here k represents the number of parameters that have to be estimated by the model and it expresses the model complexity. $log(\mathcal{L})$ quantifies the probability of observing the given data under the assumptions of the model. The higher it is, the best the model fits the data. In mathematical terms, it can be described as the sum of the likelihood functions of each term, each of which can be expressed as in Equation 2.5. In this case, σ is the variance of the residuals while e_i is the residual for the i-th observation.

$$log(\mathcal{L}_i) = -0.5log(2 * \pi) - 0.5log(\sigma^2) - 0.5\left(\frac{e_i}{\sigma}\right)^2 \tag{2.5}$$

The best model is the one that results in the lowest value of AIC. As a matter of fact, if the complexity is too high, the term k becomes higher but, on the other hand, if the estimate is not good the $log(\mathcal{L})$ increases.

2.4. Deep Learning Techniques

Given the linear nature of the method introduced in the previous Section 2.3, it may be necessary to implement a more complex algorithm better to capture the future behavior of the analyzed phenomenon. For this purpose. deep learning techniques can be applied. These are techniques based on neural networks constituted of three or more layers. This section gives a brief overview of how these models work. First, in Section 2.4.1, the NNs are presented. This is necessary to understand the concepts at the base of Section 2.4.2, where LSTM is explained. Finally, in Section 2.4.3, the encoder-decoder architecture applied to LSTM is described.

2.4.1. Neural Networks

Neural networks are a subset of data-driven machine learning techniques that are at the core of deep learning algorithms and can be used for regression and/or classification purposes [3]. They are composed of different nodes, or artificial neurons, that are organized in layers. The first and last layers correspond respectively to the input and output vectors. In between, there can be one or more hidden layers. The input layer receives the initial data and the

number of neurons in this layer depends on the dimension of the input data. Similarly, the output layer receives the transformed representation of the data from the previous hidden layer and maps it to the output vector during the training phase. In this case, the number of neurons depends on the nature of the problem that the algorithm aims to solve. The number depends on the difficulty of extracting meaningful relations between the provided data. As the amount increases, the algorithm can learn increasingly abstract and complex features. These can be represented as shown in the following Figure 2.2.

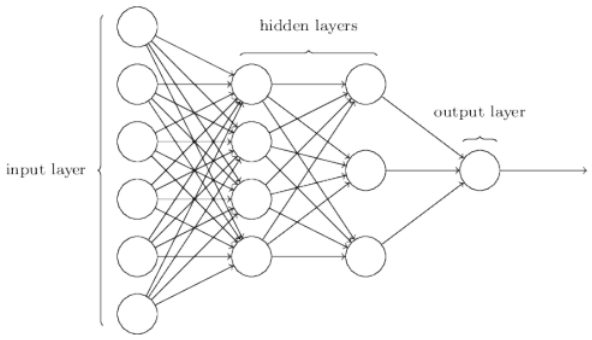


Figure 2.2.: Schematization of NN architecture [3]

Each node, or artificial neuron, has its associated weight, bias, and set threshold. Once the input reaches the node, it is multiplied by the weight, and the bias is added to the resulting value. Only if this is higher than the set threshold, then the node is activated, and the information it contains is passed on to the next layer; otherwise, no data is transferred along the network. Different activation functions can be used to calculate the value attributed to each node. These are usually non-linear expressions that are similar to each other since they provide outputs values comprised of either between 0 and 1 or between -1 and 1. The most used ones are the sigmoid, the hyperbolic tangent, and the Rectified Linear Unit (ReLU) functions. The mentioned functions are overviewed in Figure 2.3.

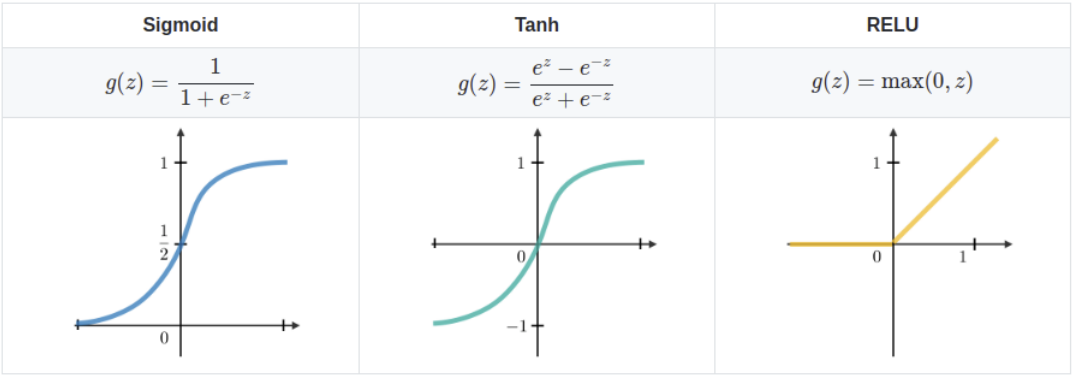


Figure 2.3.: Summary of commonly implemented activation functions

Forward propagation is used to activate each layer depending on the inputs received from the previous layer. This is outlined in the following Algorithm 2.1 [49]. Here a_j is the activation of the j-th unit; w_{ji} and b_{ji} are the weights and biases associated to the activation

of the j-th unit corresponding to the inputs of the i-th unit; z_j represents the intermediate status computed in the j-th unit and $g(\cdot)$ is the activation function.

Algorithm 2.1: Forward Propagation

Input: a_i, w_{ji}, b_{ji}
Output: a_j

1 $z_j = \sum_i w_{ji} a_i + b_{ji}$

2 $a_j = g(z_j)$

The weights and biases of each node are determined based on an algorithm that aims to minimize a cost function; thus they are the result of a minimization problem. If w denotes the collection of weights and b the biases, the cost function $C(\boldsymbol{w}, \boldsymbol{b})_n$ can be defined as shown in Equation 2.6.

$$C(\boldsymbol{w}, \boldsymbol{b})_n = \frac{1}{2} \sum_i ||y(\boldsymbol{x}_i) - t(\boldsymbol{x}_i)||^2 \tag{2.6}$$

Here n denotes the considered input used for training, $t(\boldsymbol{x}_i)$ is the vector of outputs from the network when $(\boldsymbol{x}_i)$ is used as input. The cost function is essentially a mean squared error between the observed output $y(\boldsymbol{x}_i)$ and the value for the same parameter estimated by the network, $t(\boldsymbol{x}_i)$.

Gradient descent is used to minimize the loss function. The back-propagation algorithm is applied to calculate the cost function's gradient. For the biases, the activation can be fixed at +1, thus the intermediate status can be simplified as in Equation 2.7.

$$z_j = \sum_i w_{ji} a_i \tag{2.7}$$

Next, the cost function is used to determine the derivatives of each weight by applying the chain rule from the output layer to the input layer. This is shown in the following Equation 2.8.

$$\frac{\mathrm{d}C_n}{\mathrm{d}w_{ji}} = \frac{\mathrm{d}C_n}{\mathrm{d}z_j} \frac{\mathrm{d}z_j}{\mathrm{d}w_{ji}} \tag{2.8}$$

To simplify the following algorithm, the notation shown in Equation 2.9 is introduced. Furthermore, the second part of the equation can be expressed as in Equation 2.10, remembering that a_i is the activation of the i-th unit. As a result of these considerations, the derivative of the error can be expressed as in Equation 2.11.

$$\delta_j \equiv \frac{\mathrm{d}C_n}{\mathrm{d}z_j} \tag{2.9}$$

$$a_i = \frac{\mathrm{d}z_j}{\mathrm{d}w_{ji}} \tag{2.10}$$

$$\frac{\mathrm{d}C_n}{\mathrm{d}w_{ji}} = \delta_j a_i \tag{2.11}$$

Considering that the derivative of the cost function in respect to the weights can also be written as in Equation 2.12, the previously introduced notation δ_j can be expressed as in Equation 2.13. Here x_{ni} is the input n-th input training for the i-th unit.

$$\frac{\mathrm{d}C_n}{\mathrm{d}w_{ji}} = (y_{nj} - t_{nj})x_{ni} \tag{2.12}$$

$$\delta_j = y_j - t_j \tag{2.13}$$

Now it is possible to apply again the chain rule to the term δ_j to obtain the formula of the backpropagation as in Equation 2.14. In this case, the subscript k is used to indicate the units to which the information contained in the unit j is transmitted.

$$\delta_j \equiv \frac{\mathrm{d}C_n}{\mathrm{d}z_j} = \sum_k \frac{\mathrm{d}C_n}{\mathrm{d}z_k}\frac{\mathrm{d}z_k}{\mathrm{d}z_j} = h'(z_j)\sum_k w_{kj}\delta_k \tag{2.14}$$

Thus, the results obtained from the forward propagation are used as inputs for the back-propagation algorithm reported in Algorithm 2.2 [49]. In this procedure, it is implied that all the different units adopt the same activation function $h(\cdot)$.

Algorithm 2.2: Back Propagation

Input: y_k, t_k, $g'(\cdot)$, z_j, w_{kj}, a_i
Output: $\frac{\mathrm{d}C_n}{\mathrm{d}w_{ji}}$, $\frac{\mathrm{d}C_n}{\mathrm{d}b_{ji}}$

1 $\delta_k = y_k - t_k$
2 $\delta_j = g'(z_j)\sum_k w_{kj}\delta_k$
3 $\frac{\mathrm{d}C_n}{\mathrm{d}w_{ji}} = \delta_j a_i$
4 $\frac{\mathrm{d}C_n}{\mathrm{d}b_{ji}} = \delta_j$

2.4.2. Long Short Term Memory

LSTMs are network architectures designed to model time-series data. They use a specialized system of gates that allows the model to selectively retain or forget information over multiple time steps.

LSTM is a special kind of Recurrent Neural Network (RNN). RNN have an inner loop that allows information to be passed back and forth to the previous layer or time step [50]. This is done through the storing of internal state data. It allows RNN to handle sequencing data, such as time series, since they are capable of capturing temporal dependencies by maintaining the memory of previous inputs through hidden states. A visual representation of this network can be seen in the following Figure 2.4, where it is clearly highlighted the difference with the previous Figure 2.2.

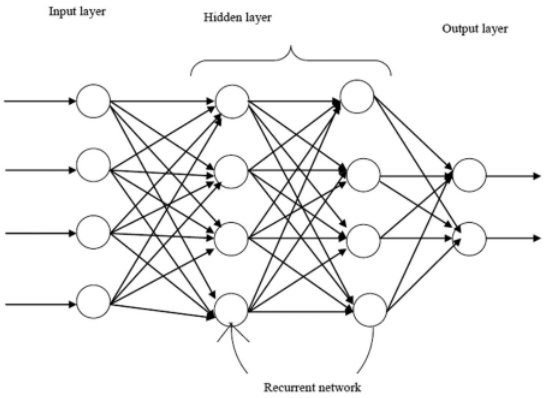


Figure 2.4.: Schematization of RNN architecture [4]

The added advantage of implementing LSTM instead of RNN is that the information can be passed through multiple layers and time steps. This happens thanks to the implementation of memory cells. The structure of a single unit can be visualized as in Figure 2.5.

Each cell can be divided into three main parts: the forget, input, and output gates. To 'move' through them, the two states of the cell are used, the hidden one and the cell state. These respectively represent the short and long-term memory stored in the cell. The unit receives as inputs the variable x_t at time t, the hidden state received from the previous cell, and the cell state from the previous time step. Once the input data and the hidden state input are received, they are multiplied by their respective weights and then summed. The result of this operation is then summed to the bias associated with the input data. The resulting value is the input value for a sigmoid function, which returns a number between 0 and 1. This represents the percentage of long-term memory that has to be carried through the cell state, thus the percentage of long memory that the cell must not forget. The output of the forget gate is then multiplied by the cell state value at time t, which is the output of the unit at $t-1$, to obtain the updated value of the long-term memory.

Similarly, the input data and hidden state input are fed into the second part of the cell. This has four weights and two biases, each of which constitutes the coefficient that respectively multiplies and is added to the values used as inputs and then fed into a sigmoid and hyperbolic tangent function. The values resulting from these two functions are then multiplied

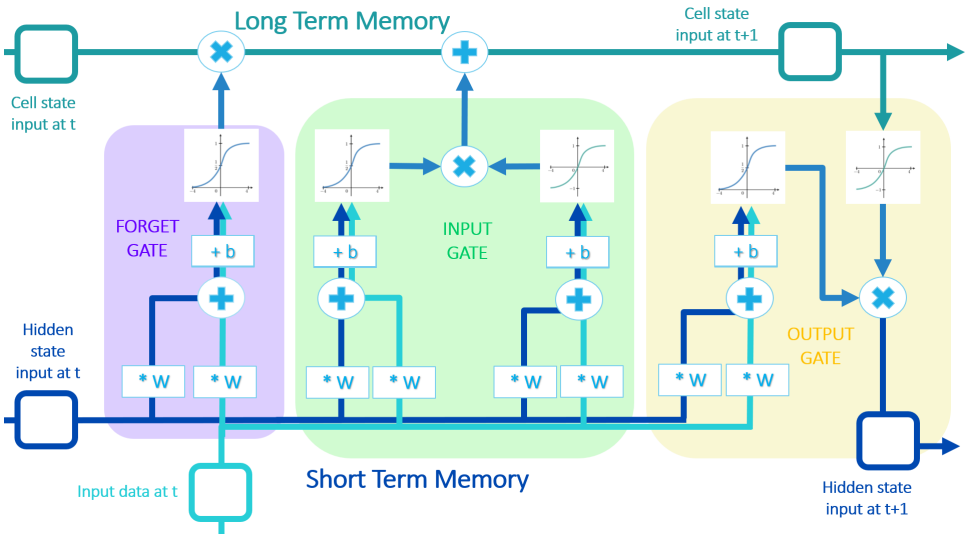


Figure 2.5.: Schematization of LSTM unit architecture

and summed to the value previously calculated for long-term memory. This cell state is used as input for the time step $t + 1$ and represents the new value of the long-term memory.

Finally, the last part of the cell, the output gate, uses the input data and the hidden state input multiplied by their corresponding weights and sums them first to each other and then to the corresponding bias. The obtained value is fed into a sigmoid function. The output of this function is then multiplied by the value obtained when feeding the updated long-term memory into a hyperbolic tangent function. The final resulting number is the hidden state output for time step $t + 1$, representing the new short-term memory.

The activation functions mentioned in the previous paragraphs are the ones displayed in the Figure 2.5. However, these can be substituted with others if desired.

These cells can be organized one in series to the other to create a chain of cells and more chains can be placed in parallel to each other, receiving all the same initial inputs. Together, these combinations constitute a layer, and each cell's cell state and hidden state can be used as inputs for the following layer. This can be better visualized in the following Figure 2.6.

2.4.3. Encoder - Decoder LSTM

In this study, we are using the encoder-decoder architecture of LSTMs. This layout is particularly useful when the length of the input differs from the length of the output sequence, for sequence-to-sequence modeling and to capture very long-term dependencies [51].

The LSTM based encoder processes the inputs to encode the sequential dependencies[52]. Next, they are passed to the decoder, where the encoded inputs will be read and a one-step prediction for each element is made in the output sequence. The main difference with the simple LSTM is that in the decoder the output training data is used, allowing the network to know both what was predicted at time step $t - 1$ and accumulate internal state, while also providing the forecast. The schematic representation of this process is shown in Figure 2.7.

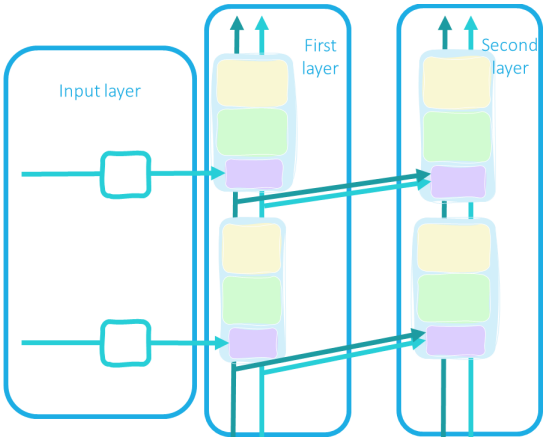


Figure 2.6.: Schematization of LSTM layers visualization

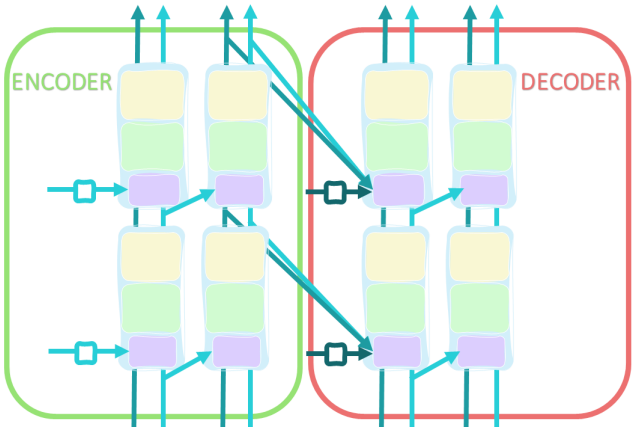


Figure 2.7.: Schematization of LSTM encoder-decoder architecture

2.5. Comparison between ARIMAX and LSTM

ARIMAX models have been widely used for time series forecasting and offer several strengths. Firstly, they provide interpretable coefficients, enabling the analysis of the impact of exogenous variables on the time series. This is valuable in domains where understanding the relationship between variables is crucial. Moreover, ARIMAX models are based on statistical assumptions, allowing for the estimation of confidence intervals and the assessment of forecast uncertainty. They also handle seasonality if needed by incorporating seasonal differencing or seasonal auto-regressive and moving average terms, accommodating periodic patterns in the time series.

However, ARIMAX models have certain limitations. They assume linear relationships between the time series and exogenous variables, which restrict their ability to capture complex non-linear patterns and dependencies. This limitation may lead to non-optimal performance in data sets with intricate relationships. Furthermore, ARIMAX models have limited memory and struggle to capture long-term dependencies or complex patterns that span a large

number of time steps.

On the other hand, LSTM models have gained importance in time series forecasting due to their ability to capture complex non-linear patterns and long-term dependencies. They offer several advantages over traditional models like ARIMAX. LSTMs excel at learning intricate relationships by modeling the interactions between time steps and leveraging their memory cells. This enables them to capture non-linear patterns and dependencies that may exist within the time series data. Additionally, LSTMs can automatically learn relevant features and representations from the raw time series data, reducing the need for extensive manual feature engineering.

However, LSTM models also have limitations. Their nature poses challenges in interpreting the internal workings of the model or understanding the impact of individual input variables. LSTMs typically require large amounts of training data to effectively learn complex patterns and prevent under-fitting, which can be a limitation in scenarios with limited data availability. Furthermore, training and evaluating LSTM models can be computationally intensive, especially for large-scale time series data sets, necessitating substantial computational resources.

In practice, it is beneficial to experiment and compare the performance of both approaches on the specific forecasting task at hand. Conducting comprehensive evaluations and considering the trade-offs in interpretability, model complexity, computational requirements, and forecast accuracy will aid in selecting the most suitable approach for time series forecasting.

Chapter 3

Training model set up

As mentioned in Chapter 2, data-driven models are not suited to extrapolate information beyond the domain within which they have been trained. This is why it is particularly important to identify the correct conditions when creating the training data set.

This process is divided into two stages. First, the environmental conditions related to the specific site are gathered using a sampling method. Then, they are used as input parameters for OpenFAST simulations. This is an open-source simulation tool that combines modules defining aerodynamics, hydrodynamics, control and electrical system dynamics, and structural dynamics to solve the aero-hydro-servo-elastic dynamic responses of a wind turbine [53].

In Section 3.1 the environmental conditions are described followed by the description of the considered sampling methods in Section 3.2. Next, in Section 3.3 the model of the turbine used, together with its considered dynamic degrees of freedom, are analyzed. Finally, in Section 3.4 an overview of the theory behind BEM and OLAF models is provided followed by a comparison of the results obtained implementing the two different aerodynamic models given the same environmental variables.

3.1. Definition of the environmental conditions

Considering the placement of the turbine offshore the coast of Ireland, in the Atlantic Ocean, it is necessary to describe both atmospheric and marine conditions. To this end, different studies from the literature are considered to list the most important site conditions and their corresponding ranges.

Nikolay Dimitrov et al., while analyzing an onshore turbine, identified as relevant parameters to describe a turbulent wind field the average wind speed at hub height, the vertical wind shear exponent, the wind veer (which is the change of wind direction with altitude), the turbulence, the air density and the mean wind inflow direction relative to the turbine [35]. K. Muller & P.W.Cheng identify as relevant sea-state parameters as wave height, wave period, and the difference between wind and wave current directions [27].

Air density, wind shear exponents, and surface roughness are considered constant and respectively equal to 1.225 kg m^{-3}, 0.11 and 0.0002 m. Their variation does not seem to have a significant impact on the damage equivalent loads as shown by Murcia et al. [36].

As a result of these considerations, the following degrees of freedom are identified to describe the environmental conditions:

- Wind parameters
  - Average wind speed at hub height
  - Turbulence intensity
  - Seed number
- Wave parameters
  - Wave direction
  - Significant wave height
  - Peak spectral wave period
  - Seed number

Where the significant wave height is the highest $\frac{1}{3}$ of waves that occur in a certain period of time (e.g., one hour) and the peak spectral period is the wave period associated with the most energetic waves in the total wave spectrum at a specific point, and the seed numbers are random numbers used by OpenFAST to initialize the random number generation.

The definition and distribution of these parameters require particular attention as they determine the successful training of the machine learning algorithm. First, real data is retrieved online [54]. In some cases, this has to be elaborated to obtain exactly the selected degrees of freedom. In particular, wind speed is provided at ten meters above sea level. To extrapolate the desired value, first, the logarithmic boundary layer law is applied considering a desired height of 60 m (h).

$$u(h) = u(h_{ref}) * \frac{ln(h/z_0)}{ln(h_{ref}/z_0)} \tag{3.1}$$

Next, to get the wind speed at hub height, which is equal to 150 m, the power law is applied. This is Equation 3.2, where h is 150 m while h_{ref} is 60 m. The resulting variable is what it will be later referred to as 'URef'.

$$u(h) = u(h_{ref}) * \left(\frac{h}{h_{ref}}\right)^{\alpha} \tag{3.2}$$

Throughout this study, wind direction is assumed to be always aligned with the nacelle. This assumption can be made thanks to the low values of the delta between the two variables and the limited impact that it has on the model [36]. As a result, wave direction is simply defined as the difference between wind and wave direction.

Finally, turbulence intensity is defined as the ambient turbulence described in IEC Class C [55]. This means that it is linked to mean wind speed as defined by Equation 3.3

$$TI = \frac{0.12 * (0.75u + 5.6)}{u} \tag{3.3}$$

Real site-specific data is retrieved for 30 years, from 1992 to 2021, with a time-step of three hours. The resulting relationships between variables along with the marginal distribution of each one, after the manipulations described above, can be observed in the following Figure 3.1.

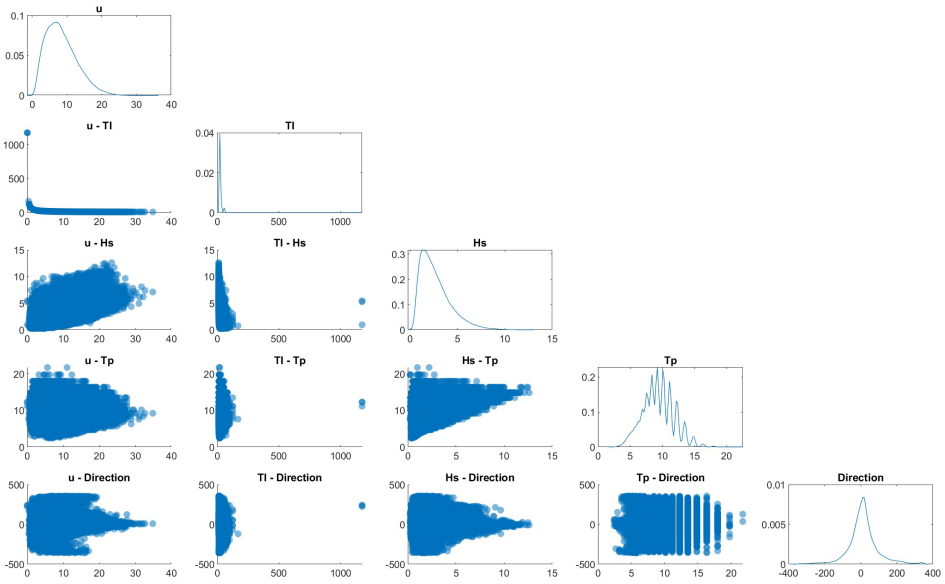


Figure 3.1.: Environmental data.

The maximum and minimum ranges of the chosen parameters are defined taking into account the physical relationships between the degrees of freedom and leaving a certain margin to ensure that the reference site is well covered in the analyzed space. To generate data within these limits, various sampling methods are considered and these are presented in the next Section 3.2.

3.2. Sampling methods

Given the high number of degree of freedom (DoF) and the fact that the computational cost of the method scales with the number of values used to discretize each DoF, it is not convenient to systematically sample every possible combination of them. Consequently,

different alternative approaches are considered to generate load cases within the previously mentioned limits.

Random number generation aims to generate a sequence of real numbers that simulates a series of independent and identically distributed random variables having a certain distribution function [56]. In particular, a uniform random number generation is considered for this study. The issue with this kind of sampling is that it does not provide an evenly distributed representation of the sample area and, if the simulation requires huge amounts of random numbers that must be available for later validation storage, difficulties may arise.

Quasi-random methods represent a possible alternative that aims to provide a spatially balanced sampling of the selected area [5]. They have been applied in many fields, including numerical integration and optimization, and environmental sampling. In particular, the Halton sequence is a quasi-random number sequence that spreads points evenly for relatively small dimensions, until 10-dimension spaces. It is particularly useful in the aforementioned study areas since it generates evenly spread points, just like a regular grid or lattice. The added advantage is that points can be added incrementally without resulting in clumping of points.

The advantages of using quasi-random methods compared to the other presented option are multiple [57]. First, quasi-random methods are more efficient in higher dimensions because they are able to cover a sample space with fewer sample points, while for random sampling the number of required points increases exponentially with the sampling space. Second, random methods may lead to some areas being over-represented or under-represented in the sample. Quasi-random methods avoid this phenomenon. Furthermore, they have faster convergence rates [58]. This means that they provide a better representation of the sampling area as this increases. Finally, they can improve the accuracy of simulations, and eventually of the predictions of the surrogate model, since they provide a more accurate representation of the sampled space. In Figure 3.2 and Figure 3.3 the differences between a random and a Halton sampling are shown.

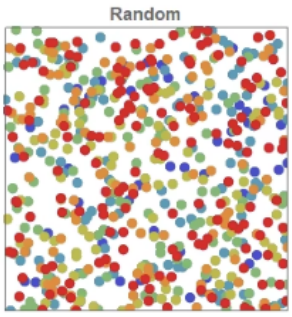


Figure 3.2.: Random sampling [5]

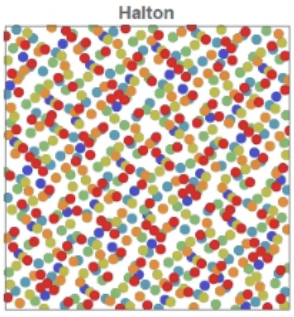


Figure 3.3.: Halton sampling [5]

Given the aforementioned reasons, quasi-random sampling is the best choice for this study. This aims to provide a methodology that can later be applied to different areas, where the sample area may be much larger than in the analyzed case. The Halton set is implemented in MATLAB to generate more environmental data, within the limits defined using the real data as a base. In particular, a Halton set in seven dimensions using the scramble method 'RR2' is applied. The RR2 scramble method consists in a permutation of the radical inverse coefficients that aims to break correlations between the inverse radical functions of different

dimensions [59]. The radical inverse function is widely used to generate uniform random patterns or sampling points within a multidimensional space. The scrambling method is derived by applying a reverse-radix operation to all the possible coefficient values. This is necessary to improve the performance of the function 'haltonset' when applied to multidimensional areas. The resulting plots can be observed in Figure 3.4.

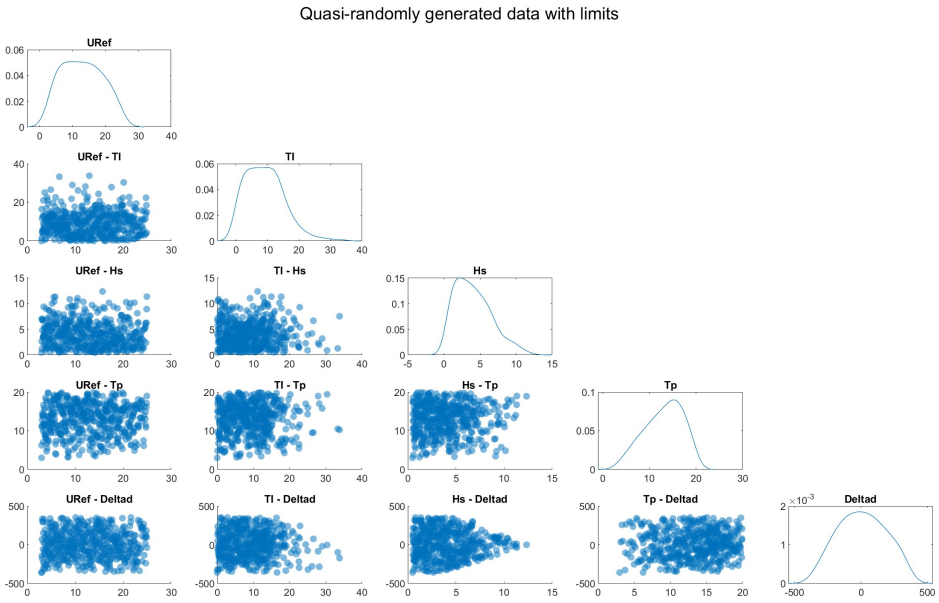


Figure 3.4.: Generated environmental data.

As it can be noticed, the marginal distributions differ from the ones of the real data. This is directly linked with the desire of having a more uniform distribution of the data points across the sampled area which is achieved by applying Halton sequencing to generate 500 sampling points for each variable. This manipulation allows to generate better quality data for the SM.

3.3. Definition of the OpenFAST model

The turbine chosen as a model within this study is the 15-megawatt offshore wind turbine with a fixed-bottom monopile support structure developed by NREL and Technical University of Denmark (DTU) [6], which can be observed in Figure 3.5.

In this section the properties of its components and the alterations applied for the definition of the load cases to generate the database are outlined. In particular, Section 3.3.1 describes the blades and rotor properties, Section 3.3.2 the tower ones, followed by Section 3.3.3 and Section 3.3.4 that highlight respectively the properties of the nacelle and controller. Finally, in Section 3.3.5 the settings of the simulations run in OpenFAST are introduced.

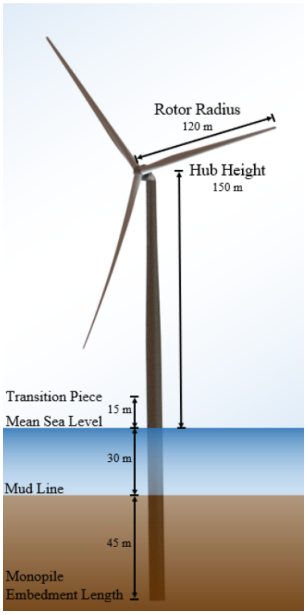


Figure 3.5.: 15 MW offshore wind turbine [6].

3.3.1. Blade and rotor properties

The rotor consists of three blades with a diameter of 240 meters [6]. These are made of two main spars, reinforced with carbon, that carry the loads and are connected through two shear webs with reinforcement along the trailing and leading edge and foam fillers. The maximum length of each blade is 117 m, with a rotor diameter of 5.2 m and a maximum chord of 5.77 m at approximately 20% span. Different blade designs are used along the blade and the aerodynamic center of the airfoils is used for the blade pitch axis.

The blades are designed with a significant pre-bend away from the tower to provide additional tip clearance [6]. As a result, there are 4 m separating the tip chord line from the root. Even though accentuating this feature even more would have given further margin, reducing stiffness requirements, this is not currently possible because of blade molding and other manufacturing challenges. The maximum tip speed that they reach is $95\,\mathrm{m\,s^{-1}}$ and their first flapwise and edgewise natural frequencies are respectively 0.555 Hz and 0.642 Hz.

To calculate the aerodynamic forces acting on the blades, BEM theory is used. This is going to be further discussed in Section 3.4.

3.3.2. Tower properties

The tower is fixed to the ground through a monopile foundation. In the modeling of the turbine, both components are designed as an isotropic steel tube [6]. In particular, the monopile has a 10-meter outer diameter, which pushes the limits of current manufacturing and installation technology, and a thickness profile that varies from 0.055 m in the pile to 0.044 m in the transition piece.

The first tower-monopile mode is 0.17 Hz and lies between the 1P and 3P blade passing frequencies for all wind speeds comprised between 3 m s^{-1} and 25 m s^{-1}. These are respectively cut-in and cut-out wind speed.

Hub height is 150 m. This allows for 30 m clearance spacing from the mean sea level from the lowest point reached by the rotating blades. Conversely to what is shown in Figure 3.5, the mean sea level is not 30 m but 65 m. This parameter is modified to fit the conditions of the target area and to place the analyzed turbine in a location suitable for both bottom-fixed and floating technologies.

3.3.3. Nacelle and drivetrain properties

For offshore wind turbines, direct-drive is the most common choice for multiple reasons [60]. First, gearboxes are susceptible to failure. The occurrence of this event increases as the wind speed increases and offshore the wind is faster, compared to the onshore location. Furthermore, the gearbox requires maintenance, which is more difficult to perform offshore. Second, they imply transmission losses. Moreover, direct-drivetrain are less complex, due to the presence of fewer parts. Finally, they allow for more flexibility in the design phase for special topologies. This feature may be useful when applying the same study to different locations. On the other hand, they require larger rotor dimensions. Nevertheless, the trend of wind turbine designing, especially for offshore applications, aims to build bigger rotors as technology progresses, since they entail a higher power production.

3.3.4. Controller properties

As a controller, the turbine model implemented in this study uses NREL Reference Open-Source Controller (ROSCO). This is calibrated to operate with a minimum rotational speed of 5 rpm to avoid 3-period interference with the tower/monopile natural frequency [6]. It reaches a rated rotational speed of 7.55 rpm when wind speed is 10.59 m s^{-1}. Furthermore, the rotor operates with a pitch setting of 0° at the design tip-speed ratio Tip Speed Ratio (TSR) but operates with positive pitch (to maximize energy extracted) at low wind speeds to track maximum power while maintaining the minimum rotor speed. At wind speeds above rated one, it pitches to face away from the wind, so that the TSR can be kept as low as possible, while maintaining the same rated generated power, to avoid over-speed.

Between 3 m s^{-1} and 6.89 m s^{-1}, a PI controller on the generator torque is applied to impose 5 rpm, the minimum rotor speed [6]. In ROSCO, the minimum blade pitch angle is defined based on a wind speed estimate, such that D_p is maximized. The angles maximizing minimum blade pitch angles are found a-priori using steady-state blade element momentum analysis. Between 6.89 m s^{-1} and 10.59 m s^{-1} the rotor speed is regulated to operate at the turbine's optimal TSR with a PI controller on the generator torque. Finally, between 10.59 m s^{-1} and 25 m s^{-1} the rotor speed is governed by another PI controller that regulates the blade pitch angle to set the TSR to its rated value, 7.55 rpm.

3.3.5. OpenFAST properties

To link these elements together and simulate the working conditions of a wind turbine, OpenFAST allows the user to choose between different options, in terms of both mathematical models and input files for environmental conditions.

The wind is modelled using a stochastic, full-field, turbulence simulator that aims to provide a numerical simulation of a full-field flow that contains coherent turbulence structures: 'TurbSim'. This allows the user to input various parameters. In this specific case, the Kaimal turbulence model is used and the mean velocity at the reference height in $\mathrm{m\,s^{-1}}$, which will be later referred to as 'URef', and the turbulence intensity expressed as a percentage are provided. No turbulence scaling method is applied because this is usually implemented when a particular standard deviation is desired from the generated wind data. In this case, this is not necessary. The analysis time is equal to 900 s with a time-step of 0.2 s and a grid measuring 285 m in both y and z direction, evaluated in 55 different points. As a result, a wind speed time series is generated, called 'WindVel' in this report.

Regarding the aerodynamic computations, for calculating loads acting on the blades (and the power generation) sprung by the generated wind stream, BEM theory is applied. OLAF is also applied to assess the performance of both methods and further discussion on this topic can be found in Section 3.4. In order to make up for its instability to deal with unsteady conditions, such as unsteady aerodynamics and dynamic stall, the most popular method and the one that has the most support throughout the community, is used: Leishman-Beddoes Model (LBM) [61]. In LBM, the different processes are modeled as first-order subsystems defined by differential equations with predetermined constants that match experimental results. The resulting forces are computed in their normal and tangential components in reference to the chord while the pitching moment at about $\frac{1}{4}$-chord location.

The water depth is fixed to 65 m. There is no offset between still-water and mean sea level. The Jonswap spectrum is applied to model irregular waves. To define the lower and upper frequency limits of the wave spectrum beyond which the wave spectrum is zeroed the inverse of the minimum and maximum peak spectral period of incident waves, respectively equal to 3 s and 20 s, is computed. Next, the wave direction is provided and no current condition is imposed. As a result, a wave elevation time series is generated. This will be used as exogenous inputs in the surrogate models and it is referred to as 'WaveElev'.

Since the turbine is modeled as having a direct drivetrain, the gearbox efficiency is imposed equal to 100% with a ratio equal to one. The generator efficiency, instead, is set equal to 95.756%.

Overall, the degrees of freedom regarding blades and tower are all preserved while the ones regarding drivetrain rotational flexibility and the yaw are blocked. A summary of the DoF analyzed in this study can be found in Table 3.1

The simulations are run for 900 s, of which the first 300 s are disregarded. This is done to avoid those possible initial instabilities to be included in the generated data set.

3.4. BEM vs OLAF

For the aerodynamic model, two different solutions are taken into account: BEM and OLAF. To understand the final choice of using BEM, the two models are described in Section 3.4.1

Table 3.1.: Summary of degrees of freedom preserved

Component	Degrees of freedom
Blades	First flapwise blade mode
	Second flapwise blade mode
	First edgewise blade mode
Generator	All
Tower	First fore-aft tower bending-mode
	Second fore-aft tower bending-mode

and Section 3.4.2, and compared in Section 3.4.3.

3.4.1. BEM

BEM was developed by Glauert in 1935 as a practical solution to the analysis and design of rotor blades [7]. The fundamental idea at the base of the model is to combine axial and angular momentum balance, so momentum theory, with loads determined from a blade element strip theory.

The one-dimension momentum theory brings to the formulation of the Betz limit starting from the equation of continuity. This is equation Equation 3.4, where $\dot{m}$ represents the mass flow and the control volume considered can be visualized in Figure 3.6

$$\dot{m} = \rho * u_0 * A_0 = \rho * u_R * A = \rho * u_1 * A_1 \tag{3.4}$$

Next, an axial momentum balance within the control volume previously defined, assuming that the influence of the pressure on the stream tube is not relevant, results in the following Equation 3.5. Here, T represents the thrust force.

$$T = \dot{m} * (u_0 - u_1) = \rho * u_R * A(u_0 - u_1) \tag{3.5}$$

By applying the Bernoulli equation on both sides of the rotor, it can be derived that the

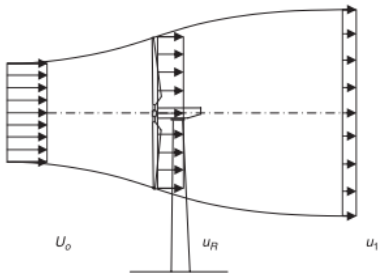


Figure 3.6.: Control volume for one-dimensional actuator disk in momentum theory [7]

difference in the total pressure between the two sides can be expressed as shown in Equation 3.6.

$$\Delta p = \frac{1}{2} * \rho * (u_0^2 - u_1^2) \tag{3.6}$$

From this, the thrust force can also be expressed as $T = A * \Delta p$. Combining this with the previous equation, the wind speed at the rotor can be computed.

$$u_R = \frac{1}{2} * (u_1 + u_0) \tag{3.7}$$

It is convenient now to define an axial inference factor, a, from which the following expressions for u_R and u_1 can be derived.

$$a = \frac{u_0 - u_R}{u_0} \tag{3.8}$$

$$u_R = (1 - a)u_0 \tag{3.9}$$

$$u_1 = (1 - 2a)u_0 \tag{3.10}$$

Consequently, the following equations for thrust and power extraction can be obtained.

$$T = 2\rho A u_0^2 a(1 - a) \tag{3.11}$$

$$P = u_R T = 2\rho A u_0^2 a(1 - a)^2 \tag{3.12}$$

Now the thrust and power dimensionless coefficient can be expressed in terms of the axial interference factor.

$$C_T = \frac{T}{\frac{1}{2}\rho A U_0^2} = 4a(1-a) \tag{3.13}$$

$$C_P = \frac{P}{\frac{1}{2}\rho A U_0^3} = 4a(1-a)^2 \tag{3.14}$$

By differentiating the power coefficient with respect to the axial interference factor, it is deduced that the maximum power can be extracted when C_P is equal to 0.593, which corresponds to a equal to $\frac{1}{3}$.

To integrate the blade element strip theory, first, the axial momentum equation has to be applied to an annulus, comprising two stream surfaces, always disregarding the influence of pressure on the considered area. This results in Equation 3.15, where $2\pi r dr$ is the area of the rotor disk on which the local thrust force, dT, acts.

$$dT = \rho u(U_0 - u_1)2\pi r dr \tag{3.15}$$

By applying the angular momentum balance, the expression for the local torque is the following.

$$dQ = \rho u_R r(u_0 - u_1)2\pi r dr \tag{3.16}$$

Assuming that $u_R = \frac{1}{2}(u_0 + u_1)$ also for this differential element and introducing the azimuthal interference factor $a' = \frac{(u_0 - u_1)}{2\Omega r}$, where Ω is the angular velocity of the rotor, the two previous equations can be rewritten as follows.

$$\frac{dT}{dr} = 4\pi\rho u_0^2 a(1-a) \tag{3.17}$$

$$\frac{dQ}{dr} = 4\pi\rho r^3 u_0 \Omega a'(1-a) \tag{3.18}$$

Employing blade-element theory, local thrust force and torque can be written as shown in Equation 3.19 and Equation 3.20, where B represents the number of blades, c is the local chord length, V_{rel} is the relative flow velocity, F_n and F_t are respectively the load on each blade in the axial and tangential direction and C_n and C_t are the corresponding force coefficients.

$$\frac{dT}{dr} = BF_n = \frac{1}{2}\rho cBV_{rel}^2 C_n \tag{3.19}$$

$$\frac{dQ}{dr} = BF_t r = \frac{1}{2}\rho cBV_{rel}^2 C_t r \tag{3.20}$$

The relationships of C_n and C_t with the lift, C_l, and drag C_d coefficients are now introduced. This is essential to highlight the relation between the velocity triangle and the force coefficients.

$$sin(\phi) = \frac{u_0(1-a)}{V_{rel}} \tag{3.21}$$

$$cos(\phi) = \frac{\Omega r(1+a')}{V_{rel}} \tag{3.22}$$

In the previous formulas, ϕ is the inflow wind angle. As a result of the aforementioned relationships, the square of the relative velocity can be expressed as follows.

$$V_{rel}^2 = \frac{u_0^2(1-a)^2}{sin^2(\phi)} = \frac{u_0(1-a)\Omega r(1+a')}{sin(\phi)cos(\phi)} \tag{3.23}$$

Inserting this expression into the previous Equation 3.19 and Equation 3.20, the following expressions are derived.

$$\frac{dT}{dr} = \frac{\rho BcU_0^2(1-a)^2}{2sin^2(\phi)} * C_n \tag{3.24}$$

$$\frac{dQ}{dr} = \frac{\rho BcU_0(1-a)\Omega r^2(1+a')}{2sin(\phi)cos(\phi)} * C_t \tag{3.25}$$

Combining equations Equation 3.19 and Equation 3.20 with Equation 3.24 and Equation 3.25 the following expressions are found, where $\sigma = \frac{Br}{2\pi r}$ is the solidity of the rotor.

$$a = \frac{1}{4sin^2(\phi)/(\sigma C_n) + 1} \tag{3.26}$$

$$a' = \frac{1}{4sin(\phi)cos(\phi)/(\sigma C_t) - 1} \tag{3.27}$$

Finally, to determine the loads, these equations are solved at different radial crossings along the blade. This allows to calculate the axial interference factor through an iterative process and it is necessary to determine how many elements each blade should be divided before the factor maintains a constant value [62]. Once convergence is achieved, the element force on each blade element is computed and through integration the total forces and moments acting on the blade are obtained.

Even though these equations may seem easy enough to apply, they can be used only for a rotor operating under steady, axisymmetric flow conditions of an incompressible, inviscid, isentropic fluid [62]. Furthermore, tangential forces do not have to influence the flow. Assuming air can be considered as a fluid that satisfies the aforementioned characteristics, the other conditions are still quite strict, thus the model has to be adjusted to cope with realistic operating conditions [7].

Momentum theory assumes that the thrust force is uniformly spread across a thin actuator disk while the blade element theory divides the blade into several elements each being subject to an infinitesimal normal force. This force is assumed to be the same for each radial location in all the other elements at the same radial coordinates. The BEM theory states that the thrust force given by the momentum theory in a certain annulus has to be the same as the thrust force acting on the same area [62]. This allows to calculate the induction factor through an iterative process. It is necessary to determine how many elements each blade should be divided before the induction factor maintains a constant value. Once convergence is achieved, the element force on each blade element is computed and through integration, the total forces and moments acting on the blade are obtained.

3.4.2. OLAF

OLAF is a free vortex method. This means that it considers the complex physics required to describe a phenomenon as complex as the wind but, at the same time, remains less computationally expensive than Computational Fluid Dynamics (CFD) methods [8]. To represent the blades, a lifting-line representation is used. This is characterized by a distribution of bound circulation that results in free vorticity being emitted in the wake because of its spatial and time variation. OLAF solves for the turbine wake in a time-accurate manner, which allows to capture the convection, stretching and diffusion of the vortices.

The OLAF model is based on a Lagrangian approach. This means that each particle's motion is described in terms of a function depending on the system's coordinates and velocities. In this case, the turbine wake is discretized into Lagrangian markers. To represent them, a hybrid lattice/filament method can be used and can be observed in Figure 3.7. Thus, the wake is represented as a collection of interconnected filaments, each of which is associated to a specific marker, while the flow field around the turbine is discretized through a regular lattice. Here the position of the Lagrangian markers is shown in terms of wake age, ζ, and azimuthal position, ψ, but in the code they are expressed in Cartesian coordinates.

For the near wake of the blade, which spans over a user-specified angle, a lattice method is applied: the filaments are connected to the lattice points in this area and are advected by the fluid flow, allowing for a description of the time evolution of the wake. After the near wake region, the wake is assumed to instantaneously transform into a tip and root vortex. Each Lagrangian marker is assumed to be connected to the adjacent markers by a straight-line vortex filament.

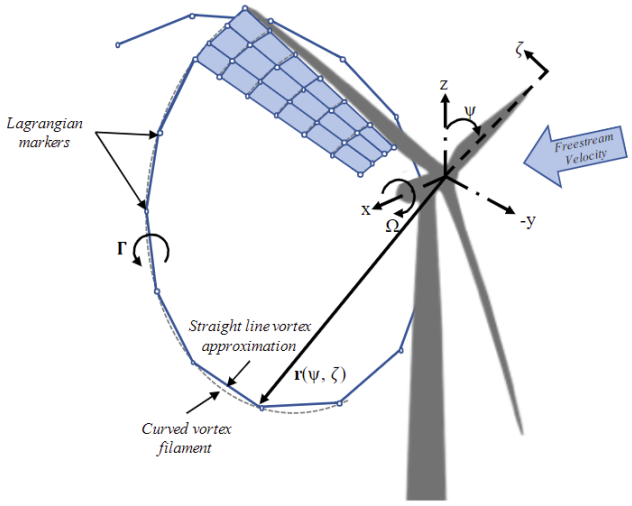


Figure 3.7.: Evolution of near-wake, blade tip vortex and Lagrangian markers [8]

Induced velocities at a certain position x are computed using Biot-Savart law. This is Equation 3.28 where Γ is the circulation strength of the vortex filament, dI is the unit versor along the filament, r is the vector from the filament to the current point x.

$$d\mathbf{v}(\mathbf{x}) = \frac{\Gamma}{4\pi}\frac{d\mathbf{l}\times\mathbf{r}}{r^3} \tag{3.28}$$

The OLAF method calculates the same parameters as the BEM one. Its advantage regards mainly the areas in which BEM assumptions are more violated, such as asymmetric rotors or in planes where the interaction between the turbine blades and the near wake of the turbine is relevant.

3.4.3. Comparison

To determine which method is more suitable for this study, both are implemented to run nine different simulations. The environmental conditions applied for each case are resumed in the following Table 3.2.

For both methods a time-step of 0.2 s is used in the 'TurbSim' file and a time-step equal to 0.005 s in the ' IEA-15-240-RWT-Monopile.fst' file. The only difference between the input files for the two models is in the 'IEA-15-240-RWT_AeroDyn15.dat' file. Here the wake induction model parameter is set to 1 to run BEM simulations and to 3 when OLAF method is implemented. The direct consequence of this change is in the type of airfoil aerodynamics model that is used. When the former method is applied, it is possible to use Beddoes-Leishman unsteady model whereas when OLAF is chosen as a solution method this parameter has to be set to 1, which corresponds to applying a steady model [63]. This is due to limitations that have been overcome in the newer version of OpenFAST, but it is still an issue for OpenFAST v3.0.0, which is the one used in this study.

Table 3.2.: Summary of the environmental conditions of the cases to which cross-correlation is applied.

File name	**URef** ($m\,s^{-1}$)	H_s (m)	**Wave Direction** (°C)	TI (%)	T_p (s)
Modified 1	24.61	5.95	88.21	9.90	8.76
Modified 11	10.17	7.45	-98.99	13.81	14.90
Modified 12	4.67	3.12	117.01	13.45	11.81
Modified 16	14.30	6.49	53.65	8.55	16.44
Modified 17	8.28	3.60	68.05	13.62	17.43
Modified 19	12.41	2.80	-47.15	5.12	15.88
Case 1	23.00	12.00	100.00	11.92	15.00
Case 2	4.00	1.00	120.00	25.80	4.00
Case 3	10.59	7.00	-130.00	15.35	11.50

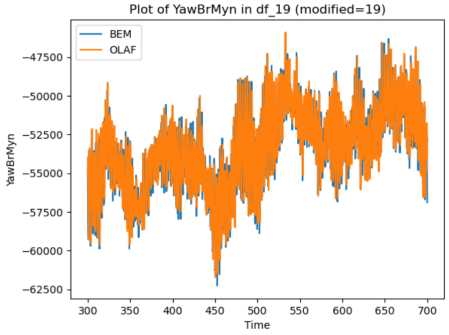
To perform these simulations in HPC, different nodes can be used. Each has different properties and multiple of the same type can be used at the same time to run a specific simulation. Each node type is then assigned to a different queue, each of which has different specifications. Overall, the queuing system gives priority to jobs requiring fewer cores and time. With these settings, to run 700 s of simulated functioning, it takes approximately one hour with BEM and almost three days to run the same input files with OLAF.

In the nine analyzed cases, the results of the comparison between BEM and OLAF are of three different types. First, the two resulting time series can be very similar, almost overlapping with each other. This is the case for variables such as rotating tower top roll moment, blade in-plane moment at the blade root, or blade edgewise moment at the blade root. Some examples of the resulting graphs for these cases can be observed in the following Figure 3.8. In the descriptions of the figures, the variables given as inputs in OpenFAST are reported in a vector $\boldsymbol{x}$ in the same order shown in Table 3.2.

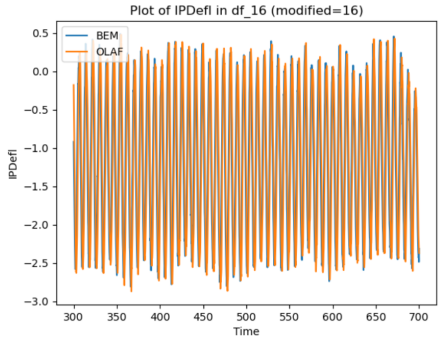
In other cases, the trends observed in the time series are the same but the values are different between BEM and OLAF simulations. This happens for the angle of attack, the blade out-of-plane moment at the blade root, the blade pitching moment at the blade root, the blade flapwise moment at the blade root, and the tower base moments. Also in this case, some examples are reported in the following Figure 3.9.

It can also happen that the variables calculated with BEM and OLAF simulations behave in either of the previously described ways depending on the environmental conditions applied. Nevertheless, for some specific conditions, it can happen that the two simulations' outputs are very different. This is what happens for simulations 'Modified 1' and 'Case 1', which are both characterized by high wind speeds, close to the cut off value. In particular, when these two cases are run with OLAF, OpenFAST is not able to compute the solution since interference between the blade and the tower is detected. This is possibly due to the fact that BEM simulations do not fully capture the complex interactions between the tower and the blades, especially at the beginning of the simulations, which is when the failure occurs. OLAF instead can explicitly model the wake dynamics and account for the effects of the tower on the flow. As a result, it is able to detect the interactions between the two components more accurately, especially when wind speed is close to the cut-off value and the flow around the turbine is highly unsteady.

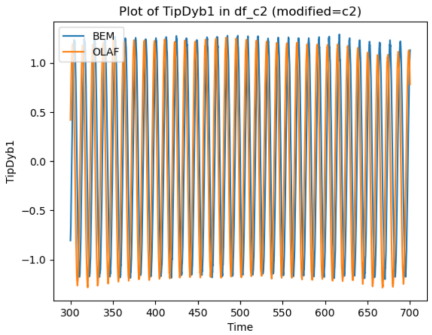
Also in other cases, the simulations reflect a difference in the loads. One example is 'Case 3', when the average wind speed at hub height given as input to the model is the rated one,



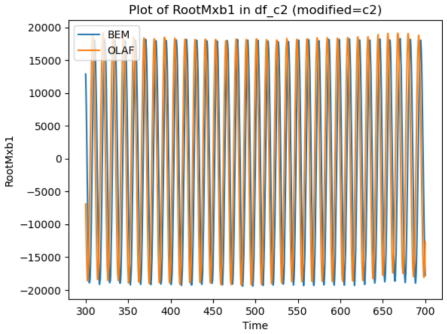
(a) Rotating tower top pitch moment from 'Modified 19' (x = [12.41, 2.80, -47.15, 5.12, 15.88])



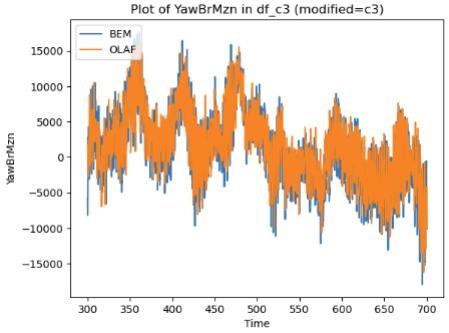
(b) In plane deflection from 'Modified 16' (x = [14.30, 6.49, -53.65, 8.55, 16.44])



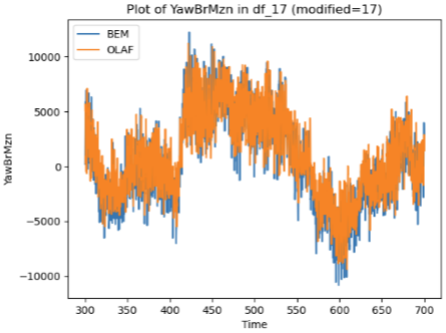
(c) Blade edgewise tip deflection from 'Case 2' (x = [4.00, 1.00, 120.00, 25.80, 4.00])



(d) Blade edgewise moment at the blade root from 'Case 2' (x = [4.00, 1.00, 120.00, 25.80, 4.00])



(e) Tower-top yaw moment from 'Case 3' (x = [10.59, 7.00, -130.00, 15.35, 11.50])

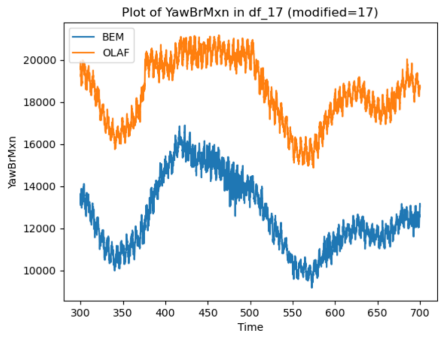


(f) Tower-top yaw moment from 'Modified 17' (x = [8.28, 3.60, 68.05, 13.62, 17.43])

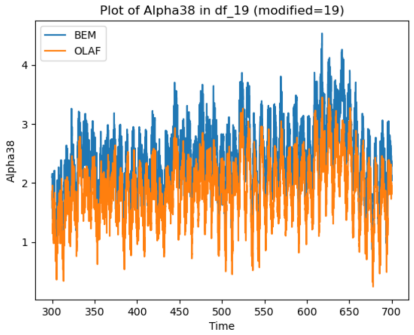
Figure 3.8.: Examples of overlapping BEM and OLAF results

10.59 m s^{-1}. This is the area where the controller switches between the two implemented PI ones, thus, depending on the grade to which the unsteady aerodynamic effects are detected, either one or the other is implemented. This contributes to accentuating the computational differences between the two models which results in different graphs. Some examples are reported in the following Figure 3.10.

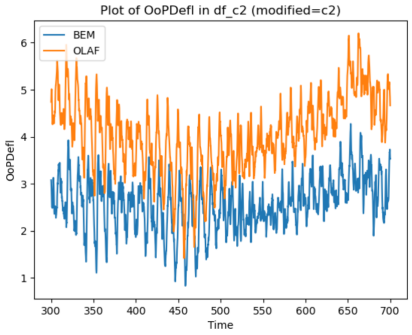
Even though OLAF has the advantage of being more precise, BEM is still considered a better



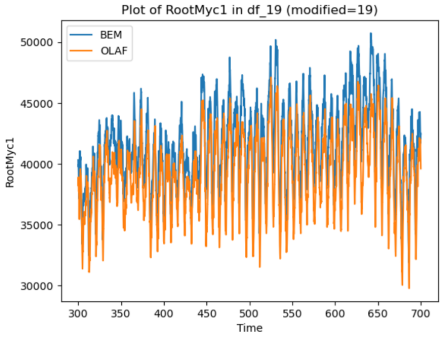
(a) Rotating tower-top bearing roll moment from 'Modified 17' (x = [8.28, 3.60, 68.05, 13.62, 17.43])



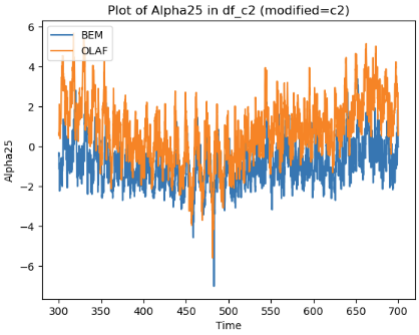
(b) Angle of attack from 'Modified 19' (x = [12.41, 2.80, -47.15, 5.12, 15.88])



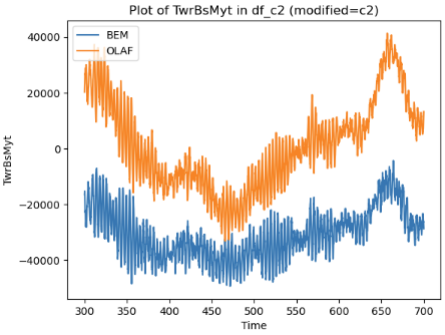
(c) Out-of-plane deflection from 'Case 2' (x = [4.00, 1.00, 120.00, 25.80, 4.00])



(d) Blade out-of-plane moment at the blade root from 'Modified 19' (x = [12.41, 2.80, -47.15, 5.12, 15.88])



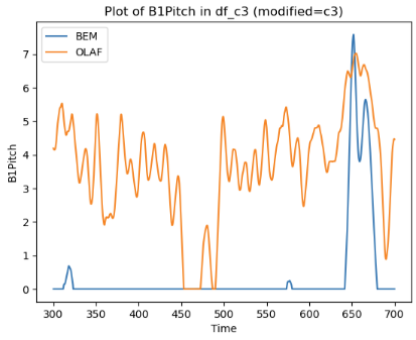
(e) Angle of attack section 25 from 'Case 2' (x = [4.00, 1.00, 120.00, 25.80, 4.00])



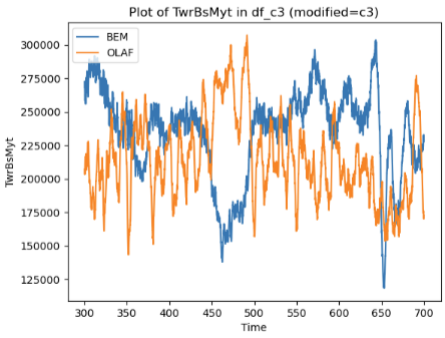
(f) Tower base pitching moment from 'Case 2' (x = [4.00, 1.00, 120.00, 25.80, 4.00])

Figure 3.9.: Examples of shifted BEM and OLAF results

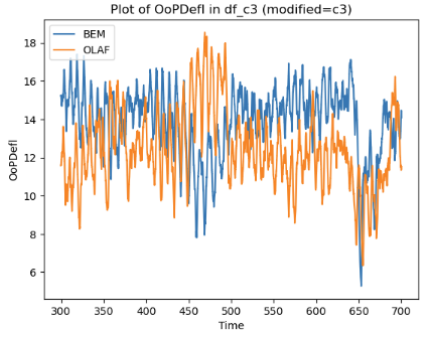
choice given its minor computational time. Taking into account the aim of this study, which is to provide a methodology to train machine learning algorithms for time series analysis, it is not particularly interesting how accurate the results are, as long as the trends are correct. As a result, for the rest of this study, BEM is used to model the turbine's aerodynamics.



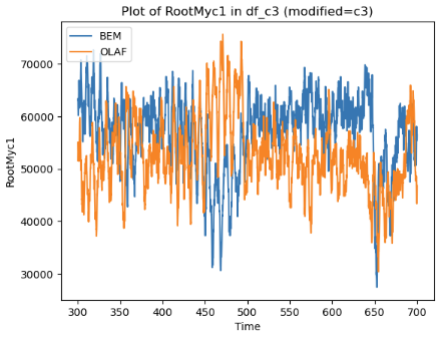
(a) Pitch angle



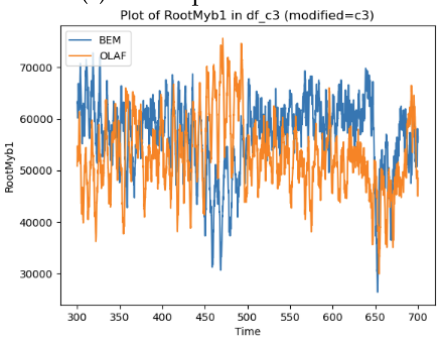
(b) Tower base pitching moment



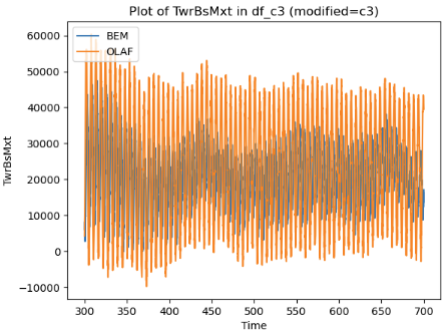
(c) Out-of-plane deflection



(d) Blade out-of-plane moment at the blade root



(e) Blade flapwise moment at the blade root



(f) Tower base roll moment

Figure 3.10.: Examples of different BEM and OLAF results from 'Case 3' simulations ($\boldsymbol{x}$ = [10.59, 7.00, -130.00, 15.35, 11.50])

Chapter 4

Feature selection

In this chapter, relevant features to train the machine learning algorithm are selected based on cross-correlation. The procedure followed to select the variables will be explained in this chapter. First, the logic behind the choice of the features, together with the identified variables, is described in Section 4.1. Next, the method applied to identify the relationship between variables, cross-correlation, is explained in Section 4.2. Later, in Section 4.3 the application of the previously described theoretic framework, i.e. cross-correlation, is shown and its application to the generated data is presented in Section 4.4. Finally, the identified 'cases' that will be assessed in this study are introduced in Section 4.5.

4.1. Feature selection

When choosing the variables that should be used to train the database, careful consideration has to be given to the availability of these from real measurement data. Since this study aims to develop a methodology that can be practically used to forecast the time series of the parameters highlighted in Table 4.2, it is preferred to consider only features available from measured data. The considered variables consist of SCADA measurements, geometric properties, Linear Variable Differential Transformers (LVDT), and accelerometer data at the nacelle. These are summarized in Table 4.1.

Note that the main focus for load calculation is on the blade root section and the tower base, as these are the sections on each component where loads are most significant [64]. Nevertheless, to consider the aerodynamics throughout the whole blade, different angles of attack and pitch angles are considered. These are, in particular, the ones referring to sections 9, 25, and 38. In this wind turbine model, each blade is divided into 50 different sections, with numbering increasing as the blade's tip is approached. The previously mentioned sections are chosen based on the study conducted by G. Wu et al. [65]. Following the methodology adopted in their study, two points are identified along the blade's span: one at 18% of the blade length, close to the blade root, and another one at 65%. The reason behind this choice is that different physics phenomena are predominant in these two different regions. Close to the blade root 3D effects including stall delay, influence the aerodynamics of the turbine the most while close to the tip the aerodynamics behaviors are dominated by the 2D chordwise flow and the 3D effects are almost negligible. In addition to these two points, the section in

Table 4.1.: Considered input variables.

Variable	Source	Name
Wind speed	SCADA	WindVelX
Out of plane deflection	LVDT	OoPDefl
In-plane deflection	LVDT	IPDefl
Pitch angle	SCADA	B1Pitch
Rotor tip speed ratio	SCADA	RotTSR
Thrust force acting on the whole rotor	Accelerometer	RotThrust
Pitch + twist angle section 9	Geometric properties & SCADA	Theta9
Pitch + twist angle section 25	Geometric properties & SCADA	Theta25
Pitch + twist angle section 38	Geometric properties & SCADA	Theta38
Tower-top / yaw bearing fore-aft (translational) acceleration (absolute)	Accelerometer	YawBrTAxp
Tower-top / yaw bearing side-to-side (translational) acceleration (absolute)	Accelerometer	YawBrTAyp
Tower-top / yaw bearing axial (translational) acceleration (absolute)	Accelerometer	YawBrTAzp
Tower-top / yaw bearing angular (rotational) roll acceleration (absolute)	Accelerometer	YawBrRAxp
Tower-top / yaw bearing angular (rotational) pitch acceleration (absolute)	Accelerometer	YawBrRAyp
Tower-top / yaw bearing angular (rotational) torsion acceleration	Accelerometer	YawBrRAzp

the middle of the turbine is also studied. Thus, sections 9, 25, and 38 are located respectively at 18%, 50%, and 65% of the blade length.

The variables considered are from nine different simulations, each representing wind and wave different conditions. This is done to represent a range of operating conditions and verify that the relationships between variables are not only due to casualty but that there is a real correlation. In particular, the input parameters for the considered cases are resumed in Table 3.2.

Table 4.2.: Output parameters

Variable	Name
Angle of attack section 9	Alpha9
Angle of attack section 25	Alpha25
Angle of attack section 38	Alpha38
Blade in-plane moment at the blade root	RootMxc1
Blade out-of-plane moment at the blade root	RootMyc1
Blade pitching moment at the blade root	RootMzc1
Blade edgewise moment at the blade root	RootMxb1
Blade flapwise moment at the blade root	RootMyb1
Tower base roll (or side-to-side) moment	TwrBsMxt
Tower base pitching (or fore-aft) moment	TwrBsMyt
Tower base yaw (or torsional) moment	TwrBsMzt
Rotating (with nacelle) tower-top / yaw bearing roll moment	YawBrMxn
Rotating (with nacelle) tower-top / yaw bearing pitch moment	YawBrMyn
Tower-top / yaw bearing yaw moment	YawBrMzn

4.2. Cross-correlation theory

Cross-correlation is a technique used to quantify the degree of similarity between two sets of numbers [66]. Even though the procedure can seem simple, the general concept is used in various advanced analysis techniques. These are all based on the idea that if one carries out a point-by-point multiplication of the two time series, the sum of the products will be a quantification of their relationship.

To find the relationship between the data sets, it is common practice to shift the curves. In the following Equation 4.1, n is the number of data points in each time series, x is the i^{th} data point of the first data series and y of the second data series. Finally, r_{xy} is the correlation. The number of data points by which the signal is shifted is called lag and is represented by l in the following Equation 4.1 [66].

$$r_{xy}(l) = \sum_{i=0}^{n-1} x_i y_{i+l} \tag{4.1}$$

Since the cross-correlation given in the previous equation is not dimensionless but depends on the units of x and y, it is difficult to compare cross-correlations from different data sets. To prevent a reduction in the sum of products with increasing lags and to make the relation

unitless, it is necessary to normalize Equation 4.1 [66]. This is achieved by dividing r_{xy} by the square root of the product of the auto-correlation of x at zero lag and the square root of the auto-correlation of y at zero lag. Auto-correlation expresses the cross-correlation of a set of numbers with itself, and it is represented in the following Equation 4.2 by $r_{xx}(0)$ and r_{yy} respectively for the first and second time series.

$$\rho_{xy}(\ell) = \frac{r_{xy}(\ell)}{\sqrt{r_{xx}(0)}\sqrt{r_{yy}(0)}} \tag{4.2}$$

From the previous equation, it is possible to deduce that the denominator is always greater than the numerator since it represents the perfect cross-correlation between the time series [66]. As a result, the maximum value that ρ_{xy} can have is 1.

If the mean values are subtracted from the previous Equation 4.2, the normalized cross-correlation expression Equation 4.3 will be negative only when the curves have an inverse relationship [66]. This is the most commonly used equation to express cross-correlation.

$$\rho_{xy}(\ell) = \frac{\sum_{i=0}^{n-1}(x_i - \bar{x}) * (y_{i-\ell} - \bar{y})}{\sqrt{\sum_{i=0}^{n-1}(x_i - \bar{x})^2}\sqrt{\sum_{i=0}^{n-1}(y_{i-\ell} - \bar{y})^2}} \tag{4.3}$$

To describe it, it can be said that it is the degree to which x and y vary together divided by the degree to which they vary separately. It is advisable not to lag the time series for values approaching n since unless the data is circular, the resulting data points would be too few as ℓ increases [66]. Good practice suggests to only using lags up to $\frac{n}{2}$.

4.3. Performing cross-correlation

A pre-requisite for performing cross-correlation is for the time series to not have an underlying trend and be stationary. In other words, it has to be verified if 'detrending' is necessary.

An Augmented Dickey Fuller (ADF) Test is performed in Python to achieve this goal. This is a root test, a statistical significance test for testing the stationarity of time series [67].

A stochastic process of form $y_i = \phi y_{i-1} + \epsilon_i$ where $|\phi| \leq 1$ and ϵ_i is white noise is considered. If $|\phi| = 1$, it is a unit root, thus a non-stationary time series. On the other hand, if $|\phi| < 1$, the process is stationary, whereas if $|\phi| > 1$ it is called explosive [68]. Only the former case will be considered in this analysis since explosive behavior in environmental conditions can only be observed in extreme events such as hurricanes, floods, or wildfires, which do not concern the situations where and when wind turbines operate. In the Dickey-Fuller test, the first difference is calculated as follows in Equation 4.4.

$$\Delta y_{i-1} = y_i - y_{i-1} = \phi y_{i-1} + \epsilon_i - y_{i-1} = (\phi - 1)y_{i-1} + \epsilon_i = \beta y_{i-1} + \epsilon_i \tag{4.4}$$

At this point, it is possible to perform the statistical test where the null hypothesis (H_0) is that $\beta = \phi - 1 = 0$ and H_1 is $\beta < 0$ [10]. To have a stationary time series the latter hypothesis has to be verified. Since this t-coefficient follows a τ distribution, the test consists in determining whether the τ statistic is less than τ_{crit} which is tabularized.

The ADF expands the Dickey-Fuller test by including higher-order regressive processes in the model. This results in the definition of other differences which can be generalized as shown in Equation 4.5.

$$\Delta y_{i-j} = y_i - y_{i-j} \tag{4.5}$$

Finally, three different versions of the ADF test can be defined, depending on what feature has to be identified. These are summarized in the following Table 4.3. In the analyzed case, type 2 is implemented since it is the most complete.

Table 4.3.: Three versions of Augmented Dickey-Fuller Test [10]

Type 0	No constant, no trend	$\Delta y_i = \beta_1 y_{i-1} + \sum_{j=1}^{p} \gamma_j \Delta y_{i-j} + \varepsilon_i$
Type 1	Constant, no trend	$\Delta y_i = \beta_0 + \beta_1 y_{i-1} + \sum_{j=1}^{p} \gamma_j \Delta y_{i-j} + \varepsilon_i$
Type 2	Constant and trend	$\Delta y_i = \beta_0 + \beta_1 y_{i-1} + \beta_2 i + \sum_{j=1}^{p} \gamma_j \Delta y_{i-j} + \varepsilon_i$

Finally, to determine the number of lags necessary, either AIC or Bayesian Information Criterion can be used. In this study, the former is implemented.

The test provides five values. First, the test statistic measures how strongly the time series violates the null hypothesis. Then the p-value is again a method to determine if the null hypothesis can be rejected. Next the number of lags, automatically determined based on AIC by the implementation of the option 'autolag', and the number of observations are displayed. Finally, the critical value at 1% significance level is shown. This is the value of the threshold used to determine if the null hypothesis should be rejected or not with a 1% probability of rejecting the H_0 if it is true.

Once the stationarity of the time series is verified, it is possible to perform the cross-correlation. To this aim, the function 'numpy.correlate' is applied. This performs the standardization of the data and returns the full cross-correlation sequence.

4.4. Results

The tests described in Section 4.2 and Section 4.3 are performed on the results of nine different simulations. The characteristics of the environmental conditions imposed in these cases

are reported in the previously shown Table 3.2. These are chosen to cover various operating conditions of the controller, which depend on the wind speed, combined with different environmental conditions.

First, the results of the ADF are analyzed. Test statistic values are all more negative than the threshold set for the 1% significance level. Thus the null hypothesis that the series is non-stationary can be rejected. Also, the p-value results confirmed the rejection of H_0 as they are much smaller than 0.05. This is commonly chosen as the threshold to reject the null hypothesis since it assures a confidence level in the results of 95%.

Next, the cross-correlation results are analyzed to determine which variables have a strong relationship. The results are displayed in graphs such as the following Figure 4.1. Similar charts are generated for all the combinations of variables' couples, one from input and one from output, for all nine simulations. As can be observed, the maximum and minimum values of cross correlation coefficient are displayed in red on the left while the corresponding lag values are displayed in black.

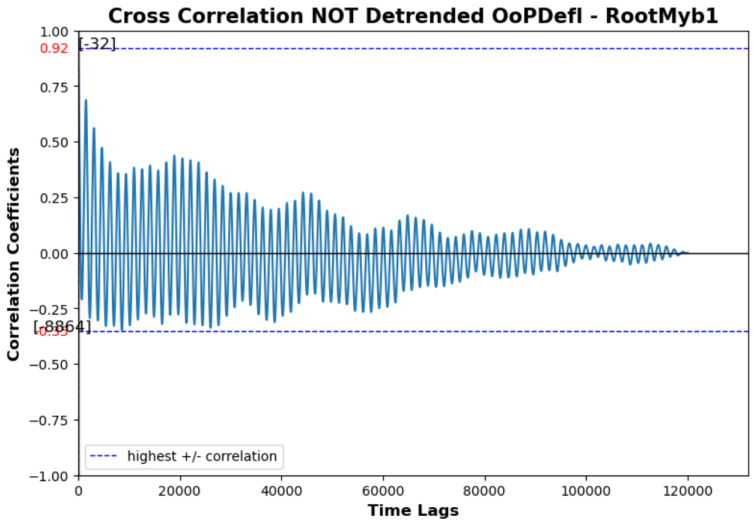


Figure 4.1.: Cross-correlation analysis result for blade out-of-plane deflection - Blade flapwise moment at the blade root

The results of the analysis are summarized in nine different heat maps, one for each considered simulation. These show the highest absolute value of the correlation coefficient between each couple made of one input, reported on the vertical axis, and one output, shown in the horizontal column. In the Appendix A the results of simulations files 'Case 1' and 'Case 2', which correspond respectively to wind speeds very close to cut-off and cut-in values, are shown.

A relationship is considered conventionally strong if the cross-correlation coefficient is at least 0.7 [69]. As it can be visualized in the heatmaps, the results are quite different between the different simulations. Nevertheless, some variables have consistently proven to be related. The resulting input variables related to the respective output are summarized in the following Table 4.4 and represented in Figure 4.2.

Table 4.4.: Relationship between input and output variables (please refer to Table 4.1 and Table 4.2 for variables' names)

Output	RootMxc1	RootMyc1	RootMzc1	RootMxb1	RootMyb1
Inputs	IPDefl	OoPDefl	IPDefl	IPDefl	OoPDefl

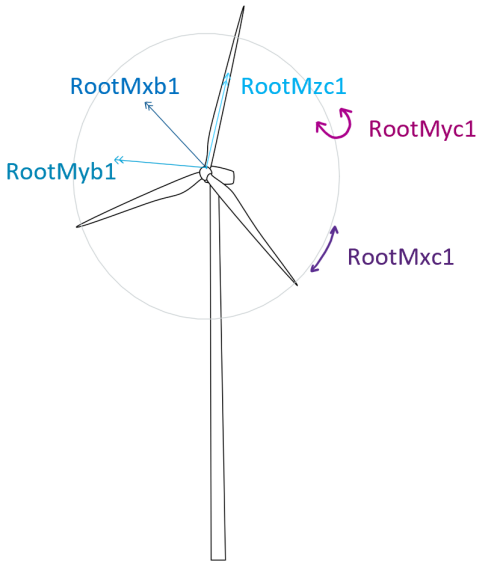


Figure 4.2.: Desired output moments

It has to be noted that also variables correlating to 'RootMzc1' are included, even though the cross-correlation coefficients are not higher than 0.65 in all considered cases. This is because the coefficients are very strong in more than 75% of the analyzed results. Furthermore, the relationship becomes weaker as the wind speed approaches the cut-off speed.

4.5. Training strategies

ARIMAX and LSTM models are trained with different combinations of exogenous inputs. Given that 500 OpenFAST simulations are run and their results can be used to train the algorithm, an analysis is performed to identify the most significant cases. Eventually, only 200 simulations are used. These are the ones that are based on all different 'URef' parameters. As mentioned in Chapter 3, it is the wind speed value indicated in 'TurbSim' to generate the stochastic wind speed time series.

The variables inserted in the data frame change depending on the targeted feature. A sum-

mary of the considered cases for both ARIMAX and LSTM can be found in the following Table 4.5. These are going to be analyzed more in detail in Section 4.5.1, Section 4.5.2 and Section 4.5.3.

Table 4.5.: Summary of training strategies features

Case name	Targeted variable	Exogenous variable (environmental conditions)	Exogenous variable (turbine's dynamics)
Wave	RootMxc1	URef, WindVel, WaveElev	IPDefl
	RootMyc1		OoPDefl
	RootMzc1		IPDefl
	RootMxb1		IPDefl
	RootMyb1		OoPDefl
NoWave	RootMxc1	URef, WindVel	IPDefl
	RootMyc1		OoPDefl
	RootMzc1		IPDefl
	RootMxb1		IPDefl
	RootMyb1		OoPDefl
PI	RootMxc1	WindVel	RotSpeed
	RootMyc1		
	RootMzc1		
	RootMxb1		
	RootMyb1		

4.5.1. Wave

The case 'Wave' considers as exogenous variables the wave elevation time series, 'WaveElev', the wind time series, 'WindVel', and the reference value for wind speed adopted by 'TurbSim' to generate the stochastic wind speed time series, 'URef'. The added additional parameter is either in-plane or out-of-plane deflection at the blade root depending on the target parameter and the results obtained in the previous Section 4.4. It is the case that considers most exogenous variables, thus the one where the environmental conditions are most represented.

4.5.2. NoWave

The 'NoWave' case considers as exogenous inputs the same as the 'Wave' one except for the 'WaveElev'. Thus, the wind time series and the reference wind speed represent the environmental conditions. The aim of this case is to assess the impact that hydrodynamic loads have on the blade features. As for the 'Wave' case, either the in-plane or out-of-plane deflection at the blade root, depending on the target variable, is also in the database.

4.5.3. PI

The last case is the 'PI' one. Here the results of the analysis described in Section 4.4 are deliberately excluded from consideration. This decision is made to evaluate the models' performance using only the inputs currently utilized in the PI controllers. Specifically, the existing controllers solely rely on the rotor speed ('RotSpeed') as input and, based on this information and the known turbine model, estimate the wind speed. As a result, this case's evaluation focuses on the models' effectiveness when operating under these specific input conditions.

Chapter 5

Implementation of ARIMAX and LSTM models

Having modeled the environmental conditions, identified the target parameters and selected the related exogenous features, it is possible to proceed to the training of the surrogate model algorithm. The whole procedure, which will be explained in this chapter, is shown in Figure 5.1.

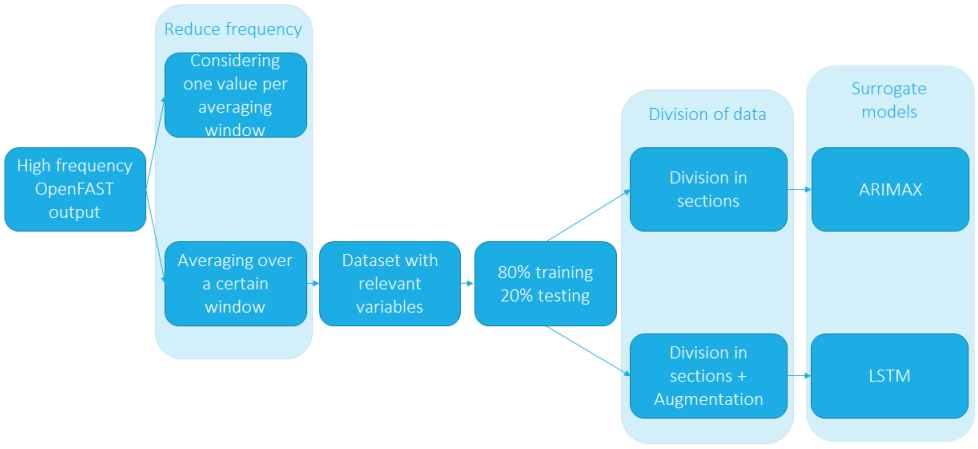


Figure 5.1.: Procedure followed to implement surrogate models

The first step, given the very high frequency used in the OpenFAST simulations, is to implement a discretization method to achieve a suitable frequency for each parameter. This procedure is described in Section 5.1. Successively, the algorithm used to implement ARIMAX and LSTM are described respectively in Section 5.2 and Section 5.3. Finally, the procedure followed to determine the hyper-parameters for the LSTM model is shown in Section 5.4.

5.1. Frequency definition

The data obtained from the OpenFAST simulations is expressed with a time-step of 0.005 s. This corresponds to a frequency of 200 Hz, which is too high for practical implementation. For this reason, a specific frequency is identified for the evaluated variables.

The sampling frequency for dynamic induction models has to be calibrated in order to capture the complex interactions incurring between the incoming wind, the rotating turbine blades and the wake generated by the blades. These interactions result in alterations in the angle of attack, lift and drag on the turbine blades as they move through the wind, leading to changes in the performance and behavior of the turbine.

As a result, the sampling frequency has to be different depending on which variable is being considered. If static forces, such as the thrust force, are studied, then lower frequencies can be considered. For periodic forces, such as the ones acting on the blades or the tower, the minimum necessary frequency is determined by the harmonics, which are an intrinsic property of the model.

To define the relationship between continuous-time signals, the ones observed in real life, and discrete-time signals, the ones modeled, the Nyquist theorem should be applied. According to it, to accurately reproduce a phenomenon the sampling frequency should be at least twice the one you want to capture. This is known as Shannon frequency.

As mentioned in Chapter 3, the natural frequencies for the blades are 0.555 Hz for flapwise and 0.642 Hz for edgewise deflection. The result of implementing Nyquist theorem is that the minimum frequency for the blade loads should be around 1.3 Hz. This corresponds approximately to a time-step of 0.77 s.

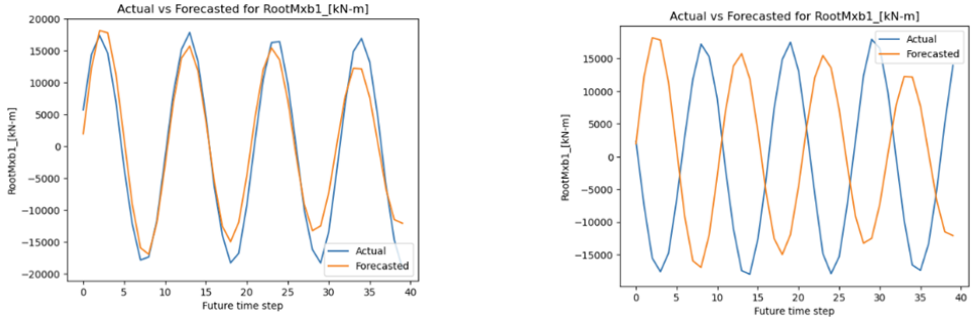
Given that the OpenFAST output is 0.005 s, the previously identified time-step corresponds to circa 154 values. To be conservative, 150 is chosen as averaging window. At this point, there are two alternatives. The first option is to consider only one value every 150 to be in the training algorithm. This means completely ignoring the fluctuation expressed by the other 149 points. The other option is to average the data points every 150 and insert in the database this value.

Simulations were launched for both options but the second was chosen since it resulted in a better forecast. This can be due to various aspects. First, averaging the data can reduce the noise, thus making it easier for the algorithm to capture the overall trend, leading to a more robust and stable representation of the data. Another advantage of averaging the values instead of sub-sampling is that a more representative sample is fed into the model. This is because instead of selecting one "random" point, which can easily be an outlier and thus not provide a good representation of the interval, a synthesis of the information contained in the selected interval is considered. Finally, the model generalization is enhanced because the single small fluctuations are not taken into account. Thanks to this operation, the computational complexity is significantly reduced, since it is proportional to the cube of the number of data points.

5.2. ARIMAX implementation

ARIMAX is a model that aims to capture temporal structure based on linear regression. As a result, it works best for one-step forecasting but it can also be used for multiple-step forecasting. Nevertheless, to build the algorithm that will result in the best prediction, the model has to be re-fitted depending on the target section [70]. For this reason, two main options are identified for ARIMAX implementation.

The first one aims to build one model that could provide a meaningful forecast for all the considered 30 s sections. This results in a single 30-second forecast, which stays the same independently from the considered history or analyzed simulation. It eventually results in plots comparing testing and forecast data that can range between the two extreme options depicted in the following Figure 5.2. In this chapter, the variables given as inputs in OpenFAST are reported in a vector x in the same order shown in Table 3.2 in the descriptions of the figures showing forecast examples.



(a) x = [15.5, 1.4, -145.4, 13.4, 7.3] at t = 690 s (b) x = [11.4, 7.5, 95.6, 14.9, 12.6] at t = 360 s

Figure 5.2.: Comparison of predicted and test data when the same ARIMAX model is used for multiple sections

The other option is to retrain the model for each target section, thus obtaining different forecasts for each one. This means that a different forecast is going to be provided for each section and, before training the following section, the test data of the previous one is going to be given to the model as input.

It is important to note that neither of the proposed solutions has real-life application given the long required computational times. Nevertheless, this last option provides a good example of how a linear model can predict these types of time series. For this reason, it is the chosen option. The next subsections provide a more in-depth explanation of the algorithm's logic. In particular, Section 5.2.1 describes the data-preprocessing, Section 5.2.2 how the forecast is computed and Section 5.2.3 the method used to evaluate the performance of the model.

5.2.1. Data pre-processing

As for the application of all SM, the first step is data pre-processing. This includes cleaning, scaling, and splitting the database into training and testing data sets.

Data cleaning is generally essential when the information is retrieved through real-life measurements. Since in this study the data is generated through the application of another algorithm, it is less fundamental. Nevertheless, the generated database is analyzed to assess if there are any missing values. This is not the case but it is to note that within the 500 simulations, only 200 have different values of 'URef'. These are considered enough to effectively represent the whole sampled area, thus the results of these OpenFAST simulations are first averaged every 150 values to obtain the desired frequency, as described in Section 5.1, and then saved in a separate database. This is later used as input in the ARIMAX code.

The second step involves scaling, which is performed by applying standardization to each column of the newly created database. In practical terms, this entails applying the following equation (referred to as Equation 5.1) to each value x in column X. In the structure of the database, this operation is applied to each time step value within the column that encompasses all the values from the considered simulations of the respective variable. The terms $\mu(X)$ and $\sigma(X)$ represent the mean and standard deviation of column X, respectively.

$$x_{rescaled} = \frac{x - \mu(X)}{\sigma(X)} \tag{5.1}$$

In general, scaling is useful for forecasting using SM because it helps ensure that the input variables have similar ranges or distributions. This has several benefits for the performance and accuracy of the surrogate models in the forecasting process. It improves numerical stability by avoiding large variations, enhances convergence speed because it avoids the additional task of taking into consideration different scales, and ensures that the model in the forecasting phase can perform well also on unseen data that may have a different scale from the one used in the training phase.

Finally, the data is divided into training and testing data. In this study, 80% of the data is used for training and 20% for testing. Furthermore, for both train and test data, the 800 time steps available for each simulation are divided into 20 different sections, each composed of 40 time steps, corresponding to 30 seconds. A four-dimension matrix is created to avoid mixing data from different simulations in the same section and providing as input data only points belonging to the same one that is being analyzed. As briefly described in Chapter 2, the first dimension differentiates between one simulation and the next. The second and third dimensions divide the 800 data points into sections, each with 40 time steps. Finally, the last dimension refers to the variables considered.

5.2.2. Forecasting

Two models are implemented for forecasting: AutoRegressive Integrated Moving Average (ARIMA) and ARIMAX. ARIMA is used initially, without considering external inputs, and relies solely on the values of the target variable. Its purpose is to predict the future values of each exogenous variable for the next 40 time steps. To accomplish this, the 'pm.auto_arima' function from the 'pmdarima' library is employed. This function automates the ARIMA algorithm and determines the appropriate orders based on specified criteria.

The order of auto-regressive components, 'p', represents the number of lagged values of the target variable included in the model [47]. Each lagged value contributes to the model

separately, with its associated coefficient. The significance of the auto-regressive order determines the influence of past values of the target variable on its present value. As 'p' increases, the model becomes more flexible but, at the same time, the complexity and the risk of over-fitting data increases. A high 'p' indicates that the target variable's past value significantly impacts its current value.

Next, 'q' is the order of the moving average components. It represents the number of lagged residuals included in the model. Each of them contributes separately to the model; thus each is assigned a different coefficient that has to be determined. This term accounts for the time series's short-term dependencies and random shocks. The greater 'q' is, the more considering the past errors helps improve the model's accuracy. However, if the order of moving average components becomes too high, there is a risk of over-fitting and increasing too much the complexity of the model, making the computational cost too high.

The inputs for the function consist of the values of the variable to be forecasted from the previous 40 time steps, along with algorithm implementation specifications. In this study, the maximum order for 'p' and 'q' is set to 20, 'd' is set equal to zero given the results of the previously implemented ADF test and no seasonal variations are considered due to the nature of the time series being analyzed.

ARIMAX is used to forecast the target variable. It takes the exogenous variables, calculated using ARIMA, as inputs and generates another forecast for the target variable for the next 40 time steps. Similarly, the maximum order values for auto-regression and moving average are set to 20, while the differentiation parameter is null.

Both ARIMA and ARIMAX models are applied within for loops that iterate through all the sections. When forecasting a specific section, the model is informed using the test data set values from the previous section. Finally, the data is re-transformed before evaluating the forecasts to restore the original scale.

5.2.3. Model evaluation

The evaluation of the quality of the forecasts is twofold. First, the forecast for each section and the corresponding test data for the target variable are plotted to visually examine the effectiveness of the implemented model. This visualization provides a clear understanding of how well the SM captures the actual data.

A quantitative analysis of the performance on the complete validation dataset is done by calculating the RMSE at each time step. The RMSE is determined by averaging the values obtained for the same time step in each section.

Root Mean Squared Error (RMSE)

RMSE is a way to assess the forecast accuracy using SM. It derives from the mean squared error but extracts the square root of it to make the value more interpretable in the original units of the data.

Exponentiation provides various advantages including guaranteeing that all the values are positive and attributing greater importance to larger errors [71]. Later, by calculating the average of these squared errors, the RMSE provides an evaluation of the difference between the predictions and the true values.

It is calculated as shown in Equation 5.2. Here N is the number of considered data points, y_i is the actual value of the considered data point, and $\hat{y}_i$ is the corresponding predicted value.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{5.2}$$

RMSE provides a measure of the average prediction error, indirectly assigning more weight to larger errors. This metric ignores the direction of the errors and only highlights the magnitude of the difference between the predicted and real points. As a result, overestimation and underestimation are reduced to the same concept. SM that result in a lower RMSE are better.

5.3. LSTM implementation

Chapter 2 presented the theoretical foundations of LSTM models for time series forecasting. In this chapter, the practical implementation of LSTM is explained. The main focus will be on the code structure required to build and deploy the model, including data pre-processing, model setup, training, and evaluation. Finally, some definitions needed to understand how the SM is built are given.

As for ARIMAX, also in this case first data preprocessing is explained in Section 5.3.1, followed by the logic of the algorithm in Section 5.3.2. Next, the hyper-parameters are defined in Section 5.3.3, and the adopted optimizer is presented in Section 5.3.4. Finally, the methods used to assess the accuracy of the model are shown in Section 5.3.5

5.3.1. Data pre-processing

As mentioned in the previous Section 5.2.1, the data is retrieved through OpenFAST simulations and gathered in a new database with values averaged every 150, including only the 200 simulations presenting different 'URef'. Also for LSTM, the data is standardized as explained earlier and divided into training and testing sets applying the 80-20% common practice.

When applying LSTM, an additional step is included. This is called augmentation. The goal of performing it is to increase the size and diversity of the training data set. It helps mitigate over-fitting and improve the model's ability to generalize unseen data. This is often used in conjunction with the sliding window process to create multiple samples from a single time series.

In particular, the sliding window process involves creating overlapping sub-sequences or windows from the original time series data [9]. Each window represents a training sample with a fixed number of consecutive time steps as input, in this case 20, and the subsequent time step as the output or target variable. The sliding window process helps capture temporal dependencies within the time series by maintaining the sequential order of the data.

Furthermore, using only the original time series data might limit the diversity of the training samples. This process can be visualized as shown in Figure 5.3.

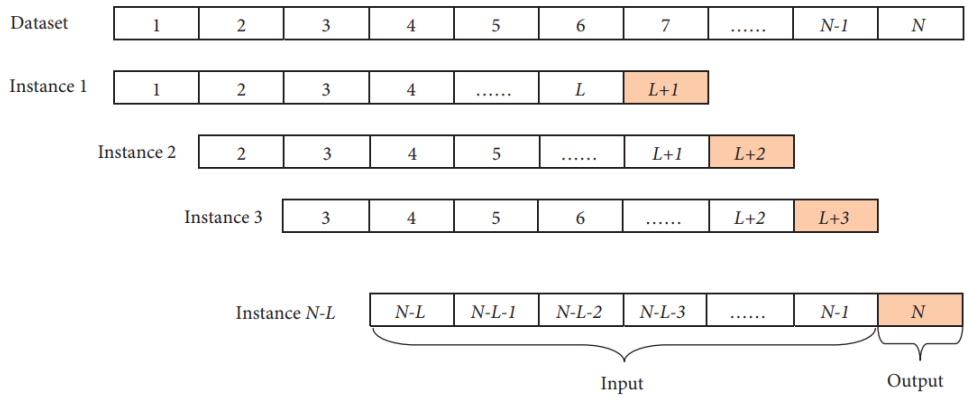


Figure 5.3.: Sliding windows process [9]

It is important to note that in this study, the data set shown in the picture represents only data from one simulation. This is inserted in a 4D matrix where the first dimension refers to the considered simulation, the second one to the number of the section analyzed, the third one represents the time steps considered, 40 in this case, and the final one represents the variables taken into account.

As a result of the implementation of the sliding windows process, the number of data points considered is increased, thus augmentation is performed. This allows the LSTM model to learn from a more diverse set of patterns, incorporating variations and temporal dependencies within the time series data. This enhances the model's ability to capture complex patterns and accurately predict unseen data.

5.3.2. Network architecture

The forecasting model is based on the encoder-decoder architecture explained in Section 2.4.3. It is built using the 'keras' library in Python.

The whole structure of the SM can be visualized in the following Figure 5.4. In particular, the encoder is composed of the first LSTM layer, while the decoder is made of the second LSTM layer and two 'Dense' layers. The whole structure ultimately produces the forecast vector which is a one-dimensional vector made of 40 values.

The model is created using an LSTM layer with 30 units and ReLU activation function to process the input sequence and learns to extract the relevant features. The learned features are then repeated in a 'repeated context vector', generated through the 'RepeatVector' layer. It can be interpreted as a summary of the input sequence's information in a condensed form. It repeats this information multiple times to create a sequence the same length as the desired output sequence. The goal of using this layer is to provide a global context or summary of the input sequence to the second LSTM layer. This context vector helps the model make predictions for each step of the output sequence by incorporating the learned information from the input sequence. In this way, it is sure that each step of the output sequence has access to the same global context.

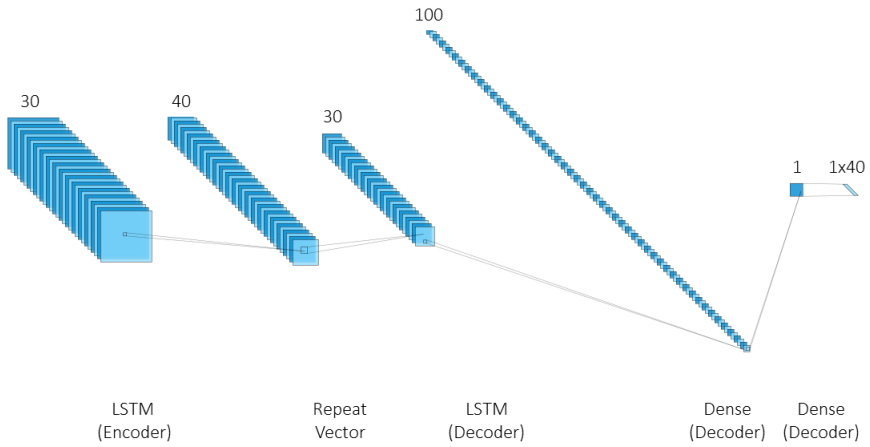


Figure 5.4.: Architecture of the applied LSTM model

The information summarized via the repeated context vector is then passed to the second LSTM layer, composed of 30 units and implements ReLU activation function. This layer processes the input sequence and learns to extract relevant features.

Further, two 'TimeDistributed' layers are implemented to enable the model to make predictions for each step of the output sequence independently. To this aim, this step is necessary because the output of a LSTM is a sequence of hidden states that have to be used to make individual predictions for each step of the output sequence rather than producing a sequence of predictions. The first 'TimeDistributed' layer uses 100 units and ReLU activation. It introduces non-linearity and additional complexity to the model, enabling it to learn more complex patterns and relationships within each time step of the output sequence. The second 'TimeDistributed' layer applies a dense layer to reduce the dimensionality of the hidden representations obtained from the previous layer to a single output value for each time step. It transforms the higher-dimensional representations into a sequence of scalar predictions, one for each time step. Therefore, each time step is treated as a separate instance and makes predictions for each future time step separately.

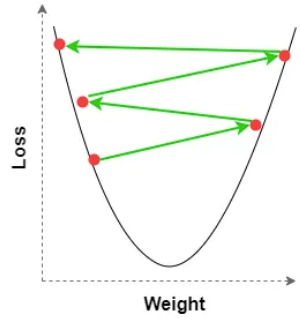
Additionally, the model requires the updated 'history' of the model implementation as inputs. This needs to be updated before each section is forecast with the actual values of the target variable of the previous section. In the beginning, it contains the first 40 time steps of the considered simulation; as the forecast proceeds it is updated, and thus it becomes longer at each iteration.

5.3.3. Hyper-parameters definition

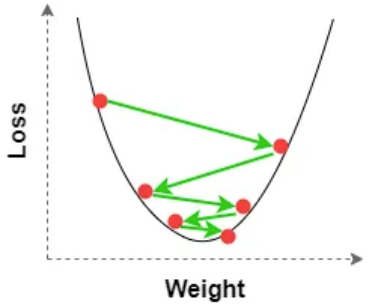
One of the challenges of implementing LSTM is determining the hyper-parameters. These are different from the parameters. While the model parameters are learned by the SM from the provided data set, the hyperparameters are variables that must be set before training

the model [72]. The optimal values for these 'settings' of the deep learning algorithm also depend on the size and nature of the dataset and the type of problem that has to be solved.

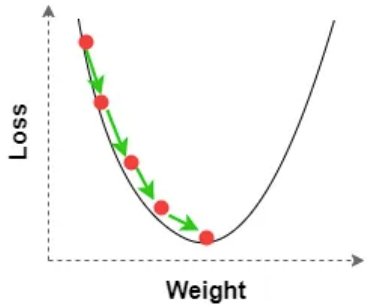
The learning rate α determines the width of the step that is taken by the optimizer to descend along the error curve [72]. The direction of the step is determined by the gradient, as described in Algorithm 2.2. A larger value of the learning rate entails that the SM can be trained faster but, on the other hand, it might lead to an oscillation around the minimum, failing to identify the precise value, thus never completing the model's training. At the same time, if α is too small, the model will require too much time to train because the convergence is reached very slowly. A visualization of the previously described behavior can be seen in Figure 5.5. Determining the appropriate learning rate often involves a process of experimentation and fine-tuning. Common techniques include manually testing a range of learning rates, monitoring the model's performance, and observing the convergence behavior. Additionally, adaptive learning rate algorithms such as 'RMSProp' optimizer, 'Adagrad' optimizer, or Adam optimizer can be implemented [73].



(a) Too high learning rate that does not lead to convergence [74]



(b) High learning rate that leads to convergence [74]



(c) Low learning rate that leads to convergence [74]

Figure 5.5.: Comparison of different behaviors of the training process of a NN given different learning rates

Another important hyper-parameter is the number of epochs, which refers to the number of times the entire training data set is passed forward and backward through the neural network. It represents the number of iterations or cycles the training process will undergo to update the model's weights. Increasing the number of epochs allows the model to potentially learn more complex patterns in the data but can also increase the risk of over-fitting

if set too high. Too few epochs may result in under-fitting, which means that, eventually, the model fails to capture all the complex patterns in the provided data. Determining the correct number of epochs often requires a trade-off between computational resources and model accuracy. It is common practice to monitor the model's performance on the testing data set and stop training the model once the error stops decreasing or starts to increase.

Batch size defines the number of samples or instances from the training dataset propagated through the network before the weights are updated.

The hyper-parameters are not independent of each other. If the learning rate is low, more epochs will be necessary since more steps will be necessary to reach the minimum of the cost function [72]. On top of that, also the batch size, another hyper-parameter, influences the number of epochs. It represents the number of training instances in the batch, thus the number of inputs processed in parallel. The smaller it is, the higher the number of epochs needs to be.

5.3.4. Adam optimizer

When implementing LSTM, the optimization algorithm plays a crucial role in the training of SM. One widely used option is the Adam optimizer. This is the abbreviation of Adaptive Moment Estimation and it is a very powerful algorithm that combines the benefits of the two most popular optimization techniques: Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp)

Adam optimizer can be described as a modified Stochastic Gradient Descent (SGD) that aims to update the network's weights based on the input data [75]. It adapts the parameter learning rates by combining AdaGrad and RMSProp. It does this by keeping separate learning rates for different parameters and adjusting them dynamically during training. The adaptive learning rate is calculated then using a running average of both the squared gradients (second moment, coming from AdaGrad) and the gradients (from RMSProp). The moments come from implementing the momentum technique, which is introduced to accelerate convergence and overcome oscillations by adding a fraction of the previous update to the current update.

Finally, the Adam optimizer also implements L2 regularization, also known as weight decay. This is a common technique to avoid over-fitting.

To sum it all up, this optimizer is used to estimate the first and second moments of the gradients, which are eventually applied to determine the parameters of the model, which are the weights and biases [76]. At the same time, it applied regularization to avoid over-fitting and adaptive learning rate to accelerate convergence. This is summarized in the following Algorithm 5.1. Here m and v are, respectively, the first and second moments of the gradients; β_1 and β_2 are the corresponding exponential decay rates; α is the learning rate; ∇L is the gradient of the loss function; ϵ is a constant added for the numerical stability and λ is the regularization constant.

5.3.5. Model evaluation

Finally, also in this case, it is necessary to bring the predictions back to the original scale and evaluate the model. As for ARIMAX, also for LSTM, the forecast of each section is displayed

Algorithm 5.1: Adam Optimizer

Input: $m(t)$, $v(t)$, $\theta(t)$, β_1, β_2, ∇L, α, λ
Output: $m(t+1)$, $v(t+1)$, $\theta(t+1)$

1 $m(t+1) = \beta_1 * m(t) + (1-\beta_1) * \nabla L(\theta(t))$

2 $v(t+1) = \beta_2 * v(t) + (1-\beta_2) * (\nabla L(\theta(t)))^2$

3 $\theta(t+1) = \theta(t) - \alpha * (\nabla L(\theta(t)) + 2\lambda\theta(t))$

compared to the corresponding testing data. However, the forecasts are all produced using the same model in this case.

For LSTM, there is also an added parameter to evaluate the performance of the model, in particular, to verify if over- or under-fitting occurred. This is done by plotting the relationship between the number of epochs adopted and the 'loss function' for the training and cross-validation data sets. The loss function measures the discrepancy between predicted and actual values during training. During the training process, the model aims to minimize this loss function by adjusting its weights and biases. The 'loss' variable stores the loss function's value at each training epoch. By plotting this value against the number of epochs, it is possible to visualize how the loss decreases over time, indicating the improvement of the model's accuracy. Not all of the training set is used in this case, so the same procedure can be repeated on data unknown to the model, called the validation set. This assessment of the model is known as cross-validation, and in particular, the one described here is named 'hold out' validation [77]. For the model to be acceptable, the curves describing the relationship between epochs and loss should be similar and close to each other. The plots resulting from this analysis and all the others previously referred to in this chapter can be found in the next Chapter 6.

5.4. Hyper-parameters tuning

A trial and error procedure is followed to tune the hyper-parameters of the applied LSTM model. This is performed using the variable 'RootMxb1' as an example. An overview of the performed analysis is shown in the following Figure 5.6 and explained in this section.

The first hyper-parameter that is set in this study is the batch size. Typical values for this variable are multiples of 16 until 1024 [72]. In this case, the smallest alternative is chosen, 16. A small batch size is memory efficient, which is convenient if large datasets are used, as in this case. It entails more frequent parameter updates, eventually leading to faster convergence. Finally, training with a small batch size allows the SM to 'see' a larger variety of samples. This increased diversity can help the model to learn more robust and generalized representations. On the other hand, the drawback of using 16 as batch size is that it entails longer training times due to the higher number of iterations required to process the entire data set.

Next, the Adam optimizer's initial learning rate is left as the default one in 'keras', equal to 0.001. Finally, the number of epochs is set to 20. In this section, 'RootMxb1' in the 'Wave' case is taken as an example to describe the behavior of the SM for the considered combinations of hyper-parameters. For the previously expressed settings, the loss function, which represents

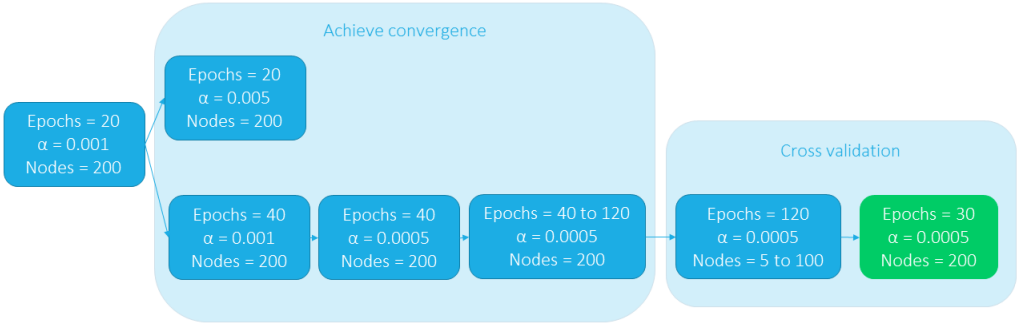


Figure 5.6.: Procedure followed to determine the LSTM model's hyper-parameters

the target function the algorithm tries to minimize, behaves as represented in Figure 5.7 as the number of epochs progresses.

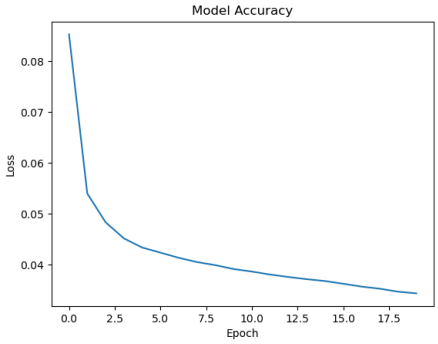
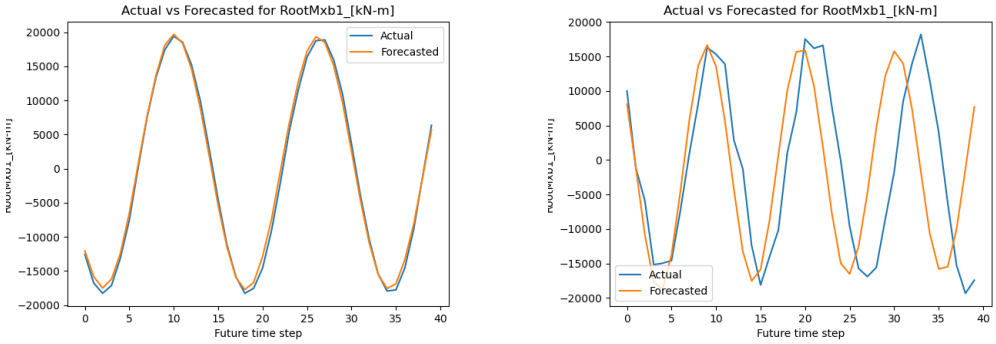


Figure 5.7.: Loss function of 'RootMxb1' in 'Wave' case when trained with SM using 20 epochs, learning rate equal to 0.001 and 200 nodes per layer

As can be observed, the model doesn't converge, as the line doesn't become flat and doesn't set on a constant loss value. Despite this, the forecast is computed. Two examples of forecasts, one very precise and one less rigorous, are shown in the following Figure 5.8.

The good performance of the SM is also confirmed by the computed RMSE. This is shown in Figure 5.9. It coherently increases as the time step progresses and assumes a maximum value of 1590 kN m^{-1}. If compared to the actual values, which range between $-21\,774$ kN m^{-1} and 26 853 kN m^{-1}, the percentage error is approximately 3.3%.

The computational time required using 1 node and 1 job on HPC equals a couple of hours to achieve this result. The same behavior is also reflected in the other considered cases,



(a) x = [11.4, 7.5, 95.6, 14.9, 12.6] at t = 360 s (b) x = [7.7, 1.4, -214.2, 13.6, 15.4] at t = 720 s

Figure 5.8.: Comparison of predicted and test 'RootMxb1' data in 'Wave' case when trained with SM using 20 epochs, learning rate equal to 0.001 and 200 nodes per layer

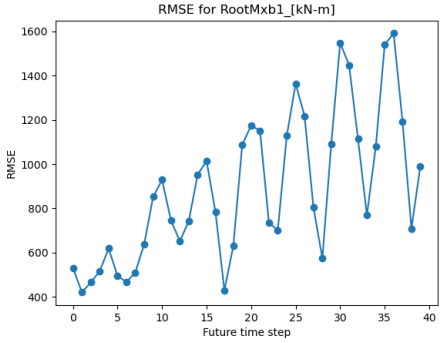


Figure 5.9.: RMSE of 'RootMxb1' in 'Wave' case when trained with SM using 20 epochs, learning rate equal to 0.001 and 200 nodes per layer

'NoWave' and 'IP', usually with a RMSE slightly higher. Given this consideration, from now on, only the 'Wave' case will be discussed in this section.

Given the desire to achieve convergence in the loss function, other combinations of learning rates and the number of epochs are also considered. First, the number of epochs is kept constant, and the learning rate is increased to 0.005. This leads to forecasts equal to a series of 'NaN'. The reason for this result is probably due to the behavior explained in Section 5.3.3, when it is shown how a learning rate too high may lead to the perpetual oscillation of the model that fails to minimize the cost function.

As a result, the learning rate is returned to its default value, 0.001, and the number of epochs is increased to 40. The resulting loss function of this case is reported in Figure 5.10, and it can be seen how the loss value decreases compared to the one reported in Figure 5.7, but still, the function doesn't converge.

Similarly to what is observed before, the forecast is accurate in some cases and less in others. Overall, the performance of the SM implementing these settings can be summarized with the following Figure 5.11.

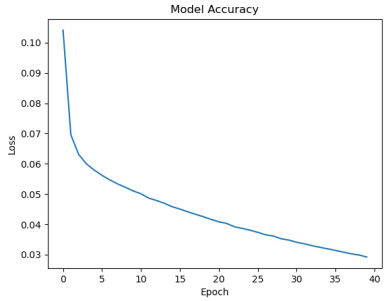


Figure 5.10.: Loss function of 'RootMxb1' in 'Wave' case when trained with SM using 40 epochs, learning rate equal to 0.001 and 200 nodes per layer

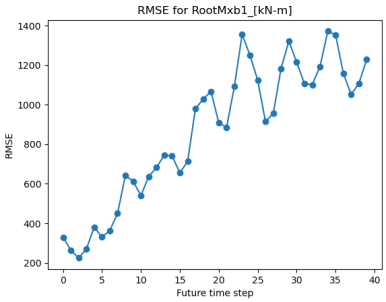


Figure 5.11.: RMSE of 'RootMxb1' in 'Wave' case when trained with SM using 40 epochs, learning rate equal to 0.001 and 200 nodes per layer

To compare the performance of the models adopting different learning rates but the same number of epochs, the learning rate is decreased to 0.0005, and the number of epochs is kept constant at 40. In this case, convergence is not achieved, and the loss function looks as shown in Figure 5.12. Compared to the previous Figure 5.10, it can be noticed that the slope is approximately the same. However, the computed RMSE is significantly lower in Figure 5.11 compared to Figure 5.13.

Also in this case, the forecasts are quite precise. As shown in the following Figure 5.14, some sections are predicted better than others, but overall the RMSE is still within the 4.2%. This is reported in Figure 5.13.

Given the results shown so far, the analysis is repeated, adopting 0.0005 as the learning rate while increasing 10 values at the time the number of epochs from 40 to 120. In the results reported in Figure 5.15, obtained when adopting the highest number of epochs for the 'Wave' case, it is possible to notice that the loss function converges to a flat line. This means that 120 is an appropriate number of epochs to adopt when combined with a learning rate equal to 0.0005 for this data set.

Moreover, the RMSE is calculated. As shown in Figure 5.16, the maximum value is equal to $1438\,\mathrm{kN\,m^{-1}}$. This aligns with the values obtained for the previously considered combinations of hyper-parameters.

To assess the model's efficiency, cross-validation is performed on the SM aiming to forecast

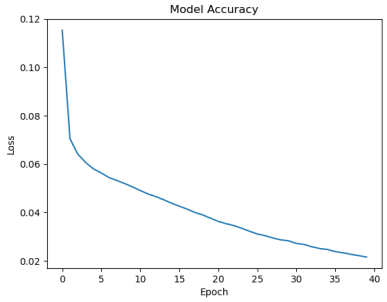


Figure 5.12.: Loss function of 'RootMxb1' in 'Wave' case when trained with SM using 40 epochs, learning rate equal to 0.0005 and 200 nodes per layer

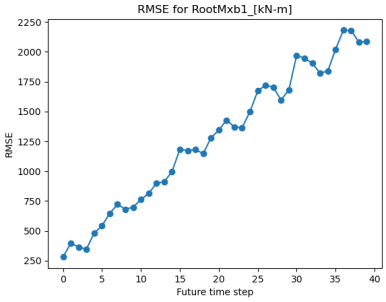
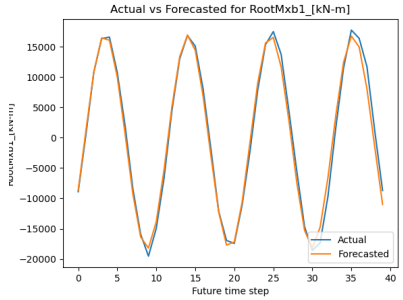
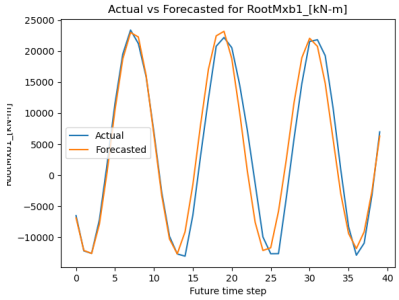


Figure 5.13.: RMSE of 'RootMxb1' in 'Wave' case when trained with SM using 40 epochs, learning rate equal to 0.0005 and 200 nodes per layer



(a) x = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 750 s



(b) x = [22.5, 5.3, 139.8, 12.0, 7.1] at t = 600 s

Figure 5.14.: Comparison of predicted and test 'RootMxb1' data in 'Wave' case when trained with SM using 40 epochs, learning rate equal to 0.0005 and 200 nodes per layer

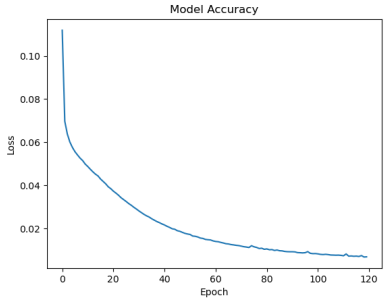


Figure 5.15.: Loss function of 'RootMxb1' in 'Wave' case when trained with SM using 120 epochs, learning rate equal to 0.0005 and 200 nodes per layer

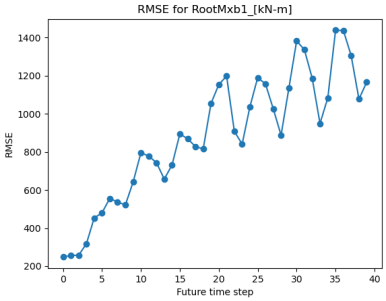


Figure 5.16.: RMSE of 'RootMxb1' in 'Wave' case when trained with SM using 120 epochs, learning rate equal to 0.0005 and 200 nodes per layer

the same variable, 'RootMxb1', in the same case, the 'Wave' one, using the same hyper-parameters, 120 epochs and learning rate equal to 0.0005. The resulting loss curves for the

test and validation data sets are shown in Figure 5.17.

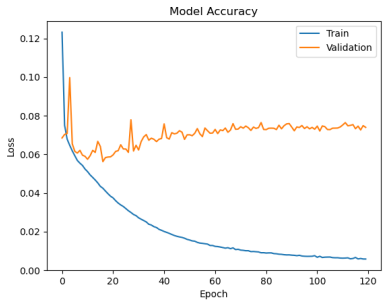
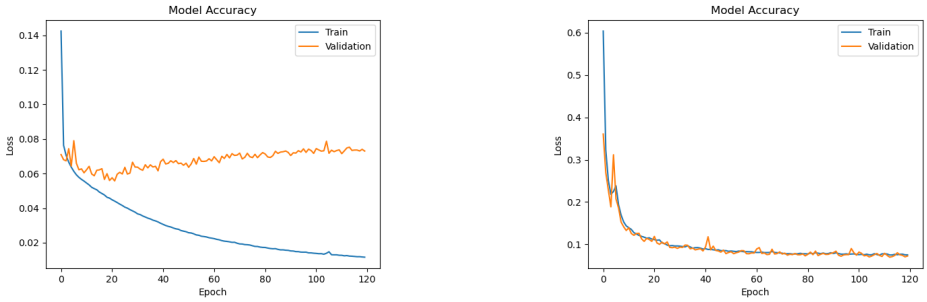


Figure 5.17.: Cross validation of 'RootMxb1' in 'Wave' case when trained with SM using 120 epochs, learning rate equal to 0.0005 and 200 nodes per layer

As can be seen, the two curves are both flat, thus confirming the appropriate choice of epochs and learning rate, but they are very distant from each other. This signifies that the model's definition has to be changed [78]. In this case, the number of nodes is adjusted. All the simulations' results shown until now are obtained using an architecture built using 200 nodes per layer. This is reduced first to 100 and then to 5, reducing 10 values each time until 10 and then 5 values in the last test. The resulting cross-validation graphs for the extreme options are shown in the following Figure 5.18.



(a) 100 nodes per layer

(b) 5 nodes per layer

Figure 5.18.: Comparison of cross-validation results for 'RootMxb1' in 'Wave' case when trained with SM using 120 epochs and learning rate equal to 0.0005

These two cases correspond respectively to an overfitting and underfitting of the data set. This can be easily understood by analyzing the corresponding RMSE graphs, which are reported in the following Figure 5.19 together with other intermediate cases. When the SM is built using 100 nodes, the maximum error is equal to 1438 $\mathrm{kN\,m^{-1}}$, while in the latter case, it is 1421 $\mathrm{kN\,m^{-1}}$.

As a result, the number of nodes adopted is an option between the two extremes, resulting in the lowest RMSE. It equals 30, which eventually still displays a difference between the training and validation curves. This is a result of overfitting. However, the difference between the curves is generally quite small [79]. Overall, reducing the number of nodes not only improves the accuracy of the forecast but also simplifies the model, reducing the com-

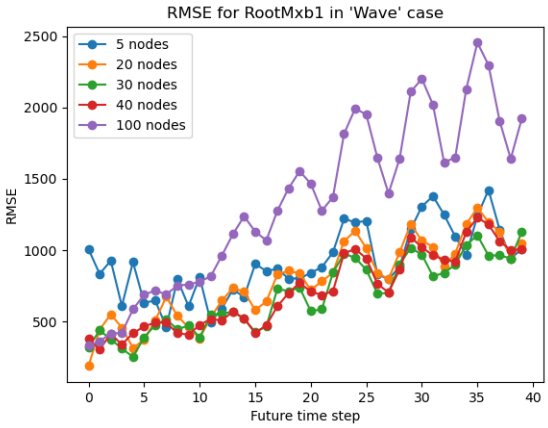


Figure 5.19.: Comparison of RMSE for 'RootMxb1' in 'Wave' case when trained with SM using 120 epochs and learning rate equal to 0.0005

putational time. The results obtained when the model is built using 30 nodes. 120 epochs and a learning rate equal to 0.0005 are shown in the following Chapter 6.

Chapter 6

Results

To implement the procedures described in the previous chapter, Python codes are written and run on HPC, where one node in the queue 'fpt-small' is used. The results of the forecasts performed applying ARIMAX and LSTM are presented in this chapter, with a particular focus on the latter, as the non-linear solution seems to be more promising for this analysis.

To forecast using ARIMAX the 'pm.auto_arima' function is used. As mentioned in Section 5.2.2, this automatically detects the best order to apply for the provided data.

In this study, given the results of the previously performed ADF described in Chapter 4, the order for differentiation is set to zero, while the maximum orders of auto-regression, 'p', and moving average, q, operations are set to 20. Despite these settings, the model never adopts an order higher than 4 for either 'p' or 'q', for both the exogenous and endogenous inputs. A clear pattern in the choice of these values could not be identified, they solely depend on the nature of the input to the model. This consists of the previous 40 real values of the considered variable.

For all the analyzed cases, the plotting of the forecast compared to real data results in very diverse graphs. In some cases, the prediction agrees well with the reference, while the scale is rather wrong in others. This is due to the limited ability of ARIMAX to handle non-linear relationships. The best performances for ARIMAX implementation are achieved when the variable tends to repeat the same behavior, thus maintaining the same period and oscillation range.

Regarding LSTM, after the studies described in the previous Section 5.4, the chosen architecture is an encoder-decoder structure made of two LSTM layers of 30 nodes using 120 epochs with an initial learning rate equal to 0.0005, as summarized in the following Table 6.1. This chapter reports the results of the variable 'RootMxb1', the edgewise moment at the blade's root, and 'RootMyb1', the blade flapwise moment at the blade root. These are examples of the two types of behaviors identified when analyzing the results of this type of SM.

All the other variables are assessed in the appendix: the blade pitching moment and the blade in-plane moment, which behave similarly to 'RootMxb1', and the blade out-of-plane moment, which has behaviors analogous to the ones of 'RootMyb1', are shown for the 'Wave', 'NoWave' and 'PI' case respectively in Appendix B, Appendix C and Appendix D.

Table 6.1.: Summary of the selected hyper-parameters for LSTM

Hyper-parameter	Value
Batch size	16
Number of epochs	120
Learning rate	0.0005
Number of nodes per layer	30

The reason for these correlations can be identified in two main points. First, while the former variables operate on a scale ranging over approximately 50 000 kN m^{-1}, the latter ones span over 90 000 kN m^{-1}, which is almost double. Second, the blade edgewise, pitching, and in-plane moment are correlated to the in-plane deflection while the blade flapwise and out-of-plane moment receive the out-of-plane deflection as exogenous inputs during training.

As far as the computing time is considered, the required time to run each simulation on HPC using 1 node is comprised between two and three hours for ARIMAX and nine and ten hours for LSTM. As previously mentioned, while the linear solution has to build a new model for each forecast, the non-linear one requires more time to build a single model that can be applied for more sections and only takes a few seconds to forecast.

The results obtained by implementing the two SM are shown in this chapter, which is structured as follows. First, in section Section 6.1 the output of the 'Wave' case implementation is shown. Next, in Section 6.2 the results of the 'NoWave' case are reported followed by those of the 'PI' case in Section 6.3. In all these sections, only the results of edgewise and flapwise moments at the blade root are assessed, and a comparison between the linear and non-linear data-driven methods is provided for each of them. Finally, a comparison and overview of all the obtained predictions and results are summarized in Section 6.4.

To display examples of the obtained forecasts, some predictions together with the data from the corresponding test sections are shown. To clarify the origin of the data, in the description of each figure the environmental variables given as inputs in OpenFAST, as explained in Chapter 3, are reported in a vector $\boldsymbol{x}$ in the same order shown in Table 6.2.

Table 6.2.: Ordered OpenFAST inputs

Reference wind speed (m s^{-1})	Significant wave height (m)	Difference between wave and wind directions (°C)	Turbulence intensity (%)	Wave period (second)
URef	H_s	$Delta_d$	TI	T_p

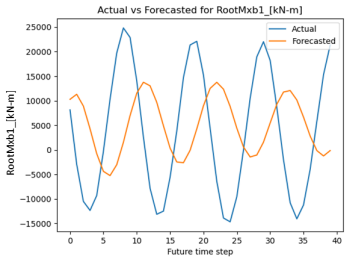
6.1. 'Wave' case

The first analyzed case is the 'Wave' one. This considers as exogenous variables to simulate the atmospheric conditions the wave elevation, the wind time series, and the reference value for wind speed adopted by 'TurbSim' to generate the stochastic wind speed time series. Moreover, an additional turbine parameter, the blade deflection of the considered blade, is also provided as exogenous input. In this section, first the results obtained for the blade

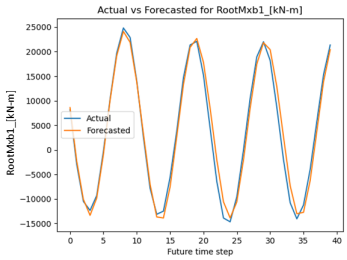
edgewise moment are presented in Section 6.1.1, followed by those of the blade flapwise moment in Section 6.1.2.

6.1.1. Blade edgewise moment - 'RootMxb1'

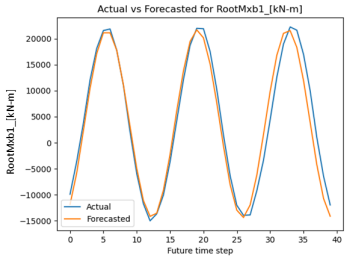
For both ARIMAX and LSTM, the results of the forecast are plotted in comparison to the values of the simulations results obtained from OpenFAST and inserted in the test database. Some of the generated results can be found in the following Figure 6.1.



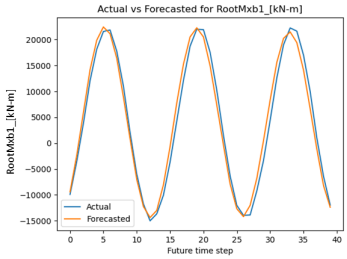
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 480 s



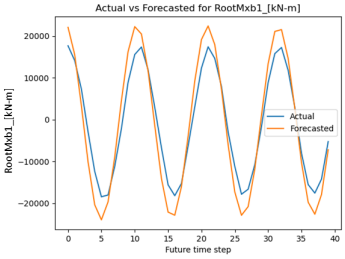
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 480 s



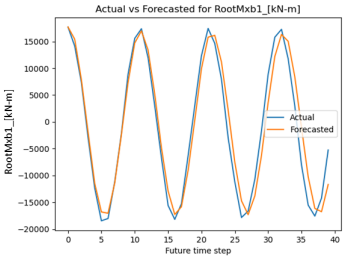
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 330 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 330 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 540 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 540 s

Figure 6.1.: Examples of forecast results for 'RootMxb1' in 'Wave' case

Here three different sections are taken as examples to show the performance of the two SM in different operating conditions. As a matter of fact, the forecast simulations are selected to exemplify the functioning of the turbine below, at, and above rated wind speed. The

same simulations, even though considering different sections, will also be used for all the other variables and cases. In particular, the first one, operates at $5.9\,\mathrm{m\,s^{-1}}$. As mentioned in Chapter 3, rated wind speed is $10.59\,\mathrm{m\,s^{-1}}$, thus the second example reference wind speed is equal to $10.8\,\mathrm{m\,s^{-1}}$. Finally, for the operations above rated wind speed, the simulation corresponding to a 'URef' equal to $20.6\,\mathrm{m\,s^{-1}}$ is chosen.

As shown, it can happen that the two models provide very similar results, as it happens for rated wind speed, but this is generally not true. As previously mentioned, ARIMAX displays the lowest discrepancy between the actual and forecasted curves when the variable behaves exactly the same way as in the previous section. When this is not the case, behaviors such as the one shown for below rated wind speed may occur. On the other hand, LSTM can better capture the trend changes since it can also rely on long-term memory to make predictions.

To assess the accuracy of the trained model more analytically, the RMSE is calculated for both ARIMAX and LSTM for each time step of each simulation. Then, this is averaged and the results are plotted in the following Figure 6.2. As can be seen, the values obtained via LSTM are approximately six times less than the ones computed when ARIMAX is implemented. In the former case, the maximum RMSE is $1102\,\mathrm{kN\,m^{-1}}$. Compared to the reference scale of 'RootMxb1' comprised between $-21\,774\,\mathrm{kN\,m^{-1}}$ and $26\,853\,\mathrm{kN\,m^{-1}}$, this is equal to a percentage error of 2.3%.

Finally, to evaluate the convergence and accuracy of the LSTM model even further, a cross-validation analysis is performed, as shown in Figure 6.3. As mentioned in Chapter 5, it can be seen how the training curve becomes flat towards the end but this doesn't overlap with the validation curve. This is because the model is slightly overfitting the data series. Thus, it is possible to optimize the algorithm and further improve the forecast.

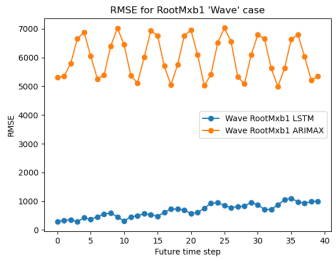


Figure 6.2.: RMSE of 'RootMxb1' in 'Wave' case

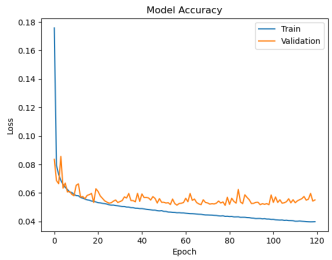
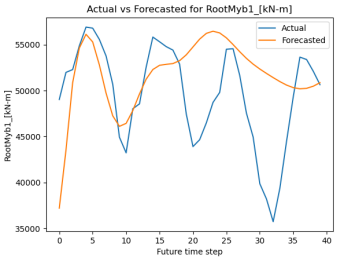


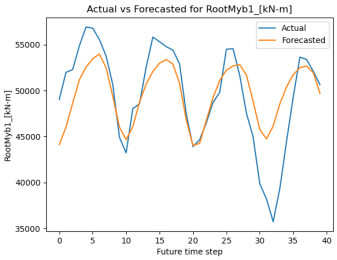
Figure 6.3.: Cross validation analysis of 'RootMxb1' in 'Wave' case

6.1.2. Blade flapwise moment - 'RootMyb1'

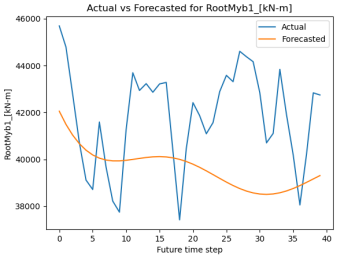
Compared to the trend seen for the edgewise moment, the results of the forecast of the blade flapwise moment are quite different. These are shown in the following Figure 6.4. It can be noticed how the behavior of this variable is much more irregular compared to the one observed in Figure 6.1, as the flapwise loads are driven by turbulence below rated wind speed. As a result, it is more difficult for both algorithms to compute accurate results.



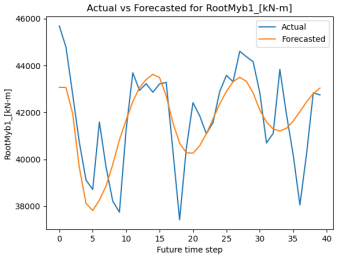
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 330 s



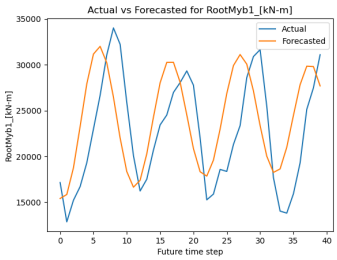
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 330 s



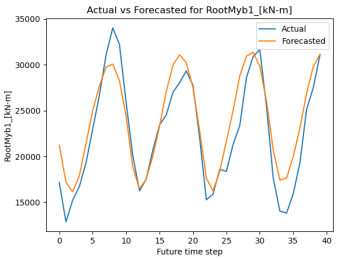
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 450 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 450 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 510 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 510 s

Figure 6.4.: Examples of forecast results for 'RootMyb1' in 'Wave' case

Given the characteristics of the blade flapwise moment, it can be noticed not only from the reported pictures, but also from all the others obtained and not shown in this study for space constrictions, that better forecasts are obtained above rated wind speed when the blades are pitched and the variable follows a more regular trend.

To improve the accuracy of the predictions, more data can be included in the training data set to provide the forecasting model with more examples of past load conditions, which

can help it identify and learn patterns, trends, and relationships related to the moments at the root blade. On top of that, a possible improvement in the forecast can be achieved by including measurements of the same target variable along different locations on the blade as exogenous features. However, this topic should be further investigated in future studies.

Despite the increased difficulty for both algorithms to produce accurate forecasts, LSTM is able to correctly capture the mean trend of the evolution of the loads. As a result, the analytical expression of the accuracy of the forecast through RMSE is significantly different for the two SM. The results of the average value obtained over all the 800 generated sections can be observed in Figure 6.5 and highlight two main points. First, the orange curve, related to ARIMAX implementation, is approximately four times higher than the blue one. Furthermore, the latter increases correctly as the time step is further in the future while the former has a more random behavior. This confirms that non-linear modeling is more successful for this type of application. The maximum RMSE value for LSTM in this case, is 1252 $\mathrm{kN\,m^{-1}}$. However, this corresponds to a percentage error of only 1.4%, given that the variable oscillates between -13 336 $\mathrm{kN\,m^{-1}}$ and 77 073 $\mathrm{kN\,m^{-1}}$, thus on a larger scale compared to the one of 'RootMxb1'.

Finally, the cross-validation analysis is also performed in this case. As seen in Figure 6.6, the validation curve is not always above the train one. This can be due to overfitting since the validation curve tends to increase in value as the number of epochs increases. This behavior highlights that the parameters are not well-tuned for this specific case.

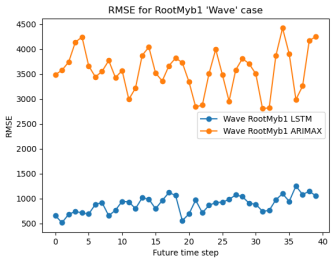


Figure 6.5.: RMSE of 'RootMyb1' in 'Wave' case

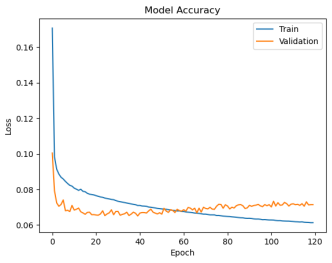


Figure 6.6.: Cross validation analysis of 'RootMyb1' in 'Wave' case

6.2. 'NoWave' case

To evaluate the impact of hydrodynamics on the loads perceived by the blades, the second analyzed case does not consider wave elevation as an external parameter. Still, it considers all the other variables mentioned in the previous Section 6.1 as exogenous variables. This means that the exogenous inputs, in this case, are the reference and time series wind speed and the blade deflection. As for the 'Wave' case, also in this section, first the 'RootMxb1' variable is analyzed in Section 6.2.1, followed by the 'RootMyb1' feature in Section 6.2.2.

6.2.1. Blade edgewise moment - 'RootMxb1'

In the following Figure 6.7, three different sections' forecasts are shown for both ARIMAX and LSTM. As for the previous case, these represent three different operating conditions: the first shows the behavior below wind speed, the second at rated wind speed, and the last above it.

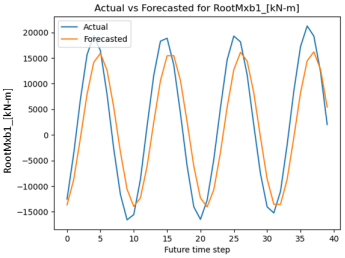
As expected, also in this case LSTM emulates the behavior of the test data more accurately, predicting correctly the period and the oscillations. On the other hand, ARIMAX either overestimates or underestimates the peaks and valleys of the oscillations and in most cases does not correctly identify when they occur, thus shifting them over time. This is once more due to the limited ability of the linear model to retain long-term memory. As a matter of fact, the predictions of ARIMAX are strictly linked to the behavior of the target variable in the previous section, thus it is highly probable that the frequency and amplitude of the oscillations are not accurate, as in this case.

The direct consequence of this observation is the difference between the computed RMSE. As shown in Figure 6.8, the values of the discrepancies measured for ARIMAX are almost eight times more than the ones resulting from the implementation of the non-linear SM. Furthermore, the trend followed by the orange curve is incorrect since it doesn't increase as the time step is further away. The error for LSTM is always lower than $1048\,\mathrm{kN\,m^{-1}}$. This is equal to only 2.2%, thus it is slightly lower compared to the one obtained for the 'Wave' case.

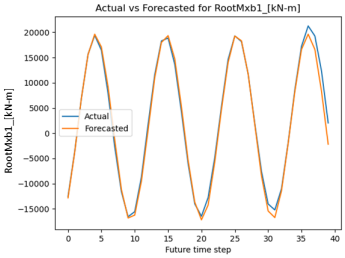
This difference may become even greater once the hyper-parameter tuning is optimized. As can be seen in Figure 6.9, the validation curve is slightly higher than the train one. Thus the model still overfits the data and the forecast achieved through LSTM might improve even more.

6.2.2. Blade flapwise moment - 'RootMyb1'

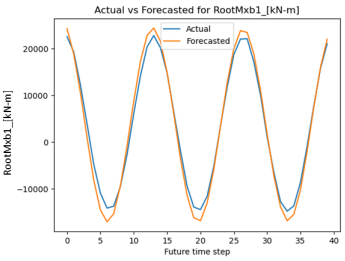
Figure 6.10 reports three forecast examples referring to three different operating conditions for both ARIMAX and LSTM implementation. As can be noticed by observing the blues lines, the behavior of the variable in the selected section is not periodical. This is expected given the characteristics of the variables mentioned in Section 6.1.2 and, as a result, both methods fail to accurately emulate the test data. However, while the non-linear model follows the trend of the target feature, the ARIMAX curve becomes flat when operating below and at rated wind speed, and the oscillations are heavily damped for the last case. This is probably due to the inability of the SM of extracting significant data. When the linear model is applied, the forecast is unlikely to improve even when the techniques mentioned in Section 6.1.2



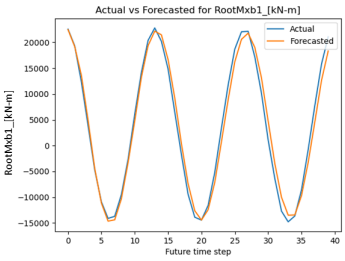
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 720 s



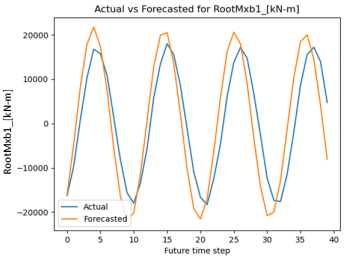
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 720 s



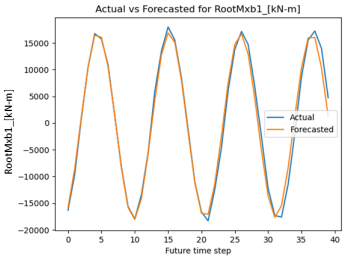
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 480 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 480 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 600 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 600 s

Figure 6.7.: Examples of forecast results for 'RootMxb1' in 'NoWave' case

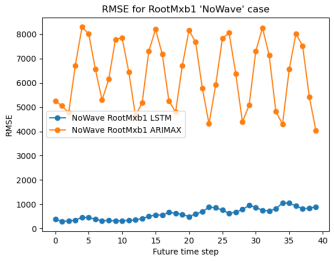


Figure 6.8.: RMSE of 'RootMxb1' in 'NoWave' case

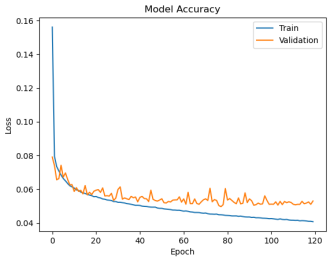
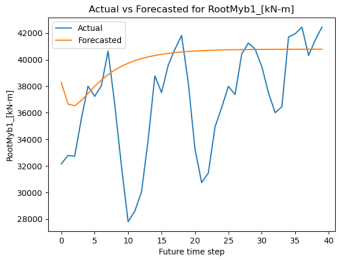
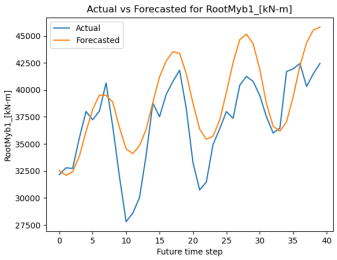


Figure 6.9.: Cross validation analysis of 'RootMxb1' in 'NoWave' case

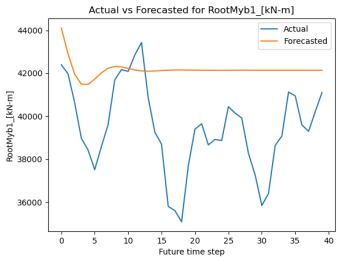
are implemented, but for the LSTM one or the combination of more of the aforementioned solutions may lead to more accurate results.



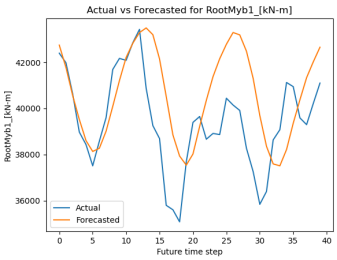
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 630 s



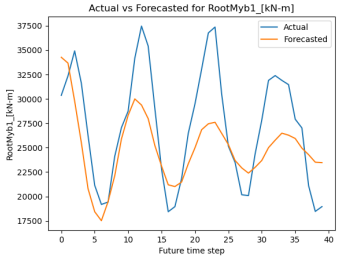
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 630 s



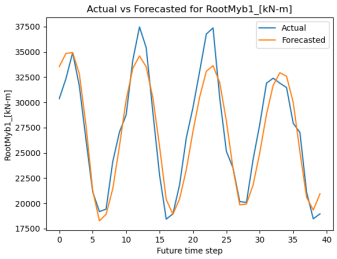
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 510 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 510 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 420 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 420 s

Figure 6.10.: Examples of forecast results for 'RootMyb1' in 'NoWave' case

The RMSE computed when ARIMAX is applied is significantly higher, almost eight times more, than the one resulting from LSTM implementation, as shown in Figure 6.11. The maximum error value for RMSE when LSTM is implemented is much higher compared to the case shown before since it is equal to 1978 kN m^{-1}, which corresponds to a percentage error of 2.1 %. Thus for the blade flapwise moment the 'Wave' case results in a lower error than the currently analyzed one when the hyper-parameters previously mentioned are applied for both cases.

Finally, from the evaluation of the cross-validation analysis performed for this application, it can be seen that the model is overfitting the data. As shown in Figure 6.12, the two curves intersect each other, and the validation one, in particular, assumes higher values as the number of epochs increases.

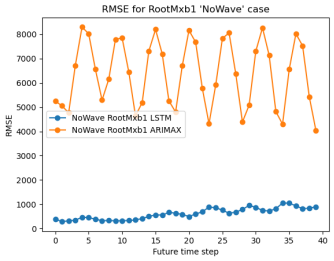


Figure 6.11.: RMSE of 'RootMyb1' in 'NoWave' case

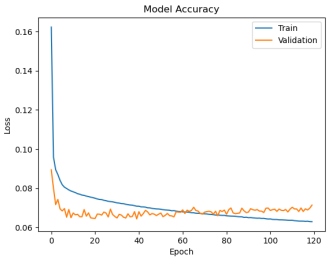


Figure 6.12.: Cross validation analysis of 'RootMyb1' in 'NoWave' case

6.3. 'PI' case

The last case considered in this study is the 'PI'. As mentioned earlier, it considers only the wind speed time series and the rotor speed as exogenous variables. This scenario is analyzed to assess the performance of the SM when it receives only the same variables currently received by PI controllers as exogenous inputs. Once more, the blade edgewise moment analysis is in Section 6.3.1, followed by the blade flapwise moment in Section 6.3.2.

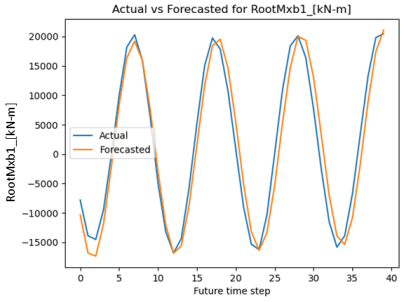
6.3.1. Blade edgewise moment - 'RootMxb1'

For this last case, once more operating conditions below, at, and above rated wind speed are represented through three different sections. These are displayed for both ARIMAX and LSTM results in the following Figure 6.13.

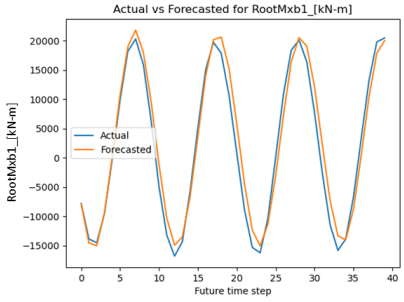
The chosen sections show that also the linear model is quite accurate in all the displayed graphs, but this is not the case in reality. The results shown in Figure 6.13 are achieved thanks to the periodicity and regularity of the target variable but, as can be seen in the following Figure 6.14, the amplitude and frequency of the oscillation are not always correctly detected by ARIMAX, while they are when LSTM is implemented. The behavior shown in this figure for the linear model is more similar to what is expected for this SM given not only the high dependency of the forecast from the measurements of the previous section but also the weak relations obtained from cross-correlation analysis performed between the target and exogenous features.

As a matter of fact, also for the 'PI' case, the RMSE is significantly higher when computed for ARIMAX application compared to the one of LSTM. In this case, also an anomaly can be seen in the curve of the ARIMAX RMSE: this is characterized by a peak for the values closer to the present time step. Conversely, the blue curve assumes higher values as the time step progresses, highlighting the increasing inaccuracy in results for time-distant instants. For LSTM, the maximum error is equal to $1253\,\mathrm{kN\,m^{-1}}$ which corresponds to a percentage error of 2.6%. This is higher than the one obtained in both 'Wave' and 'NoWave' cases.

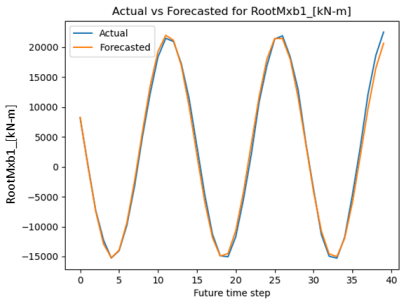
Finally, the cross-validation performed to assess the tuning of the LSTM hyper-parameters in Figure 6.16, shows a behavior similar to what has been observed for the blade flapwise moment: the validation curve is partly below the train one and partly above it. This can be due once more to overfitting.



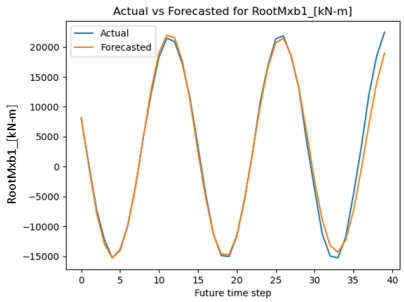
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 750 s



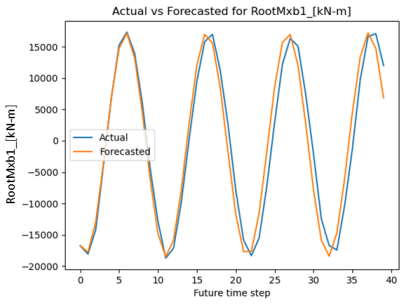
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 750 s



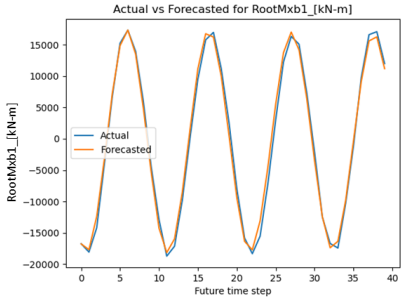
(c) ARIMAX$\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 420 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 420 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 480 s

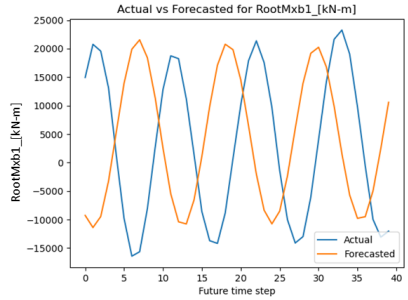


(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 480 s

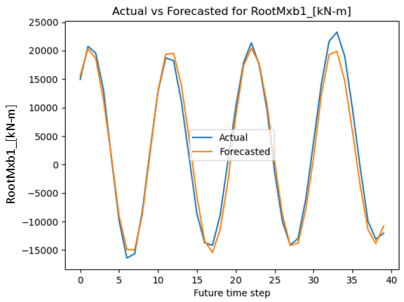
Figure 6.13.: Examples of forecast results for 'RootMxb1' in 'PI' case

6.3.2. Blade flapwise moment - 'RootMyb1'

In the following Figure 6.17, the forecasts of three different sections provided by both ARIMAX and LSTM are reported. Given the non-periodical behavior of the target variable and the weak correlations detected in Chapter 4 with the exogenous variables, it can be clearly seen how ARIMAX fails to provide accurate predictions. Furthermore, in this case, also LSTM fails to correctly emulate the trend followed by the variable in all cases. This is due to both the weak correlation between the exogenous variables and the target output and



(a) ARIMAX $\boldsymbol{x}$ = [15.5, 1.4, -145.4, 13.3, 7.3] at t = 420 s



(b) LSTM $\boldsymbol{x}$ = [15.5, 1.4, -145.4, 13.3, 7.3] at t = 420 s

Figure 6.14.: Examples of forecast results for 'RootMxb1' in 'PI' case

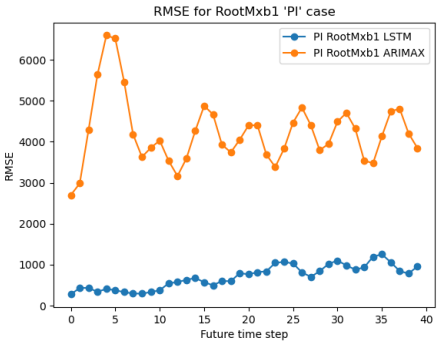


Figure 6.15.: RMSE of 'RootMxb1' in 'PI' case

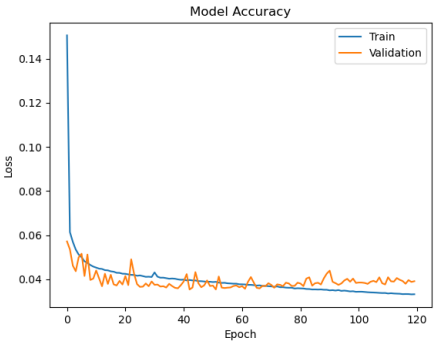
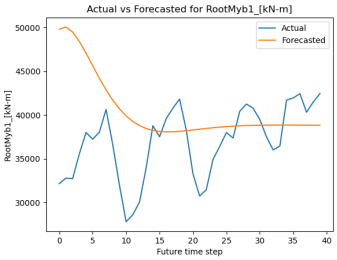


Figure 6.16.: Cross validation analysis of 'RootMxb1' in 'PI' case

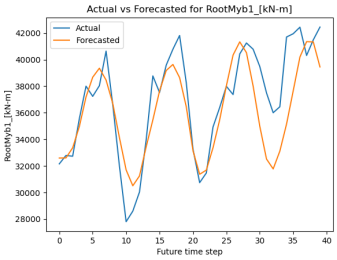
the non-periodic behavior of 'RootMyb1'. To achieve better results, it may be particularly useful to implement a model that can include uncertainty. This would help to capture the variations in wind speed, which in this case is one of the only two exogenous variables implemented, thus assuming a fundamental role for the forecast.

As a consequence, the discrepancy in RMSE evaluated for the ARIMAX and LSTM model is lower here compared to the other cases. As shown in Figure 6.19, the curves overlap around the 10^{th} time step, even though the curve obtained from the non-linear SM application is still generally lower. The maximum error value obtained in this case is 1569 kN m^{-1}, corresponding to a percentage error equal to 1.7%.

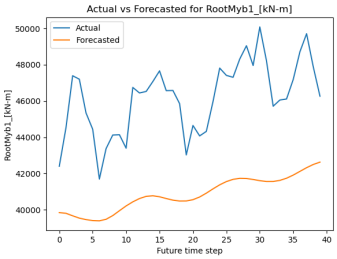
Finally, also for this analysis the cross-validation results are shown in Figure 6.19. As for 'RootMxb1', the two curves overlap, and the validation one increases as the number of epochs progresses. This is a clear sign of overfitting.



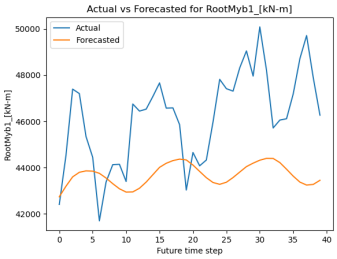
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 630 s



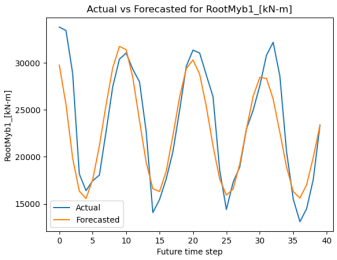
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 630 s



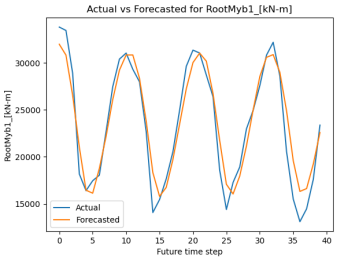
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 750 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 750 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 540 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 540 s

Figure 6.17.: Examples of forecast results for 'RootMyb1' in 'PI' case

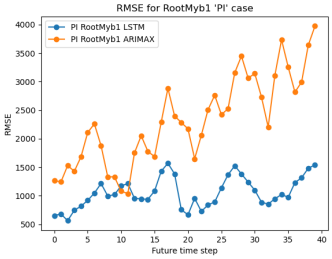


Figure 6.18.: RMSE of 'RootMyb1' in 'PI' case

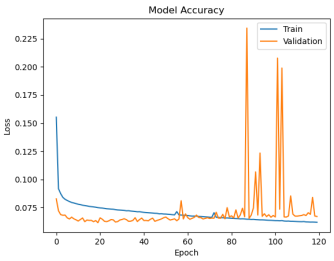


Figure 6.19.: Cross validation analysis of 'RootMyb1' in 'PI' case

6.4. Comparison of the forecasting methods

Overall, the implementation of LSTM has proven to be more successful compared to the one of ARIMAX. This is linked to multiple factors. The first consideration regards the real-life application of this study. While for the non-linear model it is possible to define a single model that can be updated each time with the most recent forecast to obtain the values of the target feature in the next 30 seconds, for the linear solution, this is not possible. Thus, rebuilding and retraining the model at each time step is necessary. This results in a computational time that is too high for practical implementation.

Furthermore, the RMSE from ARIMAX is up to seven times greater compared to the one obtained from the implementation of LSTM. The main reason for this performance relies on the ability of LSTM to retain the memory of behaviors in the training data set that occurred further away in the past compared to ARIMAX, which only takes into account the last 40 time steps in this case.

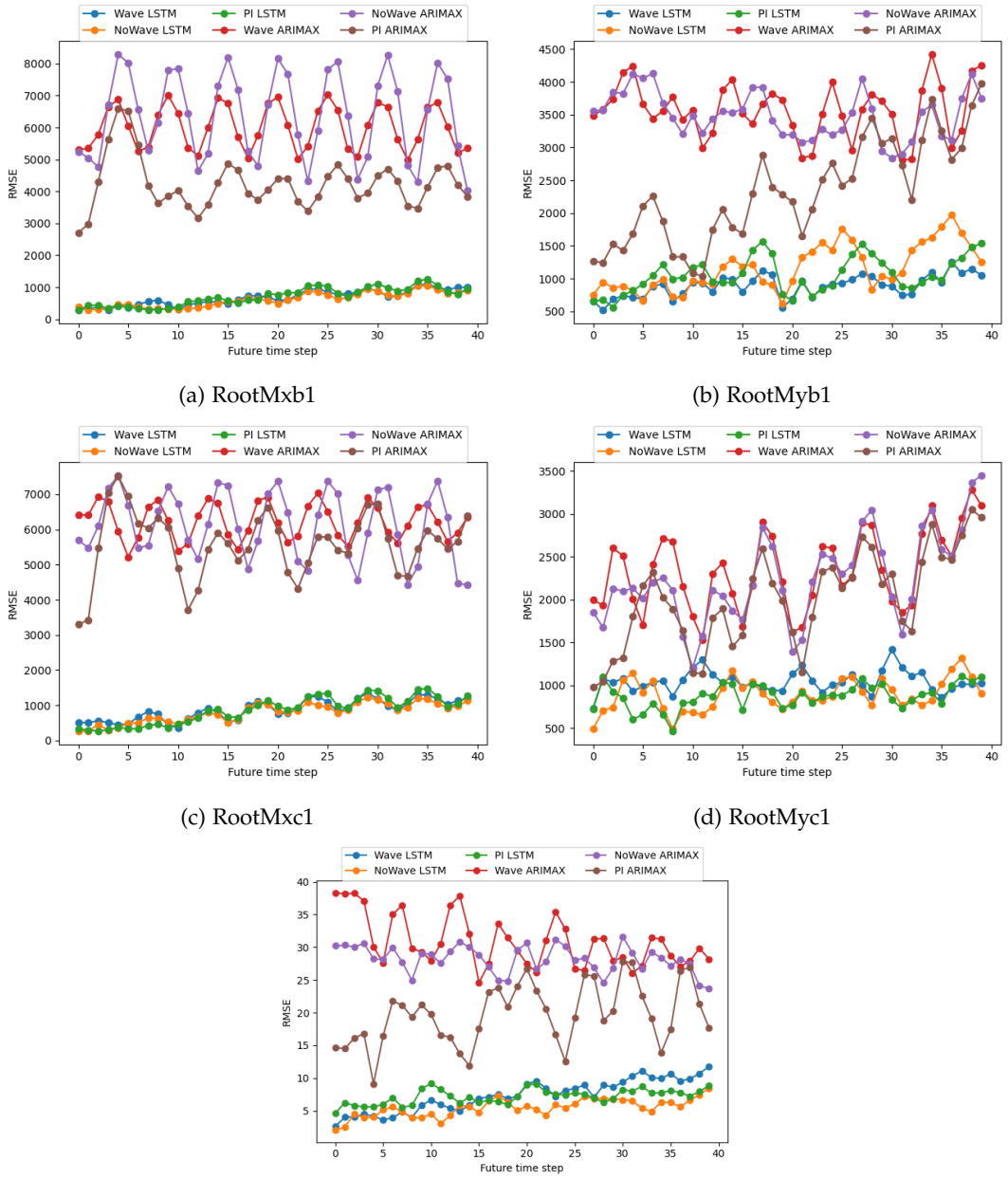
Within the application of LSTM, the case that obtained the lowest RMSE is 'NoWave' for the blade edgewise moment, the blade pitching moment, and the blade in-plane moment. However, the resulting percentage error for 'RootMxb1' compared to the 'NoWave' case is only 0.1% higher for the 'Wave' case and 0.4% for the 'PI' one. The latter is the maximum one overall, and it is equal to 2.6%.

For the blade flapwise and out-of-plane moment, the best performance is achieved when the 'Wave' case and the 'PI' case are respectively implemented. Since no consistent trend is detected and the hyper-parameters are poorly calibrated for this target feature, no conclusion is certain. In particular, the percentage error obtained in the 'Wave' case for 'RootMyb1' is equal to 1.4%, while the one for the 'NoWave' is 2.1%. This value represents the worse performance for the flapwise moment, given that for the 'PI' case the percentage error equals 1.7%.

It should be noted that the percentage errors are calculated over the variation span of data coming from 200 different simulations, thus while a certain value of RMSE may have little impact in a specific case, the same absolute value can correspond to completely wrong forecasts in another section. This should also be considered when evaluating the performance of the LSTM implementation. A summary of the obtained errors for all the variables analyzed in this study can be found in the following Figure 6.20.

Given the described computed mechanism, all the errors achieved through the implementation of LSTM are still very low, and such a result may be due to the use of a dataset originating from training instead that from real data: it is based on another mathematical model, which also makes assumptions to model real-life events. In particular, in this study, BEM is used to compute the aerodynamics of the turbine. This is already a simplification compared to OLAF, as seen in Chapter 3. However, even if the accuracy of the model may become worse as other data sets are used, the main results highlighted in this section, thus the limited impact of wave elevation on the edgewise, pitching, and in-plane moment at the blade root and the effectiveness of LSTM and non-linear models in general compared to ARIMAX, should be confirmed.

Finally, the encoder-decoder architecture adopted when implementing in LSTM is not strictly necessary given the nature of the data. However, it is generally useful to improve the performance of the forecast since the encoder processes historical load data and summarizes it



(a) RootMxb1

(b) RootMyb1

(c) RootMxc1

(d) RootMyc1

(e) RootMzc1

Figure 6.20.: Summary of RMSE for all the variables analyzed

into a context vector, which is then used as the initial state for the decoder. This can be especially beneficial for load forecasting, as it enables the model to consider the whole historical context when generating forecasts.

Chapter 7

Conclusions and future research

This chapter presents a summary of what has been analyzed so far, along with considerations on the whole study and suggestions for future research. In particular, in Section 7.1, a resume is defined following the questions defined in Chapter 1. Next, in Section 7.2, a general impression of the obtained results, along with ideas for future development, are given.

7.1. Conclusions

This study aimed to present a methodology to create a site-specific surrogate model that provides a time series forecast of the loads during normal operating conditions at different load-sensible spots along the tower and blades of a 15 MW bottom-fixed offshore turbine located 30 km off the west coast of Ireland. To effectively address the topic, different sub-questions were identified. These are here reported and answered. The first sub-question was:

What variables should be chosen to represent the environmental conditions and how should they be sampled to generate the best training database?

Throughout the literature review, average wind speed, turbulence intensity, wave direction, significant wave height, and peak spectral wave period were the most common features used to describe environmental characteristics when offshore wind turbines were studied. This research collected these features from a real database of the identified region. This data was then used to define the domain of each variable and express it as relationships between the identified features. As a result, a sampling area was defined. Quasi-random sampling through Halton sequencing was then used to generate different simulated environmental conditions uniformly distributed throughout the identified area. Next, the second question was:

Which of the computed features influence the target loads the most?

To perform this study, the 15 MW NREL offshore wind turbine with a fixed-bottom monopile structure model was used. Simulations were run using both BEM and OLAF aerodynamic models. Since the difference in accuracy between the two models was not particularly relevant for this study, BEM was eventually preferred because of its reduced computational time. Later the results were analyzed, the stationarity of the data was verified and cross-correlation was applied to determine which variables were correlated. Through cross-correlation, five different target features were selected, considering five different exogenous variables, which are retrievable through different instrumentation in reality. The selected variables to be forecast were the blade in-plane and out-of-plane moment, the blade pitching moment, and the edgewise and flapwise moment. These showed a strong correlation with the out-of-plane and in-plane deflection. On top of these, the reference wind speed, the wind speed, and significant wave height time series were also considered. Next, the most significant locations to study these features were inquired about:

What are the most load-sensible points on the blade?

The most important zones for the design of the turbine are the blade roots. For this reason, these are the areas where the loads were forecast. Once all these aspects have been inspected, the training of the surrogate model began.

What features should be chosen for the implemented SM and which exogenous variables should be considered as inputs for each target parameter?

To assess the impact of the hydrodynamic loads on the moments at the root of the blade, the cases 'Wave' and 'NoWave' were defined. The former considers as exogenous variables the wave elevation, the wind time series, and the reference value for wind speed adopted by 'TurbSim' to generate the stochastic wind speed time series to simulate the atmospheric conditions and an additional turbine's parameter which is either the in-plane or out-of-plane deflection, depending on the target feature. The latter includes the same variables, except for the wave elevation time series. Next, a case was investigated to relate the future implementation of MPC to the current PI controllers. This was named 'PI' case and considered as inputs only the wind speed time series, currently obtained as input of the controller via an estimator, and the rotor speed, which is the only measured exogenous inputs of the currently implemented control strategy. Overall, an evaluation of the implemented strategies to train the SM had to be made:

What time series forecast model works best to accurately predict the loads and why?

Given the results obtained by the error estimates and the real-life application of the study, it is clear that the LSTM, the non-linear model, performs better than ARIMAX. This is true from numerous points of view. First, it is possible to train the model only once and obtain different forecasts depending on the new input received by the most recent measurements. Next, the overall performance of the model is almost seven times more accurate in all cases. Finally, once the model is built, it is faster to fit the results. Within the LSTM application, two types of behaviors were identified, thus they were represented by the blade edgewise and flapwise moments. For the edgewise moment, the most accurate case was the 'NoWave' one but this was estimated to be only 0.1% better compared to the 'Wave' one. The 'PI' case error is estimated to be approximately 2.6% and it is the one performing less accurately, 0.4%

worse compared to the best predictions. For the flapwise moment, the 'Wave' case was the most precise. However, given the poor tuning of the selected hyper-parameters for the LSTM when applied to this variable, no real conclusion can be achieved. Thus, the main research question can be answered. This was:

Can data-driven surrogate models be used for forecasting load time series on offshore wind turbines?

Through the definition of the proposed methodology, accurate forecasts were obtained. Furthermore, it was proved that linear models are not adequate to perform this prediction and more powerful, non-linear solutions have to be implemented. LSTM proved to be a good solution, given its ability to also retain the memory of events that are further away in time with respect to the present.

7.2. Further research

Even though this work successfully answered the research question initially proposed, many aspects of time series load forecasting still have to be explored. In this section, the limitations of this thesis are used as a guideline for future research.

First, it would be interesting to repeat the hyper-parameter analysis using another variable, such as the blade flapwise moment, to obtain values of the learning rate, number of epochs, and nodes more suited to this variable. The same study can also be repeated on the blade edgewise moment, refining the adopted values and differencing them depending on the considered case.

Next, as far as the model's architecture is concerned, the encoder-decoder architecture adopted for LSTM is not essential even though it should improve the accuracy of the SM. It would be interesting to repeat the study implementing simple LSTM or another type of NN to assess the importance of long-term memory in this type of forecast. The goal of this research would be to assess the impact that the encoder-decoder architecture has on the accuracy of the forecast. Furthermore, it would be interesting to investigate this aspect to determine if it is possible to reduce the computational effort, making the implementation of loads forecasting in MPC easier.

Another area to investigate for further development pertains to extending the forecast time horizon. The current study focuses on thirty-second forecasts, which is a time horizon long enough to fit the model with the updated values from the previous section and obtain the ones for the future time frame. Despite this, it remains uncertain whether this duration will prove sufficient given future advancements and the subsequent implementation of the model predictive controller. Investigating and determining the optimal forecast horizon that strikes a balance between accuracy and practical applicability is essential.

A further deepening of the analysis can focus on using OLAF instead of BEM to create the training algorithm. The results of such a surrogate model should then be compared to the one presented in this research to confirm the assumptions of the limited impact of this decision on the final performance. To expand on this point, testing the methodology using real-life measurements in the database would also be useful. This would enable to assess the performance of LSTM when trained with data that is not the result of other simulations and evaluate if the errors in estimations stay very low or increase significantly.

Lastly, replicating the same study on offshore floating wind turbines represents an important step forward. This would enable an assessment of the performance of LSTM models in scenarios where hydrodynamic loads play a more prominent role compared to the cases analyzed in the present report. Such a study would provide valuable insights into the adaptability and effectiveness of these models in varying wind turbine configurations and environmental conditions. This would be extremely useful in the implementation of MPC, which is being further investigated in relation to wind turbines especially to be applied in FOWT.

The presented suggestions should be updated and redefined also depending on the improvement in the definition of the model predictive controller. By embarking on these suggested research endeavors, a deeper understanding of time series load forecasting can be cultivated, facilitating advancements in the field and the development of more accurate and reliable forecasting methodologies.

Appendix A

Heat maps summarizing cross-correlation results

As anticipated in Chapter 4, in this appendix two examples of heatmaps summarizing the results obtained for cross-correlation are shown. In particular, Figure A.1 shows the result obtained in the simulation denominated 'Case 1' while Figure A.2 refers to the one called 'Case 2'.

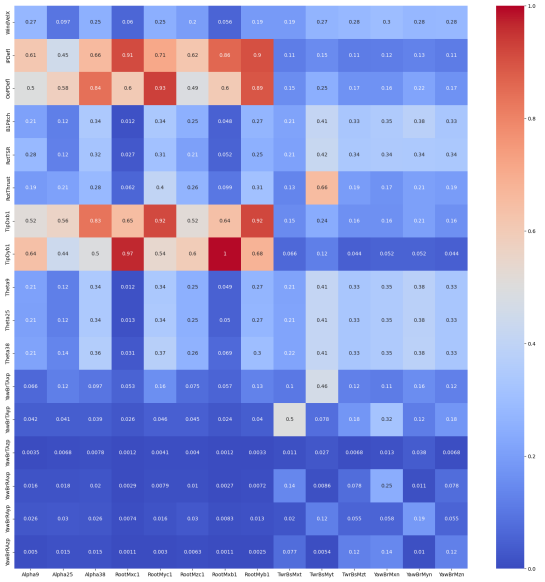


Figure A.1.: Heatmap summarizing cross-correlation analysis results for 'Case 1'

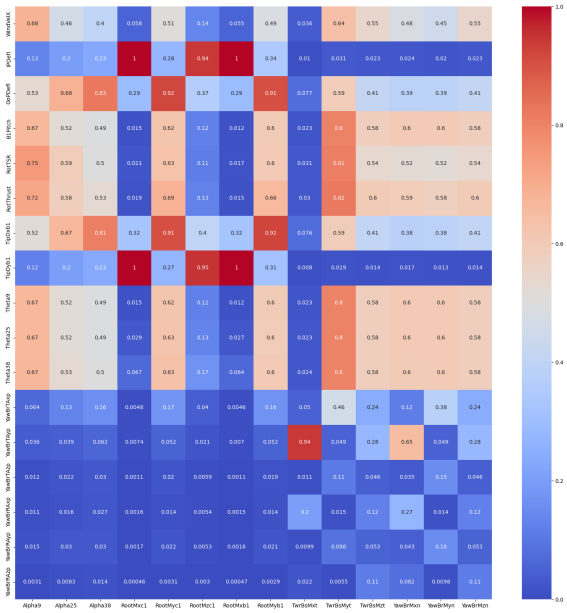


Figure A.2.: Heatmap summarizing cross-correlation analysis results for 'Case 2'

Appendix B

Results 'Wave' case forecasts

This appendix reports the results obtained for the 'Wave' case for all the target variables selected in Chapter 4, excluding the blade edgewise and flapwise moment. These are shown in Chapter 6. In this case, the wave elevation, the wind time series, and the reference value for wind speed adopted by 'TurbSim' to generate the stochastic wind speed time series to simulate the atmospheric conditions, and an additional turbine parameter depending on the target output are considered as exogenous inputs.

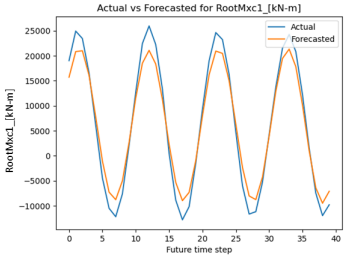
Three examples of the forecast achieved through both ARIMAX and LSTM compared to test data, the RMSE graphs and the cross-validation performed on the non-linear model are shown for each feature. The examples depict the same simulations shown in Chapter 6, thus illustrating the behavior of the turbine in different operating conditions. The results for the blade in-plane moment are in Section B.1, for the blade pitching moment in Section B.2, and for the blade out-of-plane moment in Section B.3.

B.1. Blade in-plane moment

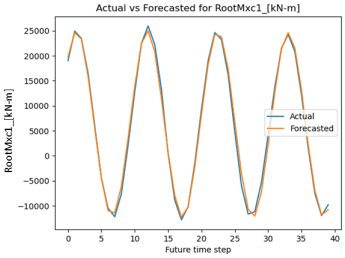
The blade in-plane moment behaves similarly to the blade edgewise moment. When the graphs of the visualization of the forecasts in Figure B.1, the RMSE in Figure B.2 and the performed cross-validation analysis reported in Figure B.3 are observed, the analogies are clear. The maximum value of RMSE obtained for 'RootMxc1' when forecasted applying LSTM is 1301 $\mathrm{kN\,m^{-1}}$, which corresponds to a percentage error of 2.4% given that the variable oscillates between $-19\,614\,\mathrm{kN\,m^{-1}}$ and $33\,881\,\mathrm{kN\,m^{-1}}$.

B.2. Blade pitching moment

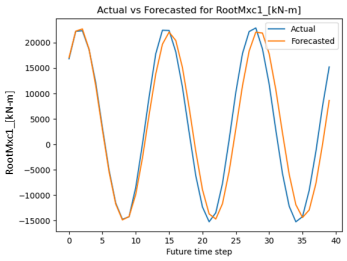
The blade pitching moment also displays a behavior similar to the blade edgewise moment. As a matter of fact, the graphs of the visualization of the forecasts in Figure B.4, the evaluation of the RMSE in Figure B.5 and the cross-validation analysis in Figure B.6 are very similar to the one shown in Chapter 6 even though the scale of this variable is different. 'RootMzc1'



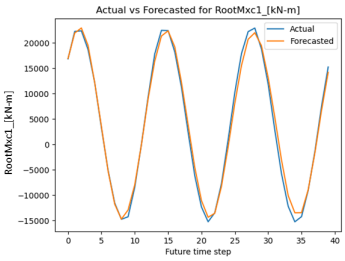
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 540 s



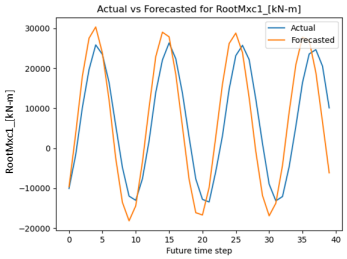
(b) LSTM$\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 540 s



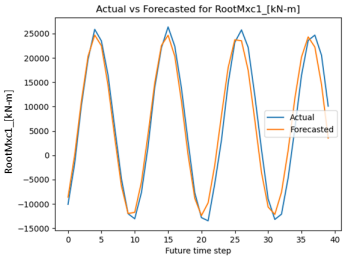
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 570 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 570 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 600 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 600 s

Figure B.1.: Examples of forecast results for 'RootMxc1' in 'Wave' case

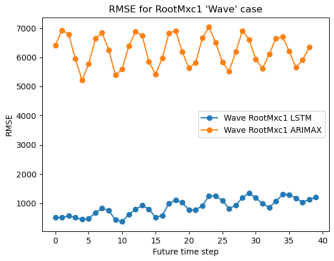


Figure B.2.: RMSE of 'RootMxc1' in 'Wave' case

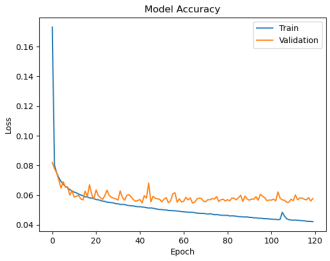
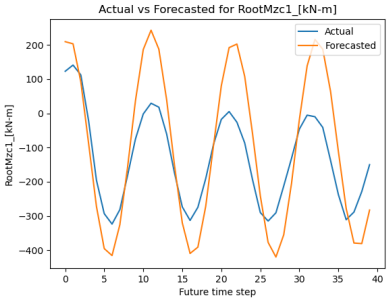
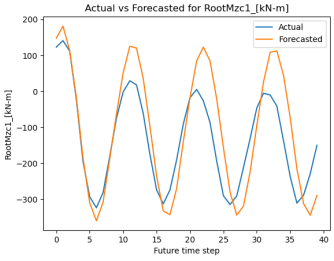


Figure B.3.: Cross-validation analysis of 'RootMxc1' in 'Wave' case

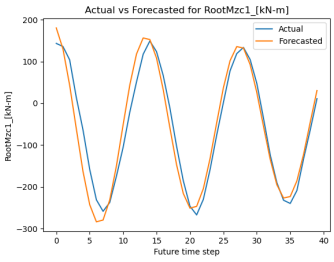
varies between $-508\,\mathrm{kN\,m^{-1}}$ and $240\,\mathrm{kN\,m^{-1}}$, thus, if the maximum RMSE obtained from LSTM application is equal to $11.8\,\mathrm{kN\,m^{-1}}$, the percentage error is 1.6%.



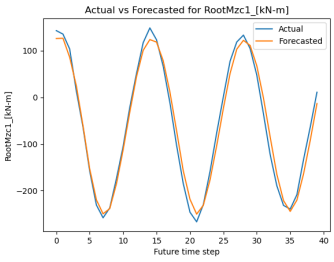
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 600 s



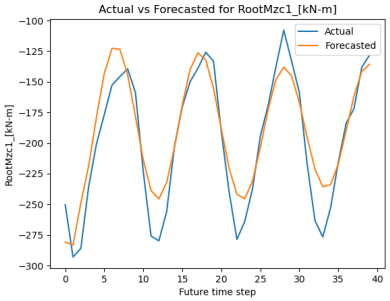
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 600 s



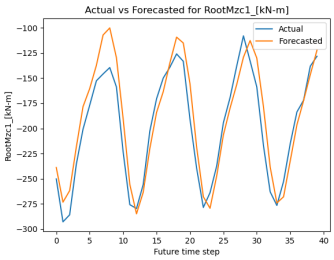
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 360 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 360 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 690 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 690 s

Figure B.4.: Examples of forecast results for 'RootMzc1' in 'Wave' case

B.3. Blade out-of-plane moment

The blade out-of-plane moment has a behavior similar to the one of the blade flapwise moment. This is clearly visible when forecasts shown in Figure B.7 is analyzed. Moreover, the RMSE in Figure B.8 and the cross-validation reported in Figure B.9 are analogous to those of 'RootMyb1' shown in Chapter 6. The maximum RMSE value for LSTM application in this

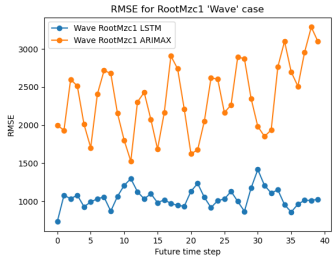


Figure B.5.: RMSE of 'RootMzc1' in 'Wave' case

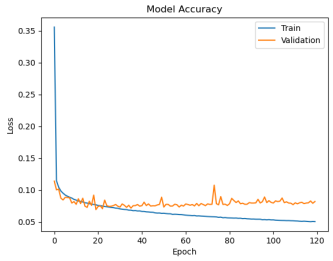
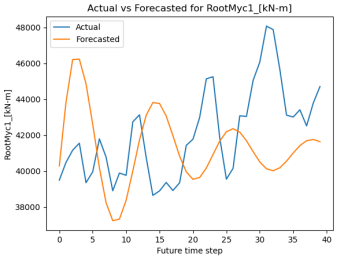
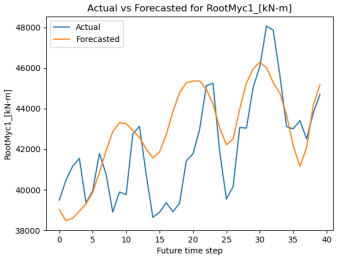


Figure B.6.: Cross-validation analysis of 'RootMzc1' in 'Wave' case

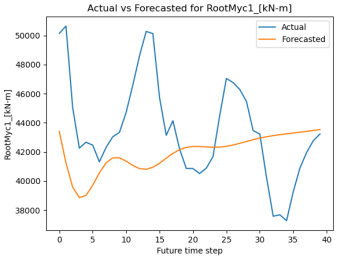
case is equal to 1416 kN m^{-1} which corresponds to a percentage error of 1.6% given that the target output oscillates between $-13\,386$ kN m^{-1} and 77 099 kN m^{-1}.



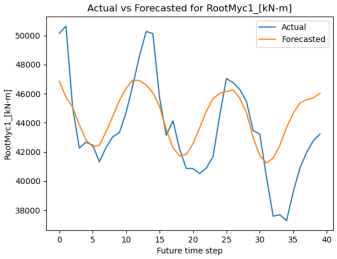
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 690 s



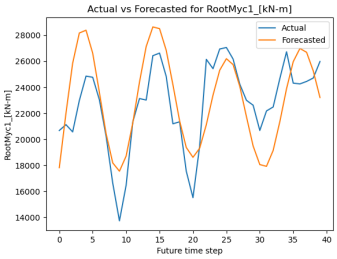
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 690 s



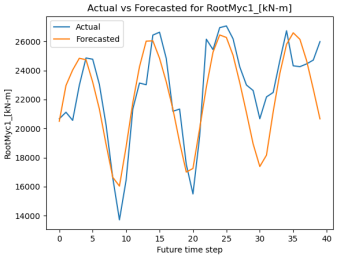
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 570 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 570 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 750 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 750 s

Figure B.7.: Examples of forecast results for 'RootMyc1' in 'Wave' case

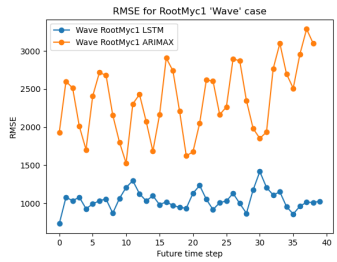


Figure B.8.: RMSE of 'RootMyc1' in 'Wave' case

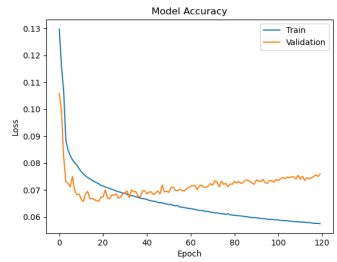


Figure B.9.: Cross-validation analysis of 'RootMyc1' in 'Wave' case

Appendix C

Results 'NoWave' case forecasts

This appendix reports the results obtained for the 'NoWave' case for all the target variables selected in Chapter 4, excluding the blade edgewise and flapwise moment. These are shown in Chapter 6. In this case, the wind time series, and the reference value for wind speed adopted by 'TurbSim' to generate the stochastic wind speed time series to simulate the atmospheric conditions, and an additional turbine parameter depending on the target output are considered as exogenous inputs.

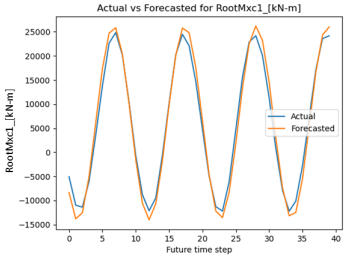
For each feature, three examples of forecast achieved via ARIMAX and LSTM implementation compared to test data, the RMSE graphs, and the cross-validation analysis of the LSTM model are shown. The chosen simulations are the same used in Chapter 6, thus depicting different operating conditions. The results for the blade in-plane moment are reported in Section C.1, for the blade pitching moment in Section C.2, and for the blade out-of-plane moment in Section C.3.

C.1. Blade in-plane moment

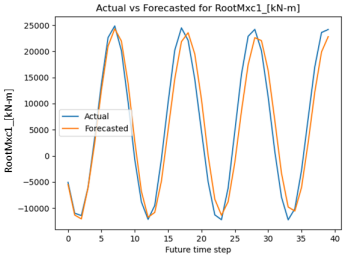
The blade in-plane moment behaves similarly to the blade edgewise moment. This is confirmed by all the graphs shown. In particular, the visualization of the forecasts in Figure C.1, the RMSE in Figure C.2, and cross-validation analysis is reported in Figure 6.9. The maximum value of RMSE obtained for 'RootMxc1' when LSTM is applied is $1197\,\mathrm{kN\,m^{-1}}$, which corresponds to a percentage error of 2.24%. As for the blade edgewise moment, also for the blade blade in-plane moment the 'NoWave' case results in a lower error compared to the 'Wave' one.

C.2. Blade pitching moment

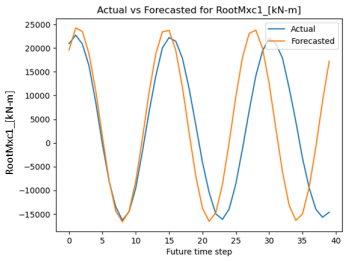
The blade pitching moment also displays a behavior similar to the blade edgewise moment. As shown in the graphs of the visualization of the forecasts in Figure C.4, the evaluation of the RMSE in Figure C.5 and the cross-validation analysis in Figure C.6, the characteristics of the aforementioned studies are analogous. In particular, for 'RootMzc1' in the 'NoWave'



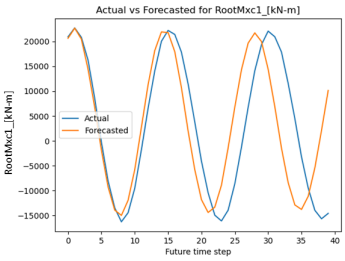
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 750 s



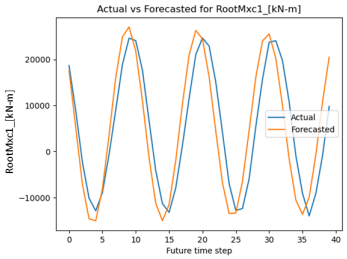
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 750 s



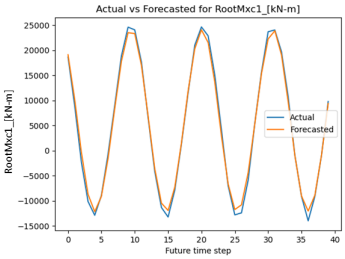
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 600 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 600 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 660 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 660 s

Figure C.1.: Examples of forecast results for 'RootMxc1' in 'NoWave' case

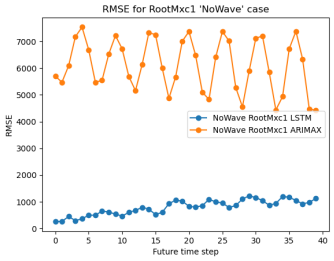


Figure C.2.: RMSE of 'RootMxc1' in 'NoWave' case

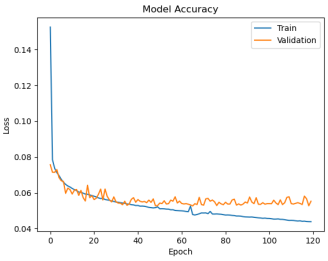
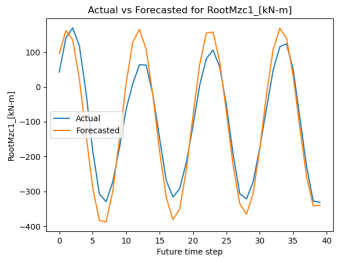
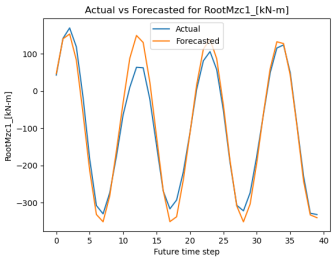


Figure C.3.: Cross-validation analysis of 'RootMxc1' in 'NoWave' case

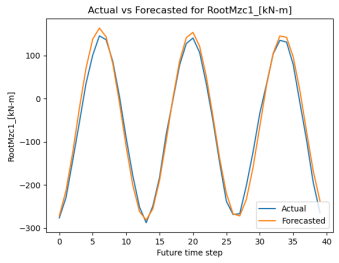
case, the maximum value of the RMSE for LSTM application is equal to 8.45 kN m^{-1}, the percentage error is 1.13%. Once more, the performance of the 'NoWave' case for this target feature is better compared to the one in the 'Wave' case.



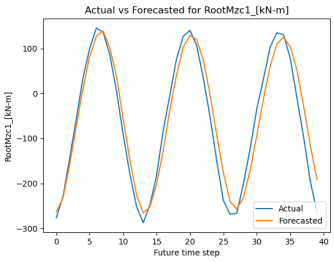
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 750 s



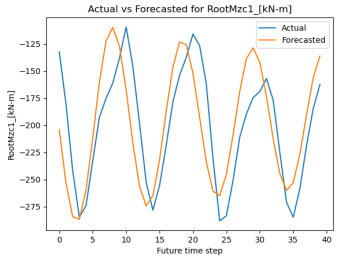
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 750 s



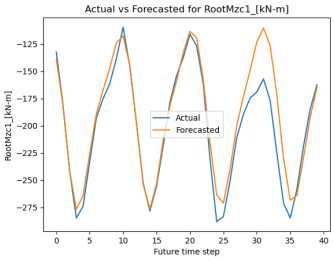
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 480 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 480 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 720 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 720 s

Figure C.4.: Examples of forecast results for 'RootMzc1' in 'NoWave' case

C.3. Blade out-of-plane moment

The blade out-of-plane moment shows characteristics similar to the blade flapwise moment. This is visible when the forecasts visualization in Figure C.7, the RMSE in Figure C.8 and the cross-validation reported in Figure C.9 are analyzed. They are all analogous to those of 'RootMyb1' shown in Chapter 6 but in this case the maximum RMSE value for LSTM implementation is equal to 1416 kN m^{-1} which corresponds to a percentage error of 1.5%. This is slightly lower than the one obtained in the 'Wave' case, which is not in line to what

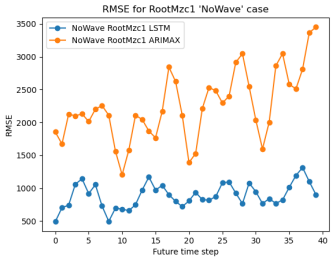


Figure C.5.: RMSE of 'RootMzc1' in 'NoWave' case

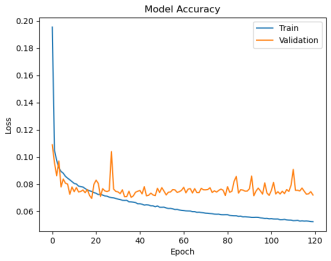
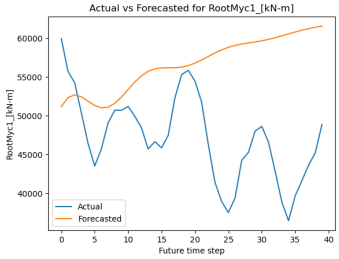
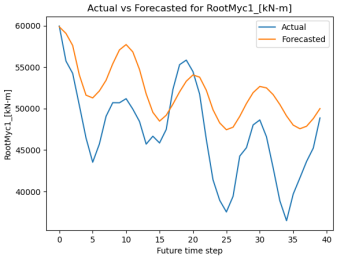


Figure C.6.: Cross-validation analysis of 'RootMzc1' in 'NoWave' case

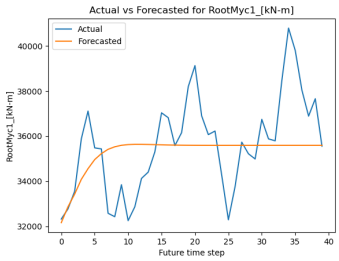
is highlighted for the flapwise moment. The divergence in this parameter can be due to the poor calibration of the hyper-parameters for these variables.



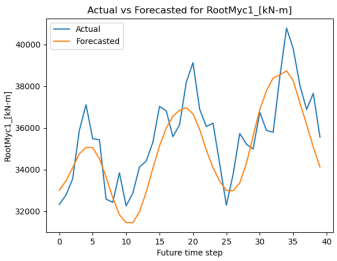
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 810 s



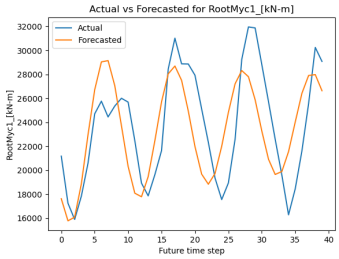
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 810 s



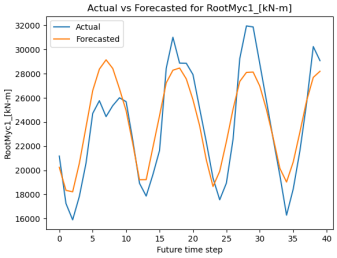
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 630 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 630 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 660 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 660 s

Figure C.7.: Examples of forecast results for 'RootMyc1' in 'NoWave' case

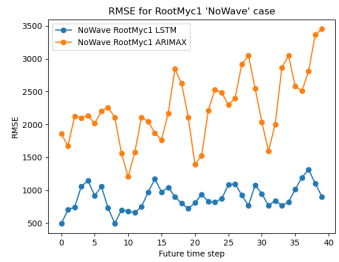


Figure C.8.: RMSE of 'RootMyc1' in 'NoWave' case

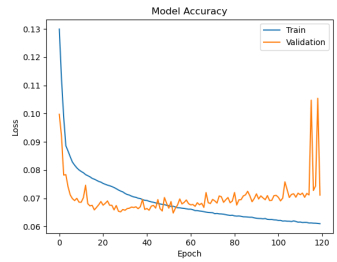


Figure C.9.: Cross-validation analysis of 'RootMyc1' in 'NoWave' case

Appendix D

Results 'PI' case forecasts

This last appendix reports the results obtained for the 'PI' case. Here all the target variables selected in Chapter 4, excluding the blade edgewise and flapwise moment are analyzed. The missing results are shown in Chapter 6. In this case, only the wind speed time series and the rotor speed are received by the SM as exogenous variables.

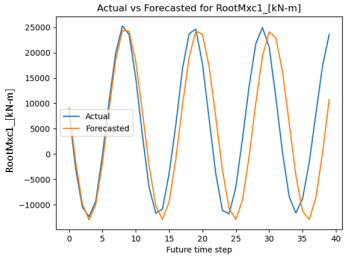
For each variable, three examples of forecast compared to test data that exemplify diverse operating conditions, the RMSE graphs and the cross-validation analysis performed on the LSTM model are reported. In particular, the results for the blade in-plane moment are shown in Section D.1, for the blade pitching moment in Section D.2, and for the blade out-of-plane moment in Section D.3.

D.1. Blade in-plane moment

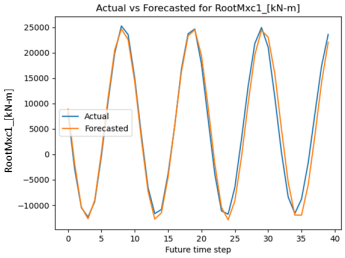
The blade in-plane moment is analogous to the blade edgewise moment. The visualization of the forecasts in Figure D.1, the RMSE in Figure D.2 and cross-validation analysis reported in Figure D.3 all confirm this statement. The maximum value of RMSE obtained for 'RootMxc1' when LSTM is applied is 1477 $\mathrm{kN\,m^{-1}}$, which corresponds to a percentage error of 2.7%. As for 'RootMxb1', this value is higher than the one obtained for both the 'Wave' and 'NoWave' cases, thus the trend is confirmed and the best performance for the blade in-plane moment is achieved in the 'NoWave' case.

D.2. Blade pitching moment

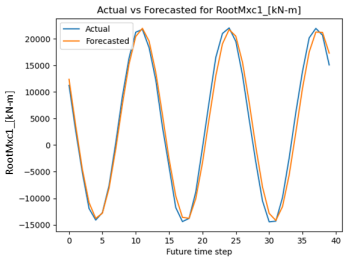
The blade pitching moment is also similar to the blade edgewise moment when the visualization of the forecasts in Figure D.4, the evaluation of the RMSE in Figure D.5 and the cross-validation analysis in Figure D.6 are analyzed. For 'RootMzc1', however, the scale is very different. Thus, even though the maximum RMSE for LSTM application is equal to 9.12 $\mathrm{kN\,m^{-1}}$, the percentage error is still 1.2%. This is higher than the value obtained for the 'NoWave' case, which is confirmed to be the one performing better, but it is lower than the 1.6% of the 'Wave' case.



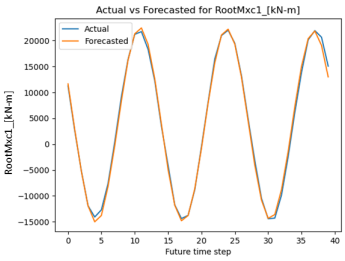
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 480 s



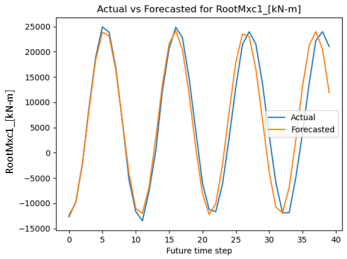
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 480 s



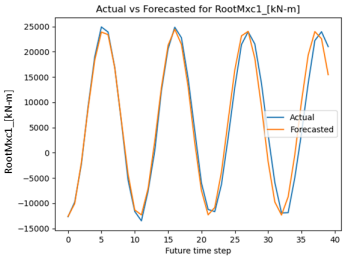
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 870 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 870 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 750 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 750 s

Figure D.1.: Examples of forecast results for 'RootMxc1' in 'Wave' case

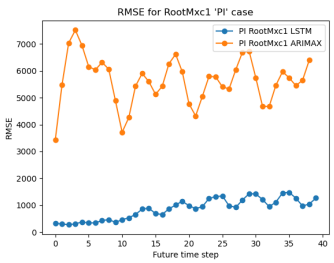


Figure D.2.: RMSE of 'RootMxc1' in 'PI' case

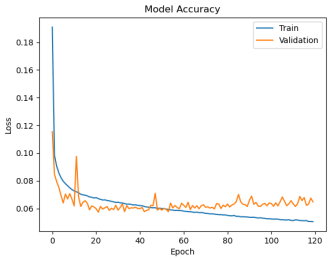
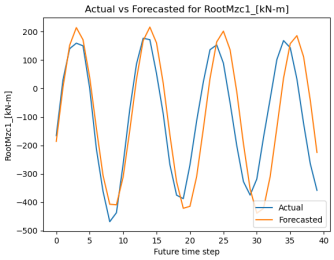
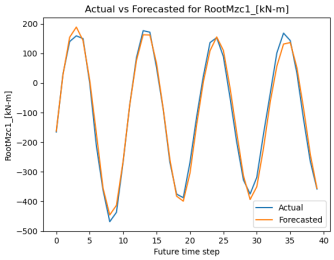


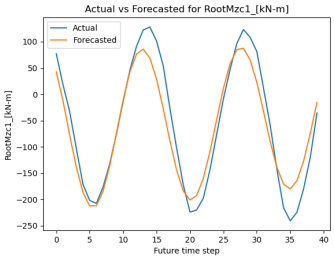
Figure D.3.: Cross-validation analysis of 'RootMxc1' in 'PI' case



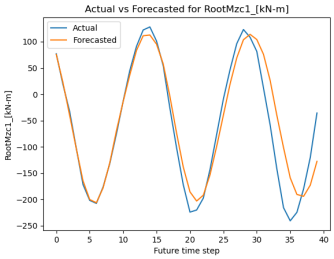
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 480 s



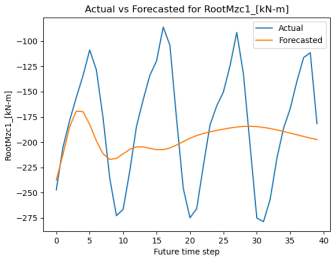
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 480 s



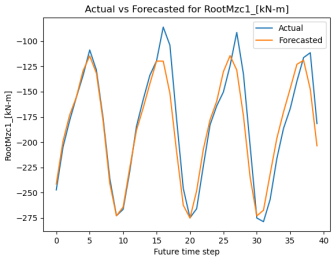
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 630 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 630 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 660 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 660 s

Figure D.4.: Examples of forecast results for 'RootMzc1' in 'PI' case

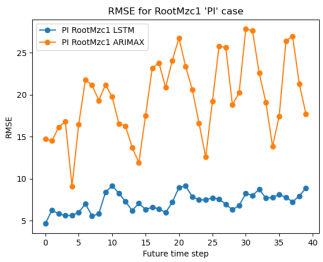


Figure D.5.: RMSE of 'RootMzc1' in 'PI' case

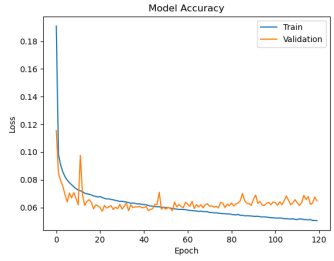
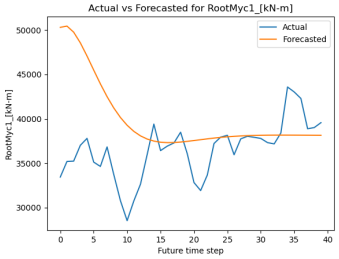


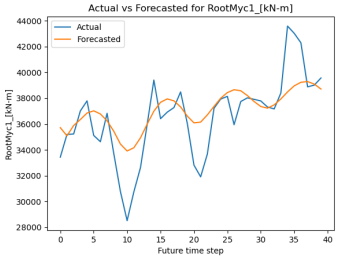
Figure D.6.: Cross-validation analysis of 'RootMzc1' in 'PI' case

D.3. Blade out-of-plane moment

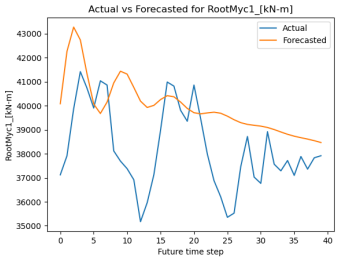
Finally, the blade out-of-plane moment has a behavior similar to the one of the blade flapwise moment. This is confirmed by the forecasts shown in Figure D.7, the RMSE in Figure D.8 and the cross-validation reported in Figure D.9. In particular, the error curve is almost flat, thus confirming the poor calibration of the hyper-parameters for this variable. The maximum obtained value for RMSE within LSTM results is equal to 1106 $\mathrm{kN\,m^{-1}}$ which corresponds to a percentage error of 1.2%. For this target variable, the 'PI' case is the one performing the best.



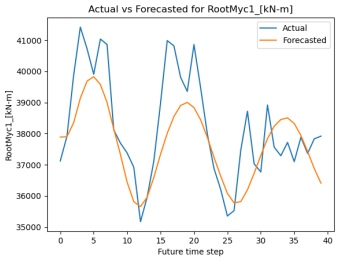
(a) ARIMAX $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 630 s



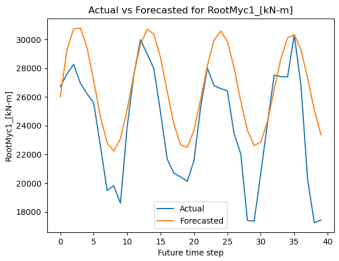
(b) LSTM $\boldsymbol{x}$ = [5.9, 3.1, -47.8, 20.4, 6.4] at t = 630 s



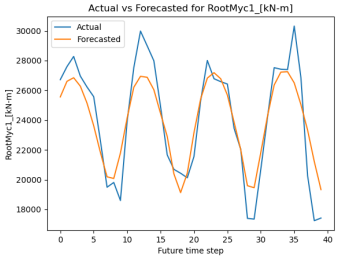
(c) ARIMAX $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 390 s



(d) LSTM $\boldsymbol{x}$ = [10.8, 0.89, -281.9, 15.2, 13.4] at t = 390 s



(e) ARIMAX $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 450 s



(f) LSTM $\boldsymbol{x}$ = [20.6, 5.2, -192.0, 12.3, 7.5] at t = 450 s

Figure D.7.: Examples of forecast results for 'RootMyc1' in 'PI' case

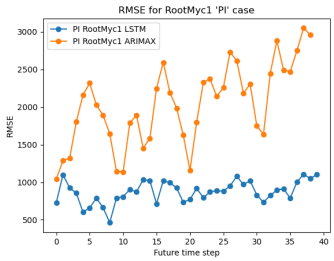


Figure D.8.: RMSE of 'RootMyc1' in 'PI' case

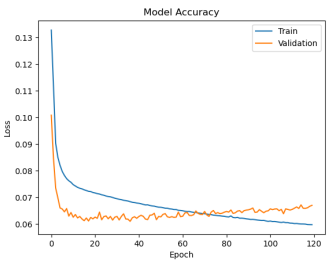


Figure D.9.: Cross-validation analysis of 'RootMyc1' in 'PI' case

Bibliography

[1] D. Hawila et al., *Renewable energy targets in 2022 Valuable review and feedback were provided by IRENA colleagues*. IRENA, 2022.

[2] R. P. Liem, *Surrogate modeling for large-scale black-box systems*. PhD thesis, Massachusetts Institute of Technology, 2007.

[3] M. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2005.

[4] B. Kumaraswamy, "Neural networks for data classification," in *Artificial Intelligence in Data Mining: Theories and Applications*, pp. 109–131, Elsevier, 1 2021.

[5] B. Robertson, J. Brown, T. Mcdonald, and C. Price, "Spatially Balanced Sampling using the Halton Sequence," *JSM*, 2019.

[6] E. Gaertner, J. Rinker, L. Sethuraman, F. Zahle, B. Anderson, G. Barter, N. Abbas, F. Meng, P. Bortolotti, W. Skrzypinski, G. Scott, R. Feil, H. Bredmose, K. Dykes, M. Shields, C. Allen, and A. Viselli, "Definition of the IEA Wind 15-Megawatt Offshore Reference Wind Turbine Technical Report," tech. rep., NREL, 2020.

[7] J. Sørensen, "Wind turbine wakes and wind farm aerodynamics," in *Wind Energy Systems*, pp. 112–e131, Elsevier, 2011.

[8] K. Shaler, B. Anderson, L. A. Martínez-Tossas, E. Branlard, and N. Johnson, "Comparison of Free Vortex Wake and BEM Structural Results Against Large Eddy Simulations Results for Highly Flexible Turbines Under Challenging Inflow Conditions," *Wind Energy Science Discussions*, 2022.

[9] H. Zhang, S. Li, Y. Chen, J. Dai, and Y. Yi, "A Novel Encoder-Decoder Model for Multivariate Time Series Forecasting," *Computational Intelligence and Neuroscience*, vol. 2022, 2022.

[10] C. Zaiontz, "Augmented Dickey-Fuller Test ," 2018.

[11] "Goal 7: Affordable and clean energy - The Global Goals."

[12] J. Tollefson, "What the war in Ukraine means for energy, climate and food," *Nature*, vol. 604, pp. 232–233, 4 2022.

[13] "The Paris Agreement — UNFCCC."

[14] "GE Haliade-X 14.7 MW offshore wind turbine is officially the world's most powerful."

[15] "Wind farm costs – Guide to an offshore wind farm."

[16] "Map of the Week – Locations of wind farms — European Marine Observation and Data Network (EMODnet)."

[17] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, "Review on model predictive control: an engineering perspective," 11 2021.

[18] G. Roig, "Application of Model Predictive Control to Wind Turbines," *ETH Library*, 2017.

[19] K. Bieker, S. Peitz, S. L. Brunton, J. N. Kutz, and M. Dellnitz, "Deep Model Predictive Control with Online Learning for Complex Physical Systems," *Cornell University*, 5 2019.

[20] C. Li, J. Hu, J. Yu, J. Xue, R. Yang, Y. Fu, and B. Sun, "A Review on the Application of the MPC Technology in Wind Power Control of Wind Farms," *Journal of Energy and Power Technology*, vol. 03, pp. 1–1, 4 2021.

[21] N. Hure, "Model Predictive Control of a Wind Turbine," tech. rep., University of Zagreb, 2013.

[22] M. Ali, A. M. Khattak, Z. Ali, B. Hayat, M. Idrees, Z. Pervez, K. Rizwan, T.-E. Sung, and K.-I. Kim, "Estimation and Interpretation of Machine Learning Models with Customized Surrogate Model," *Electronics*, vol. 10, p. 3045, 12 2021.

[23] A. Jain, "Methods For Data-Driven Model Predictive Control," tech. rep., Univeristy of Pennsylvania, 2020.

[24] Cosack N, "Fatigue Load Monitoring with Standard Wind Turbine Signals," tech. rep., Universitat Stuttgart, 2010.

[25] F. de Nolasco Santos, W. Weijtjens, C. Devriendt, and N. Noppe, "Two-tier model for wind turbine fatigue assessment based on SCADA-dependent neural networks," tech. rep., Vrije Universiteit Brussel, 2021.

[26] N. Dimitrov and T. Göçmen, "Virtual sensors for wind turbines with machine learning-based time series models," *Wind Energy*, vol. 25, pp. 1626–1645, 9 2022.

[27] K. Müller and P. Wen Cheng, "A Surrogate Modelling Approach for Fatigue Damage Assessment of Floating Wind Turbines," in *Proceedings of the ASME 2018 37th International*, 2018.

[28] J. Lee, "Fatigue Assessment of Offshore Wind Turbines Using Measurements of Individual Turbines and Machine Learning Techniques," tech. rep., Delft University of Technology, 2019.

[29] T. Obdam, L. Rademakers, and H. Braam, "Flight leader concept for wind farm load counting: Offshore evaluation," *Wind Engineering*, vol. 34, pp. 109–122, 1 2010.

[30] L. Schröder, N. K. Dimitrov, D. R. Verelst, and J. A. Sørensen, "Wind turbine site-specific load estimation using artificial neural networks calibrated by means of high-fidelity load simulations," *Torque*, p. 62027, 2018.

[31] L. Schröder, N. K. Dimitrov, and J. A. Sørensen, "Uncertainty propagation and sensitivity analysis of an artificial neural network used as wind turbine load surrogate model," *TORQUE 2020, Journal of Physics: Conference Series*, vol. 1618, p. 42040, 2020.

[32] U. Smolka, D. Kaufer, and P. W. Cheng, "Are Sea State Measurements Required for Fatigue Load Monitoring of Offshore Wind Turbines?," *Journal of Physics: Conference Series OPEN ACCESS*, 2014.

[33] M. Souliotis, "Machine learning techniques for load monitoring of offshore wind turbines," tech. rep., Delft University of Technology, 2013.

[34] A. Venu, M. Lüdde, and J. Omole, "Prediction and Validation of Fatigue loads using Artificial Intelligence on Real World Measurement Data ," *TORQUE 2020, Journal of Physics: Conference Series*, vol. 1618, p. 22006, 2020.

[35] N. Dimitrov, M. C. Kelly, A. Vignaroli, and J. Berg, "From wind to loads: wind turbine site-specific load estimation with surrogate models trained on high-fidelity load databases," *Wind Energy Science*, vol. 3, pp. 767–790, 10 2018.

[36] J. P. Murcia, P.-E. Réthoré, N. Dimitrov, A. Natarajan, J. D. Sørensen, P. Graf, and T. Kim, "Uncertainty propagation through an aeroelastic wind turbine model using polynomial surrogates," *Renewable Energy*, vol. 119, pp. 910–922, 4 2018.

[37] R. M. M. Slot, J. D. Sørensen, B. Sudret, L. Svenningsen, and M. L. Thøgersen, "Surrogate model uncertainty in wind turbine reliability assessment," *Renewable Energy*, 2020.

[38] S. Okpokparoro and S. Sriramula, "Uncertainty modeling in reliability analysis of floating wind turbine support structures," *Renewable Energy*, vol. 165, pp. 88–108, 3 2021.

[39] A. Caterina Clark and C. E. Clark, "Employing Bayesian Quadrature to Improve Fitting of Surrogate Models to Wind Turbine Loads," *TORQUE 2022, Journal of Physics: Conference Series*, vol. 2265, p. 42045, 2022.

[40] D. Singh, R. P. Dwight, K. Laugesen, L. Beaudet, and A. Viré, "Probabilistic surrogate modeling of offshore wind-turbine loads with chained Gaussian processes," in *Journal of Physics: Conference Series*, vol. 2265, Institute of Physics, 6 2022.

[41] B. Stoevesandt and J. Peinke, "Effects of Sudden Changes in Inflow Conditions on the Angle of Attack on HAWT Blades," tech. rep., University of Oldenburg, 11 2010.

[42] H. Aalborg Nielsen and H. Madsen, "Wind Power Prediction Using ARX Models and Neural Networks," tech. rep., Technical University of Denmark, 1996.

[43] M. J. Durán, D. Cros, and J. Riquelme, "Short-Term Wind Power Forecast Based on ARX Models," *Journal of Energy Engineering*, vol. 133, pp. 172–180, 9 2007.

[44] D. Kumar, H. D. Mathur, S. Bhanot, and R. C. Bansal, "Forecasting of solar and wind power using LSTM RNN for load frequency control in isolated microgrid," *International Journal of Modelling and Simulation*, vol. 41, no. 4, pp. 311–323, 2021.

[45] S. Muzaffar and A. Afshari, "Short-term load forecasts using LSTM networks," in *Energy Procedia*, vol. 158, pp. 2922–2927, Elsevier Ltd, 2019.

[46] "ARX Time Series Model," 2021.

[47] R. H. Shumway and D. S. Stoffer, *Time Series Analysis and Its Applications*. Springer, 2011.

[48] H. Bozdogan, "The General Theory and Its Analytical Extensions," Tech. Rep. 3, University of Virginia, 1987.

[49] M. Jordan, J. Kleinberg, and B. Schölkopf, *Pattern Recognition and Machine Learning*. Springer, 2006.

[50] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. 2015.

[51] A. Kumar, "Demystifying Encoder Decoder Architecture & Neural Network," 5 2023.

[52] J. Brownlee, "Multi-Step LSTM Time Series Forecasting Models for Power Usage," 10 2018.

[53] K. Cao, K. Shaler, and N. Johnson, "Comparing wind turbine aeroelastic response predictions for turbines with increasingly flexible blades ," *TORQUE 2022, Journal of Physics: Conference Series*, vol. 2265, p. 32025, 2022.

[54] "Waveclimate.com: The on-line offshore climate assessment system."

[55] International Standard, "Wind turbines: Design requirements," tech. rep., IEC, 2005.

[56] H. Niederreiter, "Random Number Generation and Quasi Monte Carlo Methods 92," *SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS PHILADELPHIA*, pp. 161–176, 1992.

[57] R. Gunesh, "Methods of sampling: Random, Quasi-random, Non-random, Simple random, Systematic, Quota, Stratified, Cluster,"

[58] Chaospy, "Quasi-random samples ."

[59] L. Kocis and W. J. Whiten, "Computational Investigations of Low-Discrepancy Sequences," *ACM Transactions on Mathematical Software*, 1997.

[60] E. Osmanbasic, "The Future of Wind Turbines: Comparing Direct Drive and Gearbox ," 2020.

[61] R. Damiani and G. Hayman, "The Unsteady Aerodynamics Module for FAST 8," *NREL*, 2013.

[62] M. Zaaijer, A. Viré, R. B. Dos, S. Pereira, and A. Daneshbodi, "Introduction to wind turbines: physics and technology," tech. rep., Delft University of Technology, 2021.

[63] "4.2.1.2. Input Files — OpenFAST v3.4.1 documentation."

[64] H. Söker, "Loads on wind turbine blades," *Advances in Wind Turbine Blade Design and Materials*, pp. 29–58, 2013.

[65] G. Wu, C. Zhang, C. Cai, K. Yang, and K. Shi, "Uncertainty prediction on the angle of attack of wind turbine blades based on the field measurements," *Energy*, vol. 200, 6 2020.

[66] T. R. Derrick and J. M. Thomas, "Time Series Analysis: The Cross-Correlation Function," in *Innovative Analyses of Human Movement*, ch. 7, Human Kinetics Publishers, 2004.

[67] S. Prabhakaran, "Augmented Dickey-Fuller (ADF) Test ," 11 2019.

[68] C. Zaiontz, "Dickey-Fuller Test," 2018.

[69] D. Mindrila and P. Balentyne, "Scatterplots and Correlation," tech. rep., University of West Georgia, 2017.

[70] R. Zhang, H. Song, Q. Chen, Y. Wang, S. Wang, and Y. Li, "Comparison of ARIMA and LSTM for prediction of hemorrhagic fever at different time scales in China," *PLoS ONE*, vol. 17, 1 2022.

[71] M. Kuhn and K. Johnson, *Applied Predictive Modeling*. Springer, 2013.

[72] R. Pramoditha, "Classification of Neural Network Hyperparameters ."

[73] J. Brownlee, "Understand the Impact of Learning Rate on Neural Network Performance ."

[74] R. Pramoditha, "How to Choose the Optimal Learning Rate for Neural Networks ."

[75] J. Brownlee, "Gentle Introduction to the Adam Optimization Algorithm for Deep Learning ," 1 2021.

[76] J. Brownlee, "Code Adam Optimization Algorithm From Scratch," 10 2021.

[77] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-Validation," *Encyclopedia of Database Systems*, pp. 532–538, 2009.

[78] J. Brownlee, "How to Diagnose Overfitting and Underfitting of LSTM Models ," 9 2017.

[79] F. Mello, "Neural Networks - Dealing with LSTM overfitting - Cross Validated," 6 2020.

Colophon

This document was typeset using LaTeX, using the KOMA-Script class `scrbook`. The main font is Palatino.