Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Marine Technology

Haakon Lyngstad
Jørgen Engelsen

Hybrid modelling for mooring system monitoring - A digital twin framework for floating wind turbines

Master's thesis in Marine Technology
Supervisor: Dong Trong Nguyen
Co-supervisor: Dag-Børre Lillestøl
June 2022

NTNU
Kunnskap for en bedre verden

Haakon Lyngstad
Jørgen Engelsen

Hybrid modelling for mooring system monitoring - A digital twin framework for floating wind turbines

Master's thesis in Marine Technology
Supervisor: Dong Trong Nguyen
Co-supervisor: Dag-Børre Lillestøl
June 2022

Norwegian University of Science and Technology
Faculty of Engineering
Department of Marine Technology

NTNU
Norwegian University of
Science and Technology

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

MSC THESIS DESCRIPTION SHEET

Name of the candidate: Haakon Lyngstad, Jørgen Engelsen

Field of study: Marine control engineering

Thesis title (Norwegian): ---

Thesis title (English): Hybrid modelling for mooring system monitoring - A digital twin framework for floating wind turbines

Background

Condition monitoring of mooring systems is a key element in many emerging floating wind solutions. Floating wind turbines may be installed in deeper water when compared to bottom-fixed solutions. This opens more locations for installations, allowing for placement in more wind-dense areas. This also often leads to more remote and isolated locations making inspections and maintenance more demanding. Floating wind farms could potentially have hundreds of floating units, all with a minimum of three mooring lines. There are no current offshore installations with an equal number of mooring lines, and the existing solutions for mooring monitoring and quality assurance might not be sufficient to make the cost of floating wind competitive. Manual inspection of all lines is time-consuming and expensive, and upgrading the mooring monitoring sensor equipment comes with an increased cost. Tension measurements are historically expensive and have issues with range and accuracy. For this thesis, we will investigate the potential degree of accuracy for which it is possible to estimate mooring line tension using a hybrid model utilizing physical simulation and other cheaper sensor packages such as GPS, IMU and Metocean data. With an alternative adequate method for tension estimates, offshore wind farms could significantly reduce the necessary expenses for tension measurement equipment while maintaining their advantage for condition monitoring.

Work description

1. Perform a background and literature review to provide information and relevant references on:
    - Mooring system and relevant failure modes for floating wind
    - GPS/IMU data quality and accuracy issues
    - Tension monitoring systems
    - Catenary mooring line analysis
    - Appropriate machine learning methods
    - Methods for hybrid modelling
    - Predictive maintenance with digital twins
    - Estimation of accumulated fatigue damage on mooring lines

    Write a list with abbreviations and definitions of terms, explaining relevant concepts related to the literature study and project assignment.
2. High level discussion on the complications of floating wind farm mooring monitoring.
3. Establish case studies on floating wind turbines
4. Apply hybrid modelling to bridge the gap between theoretical models and operational data to find an estimate of top tension
5. Verify the proposed methods with historical data.

Specifications

The scope of work may prove to be larger than initially anticipated. By the approval from the supervisor, described topics may be deleted or reduced in extent without consequences with regard to grading.

The candidate shall present personal contribution to the resolution of problems within the scope of work. Theories and conclusions should be based on mathematical derivations and logic reasoning identifying the various steps in the deduction.

The report shall be organized in a logical structure to give a clear exposition of background, results, assessments, and conclusions. The text should be brief and to the point, with a clear language. Rigorous mathematical deductions and illustrating figures are preferred over lengthy textual descriptions. The report shall have font size 11 pts., and it is not expected to be longer than 60-80 A4 pages, from introduction to conclusion, unless otherwise agreed upon. It shall be written in English (preferably US) and contain the following elements: Title page, abstract, acknowledgements, thesis specification, list of symbols and acronyms, table of contents, introduction with objective, background, and scope and delimitations, main body with problem formulations, derivations/developments and results, conclusions with recommendations for further work, references, and optional appendices. All figures, tables, and equations shall be numerated. The original contribution of the candidate and material taken from other sources shall be clearly identified. Work from other sources shall be properly acknowledged using quotations and a Harvard citation style (e.g. *natbib* Latex package). The work is expected to be conducted in an honest and ethical manner, without any sort of plagiarism and misconduct. Such practice is taken very seriously by the university and will have consequences. NTNU can use the results freely in research and teaching by proper referencing, unless otherwise agreed upon.

The thesis shall be submitted with a printed and electronic copy to the main supervisor, with the printed copy signed by the candidate. The final revised version of this thesis description must be included. The report must be submitted according to NTNU procedures. Computer code, pictures, videos, data series, and a PDF version of the report shall be included electronically with all submitted versions.

Start date: January 15th 2022 **Due date:** As specified by the administration.

Supervisor: Dong Trong Nguyen
Co-advisor(s): Dag Børre Lillestøl

Trondheim, _____

_____
Dong Trong Nguyen
Supervisor

Abstract

Floating wind has the potential to become a crucial contributor to the world's need for renewable energy. For the technology to become competitive compared to other energy sources, the cost of installation and maintenance must be reduced. This thesis proposes a digital twin framework for monitoring the mooring system of floating wind turbines. Specifically, it attempts to replace direct tension measurements with a virtual sensor prediction model to monitor accumulated fatigue damage of the mooring lines. In a large wind farm, the cost reductions of transitioning from preventative to predictive maintenance of the mooring systems could be significant.

Using a hybrid modelling approach, this thesis aims to utilise a physics-based model to create a mapping from turbine position to mooring line top tension in "as-designed" condition with a basis in physical understanding of the system. A data-driven model is used to map the difference from "as-designed" to "as-is" condition of the mooring system behaviour. The two models are combined into a hybrid model to create a reliable and accurate prediction of top tension in mooring systems.

For the physics-based model, the mooring line tensions are estimated for each line, at each time step, using quasi-static analysis. The data-driven models are based on three machine learning algorithms: Random Forest Regressors, Artificial Neural Networks and Long-Short Term Memory networks. These algorithms are compared for the more traditional data-driven approach, and for the hybrid approach proposed in this thesis.

The digital twin framework is applied to a case study on the world's first floating wind farm, Hywind Scotland. The proposed hybrid model achieves an average 50% improvement in RMSE from the physical-based models and an 11% improvement compared to the data-driven model.

The results indicate that hybrid modelling may have merits within mooring monitoring and remaining useful life forecasting for floating wind. A case study with an increased amount of data is needed to test the method on a broader range of conditions and evaluate generalising performance and the scalability of the digital twin framework. Further work is proposed to improve upon the hybrid framework presented.

Sammendrag

Flytende havvind har potensiale til å være en viktig bidragsyter mot verdens behov for fornybar energi. Vedlikeholds og installasjons kostnader må reduseres for at teknologien skal bli et konkurransedyktig alternativ til andre fornybare energi kilder. I denne masteroppgaven blir et rammeverk for digitale tvillinger foreslått for tilstandsovervåking av flytende vindmøllers forankrings liner. Kostnadsbesparelsene ved å bytte ut spenningsmålere med predikert spenning fra en virtuell sensor modell kan bli betydelige i en stor flytende vindpark.

Oppgaven kombinerer en fysisk modell av forankringssystemet med en datadrevet modell i et forsøk på å skape en robust og presis hybrid modell for prediksjon av topp-spenning i forankringslinene. Den fysiske modellen blir konstruert basert på designkriteriene til Hywind Scotland, mens den datadrevne modellen forsøker å lære hva som er forskjellen mellom den designede og faktiske topp-spenningen.

Den fysiske modellen bruker en kvasi-statisk tilnærming til å estimere topp-spenningen for hver line i ethvert punkt som vindmøllene er i. De datadrevne modellene er basert på tre forskjellige maskinlæringsalgoritmer: Random Forest Regression, Artificial Neural Networks og Long-Short Term Memory networks. Disse algoritmene presenteres for den mer tradisjonelle datadrevne tilnærmingen, og sammenlignes med den hybride modellen som er forslått i denne avhandlingen.

Det foreslåtte digital tvilling rammeverket ble testet på en kasusstudie med operasjonell data fra verdens første flytende vindpark, Hywind Scotland. Hybrid modellen oppnådde en 50% forbedring sammenlignet med den fysiske modellen alene, og en 11% forbedring sammenlignet med den datadrevne modellen.

Resultatene antyder at hybrid modellering har potensiale innenfor tilstandsovervåking av flytende vind og prediksjon av gjennværende levetid. Men en større studie med mer data og flere vind turbiner er nødvendig for å kvantifisere rammeverkets skalerbarhet og nytte.

Acknowledgments

This thesis is the culmination of the five years of studies at NTNU, and for us marks the end of countless projects performed together. We are both grateful for the companionship it has brought to write this thesis together

First of all, we would like to thank our supervisor, Professor Dong Trong Nguyen at the department of Marine Technology, NTNU for his support and guidance. Special thanks to our co-supervisor Dag-Børre Lillestøl at DNV for providing invaluable domain knowledge on position mooring systems from the industry. We would be remiss in not mentioning Ore Catapult for providing operational data from Hywind Scotland.

Lastly, we would like to thank our fellow classmates who are sharing offices with us, for enduring our endless discussions, and for all the feedback and support.

Trondheim
June 2, 2022

Haakon Lyngstad
Department of Marine Technology

Jørgen Engelsen
Department of Marine Technology

Table of Contents

List of Tables

List of Figures

Abbreviations

AI	=	Artificial Intelligence
ANN	=	Artificial Neural Net
CV	=	Cross Validation
DDM	=	Data driven modelling
dGPS	=	differential Global Positioning System
DoE	=	Design of Experiments
DOF	=	Degrees of freedom
DT	=	Digital Twin
FDV	=	Force-Displacement-Velocity
FEA	=	Finite Element Analysis
FEM	=	Finite Element Method
FLS	=	Fatigue Limiting State
FOWT	=	Floating Offshore Wind Turbine
FPSO	=	Floating Production, Storage and Offloading
FWT	=	Floating Wind Turbines
GNSS	=	Global Navigation Satellite Systems
GPS	=	Global Positioning System
IAI	=	Industrial Artificial Intelligence
IMU	=	Inertial Measurement Unit
IoT	=	Internet of Things
LCOE	=	Levelised cost of energy
LSTM	=	Long Short-Term Memory
LxBy	=	Line x, Bridle y
MAE	=	Mean Absolute Error
Metocean	=	Meteorology and oceanography
MIMO	=	Multiple Input Multiple Output
ML	=	Machine Learning
MLP	=	Multi Layer Perceptron
MRU	=	Motion Reference Unit
NN	=	Neural Net
OOS	=	Out-Of-Sample
PBM	=	Physics-based modelling
PBP	=	Physics-based Pre-proccesing
PRSL	=	Position Response Learning System
RMSE	=	Root Mean Square Error
RMSP	=	Root Mean Square Propagation
RNN	=	Recurrent Neural Network
RUL	=	Remaining Useful Life
TLP	=	Tension Leg Platform

Chapter 1

Introduction

Floating offshore wind is a rapidly growing industry. Maintenance and inspection of mooring systems at large floating wind farms is labor intensive and costly. By transitioning from preventative to predictive maintenance, costs and downtime may be reduced due to optimal intervals between repairs and replacements (Daily and Peterson 2017, pp. 268–269). Ideally, accurate and reliable tension measurements should be affordable and available. Line failures would instantly be detected, and fatigue from cyclic loading could be quantified. Instead, however, the technology currently in use for permanently moored platforms is expensive and unreliable (Hageman et al. 2019). One potential cost reduction comes from virtual sensors, replacing sensor technology with a digital twin framework that predicts tension measurements from other available data sources. Virtual tension sensors have the potential to cut both capital and operational expenditures for floating offshore wind and, by doing so, partake in making floating offshore wind more economically competitive as a renewable energy production technology.

This thesis will expand upon previous work within in-direct tension sensor technology for floating wind turbine farms by applying a hybrid approach combining a simulated physical model and a machine learning model. This methodology attempts to leverage the trustworthiness and interpretability of a physics-based model while achieving the accuracy of data driven models, resulting in a more complete approach for consistent in-direct tension measurement.

Figure 1.1 shows a wind farm setup where most of the floating turbines are only equipped with cheaper sensors for tilt and displacement, while a small subset of the turbines is equipped with direct tension measurement sensors. The direct tension measurement sensors are used to train and calibrate the hybrid method, passing trained models and data onto the other turbines so that predictive maintenance can still be performed without needing costly tension sensors.

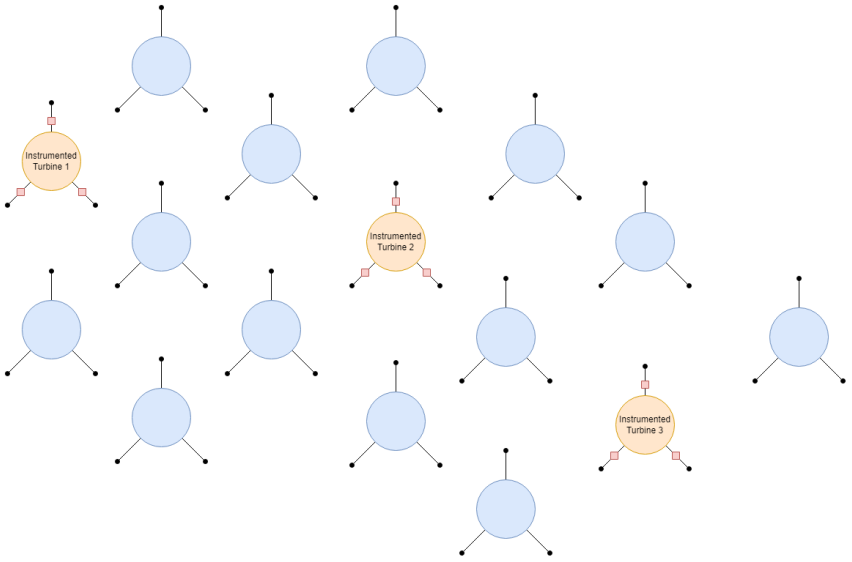


Figure 1.1: Wind farm with sparse tension measurements.

1.1 Background

According to IPCC's latest assessment report, (Clarke et al. 2022) global warming cannot be limited to less than 2°C without rapid reductions in emissions and transition to renewable energy sources. In the emerging taxonomy of green solutions to the world's energy shortage, power generated from wind is expected to have a significant market share. In the period from 2015 to 2019, the global wind energy capacity grew by 70% (Clarke et al. 2022).

Estimates by the International Energy Association (IEA) forecast that by 2030, 470 GW of wind production capacity will be added annually to the global production, and 80 GW will be offshore (IEA 2021). In comparison, only 5 GW of offshore wind capacity was added in 2020. As a result of this growth, IEA expects the cost of operating and maintaining offshore wind production to soar to 90 billion USD. In such a scenario, there will be a significant demand for technologies that enable cost savings, such as reduced manual inspection or reduced need for sensors and technologies that minimize the likelihood of damaged equipment or personnel.

For such a scenario to unfold, the cost of floating wind must become competitive with alternative renewable energy sources. Levelised Cost Of Energy (LCOE) is a metric used to compare the cost competitiveness of power production technologies. The metric is the sum of production costs over the designed lifetime, divided by the energy produced in the same period. When comparing the LCOE for Floating Offshore Wind Turbines (FOWT), Bottom-fixed offshore wind turbines, and onshore wind turbines in Europe, even the lowest current LCOE for FOWTs is significantly higher than the average for the other two technologies. The LCOE of onshore wind turbines is 59 €/MWh (Trinomics 2020). In contrast, the comparable number for FOWTs is estimated to be 95 €/MWh in the most profitable European areas (Martínez and Iglesias 2022), making the cost of offshore wind over 60% higher than its shore-based alternative.

The high cost of FOWTs is attributable to several factors, the biggest being the added complexity of floating structures in offshore regions compared to bottom-fixed or onshore. In addition, offshore wind turbines require more robust infrastructure than their onshore counterparts, such as subsea cables and floating substations which increase the total cost of FOWTs . The substructure and the mooring system are among the most significant contributors to the increased costs of floating wind (Martínez and Iglesias 2022). In addition to the above factors, the cost of offshore operations such as installation, maintenance, and decommissioning adds to the total life cycle cost (Martínez and Iglesias 2022).

Although the LCOE of floating wind is expected to decrease by the synergies that follow the economics of scale as the sector grows, innovations and new technologies are also vital to making FOWTs economically competitive (M. Learch and Berthelsen 2019).

1.2 Literature Review

1.2.1 Floating Wind

One of the biggest challenges when moving energy production offshore is the harsh environmental conditions. For floating wind, severe weather is also one of the biggest advantages. Although the offshore conditions cause a significant increase in both capital expenditures and operational costs, they also come with the added benefit of a 90% increase in wind speeds at 80 meters (Archer and Jacobson 2005). Increased wind speed means that the energy production potential is considerable, providing a promising upside given cost-efficient solutions for floating wind compared to onshore.

To date, only three floating wind farms have been commissioned. Current farms have only a hand full of turbines, but larger projects are planned. Hywind Tampen by Equinor is planned to be commissioned in Q3 2022 and will have a nearly doubled capacity (95 MW), and over double the number of wind turbines compared to the current largest floating wind farm (Equinor 2022b). Norway's government recently announced an ambition to build 1500 offshore wind turbines with a total 30 GW capacity by 2040 (Olje- og energidepartementet 2022). Other countries have also announced similar commitments to building new capacity for floating wind. The number of wind farms and their size are expected to proliferate in the coming decades.

Wind farm	Location	Commissioned	Capacity [MW]
Hywind Scotland	Scotland	2017	30
WindFloat Atlantic	Portugal	2020	25
Kincardine	Scotland	2021	50

Table 1.1: Commisioned floating wind farms to date (Cobra Group 2021; Energias de Portugal 2020; Equinor 2022a)

The forecasted growth in offshore wind production is based on several key characteristics of the energy source. Firstly, the potential scalability for offshore wind is far beyond the current energy demand. There are numerous different methodologies and estimates for the global potential. However, most are in the

range of 71 - 120 terawatt (TW) of capacity potential and 300 000 - 420 000 TWh/year of generation potential, around three times the annual global energy consumption (Bosch et al. 2018; IEA 2019; The World Bank 2019). These estimates are based on data for, among other things, wind capacity, the efficiency of wind turbines, the global water depth, spatial constraints such as distance to shore, and current offshore activities. When looking at the share of floating to bottom fixed potential within the global potential, 20% are suitable for bottom-fixed systems (0-40 m) while the rest need floating structures (Bosch et al. 2018).

Figure 1.2 presents the most common categories of floating wind turbine concepts developed. The leftmost design, the spar, is the same type of floater that has been used in the Hywind Scotland wind park, the world's first commercial floating wind farm (Equinor 2022a). The figure also illustrates a key component of floating wind turbine designs; mooring configurations.

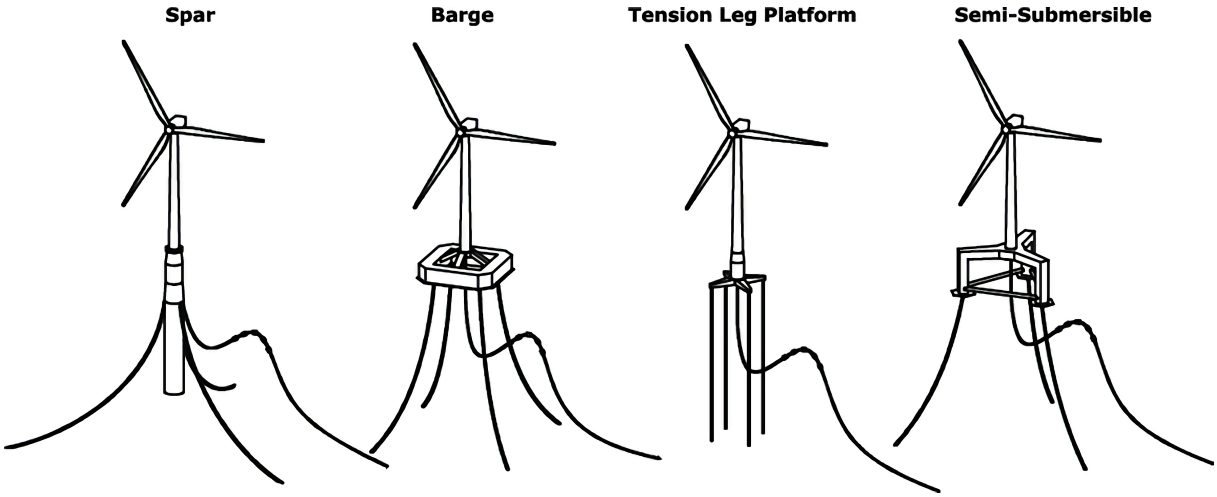


Figure 1.2: Wind turbine designs. Courtsey: Scheu et al. (2018)

1.2.2 Floater designs and mooring configurations

For FOWTs, there exists four main types of floaters designs, shown in Figure 1.2: Spar buoy, Barge, Tension Leg Platform (TLP), and Semi-Submersible. These designs are based on tried and tested solutions from the oil and gas sector, leveraging decades of offshore experience. The spar buoy, currently in use at two offshore locations, uses a low center of gravity combined with taut mooring lines or catenary mooring lines for station keeping. Compared to the other designs, the floater has a significant draft in the magnitude of 80-100 meters. The large draft limits the applicability in shallower waters and causes complications during the assembly and transportation of substructures. The TLP has other advantages, primarily the low mass needed for the floater compared to the other designs. It does, however, come with added strain to the mooring lines and anchor. Finally, the semi-submersible design is easy to install and transport but more complex on the design side. The semi-submersible utilizes active buoyancy distribution with water weight used as a stabilizing moment. Semi-submersibles require a more complex design process and often come with a more expensive and time-consuming production. (*Floating Offshore Wind*

Farms 2016, p. 6-8)

1.2.3 Mooring system components

Mooring systems may contain different components depending on the desired properties and behavior of the system. As illustrated in Figure 1.3, mooring systems contain one or more anchors, mooring lines, connection links, and points to the vessel. Mooring configurations may also include buoys or clump weights to alter the mooring system's buoyancy and weight, respectively. Buoys can be used to improve the performance of the mooring by reducing the vessel's offset and reducing the weight applied to the vessel from the mooring lines. Clump weights are used to increase the restoring forces of the mooring system (Chakrabarti 2005).

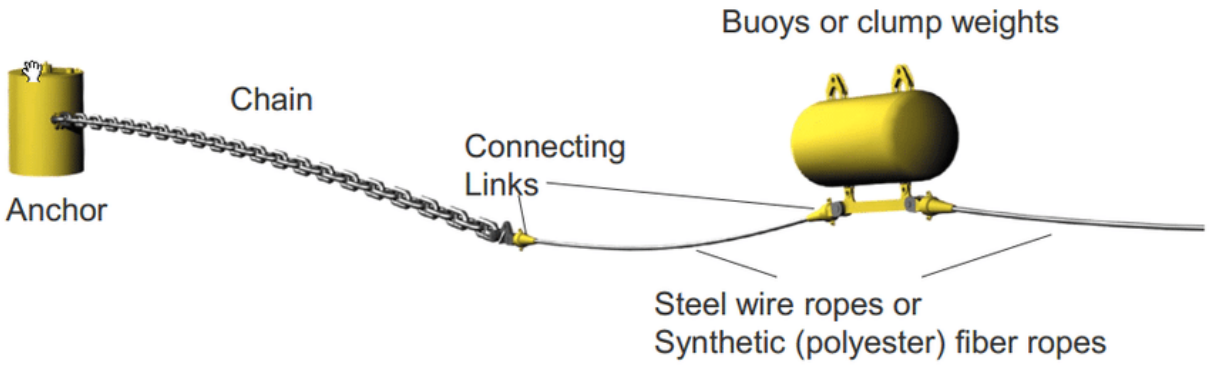


Figure 1.3: The main mooring line components. Courtsey: Hasan (2015)

The materials used for mooring lines are essential for how the mooring system will behave and change over time. Depending on the differing use-cases, the most commonly used materials are chain, steel wire rope, and fiber rope (Kai-Tung et al. 2019, p. 71-73).

Chains are used in mooring structures due to their sturdiness, resistance against seabed abrasion, and high breaking strength. Especially for structures moored in shallow waters, all-chain solutions have been frequently used in the oil and gas industry, with similar designs being introduced for offshore wind. In deeper water, the weight of chains causes the material to be a more significant challenge than an advantage, causing the catenary shape to become sub-optimal. The size and shape of chains vary, from bar diameters down to 70 mm and up to 200 mm, implying link lengths between 0.42 and 1.2 meters. Another important distinction is between studless and studlink chains, the two subcategories of chain. The studlink chains have a stud in between the arch of the chain, comparable to an extra beam, which improves the handling when deploying or retracting the lines. On the other hand, Studless are lighter and cheaper to produce due to their simplicity but are primarily helpful for permanent mooring systems. Kai-Tung et al. (2019)

The primary purpose of anchors is to fix the mooring lines to the seabed, primarily by restraining horizontal movement. Traditional drag anchors are therefore not generally designed for vertical loads (Chakrabarti 2005). Suction anchors differ from traditional anchors by also providing a vertical anchor

load. The suction anchor's holding capacity comes from a pressure differential using pumps. The under pressure in the anchor cavity submerges the anchor into the seabed as described in Figure 1.4.

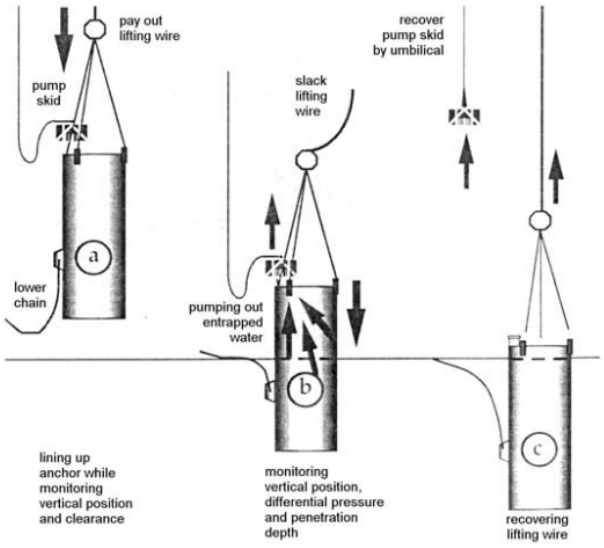


Figure 1.4: Installation of suction anchors. Courtesy: Chakrabarti (2005)

1.2.4 Mooring system failures

The failure rate of mooring systems for permanently moored units is a growing concern in the offshore industry. In the ten years from 2001-2011, the annual probability of failure was found to be ten times higher than DNV's Offshore Standards target (DNV 2021; Ma et al. 2013). In the same period, several previously unknown modes of failure were discovered from the incidents. Examples are out of plane bending, chain hockeling, pitting corrosion, and more (Ma et al. 2013). Unknown failure mechanisms are difficult to avoid since it is by nature unanticipated. Most of the new failure modes discovered are detected by visual inspection, advocating for new standards for inspection and maintenance. Mooring system monitoring is an essential tool for keeping track of mooring integrity and condition (Gordon et al. 2014). Especially to predict unforeseen failure modes and detect anomaly behaviors of the vessel before a failure occurs. Some of the most common sources of mooring failure can be seen in the list below.

- **Wear:** Long-time exposure to rubbing between line components and other instances of degradation cause the mooring lines to lose tensile strength.
- **Fatigue:** Long-time exposure to bending and axial stresses. The effect of tension and out-of-plane bending of different links can cause fatigue failures.
- **Abrasion:** The roughens of seabed sediments can cause deterioration of the mooring lines if there is contact. This is especially true when materials are designed for a mid-water touchdown on the seabed.

- **Corrosion:** Refined metals used as mooring line materials can suffer gradual destruction caused by chemical reactions with their environments. This degredation is especially true for the splash zone, the zone where the mooring lines reach the surface, as the water in this area is oxygenated.
- **Excessive tension:** Mooring systems placed in areas with more severe environmental conditions than designed for can experience failure due to excessive tension.
- **Operational failure:** Accidents and poorly executed operations can, in the worst-case scenario, lead to mooring line failure.
- **Installation:** If the system is installed differently from the design, the capacity of the mooring system may be degraded. This also applies to defected components and material flaws.

The consequences of these mooring failures are potentially catastrophic. For oil and gas installations it can cause vessel drift, which can be the source of riser rupture, production shutdown, and hydrocarbon release (Ibrion et al. 2020). Both these consequences themselves are severe, in addition to the necessary restorations of the broken mooring line. Apart from materials costs and damages, there is also the increased risk for human life on the platforms, which increases drastically with mooring failures (Ibrion et al. 2020). The consequences for the environment and human safety are smaller for mooring failures in floating wind but the economical consequences may still be large.

1.2.5 Service life monitoring of mooring systems

When considering the strength and reliability of a mooring system, both the present state and future degradation of the system must be taken into account, which in turn requires an understanding of the mechanical behaviors of all mooring components and the environmental influences on the behaviors. Understanding a mooring system requires an in-depth knowledge of everything from failure modes and tensile strengths to marine species, corrosion, and abrasions (Pham et al. 2019).

Three key concepts in service life monitoring are Ultimate limit state , Fatigue limit state (FLS), and Accident limit state . Each of these has a different purpose, as will be explained below, and are part of DNV's position mooring offshore standards DNV (2021).

- **Ultimate Limit State (ULS):** This design criterion defines requirements for individual mooring lines and what environmental loads each mooring line should be able to withstand.
- **Fatigue Limit State (FLS):** For permanent mooring, there is an expected degradation of the mooring lines. The FLS criteria define some measures for the expected service life of mooring lines, given the fatigue from decomposing materials and environmental factors.
- **Accident Limit State (ALS):** This criterion is based upon situations post-failure. Regardless of the cause, the aftermath of a line break can be detrimental or manageable, depending on the

redundancy of the design. The ALS criteria define how a mooring system should behave during failure.

DNV's standard for position mooring DNVGL-OS-E301 requires permanently moored units to monitor the integrity of each mooring line. Acceptable methods include direct load measurement and angle or depth measurement. Global Navigation Satellite Systems (GNSS), such as GPS, is an acceptable method for monitoring mooring lines if it can be documented to detect failures (DNV 2021, ch.2 sec.14) .

DNV's standard for fleet in service DNVGL-RU-OU-0300 dictates how DNV classified units are to be inspected throughout their lifetime (DNV 2018; Lillestøl et al. 2021a). For these inspections, the main goal is to quantify the fatigue of the mooring system per the FLS. If a unit has experienced more or less cyclic loading than expected, these inspections are meant to uncover this and how it has affected the mooring lines. The July 2021 Edition of the in-service DNV Class Rules, DNVGL-OS-0300, requires data collected on the actual behavior of the mooring system to be used in the recalibration of the design criteria, meaning the FLS can be refactored based on how the mooring system has behaved historically (DNV 2018; Lillestøl et al. 2021a) .

1.2.6 Digital Twin Framework

Digital twins (DT) are virtual copies of a physical system capable of replicating the behavior of the actual system in real-time by using data from sensors.

As the industries become increasingly complex, the number of unknown dynamics grows. Unknown dynamics affect the performance and output of the industrial process or system. The growth of big data, real-time monitoring, and machine learning have introduced Industrial Artificial Intelligence (IAI). IAI is utilized to create robust and autonomous methods of improving monitoring, optimization, and performance of complex industrial systems.

Figure 1.5: Digital twin framework, Courtesy: Viola and Chen (2020)

Viola and Chen (2020) proposed a framework for constructing digital twins for IAI, Figure 1.5 shows the proposed framework. The first steps build the method's foundation by gathering information to understand and model the system. Step 1, Target system definition, and step 2, system documentation, encapsulate all activities for collecting information on the system. Examples of these information sources are design drawings, system installation information, sensor package descriptions, and subsystems to

describe the physical system as accurately as possible. After information gathering, a multidomain simulation model is built to match the system definition and documentation gathered in the previous steps. The simulation model should capture all known physical laws and relations of the system, and the digital twins' scope defines the simulation model's complexity.

As the simulation model does not describe the unknown dynamics of the system, the simulation's behavior often deviates from the actual system. Therefore, behavioral matching is performed to optimize the DT to fit the physical systems' full dynamics. Methods within machine learning are used for behavioral matching.

The final part of constructing a DT framework is the model validation and deployment. The DTs' performance as a virtual copy of the physical system must be evaluated and quantified, ideally under all conditions and states the system may experience. Edge cases are essential to evaluate the models' robustness and fault detection methods.

In this thesis, a hybrid model will perform the simulation step and behavioral matching of the digital twin framework.

1.2.7 Predictive maintenance

The Internet of Things (IoT) has brought continuous data collection from a plethora of sensors and equipment. Advances in cloud-based storage and analytics have enabled maintenance to transition from preventative to predictive in many industries.

Preventative maintenance has been the norm within most industries, meaning equipment and parts are inspected and replaced after a given period, hopefully before a failure occurs. The intervals between maintenance are set by statistics of the unit's durability and the expected operating conditions. This results in most parts being inspected or replaced prematurely, while parts exposed to higher wear than expected are prone to failure before scheduled maintenance. Predictive maintenance attempts to optimize the interval based on the collected data on actual operational conditions, continuously updating the Remaining Useful Life (RUL) instead of using statistical averages for RUL. Ideally, predictive maintenance reduces costs by performing as little as possible planned maintenance while avoiding extra costs due to premature equipment failure. (Daily and Peterson 2017)

In the maritime industry, remote locations, strict safety requirements, and high downtime costs make maintenance a significant expenditure for the operational phase. In recent years, multiple concepts for predictive maintenance have emerged within shipping, oil and gas, and floating wind. Utilizing big data and machine learning methods Jimenez et al. (2020) proposed a model for predicting maintenance for vessel machinery in shipping. Moghadam et al. (2021) presented a method for building a digital twin for predictive maintenance of gearboxes in FOWT drivetrains. Floating offshore Wind farms are especially suitable for predictive maintenance due to the high amount of identical units and high maintenance cost, making the development of a predictive model more worthwhile as it can be utilized for the entire park.

1.2.8 Tension measurements

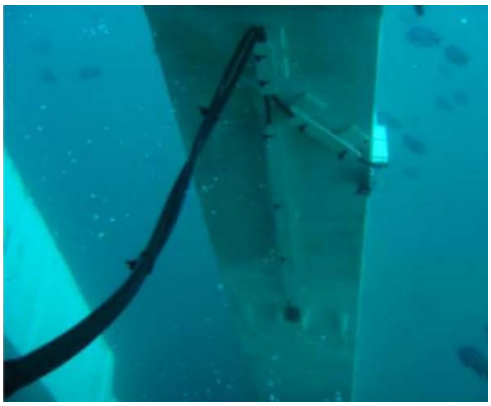
For condition monitoring of mooring systems, the induced tension for the mooring lines is an important parameter. The amount of stress the lines are experiencing provides insight into the mooring systems' condition. Ideally, accurate and reliable tension measurements of all mooring lines should be affordable and available. Line failures would instantly be detected, and fatigue from cyclic loading could be quantified. Unfortunately, with current solutions, adequately accurate and reliable tension readings are very challenging (Minnebo et al. 2014). Tension is either measured directly or indirectly on the mooring line. Direct tension measurements are provided from in-line load cells, as described by Elman et al. (2013). The main issue with current direct tension measuring equipment is the inability to provide adequate load range but still provide high-resolution data. A trade-off between capturing the highest loads and capturing high-resolution data must be made.

Additionally, the load cell is exposed to the harsh environment subsea with corrosion and abrasion, making the sensors prone to failure. Direct tension sensors either need a wire for power and data transmission or must rely on batteries. The wired solution can capture high temporal resolution, with the drawback being that the exposed wire is prone to failure. The battery-powered version is less vulnerable but must balance between high temporal resolution and frequency of battery replacements.

The alternative, in-direct tension measurements utilize inclinometers to measure the angle of the mooring line at the connection point to the unit. The line tension can be found using the in-direct angle measurement using catenary equations. The catenary equations are based on static equilibrium. They, therefore, neglect dynamic and highly non-linear effects, which may be significant, leading to the deterioration of the accuracy of in-direct tension measurements based on the catenary equations alone.

Hageman et al. (2019) argues that current solutions for direct tension measurements are insufficiently reliable for most mooring monitoring solutions. Direct tension measurements are more suitable to monitor damaged or weakened lines until replacements can be made and for validation of design models and assumptions.

(a) Inter-M Pulse H-link for direct tension measurement. Courtesy: PULSE mooring monitoring



(b) Direct load monitoring with electrical strain gauges. Courtesy: Hageman et al. (2019)

Figure 1.6: Direct tension measurement solutions

1.2.9 Alternative sensor packages

Sensor packages for monitoring FOWTs have been the focus of research last years. Many sensor solutions for the nacelle, gearboxes, umbilicals, and more have become available. The following section will introduce a focus on alternative sensor solutions for mooring monitoring.

The Global Navigation Satellite Systems (GNSS) were made available for civilian use in the mid-1990s. It first consisted of the American GPS and the Russian GLONASS systems. In the last decade, the GNSS has been expanded with the Chinese BeiDou system (completed in 2018) and, most recently, the European Galileo system (completed in December 2021) (Zidan et al. 2021). The availability of high accuracy positional data have been made possible by the growth of the GNSS system, creating a more robust and accurate sensor solution for offshore purposes. The disadvantage of the GNSS is the low temporal resolution. GNSS alone should be used with caution due to low sampling rates and wave-frequency motions issues (Ciuriuc et al. 2022).

For measuring a FOWTs acceleration, tilt, and velocities in roll, pitch, and yaw, an Inertial Measurement Unit (IMU) may be installed. An IMU consists of multiple accelerometers and gyroscopes. Unlike GNSS, an IMU requires no external reference signals and can provide a much higher sampling rate compared to GNSS (Fossen 2021, Chp. 14).

According to Hageman et al. (2019) systems using the Global Positioning System (GPS) have been used in multiple applications to detect mooring failures where outlier positions of the FOWT signal a potential mooring system failure.

Ren et al. (2019) proposed two methods for combining IMU and GNSS measurements with sensor fusion to get a real-time position and velocity estimate of offshore wind turbines. By combining a multi-rate Kalman filter with a moving horizon estimator, the issues with the low sample rate from GNSS measurements are mitigated. The resulting measurements proved applicable for a wide range of practical applications.

1.2.10 Physics-based modelling

Traditional mooring analysis has been based on physical laws and equations. Most analysis is performed in the design phase, prior to the installation of the offshore unit. The models are made from the design specifications and simulations performed with statistical distributions of sea states for the given installation location.

The industry standard for mooring system analysis is catenary equations and finite element method (FEM) solvers. The methods are built upon a trustworthy foundation from physical principles, verified with empirical data and model tests. Examples of software solutions for mooring system analysis are DNV's RIFLEX and MIMOMOSA software in the SESAM package; Orcina's OcraFlex package is an alternative. These methods apply to a wide range of configurations and floaters, making it easier to

apply the same method to multiple windmills in a wind park. The transparency of the methods enables errors and uncertainties to be quantified.

The traditional methods are computationally expensive and are not typically suitable for real-time computations. Therefore, most of the analysis has been done in the design phase of the mooring system. The downside to performing fatigue analysis in the design phase is the lack of ability to capture and correct the difference between ”As designed” and ”As installed” and a missing symbiosis between the model and the real world.

As the focus on mooring failures has increased, solutions for physics-based models (PBM) in real-time mooring monitoring have emerged. Figure 1.7 shows the architecture of a PBM for mooring monitoring. Each block in the model creates uncertainty in the estimate of mooring system response due to the design conditions that do not perfectly represent the actual system. This uncertainty causes an iterative degradation of the monitoring solutions' accuracy. Hageman et al. (2019) discusses the mooring monitoring system of Bluewater's FPSO Aoka Mizu, located west of the Shetland Islands. Due to harsh weather conditions at the site, an extensive mooring monitoring solution was installed. Inclinometers, providing measurements of mooring line tension and line failure detection, are installed at each mooring line. An indirect mooring force estimation system was also implemented using motion and environmental data. The data is fed into a detailed design model from aNySIM, providing real-time mooring system response. Compared to the architecture described in Figure 1.7, the system installed at Bluewater's Aoka Mizu only models the mooring system, as the motion measurements replace the floater motion response in the model, improving the computational expense of the mooring monitoring solution.

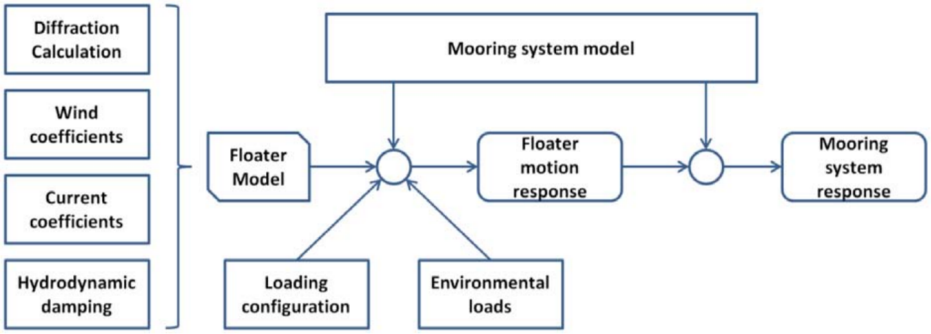


Figure 1.7: Typical Physics-based architecture for mooring monitoring, where the mooring system response is estimated. Courtesy: Hageman et al. (2019)

1.2.11 Data driven modelling

Data driven modelling (DDM) has become a popular method for modelling processes, plants, complex systems behavior, and properties by utilizing historical data. The increasing focus on data collection, combined with the advancements in machine learning methods, has enabled DDM to become a powerful analytical method (Ge 2017). Utilizing long-term historical data enables the data driven model to learn actual behavioral patterns of the system, patterns that theoretical models struggle to capture as they may

be too complex to include due to computational constraints. Examples are highly non-linear behavior, coupled effects, and complex hydrodynamics. Meanwhile, a data driven model is not based on physical laws or principles and is suitable for complex systems since the models are computationally efficient after the training phase. Data driven models have, in multiple cases, been shown to be significantly faster than physical models (Christiansen et al. 2013), enabling the models to run in real-time (Wu et al. 2014).

Another reason why DDM may be desirable is the ability to learn the ”as-installed” characteristics of the system. As discussed by Lillestøl et al. (2021b), the gap between ”as-is” and ”as-installed” conditions of long-term mooring systems can and should be quantified with data. The paper suggests using data from the Automatic identification system and historical weather data in combination with ML methods and statistical methods as a low-cost solution to provide insight into the condition of mooring systems.

Multiple papers have shown proof of concept for modelling slender marine structures with data driven methods. Christiansen et al. (2013) demonstrates how the computational cost of dynamic mooring analysis can be reduced by a factor of 600 by using an artificial neural net (ANN). Riemer-Sørensen et al. (2019) applied a data driven model to predict vortex-induced vibration response of marine risers subjected to three-dimensional current using a random forest regressor.

Walker et al. (2021) proposed two digital twin frameworks for monitoring of mooring system for floating offshore wind. The first DT concept is a proposal to detect long-term drifts in the mooring footprint. By training a DDM to predict healthy FOWT behavior, large deviations between the data driven healthy mooring system model and measured behavior may indicate a fault in the mooring system. The second DT concept proposed is a framework for predicting near-future top tension in the FOWTs mooring lines. Both DT concepts were tested on operational data from Hywind Scotland, and proved capable of providing high accuracy predictions, with a mean absolute deviation of around 15 kN for each mooring line.

The case study used by Walker et al. (2021) is also used in this thesis. The work of Walker et al. provides a good benchmark for the Hybrid frameworks of this thesis, but it is important to stress that the results achieved are not directly comparable. This is mainly due to the model validation schemes used. Walker et al. randomly select a subset of the data from the complete dataset for validating the model. In theory, this can lead to measurements a fraction of a second before and after the validation sample is included in the training set. By utilizing such a validation scheme, an overly optimistic estimate of predictive performance on unseen data is achieved. The validation scheme utilized in this thesis with a case-wise cross-validation scheme provides a more conservative estimate of predictive performance.

The downside to pure data driven models is primarily the fact that the models are so-called black boxes, meaning it is difficult or impossible to interpret the models' reasoning. Trust and confidence in AI have gotten increased attention in recent years, as DDM and AI are applied within safety-critical systems, transparency and interpretability are a growing concern (Došilović et al. 2018). Samek and Müller (2019) discussed recent advances in making explainable artificial intelligence, but compared to physics-based models, most AI is still to be categorised as a black box.

The consequences of a black box data driven model are a vulnerability to the quality and selection of training data. ML models can interpolate between data sets used in training but have minimal capabilities for extrapolating for unseen input. Lacking extrapolation means a robust model requires training data covering all potential model use cases, which poses a challenge for rare events. Especially in a marine environment, data from a 100-year storm is not likely to be included in a training set for a data driven model. This, combined with the fact that data driven models have no bounds for errors, gives the models poor performance given changes to the system or previously unseen problems.

1.2.12 Hybrid modelling

During the 1990s, the concept of hybrid modelling arose due to the potential benefits of combining both physics-based and data driven models into a combined model. Psichogios and Ungar (1992) improved a neural network model of a fed-batch bioreactor by introducing a partial first principles model of the system. The result was a more generalized and reliable model, with improved interpolating and extrapolating capabilities.

With the rapid increase in data-driven models, interest in hybrid modelling has reached new heights in recent years. There is a growing consensus that safety critical systems with ML methodologies trained on data must be coupled with physical modelling techniques to ensure reliability (Rai and Sahu 2020). Multiple fields of science are contributing to hybrid modelling resulting in inconsistent terminology. The terms ”gray-box”, ”physics guided”, ”physics fusion”, ”physics informed”, ”hybrid” are all for architectures utilizing a combination of physics-based modelling and machine learning algorithms (Rai and Sahu 2020; Sansana et al. 2021).

Implementing the coupling between PBM and DDM is not a straightforward process, and a plethora of architectures have been proposed in the literature. Rai and Sahu (2020) presented a review of current methodologies for hybrid modelling; a selection of architectures is presented in Figure 1.8. The figure is based on a figure from the overview paper, with adapted terminology and symbols used in this thesis. Figure 1.8 (a) shows how physics-based preprocessing (PBP) can be applied to a data driven model. PBP includes a wide range of methods, including domain knowledge of the system's behavior enabling improved noise and outlier removal. The preprocessing transforms the input data X into $\hat{X}$, which is then used as input to the data driven model. The output of the DDM, $\hat{Y}_{DD}$, is then used in the loss function together with the measured truth (in training) Y. Martens (2021) applied physics informed dimensional reduction of multi-variate data to extract systematic covariation patterns and sent only the residual irregularities to a black-box ML method.

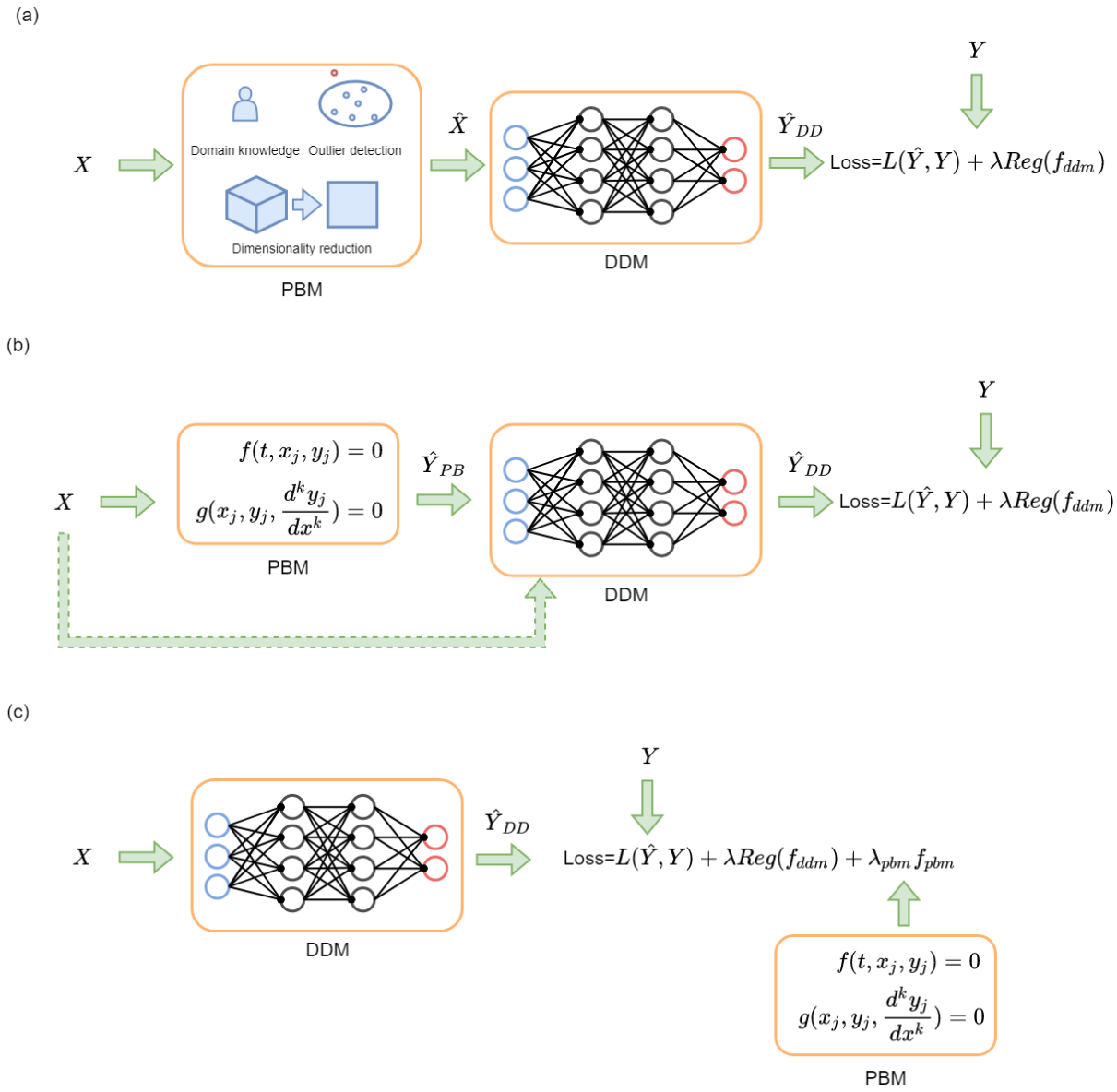


Figure 1.8: Architectures for hybrid modelling found in literature. (a) Physics based preprocessing for DDM, (b) Hybrid architecture where the output from PBM is used as input to DDM (c) DDM with Physics-based regularization. Adapted from Rai and Sahu (2020)

Architectures where the PBM predicts the output value, $\hat{Y}_{PB}$, is shown in Figure 1.8 (b). Here the physics-based prediction is used as input to the DDM. The input data can also optionally be sent into the DDM. The measured output value is used in the loss function to train the network. Fuchs et al. (2016) apply a DDM to improve the accuracy and computational time of a mathematical aircraft wake model.

Figure 1.8 (c) show physics-based regularization. Instead of the PBM being fed input data, the physics are included in the loss function, penalizing violations of physics-based constraints and governing equations. The governing equations may be first principle laws like mass, momentum, or energy conservation (Chang and Dinh 2016).

To the best of the authors' knowledge, no studies have compared multiple hybrid approaches against each other and provided a quantifiable argument for using one approach over another. The choice of method is often limited by the available PBM and the problem to be solved. In this thesis, versions of method b and method c from Figure 1.8 will be implemented and compared. Hereafter, the methods will be denoted method A and method, respectively.

The common denominator for hybrid modelling methodologies is the attempt to capture the beneficial characteristics of both physical and data driven modelling while minimising the methods' weaknesses. Achieving the trustworthiness and interpretability of a physics-based model while achieving the accuracy and computational effectiveness of data driven models.

1.3 Objectives and scope

This thesis explores the potential of combining data driven and physics-based modelling to create an accurate and cost-effective method of mooring system monitoring for floating wind turbines. The combination of these models, the hybrid model, is the basis for the research question for this thesis: How to create a trustworthy and accurate alternative to direct tension measurements for floating wind turbines?

The research question will be investigated through the following topics:

- Mooring systems for floating wind turbines and their failure modes
- The background theory for physical and data driven modelling and previous work on hybrid modelling
- The proposed digital twin framework and its implementation for monitoring and prediction of remaining useful life

A case study of the world's first floating wind farm, Hywind Scotland, is established, followed by a comparison between the results of the hybrid approach and pure DDM and PBM. The final results will then investigate and discuss the merits of hybrid modelling for in-direct tension measurement using positioning and metocean data.

1.4 Contributions

With this thesis, we aim to contribute to the ongoing work on virtual sensors and digital twins within the marine sector. This is done by building on previous work on DDM with operational data (Walker et al. 2021), and PBM for mooring system of floating wind turbines (Borg et al. 2014), and then creating a new hybrid modelling digital twin framework. The ambition of the framework is to prove that the combination of these methods is another step in the right direction for accurate and trustworthy mooring monitoring of offshore floating wind, which is expected to have both tremendous growth and subsequent need for innovation in the coming years.

To the best of the authors' knowledge, no one has previously combined PBM and DDM into a hybrid framework for mooring system monitoring. By investigating multiple physical and data driven models and architectures for combining the models, further studies within the field may have a quantitative basis for hybrid model selection.

1.5 Outline of thesis

The contents of this thesis are divided into six chapters.

Chapter 1: Introduces the background and literature review for the topic of the thesis. The motivation and scope of the thesis is also presented.

Chapter 2: Presents theory for physical-based model, mooring line analysis and catenary equations, and briefly touches on the finite element method for mooring line analysis.

Chapter 3: Theory and mathematical formulation of data driven models with machine learning.

Chapter 4: Presents the case study used in the work of the thesis, the available data and information for the case, and how the proposed hybrid methodology is implemented.

Chapter 5: Results from physical-, data driven- and hybrid-models are presented and compared. The results and differences between the models' results are discussed.

Chapter 6: Concluding remarks and proposals for further work on hybrid modelling for mooring line analysis.

Chapter 2

Theory of mooring system analysis

The mooring system is an integral part of floating offshore wind. Deep knowledge and understanding of the physical properties and behaviors of mooring systems are therefore critical for the design and monitoring of the mooring systems.

Borg et al. (2014) provides a comprehensive list and details of relevant modelling approaches for floating offshore wind.The main approaches are presented in Table 2.1 in order of increasing levels of complexity and capabilities. Simplest model is a linear force-displacement-velocity (FDV) model, where the mooring forces are described by a linear model consisting of a mooring stiffness proportional to the floater displacement, and a dampening term proportional to the floaters velocity. The mooring system is modelled as one unit in the FDV approach, it is therefore not possible to investigate individual mooring line forces. To study mooring forces of individual mooring lines a quasi-static model may be utilized. A quasi-static approach includes the forces acting on the individual lines, but not the inertia and other dynamic forces. Multibody and FEM allow for both quasi-static and dynamic analysis, but increasing complexity come at a increased computational cost (Borg et al. 2014).

	Linear F-D-V	Quasi Static	Multibody	Finite element
Static (average) tension	x	x	x	x
Individual line tension		x	x	x
Line-seabed interaction		x	x	x
Line longitudinal stiffness		x	x	x
Line bending stiffness			x	x
Line inertia/hydro-damping			x	x
Line torsional stiffness				x

Table 2.1: Mooring line modelling approaches and capabilities. Adapted from Borg et al. 2014

Analysis of mooring systems is a comprehensive field of engineering, to limit the thesis scope only theory regarding our later case study will be covered. Focus on a quasi-static analysis of catenary mooring configurations.

2.1 Catenary mooring line analysis

Figure 2.1 illustrate a catenary line with length L, anchored to the seabed at point B, and connected to a floater at point A. When the floater is positioned at point A, a part of the mooring line is resting on the seabed, not supported by the floater. The suspended part between the seabed and the floater is supported by the floater and the forces F_3 and F_4 at the floater connection point is due to the total weight in sea water of the suspended line length. As the floater moves right in the horizontal direction, increasing the horizontal length of the mooring line, a, more of the mooring line is lifted from the seabed and is suspended in water. This results in more weight supported by the floater, causing higher line tension near point A. The line angle, between the horizontal direction and the line, is decreased when a increases, shifting more of the force into the horizontal component F_3. providing a restoring force to the floaters horizontal movement. Accordingly when the floater moves to the left in the horizontal direction, decreasing the horizontal line length, more of the line is placed on the seabed, in combination with an increased line angle resulting in reduced restoring forces at the floater.

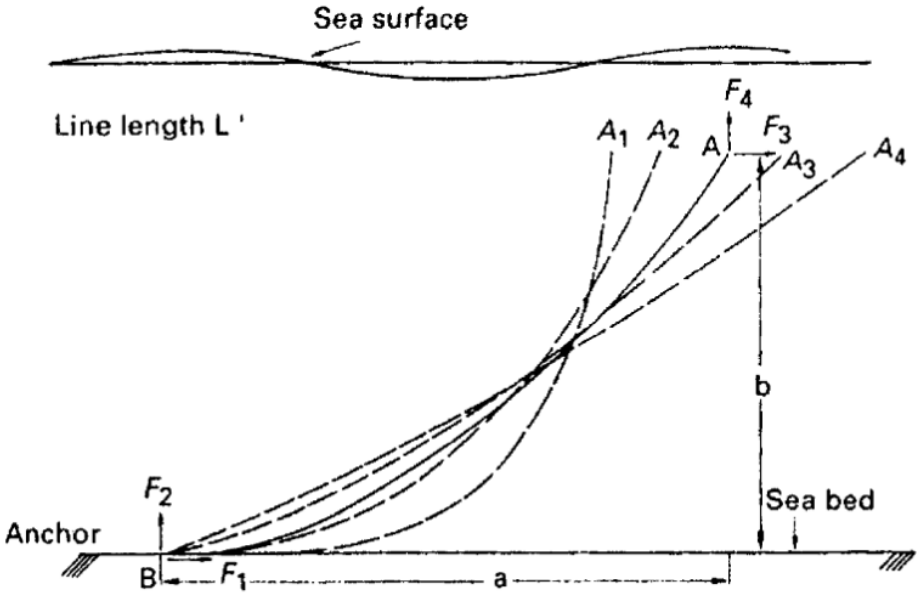


Figure 2.1: Catenary line motions caused by floater movement. Courtesy: Chakrabarti 2005

The behavior of catenary mooring lines may be described mathematically by the catenary equations. These equations describe the behavior of single mooring lines, for complete system analysis the contributions from each line in the system may be summed up. Catenary analysis of complex mooring systems can be a cumbersome task and is suitable for computer software. Several software packages for catenary analysis are available from the industry, including OrcaFlex, Mimosa, and Riflex.

Figure 2.2 show a single mooring line with symbols and reference frame used in the following analysis.

- T: Line tension

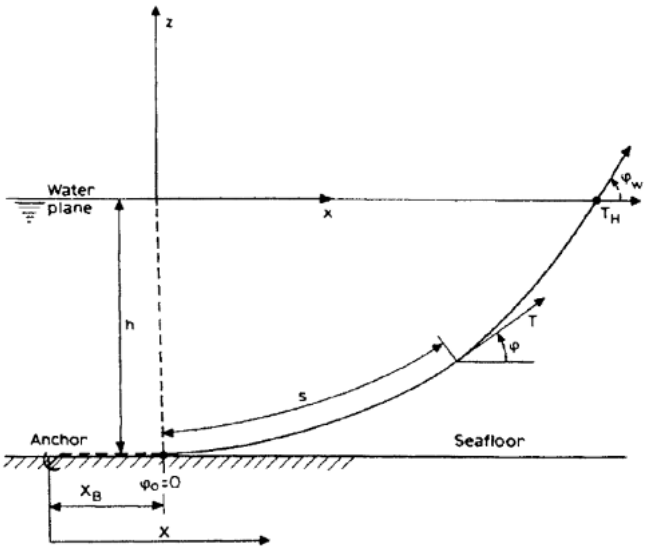


Figure 2.2: Mooring line with symbols, Courtesy: Faltinsen 1999

- T_h: Horizontal component of tension at the connection point
- s: Suspended line length
- h: vertical dimension, in this case the water depth
- x_B: Length of line resting at the seafloor
- ϕ: Line angle
- ϕ_w: Line angle at the connection point

A static approach for analysis of catenary mooring lines is described in Faltinsen (1999, Chp.8) and Chakrabarti (2005, Chp.8) and will be presented in the following section. Figure 2.3 show an element of the mooring line and the forces acting on it. The equations are formulated mathematically by Equation 2.1, where T is the line tension in the mooring line. F and D represent the hydrodynamic forces acting on the line section, F in tangential direction, and D in normal direction with regards to the line cross-section. According to Chakrabarti (2005), the elastic stretch of the line and the environmental forces acting on the mooring line may be ignored for shallow waters and non-taught lines.

$$dT - \rho g A dz = \left[w sin\phi - F\left(1 + \frac{T}{EA}\right)\right] ds \tag{2.1a}$$

$$Td\phi - \rho g A z d\phi = \left[w cos\phi + D\left(1 + \frac{T}{EA}\right)\right] ds \tag{2.1b}$$

The mooring lines characteristics are described by the submerged weight of line per unit length, w, the elastic modulus, E and A the cross-sectional area of the line segment.

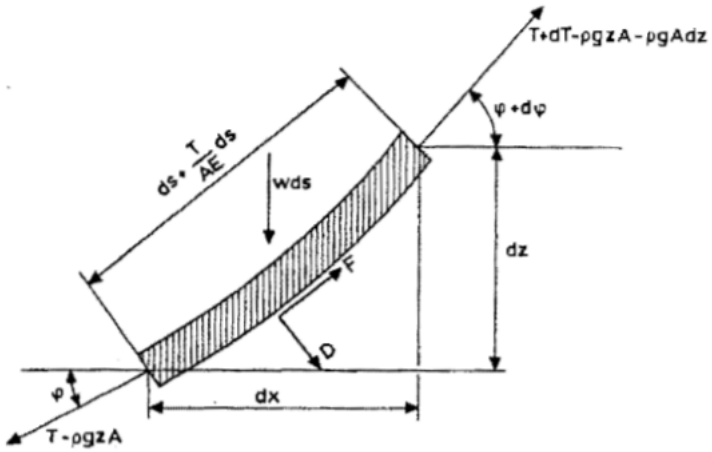


Figure 2.3: Forces acting on a line element, Courtesy: Faltinsen 1999

The catenary equations are nonlinear and not generally solvable. To simplify the equations the following assumptions are made

- Neglect F/D, no hydrodynamic forces acting on the mooring line
- No bending stiffness or torsional stiffness, a reasonable assumption for chains (Faltinsen 1999)
- Ignore elasticity, mooring line have a constant length
- Flat seabed, where the line lies horizontal at the lower end

Using the assumptions above the suspended line length s and vertical dimension h can be obtained using respectively Equation 2.2 and Equation 2.3

$$s = \frac{T_H}{w} sinh\left(\frac{wx}{T_H}\right) \tag{2.2}$$

$$h = \frac{T_H}{w}\left[cosh\left(\frac{wx}{T_H}\right) - 1\right] \tag{2.3}$$

From Figure 2.2 it can be seen that the vertical and horizontal component of the line tension, T_H and T_z, at the top end becomes:

$$T_z = ws \tag{2.4}$$

$$T_H = T cos\phi_w \tag{2.5}$$

By rewriting the equations for the top end of the line, using depth d as the horizontal dimension h. The

tension at the top end is expressed as:

$$T = \frac{w(s^2 + d^2)}{2d} \tag{2.6}$$

To account for the section of mooring lines laying on the seabed, x_B a iteration scheme is applied. Starting with the line fully suspended, $x_B = 0$, parts of the mooring line is laid at the seabed incrementally, until force equilibrium is achieved with the restoring force.

As described in subsection 1.2.3 mooring lines are typically made up of multiple segments to improve the characteristics of the mooring system. In a catenary analysis different segments are analysed separatly, where the position of the connection points between the line segments are incrementally moved till there is no force imbalance in the connection points.

2.2 Finite Element Method

Finite element method (FEM) is a numerical approach to find approximate solutions to differential equations. FEM has been widely applied within the field of engineering for decades, and the use cases vary greatly. Within the field of marine technology FEM is used for computational fluid dynamics, structural analysis of hulls, mooring analysis and more Dhatt et al. 2012.

The basics of finite element analysis (FEA) consists of first defining the physical problem, with physical laws describing the problem in terms of differential equations. The object to be analyzed in FEA is defined as the domain. Everything outside the domain is defined as boundary conditions. In a mooring analysis boundary conditions may be external pressure, forces or environmental conditions.

The initial conditions of the analysis must be defined. The initial conditions is either set, or may use the results from an simplified analysis like the caternary equations as a starting point. The shape, position, and pre-tension of mooring lines must be defined.

The 'Finite Element' part of the analysis comes from the division of the domain into small elements. The connection points between the finite elements are defined as nodes. The initial conditons and external conditions are then casted upon the nodes, and a iterative scheme is applied until convergence.

2.3 Static finite element analysis

A short summary of the theory behind the static finite element analysis is described in this chapter. Taken from RIFLEX theory manual (SINTEF Ocean 2019).

Goal of the static analysis to determine the nodal displacement vector that corresponds to the system being in static equilibrium. The equilibrium state is found as the solution to the system of equations in

Equation 2.7.

$$R^S(r) = R^E(r) \tag{2.7}$$

where

- r - Nodal displacement vector containing displacements and rotations for all nodes relative to the stress-free configuration.
- $R^S(r)$ - Internal structural reaction force vector found by assembly of element contributions
- $R^E(r)$ - External force vector assembled from all elements.

The structural ($R^S(r)$) and external forces ($R^E(r)$) will in general be nonlinear functions of the nodal displacement vector. Static equilibrium is determined numerically by incrementally applying loads with equilibrium iterations for each load step.

For catenary mooring systems the bending stiffness of the mooring lines have a negligible contribution to the static equilibrium. As a result the solution from catenary equations provide a close approximation to the static equilibrium configuration. For such systems, the catenary solution can be used as the initial guess to the FEA. Initializing the FEA with a approximate starting configuration results in a significant reduction in computational time.

The static equilibrium in each load step k corresponds to zero imbalance force in the force imbalance vector at load step k as described by Equation 2.8.

$$R_k(r) = R_k^S(r) - R_k^E(r) \tag{2.8}$$

The equilibrium configuration for each load step may then be found by applying a iterative numerical solver. The initial values at current step (k) for the iterative scheme are found from the static equilibrium at previous load step ($k-1$).

$$\Delta r_k^0 = -\left[\frac{\partial R_{k-1}}{\partial r}\right]^{-1} \left(R_{k-1}^S - R_k^E\right) \tag{2.9a}$$

$$r_k^0 = r_{k-1} - \Delta r_k^0 \tag{2.9b}$$

where Δr is the incremental displacement vector and $\frac{\partial R_{k-1}}{\partial r}$ is the tangential stiffness matrix, commonly denoted K.

$$\frac{\partial R_k}{\partial r} = K = K_M + K_G \tag{2.10}$$

K_M and K_G denote the material and geometric stiffness matrices, respectivly.

Due to the favorable quadratic convergence rate, Riflex applies an Newton-Raphson iteration procedure to Equations 2.9. For iteration cycle j the iteration scheme is:

$$\Delta r_k^j = \left[\frac{\partial R_{k-1}}{\partial r}\right]^{-1} R_k^{j-1} \tag{2.11a}$$

$$r_k^j = r_k^{j-1} - \Delta r_k^j \tag{2.11b}$$

The iteration scheme is performed until max iterations is reached or the convergence criterion is met. The convergence criterion is given by Equation 2.12.

$$\frac{||\Delta r_k^j||}{||r_k^j||} < \epsilon_d \tag{2.12}$$

The tolerance, ϵ_d is a specified requirement for convergence. $||r_k^j||$ is a modified Euclidean displacement norm, given by

$$||r_k^j|| = \frac{1}{N_t}\sum_{i=1}^{N_t} r_{ki}^2 \tag{2.13}$$

where N_t is the number of translational degrees of freedom, and the summation is only performed on the translational components in r.

2.4 Fatigue Analysis

Following section based on DNV-RP-C203 for Fatigue Design of Offshore Steel Structures and DNV-OS-E301 standard for position mooring. The accumulated fatigue damage to a mooring line due to cyclic loading is summed up from the load cycles experienced (DNV 2021, Chp.2 Section 2).

A S-N curve defines a components capacity against fatigue. It expresses the number of stress cycles $n_c(s)$ at a given stress s until failure of a component is expected.

$$n_c(s) = a_D s^{-m} \tag{2.14}$$

In Equation 2.14 a_D and m refer to the intercept parameter and the slope of the S-N curve respectivly. These are constants and values are given for chains and ropes in DNV standard DNV-OS-E301.

Assuming a linear cumulative damage model (Palmgren-Miner rule):

$$D = \Sigma_{i=1}^{k} \frac{n_i}{N_i} \leq \eta \tag{2.15}$$

where the accumulated fatigue damage D, is expressed by the sum of number of cycles at stress S_i, n_i, and the number of cycles to failure at stress S_i, N_i. η describes the usage factor.

The stress cycles in Equation 2.15 are found using a cycle counting algorithm. A common option for cycle counting is rainflow counting. Rainflow method refer to a range of algorithms where a timeseries of stresses are split into individual cycles. This reduces the statistics used to describe the stress-timeseries into a list of stress ranges and corresponding number of occurences of given stress ranges.

By keeping track of experienced stress cycles of each mooring line component the accumulated damage can be quantified and in turn be used to give an estimate of remaining useful life of component.

Chapter 3

Theory for machine learning

Machine Learning (ML) is a rapidly growing field within Artificial Intelligence (AI). ML algorithms span a wide range of regression, classification, and clustering methods. In recent years ML algorithms have been applied to problems within most fields and industries to solve problems that would be too labor-intensive or complex for conventional methods.

As Figure 3.1 shows, the main three branches of machine learning methods include supervised learning, unsupervised learning, and reinforcement learning. Supervised learning requires both input data and the desired outputs to be included in the training set. The supervised methods, in general, attempt to learn the connections between the input and output data to predict the output from the input data. Unsupervised learning methods do not require the desired outputs to be included in the training data. Instead, it learns structures and similarities. They are used for data exploration, clustering, and outlier detection. Reinforcement learning is used within the control of systems where the learning algorithm is rewarded and punished in a feedback loop for good and bad actions, respectively.

As this thesis is studying the possibility of replacing a sensor package for tension measurements with an indirect method, we wish to recreate the measured output as close as possible. Meaning we require the desired outputs to be included in the training data, making supervised training the method of choice. More specifically, we attempt to map the relation between cheaper sensor packages and the desired tensions measurement. The mapping is a regression problem. We will explore the theory behind the machine learning methods within supervised regression problems in the following sections. During literature review of DDM for tension prediction in a maritime setting the most used methods described were artificial neural nets (Christiansen et al. 2013) and Random Forest Regressors (Riemer-Sørensen et al. 2019), these methods were therefore the primary focus for this thesis.

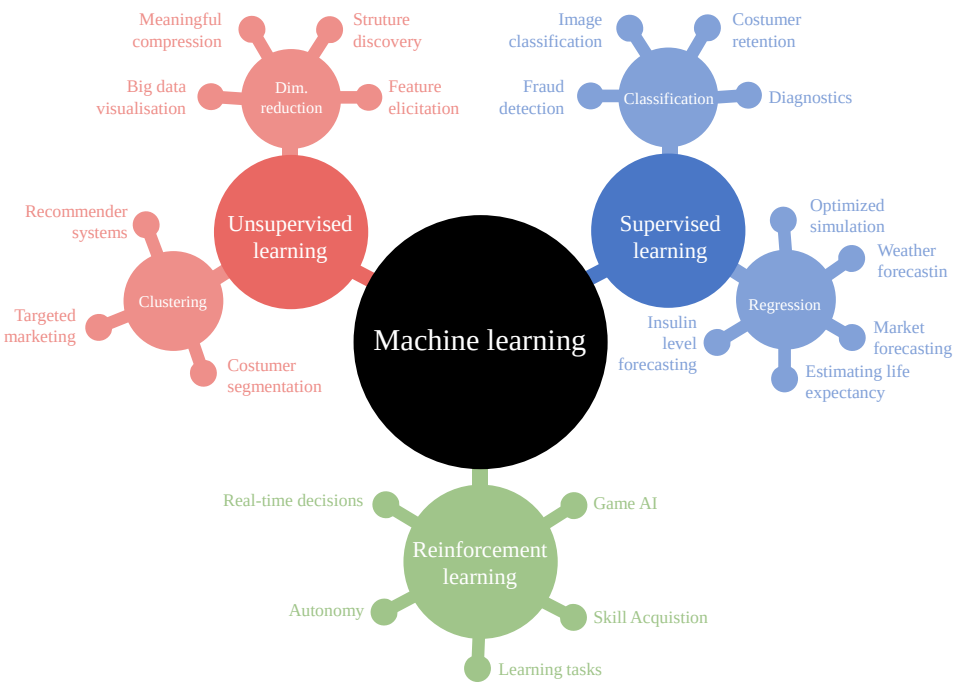


Figure 3.1: An overview of the machine learning taxonomoy. Source: Adapted from Gao et al. (2020)

3.1 Supervised learning

Supervised learning is a particular case of machine learning used when one has the opportunity to observe and supply the desired response for a model as a part of the training process. More formally, the task of supervised learning is as follows:

Consider some set of variables x, drawn from a fixed but unknown distribution $P(x)$, and subsequent y for each x from a unkown conditional dependence function $P(y|x)$. Given a set of N input-output pairs $I = (x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, use the input-output pairs to discover some function f that approximates $P(x)$.

Supervised learning is one of the largest subdomains of machine learning, and its methods have evolved in parallel with data availability and processing power over the last decades (Russell and Norvig 2003; Vapnik 1991).

Generally, supervised learning is a threefold process:

1. **Tuning the model,** where different hyperparameters for the function f are tested to approach the best approximation of P.

2. **Producing the model,** where all data available is used with the hyper-parameters from the previous step to enable our function f.

3. **Evaluating the final model,** where the performance of the model is quantified.

3.2 Performance Estimation Methods

Performance estimation is the process of evaluating the performance of a model on new and unseen observations and measurements. Quantifying how some model correctly handles unseen input-output pairs is a fundamental part of machine learning. The method and metrics applied to achieve this quantification are essential to the success of the training process. Within the evaluation domain, the concept of loss and the loss function is used to handle this quantification: How can a model's performance, good and bad, be reduced to a single number? The loss function can be defined as a performance estimation metric and is one of the two key concepts within performance estimation (Cerqueira et al. 2020).

Another important concept within performance estimation is how to train and validate the model. There are several methods for this purpose, and which to choose depends on the underlying characteristics of the data set. An independent and identically distributed data set, meaning all variables are equally distributed and have no impact on each other, will require a different strategy compared to a time-series where the temporal placement of a data point might have a significant impact on the predicted value (Arlot and Celisse 2010). There will be further discussion of the most common methods and their advantages and disadvantages below.

3.3 Risk and risk minimization

Loss, as described in section 3.2, is an important part in choosing and evaluating the best approximation function f. This section is based on Vapnik (1991). The loss function can be defined as some $L(y, f(x))$, which describes the discrepancy between the y and $f(x)$. The expected value of loss can then be defined by the risk function:

$$R(f) = \int L(y_i, f(x_i))dP(x, y) \tag{3.1}$$

where $P(x, y) = P(y|x)P(x)$ is the joint distribution introduced in section 3.1. The goal is to minimize $R(f)$, but with the unknown joint distribution, which can be solved using the training set. The risk function can be replaced with an empirical risk function $E(f)$ such that:

$$E(f) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(x_i)) \tag{3.2}$$

By minimizing Equation 3.2, it is assumed that the the the risk R is also minimized. How the minimization can be done using optimizers will be presented more in-depth later for a more specific case.

3.3.1 Loss functions

Loss functions are statistical performance indexes and are a vital part of all supervised machine learning methods. Evaluating the performance of models is needed to improve the model; the loss function grades how good a prediction is, and the model attempts to minimize the loss function by changing model parameters.

Numerous methods exist for formulating mathematically how close a prediction is to the actual value. The choice of method depends on the desired optimization goal of the machine learning model, and all loss functions come with a set of advantages and drawbacks.

Regression and classification problems use different loss functions due to different ambitions of solving the problems. In classification problems, the cost of being right and wrong may not be equal. The choice of loss function is dependent on the algorithm's confidence in predictions and performance. For regression problems, loss function is decided based on how deviations should be punished or how well the model's features explain the variance in the output. The most common loss functions for regression problems are described in detail below.

Define the regression problem as:

$$Y_i = f(X_i, \beta) + e_i \tag{3.3}$$

Where Y_i is the dependent variables in a observation, also know as the feature to be estimated. The function $f(X_i, \beta)$ is function to be estimated, with β being the regression coefficients in the regression problem. X_i is the training data, also denoted as the independent variables. The deviation, $e_i = Y_i - \hat{Y}_i$, is the difference between the estimated value, $\hat{Y}_i = f(X_i, \beta)$, and the measured value, Y_i,

Root Mean Squared Error

Root mean square error (RMSE) is one of the most common statistical performance indexes presented for evaluating machine learning algorithms in the literature. For N samples RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{\Sigma_i^N (Y_i - \hat{Y}_i)^2}{N}} \tag{3.4}$$

It represents the quadratic mean of the difference between estimated and observed values. The unit of RMSE is the same as the dependent variable, making interpretation easier. RMSE is in the range $[0, \infty)$, where lower values mean smaller average model estimate error. Since the deviations are squared, large deviations are punished harder than smaller ones. Implying that the RMSE as a loss function emphasizes the largest deviations, this property is valuable in cases where an error of 2ϵ is more than twice as bad as an error of ϵ. Chai and Draxler (2014) argue that a weakness of RMSE as a loss function is it's sensitivity to outliers in the data set, but concludes that RMSE is the best option when the error distribution is

expected to be gaussian.

Mean Absolute Error

Mean Absolute Error (MAE) is similar to RMSE but has a few key differences. MAE for N samples is defined as:

$$\text{MAE} = \frac{\Sigma_i^N |Y_i - X_i|}{N} \tag{3.5}$$

Most notably, the absolute value of the error is summed instead of the squared value. Consequently, an error of 2ϵ is twice as bad as an error of ϵ, giving larger errors no more weight than smaller ones, making MAE as a loss function less sensitive to outliers in the data than RMSE. The downside to MAE is in the definition; with absolute value, mathematical calculations become more difficult. Consequently, finding the gradient of MAE for model parameters is more computationally difficult or unfeasible (Chai and Draxler 2014).

Coefficient of determination

The coefficient of determination (R^2) expresses a regression model's goodness of fit. R2 typically lies between 0 and 1, where 1 means the regression model perfectly fits the data. Negative values can occur when the model's predictive performance is worse than just predicting the mean of the data. More formally, the proportion of the variance in the dependent variable is predictable from the independent variables. Formulated mathematical:

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} = 1 - \frac{\Sigma_i^N (Y_i - \hat{Y}_i)^2}{\Sigma_i^N (Y_i - \bar{Y})^2} \tag{3.6}$$

Where SS_{res} is the residual sum of squares in the model and SS_{tot} is the total sum of squares. $\bar{Y}$ is the mean value of the dependent variables.

3.3.2 Model validation methods

To get a estimate of the predictive performance of a machine learning model, the model must be validated. The available data is split into a training set, a test set and a optional validation set. The training set is used to fit the model to the data. The optional validation set is sometimes used to evaluate the fit of the model while training, to improve not only the models fit to the data but also its predictive performance on unseen data. The validation set can help mitigate risk of overfitting. The test set is used for final validation of the models performance on unseen data. The test set should be completely independent of the rest of the data, and it is vital that no data leakage occur from the training set into the test set.

A wide range of methods of splitting the data into training and test set exist. Out-Of-Sample (OOS) method, commonly known as the holdout method, is an approach to performing model evaluation within machine learning. In its simplest form, OOS validates by taking out some subset of the available data and keeping it to perform testing after the model has finished training. More formally:

Let I be the set of input-output pairs defined in section 3.1 with a size N. For some I^t, require it to be a proper subset $I^t \subset I$, selected using some function P such that $P(I) = I^t$, and define its complement as $I^v = (I^t)^\complement$. The training set is then OOS if there exists some temporal point ϵ such that all input-output pairs in I^t are temporally spaced before ϵ, but all pairs in I^v are after. When a model is trained on I^t and validated on I^v, an OOS approach has been used.

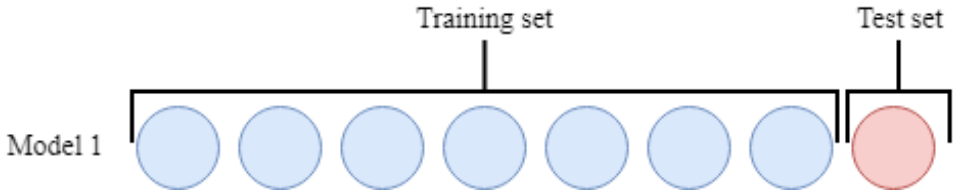


Figure 3.2: Out of sample model validation scheme

This method is beneficial for evaluating models meant to be used in real-time, as it properly introduces data more realistic than a deployment scenario. OOS approaches also give a more accurate representation of the model's grasp of non-stationarity. As only the in-sample temporal dependencies are captured, a non-stationary change outside of ϵ would need to be captured based on what is learned inside ϵ.

3.3.3 Cross-validation approaches

Cross-validation (CV) is another technique for model validation. When performing machine learning, one typically wishes to train the model on as much data as possible to improve the model's predictive performance. While still saving enough data to get a good estimate of the true predictive performance on unseen data. Compared to OOS approaches, CV utilizes more available data for model validation. By training multiple models on subsets of the data, then predicting the remaining data, a better estimate of the predictive performance of a model is achieved (Berrar 2019).

Several cross-validation algorithms exist. The main categories are exhaustive and non-exhaustive schemes. As cross-validation generates one model per step, it has a much higher computational cost than OOS approaches. The most exhaustive method is leave-p-out CV, where p samples out of a set of N samples are used for validation, and the remaining samples are used for the training set. This is then repeated to all possible combinations of p samples that are used as a validation set. The amount of models, M, created is given by the binomial coefficient:

$$M = \binom{N}{p} = \frac{n!}{k!(n-k)!} \tag{3.7}$$

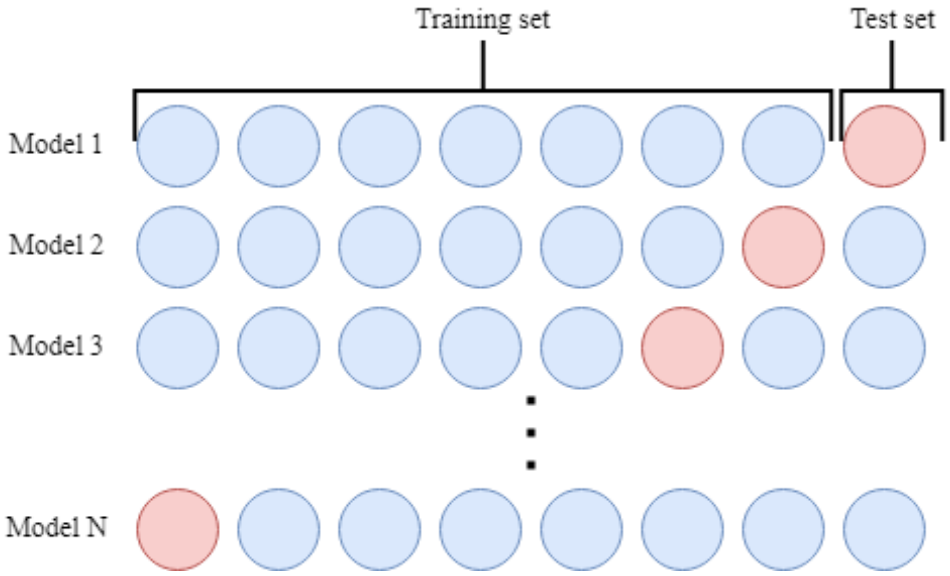


Figure 3.3: Leave-one-out cross validation scheme, blue marks samples in training set, while red samples are in the validation set

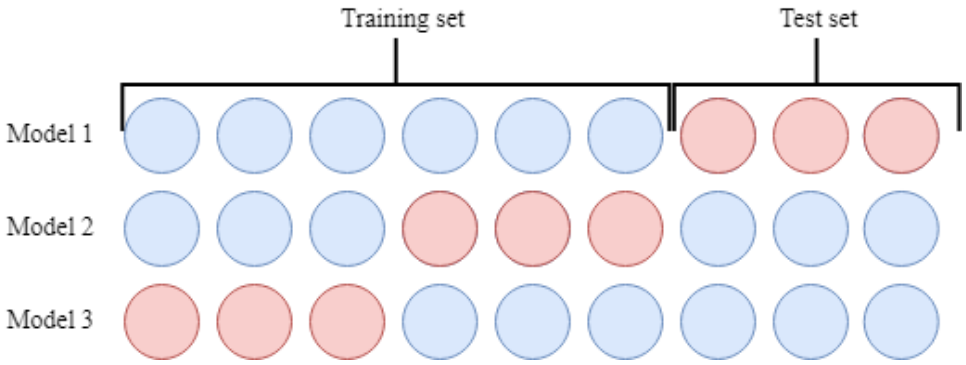


Figure 3.4: K-fold cross-validation with $k = 3$, blue marks samples in training set, while red samples are in the validation set

For a (small) data set with 100 samples and $p = 5$, over 75 million models are required to perform full leave-p-out cross-validation.A special case of leave-p-out CV is leave-one-out CV, where $p = 1$. Shown in Figure 3.3 leave-one-out reduces the number of models down substantially to $M = N$. The test set error in leave-one-out CV is approximately an unbiased estimate of the true prediction error (Berrar 2019).

For large data sets exhaustive CV approaches are unfeasible due to computational cost. Non exhaustive methods are an estimation of leave-p-out cross-validation methods. Most commonly used is k-fold CV, where the set of samples N is divided into k subsets, and each subset is used for validation once. A k-fold CV scheme is shown in Figure 3.4. For $k = 1$ the method becomes identical to leave-one-out CV.

3.4 Neural Networks

Neural networks (NN) are among the most discussed and anticipated subcategories of machine learning. Over the last 60 years, computers' ever-increasing computational power has followed Moore's law, which states that the number of transistors within the same area will double each second year. The size of transistors has decreased by a factor of 1 000 since the 1970s, giving way to far more complex computational tasks such as NNs (Popper et al. 2018).

The human brain inspired the base idea of NNs. In 1958, Rosenblatt (1958) defined a mathematical approximation for the behavior of neurons in the brain, aptly named the perceptron. The perceptron can be seen as a linear regression model with data, weights, bias, and output. More in-depth, the perceptron uses linear combination and some activation function σ to transform the input X, of size N, to some output y. For each $x_i \in X$, a corresponding weight w_i reflects how important or unimportant the given input is. Each of the weighted inputs is linearly combined and then combined with a bias b, which corrects the net sum of the combination and defines this linear combination as z. Finally, the bias and linear combination are given to the activation function, which will be introduced later. The mathematical definition of a perceptron can be seen in Equation 3.8, and an illustration in Figure 3.5.

$$y = \sigma(z) \qquad \text{where} \qquad z = \sum_{i=1}^{N} w_i x_i + b \tag{3.8}$$

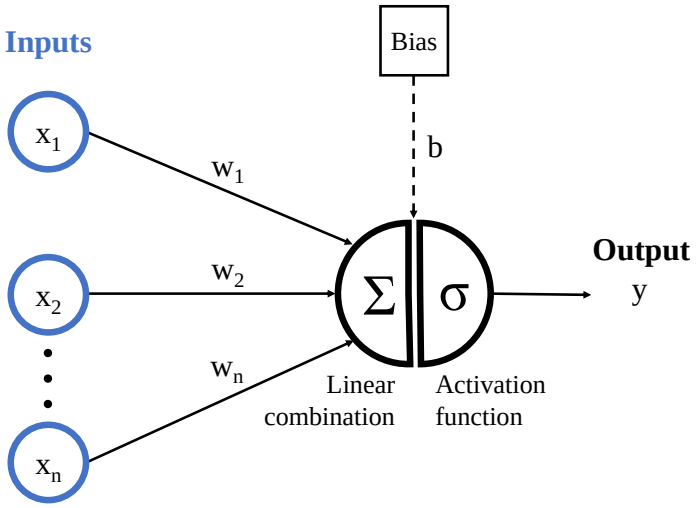


Figure 3.5: A standard perceptron.

3.4.1 Activation function

One of the most significant advantages of NNs is their capability to learn any nonlinear function (Nwankpa et al. 2018). Also nicknamed Universal Function Approximators, ANNs ability to learn any input-output

mapping comes from the activation function by introducing the network to nonlinearity.

The task of the activation function is to add nonlinearity to perceptrons. However, another important characteristic is that the activation function must be differentiable. Backpropagation, the tuning of NNs, is only possible given a differentiable activation function, which will be discussed later.

3.4.2 ANN

The simplest form of a NN is just a single perceptron, which can classify linearly separable classes. However, for most modern use cases, the perceptron is used as a building block for larger NNs. Artificial neural networks (ANN), also known as feed-forward neural networks, consist of three types of layers: the input layer, hidden layers, and the output layer, as shown in Figure 3.6 (Grossi and Buscema 2008) . Except for the input layer, all nodes that make up an ANN are perceptrons. For each layer, different activation functions can be used. So with unbound numbers of layers and subsequent nodes, ANNs can represent the interaction of multiple factors in parallel. Utilizing the ANN to its fullest potential makes it possible to derive the maximum amount of knowledge for some mapping between the input and output without explaining the potential causation connecting the two (Grossi and Buscema 2008) .

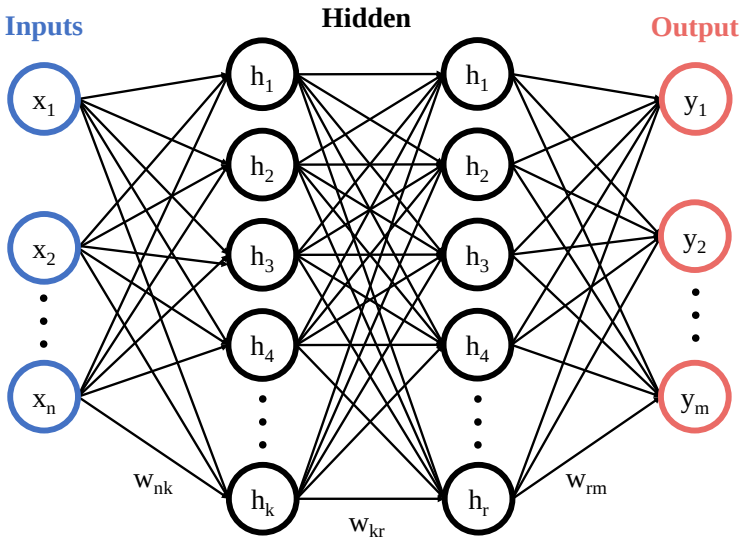


Figure 3.6: A traditional feed-forward neural network.

Backpropagation

Backpropagation, an abbreviation of backward propagation of errors, is a method for calculating the gradient of the loss function described in section 3.3 with regards to the weights in a NN. As with any machine learning model, during the training phase, NNs need to minimize the deviation between the predicted output $f(x)$ and the actual output y, i.e., a specific method for minimizing the empirical risk function for a given network. For feed-forward neural networks, this is done using backpropagation (Rumelhart et al. 1986). Backpropagation moves the opposite way of neural networks: first, the output

nodes are backpropagated, with all weights and biases adjusted for output perceptrons. The same is then done iteratively for all hidden layers until every perceptron has been backpropagated. Below, the process of backpropagating for a single perceptron will be explained.

For the first step, some loss function L is chosen (for each layer), which quantifies the difference between the desired and predicted output. The cost function C can be calculated after calculating this deviation for the training set I^t, and the cost function C can be calculated. Although the cost function is formally defined as all functions that aggregate the response from the loss function over the training set, the average of the loss function is the only method used in this paper.

The next step defines how one can reduce the cost function. This reduction requires a mapping between the change in the weights and biases and the cost function. Using the gradient of the cost function for the weights, which can be done using partial derivation and the chain rule as seen in Equation 3.9, this mapping can be extrapolated.

$$\frac{\partial C}{\partial w_i} = \frac{\partial C}{\partial f} \times \frac{\partial f}{\partial z} \times \frac{\partial z}{\partial w_i} \tag{3.9}$$

These expressions can be derived mathematically, giving a firm rule for the gradient of any cost function. For the first two expressions, $\frac{\partial C}{\partial f}$ and $\frac{\partial f}{\partial z}$, there is, however, the issue of choosing a loss function L and an activation function σ. Without the concrete functions, the partial derivatives cannot be simplified, as shown in Equation 3.10 and Equation 3.11. For the last expressions, the solved and simplified solution can be found in Equation 3.11.

$$\frac{\partial C}{\partial f} = \frac{\partial}{\partial f}\frac{1}{N}\sum_{i=1}^{N} L(x_i, f(x_i)) \tag{3.10}$$

$$\frac{\partial f}{\partial z} = \frac{\partial}{\partial z}\sigma(z) \tag{3.11}$$

$$\frac{\partial z}{\partial w_i} = \frac{\partial}{\partial w_i}\sum_{i=1}^{N}(w_i x_i + b) = x_i \tag{3.12}$$

Solving the above equations for some functions L and σ gives the cost functions C gradient with respect to w_i. The same can be shown to be possible for the bias, $\frac{\partial C}{\partial w_i}$, where the only difference is that the input is assumed to be constantly equal to 1, simplifying the expression somewhat.

Optimizers

The next step is to leverage the gradients to minimize the cost function. This optimization can be divided into two subcategories, the gradient descent optimizers, and the adaptive optimizers. The primary

defining characteristic which divides the two optimization categories is whether there is a set learning rate, as there is for the gradient descent optimizers, or if the learning rate is dynamically set throughout the training phase, as is done for the adaptive optimizers (Bottou 2012). Each will be introduced below; each is considered the best general approach within their respective categories.

The focus will be the mini-batch gradient descent for the gradient descent optimizer. The method performs gradient descent on a subset of the training set and is presented in Equation 3.13. Here, x, y is the input-output pair as previously, α is the learning rate, j is some index in the training set I^t of size N such that $0 \leq j \leq N$, and k is the size of the batch. The term θ is also introduced, which in this case is just a symbolic parameter of either w or b. The mini-batch method compromises two other gradient descent methods: stochastic gradient descent and batch gradient descent; if $j = 0$ and $k = N$, the entirety of I^t is part of the batch, while for $k = 0$, each batch is just a single input-output pair. This trade-off affects how well the method avoids local minimums, how fast the computations can be performed, how fast the method converges, and several other parameters (Bottou 2012). The learning rate, which defines how far each "step" the method takes, must also be tuned with the batch size in mind, as smaller batches are usually better paired with lower learning rates.

$$\theta = \theta - \alpha \frac{\partial C(x_{j:j+k}, y_{j:+k})}{\partial \theta} \tag{3.13}$$

As explained previously, in gradient descent algorithms, the learning rate is fixed for all the recurrent sequences, which results in slow convergence. The adaptive optimizer Adam instead has a variable learning rate for each parameter. This method will converge to a minimum of the cost function for an adequately set learning rate. In short, Adam is the combination of two other optimization functions; Root Mean Square Propagation optimization (RMSP) and Moment optimization.

Momentum optimization keeps track of the gradient at all other steps and accumulates the weighted sum of these to decide the next direction. A simplified analogy is the momentum of a ball rolling down a hill with a sudden turn. The ball would start slow, build up speed and then continue straight for some distance when the turn appears. This is expressed mathematically in Equation 3.14, where m_t is the momentum part of the Adam optimizer. The formula also includes β_1, the empirical moving average parameter. The RMSP optimizer, on the other hand, attempts to penalize changes in the parameters which cause the most oscillations in the cost function. This is done by implementing a vector of learning rates, one for each trainable parameter. This average is updated with a running average of magnitudes of squares of previous gradients, as can be seen in Equation 3.15, where v_t is the exponential average of squares along some θ.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \left[\frac{\partial C}{\partial \theta} \right] \tag{3.14}$$

$$v_t = \beta_2 m_{t-1} + (1 - \beta_2) \left[\frac{\partial C}{\partial \theta} \right]^2 \tag{3.15}$$

Each of these terms, m_t and v_t must then be bias corrected, as the moving average is biased towards 0. This can be seen in Equation 3.16 and Equation 3.17.

$$\hat{m}_t = \frac{m_t}{1 - b_1^t} \tag{3.16}$$

$$\hat{v}_t = \frac{m_t}{1 - b_2^t} \tag{3.17}$$

Finally, each of the two methods can be combined as seen in Equation 3.18. For most of the given constants, ϵ, β_1, and β_2, there are empirical values that give good generalized results. The only parameter that must be tuned is α. parenciteadam

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t} - \epsilon}\hat{m}_t \tag{3.18}$$

The Adam optimizer has been shown to improve upon both the RMSP and the momentum optimizers (Kingma and Ba 2014). The method is well tested, and for general optimization problems, Adam has a fast running time, low memory requirements, and requires only tuning for a single parameter.

3.4.3 LSTM

Long Short-Term Memory (LSTM) networks are a Recurrent Neural Network (RNN) meant to learn long-term dependencies. The RNN is itself one of the subcategories of NNs, and can be viewed as copies of ANNs executing sequentially as illustrated in Figure 3.7. For traditional RNN structure a common issue is exploding or vanishing gradients when many time steps need to be back-propagated (Hochreiter 1998).

In comparison, the LSTM cell states and memory modules, as illustrated in Figure 3.8, reduce the likelihood of exploding or vanishing. The memory modules, or gates, are responsible for what part of the data is input, what part is remembered, and what part of the data is output. The LSTM receives inputs in the figure depicted as x_t. In addition to this input, there are two types of memory or states: the hidden state y_t and the cell state c_t. The hidden state is denoted y, instead of h_t as in Figure 3.7, as it is also used as the output from the LSTM layer. There are several interpretations of what these states mean conceptually, and understanding this depends on how the model is trained and applied. The cell state can be considered as an aggregate of all previous memory. In contrast, the hidden state is usually considered the characteristic output from the previous LSTM cell and the associated input value. As the cell state c_{t-1} enters the current cell through the to, the forget gate f decides what part of the cell state is forgotten. The input gate, i, decides what is added to the cell state that continues, c_t. Finally, the output gate o decides what part of the data has to be output. (Hochreiter and Schmidhuber 1997)

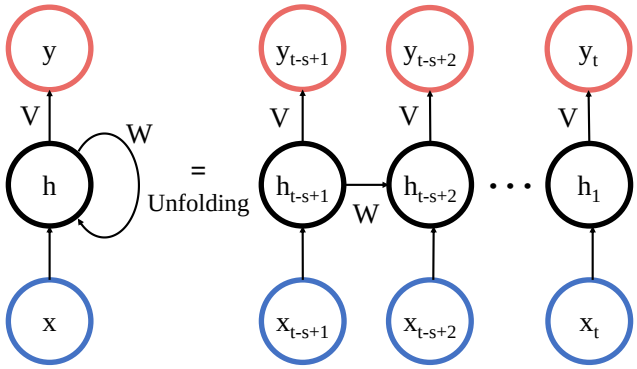


Figure 3.7: The RNN structure. Here, x is the input, y the output, h is the hidden state passed between each temporal step and W, V are weights. Adapted from Feng et al. (2017)

The gates leverage either sigmoid or tanh functions for two differing purposes. The sigmoid function squishes inputs between zero and one, which can be interpreted as either diminishing or keeping some inputs. The tanh function handles normalized encoding of the input it is given.

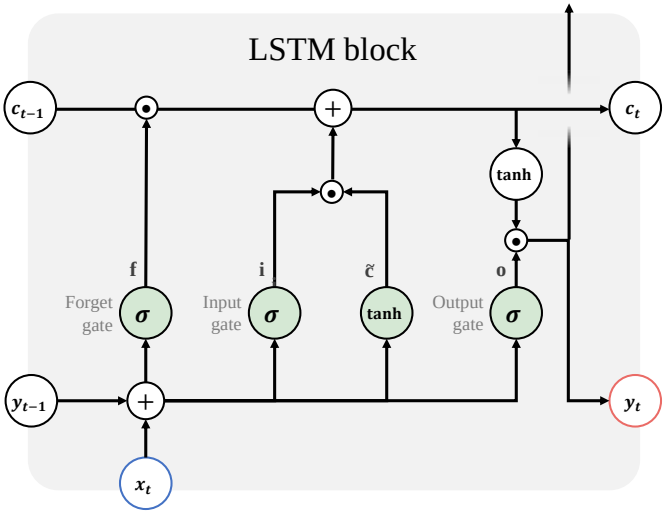


Figure 3.8: The LSTM structure. Here, the green circles denote the layer of the neural network; merging arrows denote point-wise operation; dotted circle denotes multiplication; plus denotes sum over all inputs; arrows denote concatenate; splitting arrow denotes copy. Adapted from Van Houdt et al. (2020)

Forget gate

The first gate in the LSTM architecture handles prioritization of information from the previous cell state c_{t-1}. As mentioned above, the sigmoid function diminishes parts of the state memory. The forget gate uses the hidden state from the last cell, combined with the new input x_t for this cell, to decide what part of the cell state should be removed before continuing with the calculations for the current cell.

$$f_t = \sigma\left(W_f[y_{t-1}, x_t] + b_f\right) \tag{3.19}$$

Equation 3.19 shows the forget gate f_t, where σ is the sigma function, W_f are the weights for the forget function, and b_f is the forget bias.

Input gate

The input gate is responsible for updating the cell state with the most recent data for the given block. The gate both encodes all information that should be kept further in the model and prioritizes this information. As the information is passed to the cell state, it is scaled to the given priority. Equation 3.20 and Equation 3.21 illustrate the calculations for the prioritizing and encoding, respectively, again leveraging the properties of the sigmoid and tanh functions.

$$i_t = \sigma\left(W_i[y_{t-1}, x_t] + b_i\right) \tag{3.20}$$

$$\tilde{c}_t = \tanh\left(W_c[h_{t-1}, x_t] + b_c\right) \tag{3.21}$$

For Equation 3.20 and Equation 3.21, W_i and W_c are the weights, b_i and b_c are the biases, i_t is the input gate, and $\tilde{c}_t$ is the encoded information from the hidden state and new input.

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{3.22}$$

In Equation 3.22, the forget gate is used to remove unnecessary information from the cell state while the input gate and encoded new information are added to the cell state. $\odot$ is point-wise multiplication in the equations.

Output gate

The output gate is responsible for defining how the current input should be combined with the previous hidden state. This gate functions similarly to other previously explained sigmoid gates, weighting what part of the flowing data should be prioritized and neglected as shown in Equation 3.23.

$$o_t = \sigma\left(W_o[y_{t-1}, x_t] + b_o\right) \tag{3.23}$$

$$y_t = o_t \odot \tanh(c_t) \tag{3.24}$$

Finally, the block output is calculated and passed on as the new hidden state and as the output from the LSTM layer. Here, in Equation 3.24, the cell state is used to encode the output state calculated in the previous step, completing the flow through this instance of an LSTM block.

Layer outputs

When considering a full layer of LSTM blocks, there are various ways to pass output onto the next layer or output from the entire layer. In Figure 3.8, there is the appearance of what has previously been called the output, y_t, in addition to the hidden cell state c_t. The potential outputs increase when stacking several blocks together, from time-step $t - s$ to t. Figure 3.9 shows the two most common output variants, Many-to-Many and Many-to-One. The former utilizes some subset $1 < k \leq t$ of the output states and forwards these to the next layer, the figure illustrating this for $k = t$. Many-to-one, for comparison, only forwards the last output state y_t. This can be advantageous when there are no appended layers after the LSTM, as shown in the figure, and the desire is only to predict one time-step. The hidden call state is often also returned from the LSTM layer as an added data source to improve predictions.

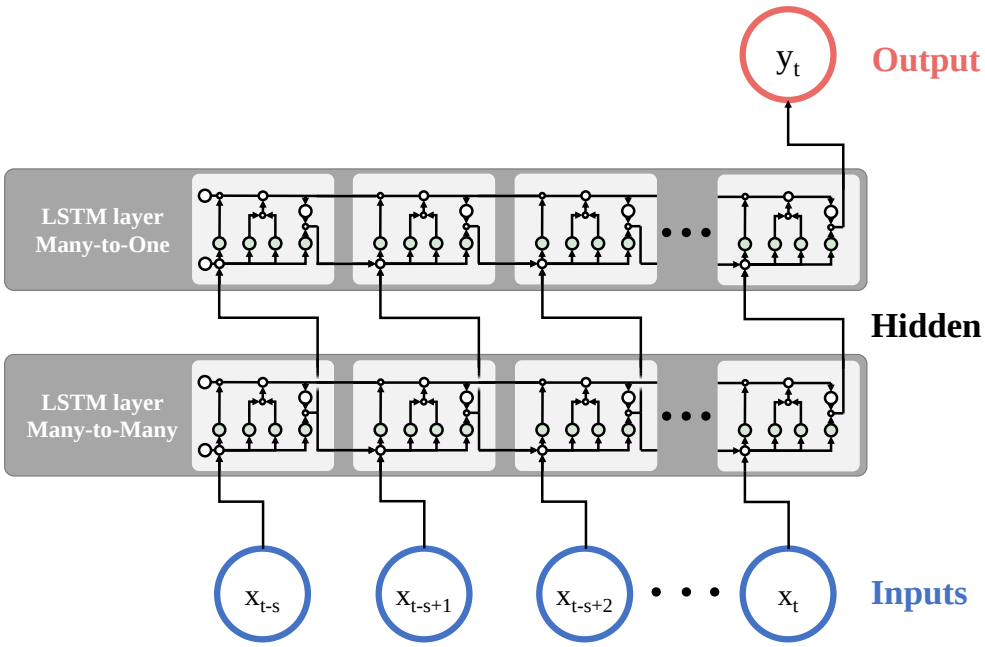


Figure 3.9: LSTM layer structures.

3.4.4 Backpropagation through time

For LSTMs, the backpropagation explained in section 3.4.2 is no longer complete. Because of the added dimension, time, LSTMs, and RNNs, in general, are both deep and ”wide”. This means one must backpropagate both backward in the deep network, and at the same time through the previous time steps. This is done by unfolding the LSTM network, which can mathematically be described as adding a time

as part of the chain rule in Equation 3.9. The loss for backpropagation through time depends on the output type of the layer, described in subsection 3.4.3. This is due to either having numerous loss metrics that must be summed for Many-to-One output, while there is only one for Many-To-One. Regardless of the output type, there are two main gradients of backpropagation through time: the output difference as computed by any subsequent layer and the output difference for each subsequent time-step, i.e., LSTM block.

3.5 Over- and under-fitting

One of the most fundamental challenges of supervised learning is how to achieve optimal fit for a model. Given a set of input-output pairs as described in section 3.1, there are countless ways of creating functions that approximate the relationship, most of which will usually not be a good fit for the actual function. Consider Figure 3.10, and the three graphs presented there. The rightmost graph is an example of over-fitting, where all the noise, measurement errors, and peculiarities of the data are considered to represent the function we are attempting to recreate. Because of this, the approximation function for the data becomes some arbitrary function that coincidentally is correct for all our input points. However, the predictions would be inaccurate given any set of new points. Over-fitting is especially common for small data sets, where the needed data to generalize properties of some domains is simply missing. Under-fitting also occurs when the data sample size is too small, or one attempts to capture complex dependencies and properties with models that cannot. An example of this is applying linear regression to capture polynomial behavior, as seen in the leftmost graph of Figure 3.10.

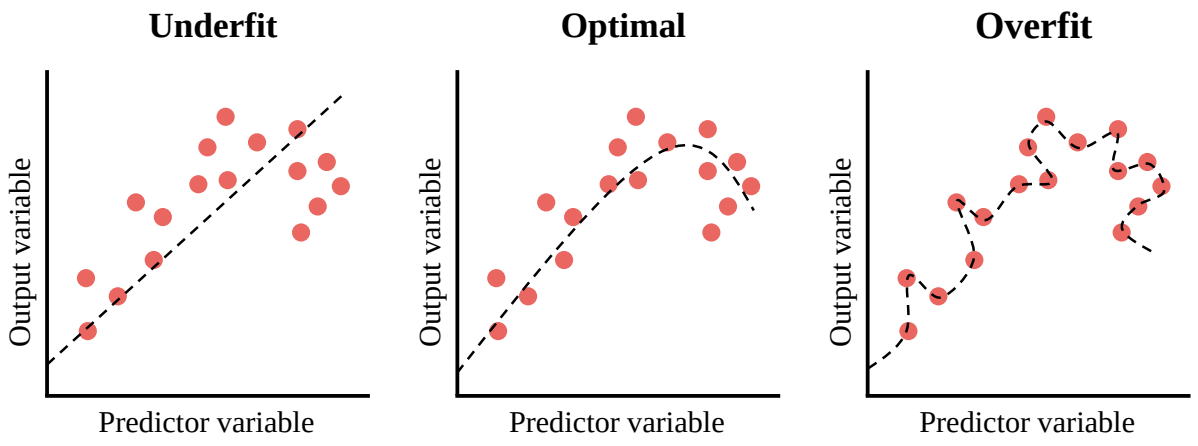


Figure 3.10: Over- and under-fitting. Adapted from Educative.io (2013)

There are a variety of approaches that, when used correctly, can help minimize the likelihood of over-and under-fitting. Methods used in this thesis is described in the following sections.

3.5.1 Early stopping

When working with real-world data in machine learning, there is a high likelihood of noise being present in the data set. Given some training data to reduce the risk of a function f, if the estimation risk function is forced to reduce to zero, then the function f is forced to represent both the desired representation and the noise present in the data (Ying 2019). This behavior is illustrated in the rightmost graph of Figure 3.10. Early stopping is one of the methods used to avoid precisely this behavior.

Figure 3.11 shows the expected errors when training a machine learning model on a data set with noise. After the initial improvement for the validation set, the model starts to represent the noise and the underlying relationship, which in turn causes the metrics on the training set to improve. In contrast, the metrics for the validation set degrade. The idea behind early stopping is to identify the turning point for the validation set and stop training at that point.

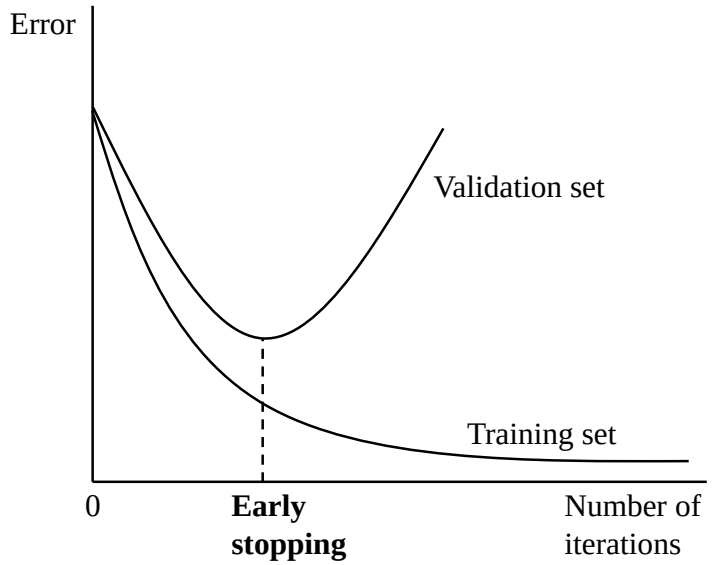
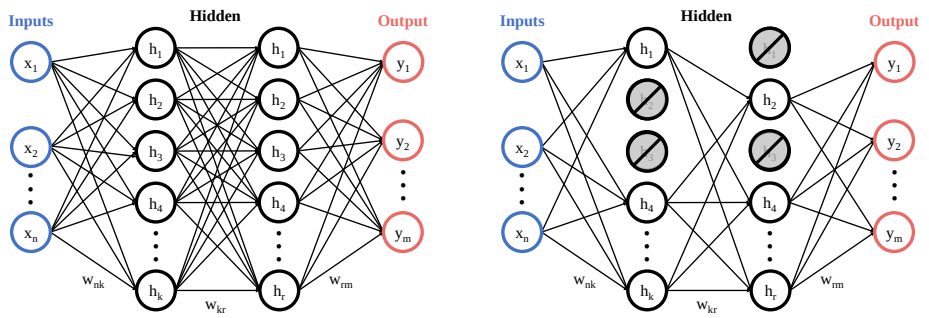


Figure 3.11: Illustration of how early stopping may improve model accuracy on unseen data. Courtesy: Gençay and Qi (2001)

3.5.2 Dropout

Another way to prevent over-fitting is using dropout, where there is a probability that each node in a neural network is ignored in the training phase. This method is a type of regularization, which is a term for all methods where some penalty is added to the loss function such that parameters that are useless gain limited control of the model. Dropout modifies the concept of learning all network weights and instead runs some fraction of weights and aggregates the results (Ying 2019). Figure 3.12 shows the difference between a regular ANN during the training phase on the left and an ANN with dropout on the right. The activated and deactivated perceptrons change each iteration, allowing all perceptrons the possibility to partake in the training process.



(a) A standard ANN structure.

(b) ANN structure after dropout.

Figure 3.12: The effect of dropout on the NN training phase. Adapted from Srivastava et al. 2014

The intuition behind dropout is based on aggregation and regularization. Ideally, when working with machine learning, each prediction done would be the aggregate of several models, which almost always improves the performance of the predictor (Srivastava et al. 2014). However, for neural nets, this is prohibitively computationally expensive. Instead, dropout is used, which samples subnetworks of the original network. and aggregates the effects of each of these smaller networks by upscaling their weights. Mathematically, the method requires defining some desired dropout rate r_d, such that r_d of the nodes in each layer is ignored. The others have their weights upscaled by $\frac{1}{1-r_d}$.

3.6 Random Decision Forest

Random Decision Forests is a supervised ensemble learning methods within machine learning. Ensemble method in this context means it utilizes multiple learning methods to improve the model's performance, obtaining better performance collectively than the methods perform on their own. The name random decision forest stems from the method consisting of multiple decision trees collectively creating a "decision forest".

Decision trees are used to replicate human decision making and have been widely applied to classification tasks and within data mining. The advantages of decision trees are their white-box properties, meaning the conclusions made are explainable and interpretable. Due to the method handling one variable at a time, there is little data preparation needed as the data set does not have to be normalized (Kotsiantis 2013).

The disadvantages of decision trees are primarily overfitting and instability. The resulting decision tree is highly dependent on the training data, and small changes in the data can result in a completely different decision tree. This also means the method does not generalize well, performance is overfitted to the training set, and often does not perform well on unseen data (Ho 1995).

Figure 3.13 show an example of a decision tree. An observation enters the top node of the tree. At each branch, a feature of the observation is evaluated against a threshold, deciding the path of the observation. This continues down the tree structure until the observation is placed in an end node, where a conclusion

about the observation is made.

Decision trees are typically constructed top-down, meaning the feature of each split is selected in the order of best performance improvement. A metric is used for evaluating the predictive performance of a split. For classification problems, Information gain and Gini Impurity are commonly used. Decision trees for regression are built using a loss function for regression problems, like RMSE, MAE, or R2. At each split in the tree, splits are selected such that the loss function is minimized.

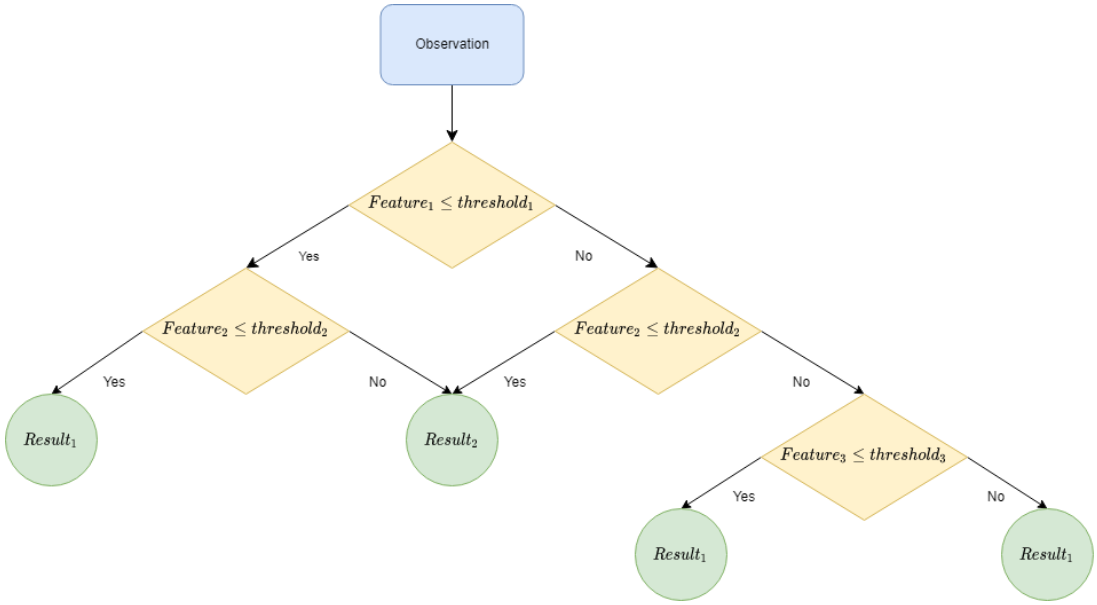


Figure 3.13: Example of decision tree

Ho (1995) first proposed Random Decision Forests for classification problems over 20 years ago. Random forest attempts to mitigate the shortcomings of singular decision trees by building multiple trees in randomly selected subspaces of the feature space. As shown in Figure 3.14 the output of the Random Forest method is a combination of all the decision trees' outputs. By creating multiple trees built with different features, the method reduces the risk of overfitting the model to the training data making the model more generalizable to unseen data. The same principles discussed by Ho (1995) are also applicable to regression problems.

Random Decision Forests are constructed using a technique called Bootstrap aggregating or bagging. Bagging selects a random sample of the training data to construct each decision tree. The amount of decision trees created is a tuning variable. Feature bagging is also applied to select a random subset of the features. These bagging techniques are utilized to ensure a wider variety of trees are constructed. If a few features dominate the predictive response, these features will be featured in most of the decision trees, causing correlated trees. Correlated trees are undesirable as it reduces the advantages of multiple trees.

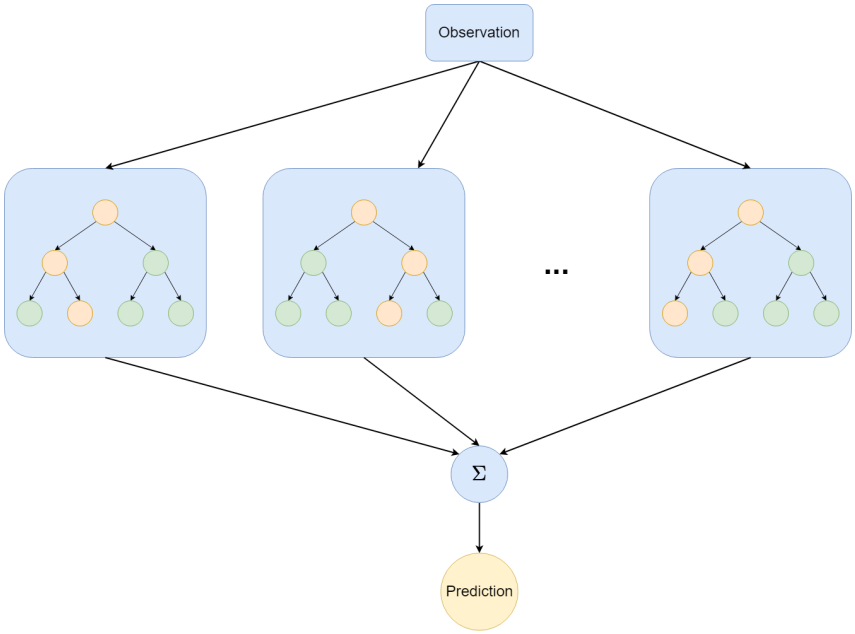


Figure 3.14: Random Forest structure

Chapter 4

Methodology

In this chapter the implementation of the digital twin framework for mooring system monitoring of floating wind turbines is presented.

The case study used to evaluate the frameworks performance is discussed and the available data and mooring configuration is described. Subsequently the physical modelling of the described mooring system in Riflex and MoorPy is shown. Then the framework for data driven modelling is elaborated on, and the process of tuning each model is discussed. Lastly the implementation of fatigue life estimation algorithms is presented.

4.1 Hywind Scotland

In October 2017, the world first floating wind farm, Hywind Scotland, started producing electricity. The wind farm resulted from a collaboration between Equinor and Masdar and was a continuation of the Hywind demo project in Norway, demonstrating the scalability of offshore floating wind. Five turbines are installed, giving Hywind Scotland a total of 30 MW capacity, powering equivalent to 20 000 homes in the UK every year since its start of production (Equinor 2022a).

Ore Catapult is a UK-based technology and research center for offshore renewable energy. Together with Equinor, they published operational data from the Hywind Scotland wind farm at their Platform for Operational Data (POD), making a wide range of sensor streams from the world's first wind farm available to the industry and academia. Thanks to Ore Catapult and Equinor, we can test our hybrid methods on operational data.

4.1.1 Mooring setup

The floating wind turbines at Hywind Scotland are moored using a spread catenary configuration. The wind park is anchored at 100m depth. Figure 4.2 shows how each unit is connected to 3 mooring lines

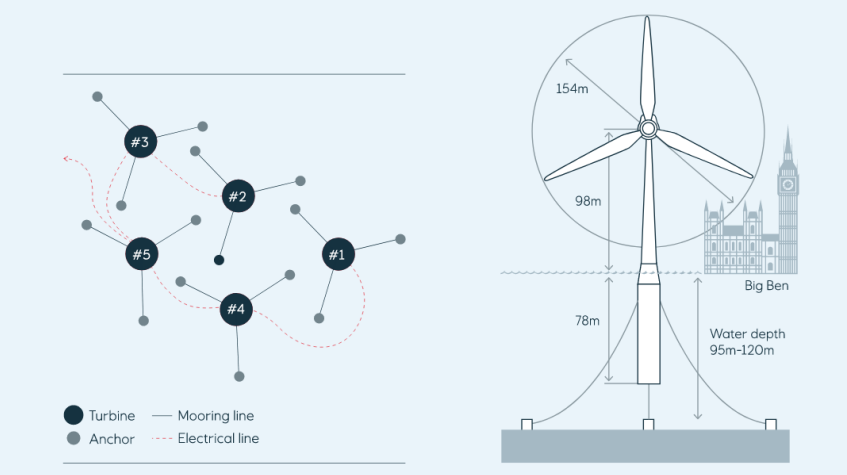


Figure 4.1: Hywind Scotland wind farm. Courtesy: Equinor (2022a)

anchored to the seabed. Suction anchors are used to fix the lower end of the mooring lines to the seabed. At the wind farm, multiple FOWTs may share seabed anchors. The upper 50 meters of the mooring lines are split using a bridle, giving the three mooring lines six connection points to the FOWT. The mooring lines are made out of steel stud-less chains. For short the 6 lines will be referred to by the line number and bridle number, LxBy, corresponds to Line x and Bridle y etc.

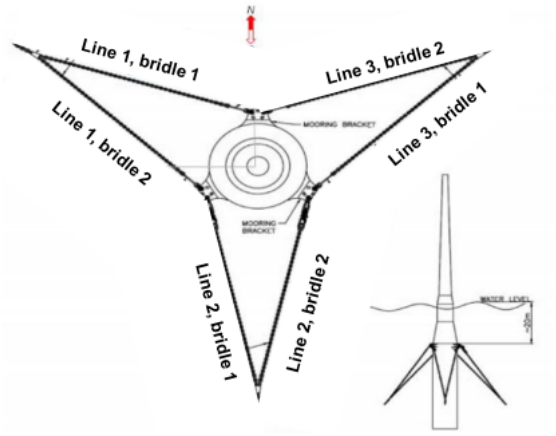


Figure 4.2: Mooring configuration

4.1.2 Sensor package

The turbines at Hywind Scotland are outfitted with a sensor suite to monitor the floating structure's behavior and dynamics. All sensor measurements available from the turbine is summarized in Table 4.1 and the sensors locations are shown in Figure 4.3. Except for heave, all movements of the floater are measured. Missing heave movement must be compensated for in the modelling approaches. The effect of missing heave measurement will be discussed later in the thesis.

The dGPS sensor measurements are converted from longitude-latitude coordinates to a measurement

of drift off in latitude and longitude direction. The zero-offset position is selected as the installation coordinates for the given FOWT. Due to the dGPS being installed 17m above sea level, some of the translational movements may be caused by rotational movements in roll and pitch. These movements will be corrected for in the physical modelling.

Sensor	Measurement	Unit
dGPS at tower	Drift of in north and east direction	m
MRU at tower	Roll, pitch, and yaw angle	deg
MRU at nacelle	Roll and pitch angle	deg
Anemometer	Wind speed	m/s
Rotor controller	Orientation of rotor (wind direction)	deg
Load cell	Tension at upper end of each bridle	kN

Table 4.1: Sensor suite at turbine

The environmental conditions for the area are also provided in the data set published at Catapult's POD. The wave and current conditions are measured from an onsite wave buoy, while the conditions for wind are aggregated from the anemometers mounted on the wind turbines. The available metocean data is summarized in Table 4.2.

Measurement	Unit
Significant wave height	m
Peak Wave period	s
Wave direction	deg
Mean wind	m/s
Wind direction	deg
Current at 4 depths	m/s
Current direction	deg

Table 4.2: Metocean data collected per case

Coordinate system

The coordinate system used for the measurements is x in the eastern direction, while y points north. The labels on rotation are somewhat unconventional, where roll is defined about the north (y) axis, while pitch is defined about the east (x) axis. All rotations have a positive direction with the clock. Figure 4.4 shows the coordinate system. Wind and wave direction is defined as the direction the wind is coming from, while the current direction is defined as the direction it is going towards. The same coordinate system is used in the physics-based simulations to avoid confusion and errors.

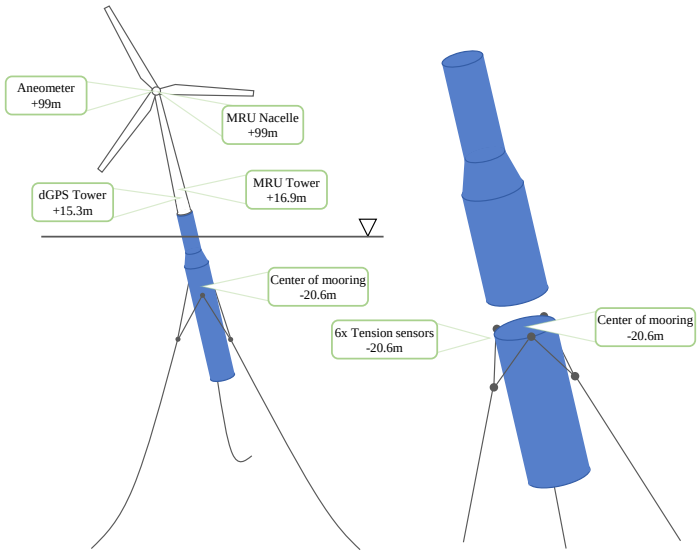


Figure 4.3: Diagram showing the Hywind Scotland FOWT, and its respective sensors.

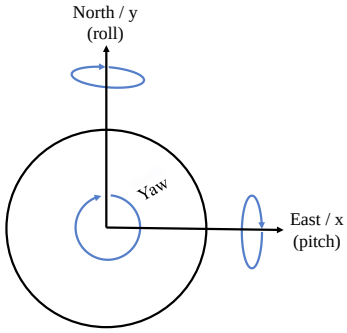


Figure 4.4: Coordinate system used for all features.

4.2 Data exploration

Data exploration is the first step of data analysis when working with complex data sets. Understanding the the quality, quantity, and completeness of the data available, and potentially understanding the relationships between the different features are valuable for the continued use of the data set. The data set consists of 11 cases of 30min intervals of data from the first half of 2018. Nine (9) of the 11 cases are from the regular operation, while 2 cases are from high wind conditions where the turbine is in idle mode. The two idle cases of the 24th of January 2018 overlap by 10 minutes and are therefore combined into a 40min long case. This combination prevents data from being used twice in the overlapping cases.

Data visualization is a valuable tool for data exploration. By looking at the distribution of data patterns, incomplete subsets may be discovered. Plotting the features against each other gives insight into their relationships and interactions. Figure 1 in Appendix A shows some of the features plotted against each other in the lower triangle. The diagonal of the same figure shows the distribution of each feature. The

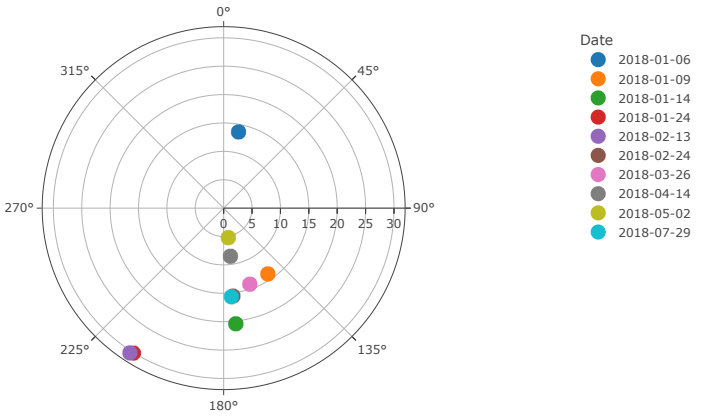


Figure 4.5: Distribution of mean wind speed [m/s] and direction in cases

color represents the dates. In Figure 1 it is clear that the idle cases (in red) and the case of the 6th of January (in dark blue) are significantly different from the rest of the cases. This difference is due to the wind conditions, where Figure 4.5 shows that the 6th of January is the only case with a northerly wind, while the rest of the cases have southerly winds. In addition, the idle cases have significantly higher wind speeds than the operational cases. Differences in environmental conditions are expected to influence the results of the hybrid methods, especially when the outlier cases are not part of the training set.

An interesting artefact found from the data exploration was the distribution of the Tower yaw angle, as shown in Figure 4.6. We found it peculiar that in all cases before April, the measurement of floater yaw angle has a non-zero mean for all cases. This causes the mean yaw position to be approximately 9 degrees off. After contacting the data distributor, we found out the yaw measurement calibration was incorrect. The offset in the yaw angle is corrected by assuming a zero mean yaw angle for the period from January to April. Zero mean yaw angle is a reasonable assumption for a circular hull with the bridle mooring configuration, as shown in Figure 4.6 from April to July.

Hywind Scotland is located east of the Scottish mainland. Prior to the installation of the wind farm, a thorough analysis of the metocean conditions at the site was performed. Figure 4.7a shows the probability contours for return periods of H_s-T_p pairs at the Buchan Deep. These contour lines are based on over 50 years of data from the Norwegian Meteorological Institute. On the other hand, Figure 4.7b shows the H_s-T_p pairs present in the data set used for this thesis. As the figures show, the span of wave conditions in the data is relatively narrow compared to the one-year return period, indicating that more data is needed to get a stable model for all expected sea conditions of the wind farm.

Due to the limited variance of weather conditions in the data set, the spread of floater positions are not as well distributed as would be desirable to acquire a more robust model. Figure 4.8 shows the distribution of recorded latitude and longitude measurements. As expected from the distribution of wind directions shown in Figure 4.5 pushes the floater north. With only one case of northern wind, the number of positions with negative latitude drift-off is limited.

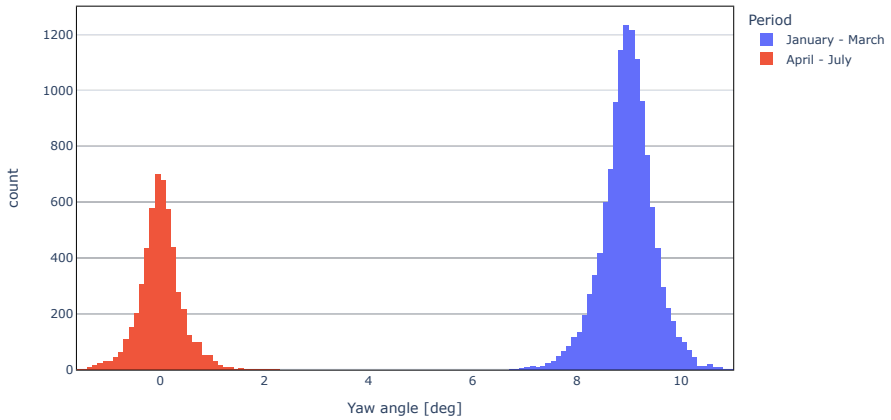


Figure 4.6: Histogram of yaw-angle measurements for all cases. Cases before april 2018 have a mean yaw angle of 9 degrees, from april and out the yaw angle have a mean of 0 degrees.

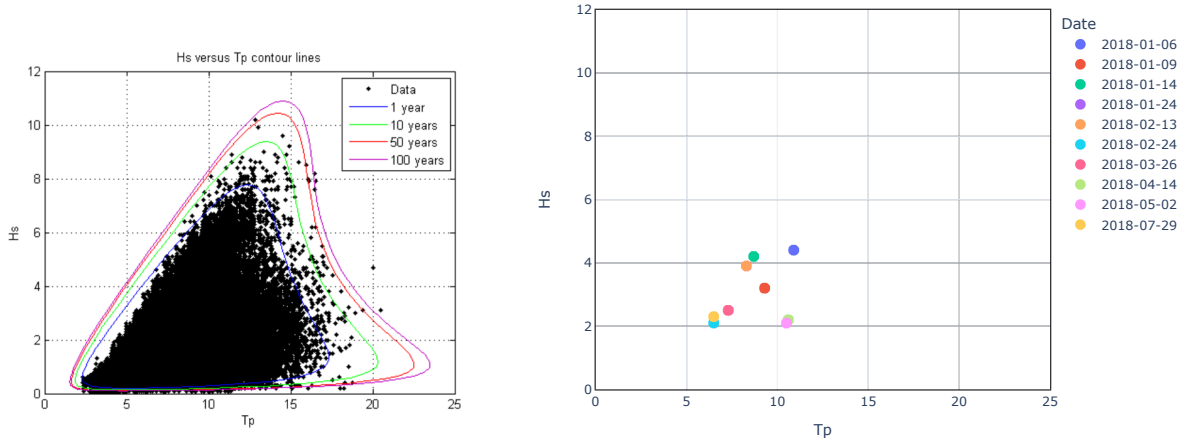
STD	L1B1	L1B2	L2B1	L2B2	L3B1	L3B2
mean	14.68	14.68	25.55	26.86	17.13	19.03
25%	5.77	5.77	8.54	12.95	6.20	7.14
50%	12.43	12.43	19.97	23.96	14.30	16.41
75%	21.17	21.17	38.95	37.68	25.26	28.36
max	84.55	84.55	125.61	141.23	79.00	91.99

Table 4.3: Measured tension standard deviation, in kN, for similar FOWT positioning. Requirement for similarity: < 0.1 meters deviation in the horisontal plane and < 0.1 degrees for pitch, roll and yaw. Mean is the mean STD for all sets of similar points for each line, 25% is the boundary for the 25% lowest STDs for all sets of points, etc.

Quality of sensor measurements

For the tension measurements, it is desirable to grasp how much noise is present for the position of the FOWT. Given that no dynamic analysis with current and wave loads will be performed for the final model, having a quantifiable range for the noise this and other factors contribute is important for how the results are evaluated later. One possible way to quantify this noise is by comparing the tension values for points with similar positions. The physical part of the hybrid modelling relies solely on position, meaning deviations for similar positions also estimate how accurate the physical analysis is. Table 4.3 contains this estimate. The values in the table are calculated by defining all positions within the boundary given for similarity and then calculating the standard deviation for each set. The mean value, and some boundary values, for the STD are then presented to show the distribution of tension.

Table 4.3 shows that the most significant deviation is in L2B1 and L2B2, which is reasonable given that these are the bridles with the highest average tension. The mean values are also within reason; they are well below one-tenth of the measured tension for each line. Given measurement errors in the different sets of sensors, some deviations are expected. What is challenging is the max deviations. It is clear that some similar positions have significantly different tensions



(a) Probability contour lines of H_s-T_p pairs for return periods of 1,10, 50, and 100 years for the location of Hywind Scotland. Courtesy: Mathiesen et al. (2014)

(b) H_s-T_p pairs in the operational data set

Figure 4.7: Expected sea states vs. Captured sea states

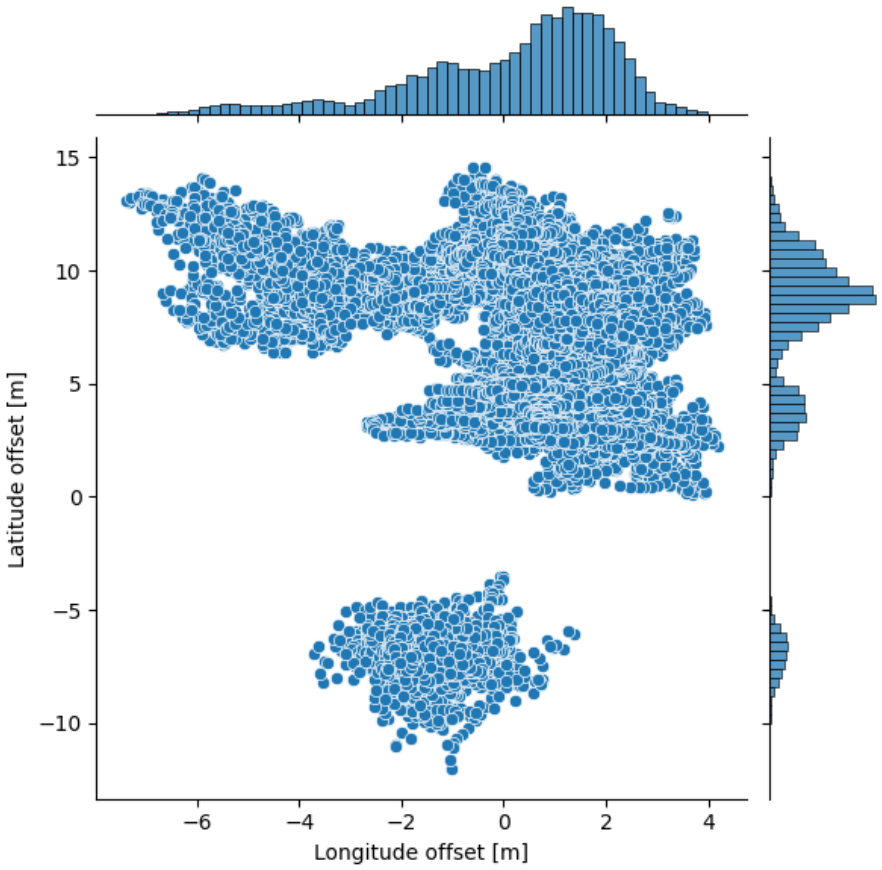


Figure 4.8: Distribution of FOWT positions recorded

4.3 Hybrid Modelling

Input	Target
Latitude driftoff	PBM deviation L1B1
Longitude driftoff	PBM deviation L1B2
Tower roll angle	PBM deviation L2B1
Tower pitch angle	PBM deviation L2B2
Tower yaw angle	PBM deviation L3B1
Wind speed in x direction	PBM deviation L3B2
Wind speed in y direction	
Metocean data (optional)	
PBM estimate L1B1-L3B2	

Table 4.4: Input and target features for Hybrid model A

Input	Target
Latitude driftoff	Measured tension L1B1
Longitude driftoff	Measured tension L1B2
Tower roll angle	Measured tension L2B1
Tower pitch angle	Measured tension L2B2
Tower yaw angle	Measured tension L3B1
Wind speed in x direction	Measured tension L3B2
Wind speed in y direction	
Metocean data (optional)	
PBM estimate L1B1 - L3B2	

Table 4.5: Input and target features for Hybrid model B

The motivation for this thesis is to provide a proof of concept for hybrid modelling for mooring system monitoring of floating wind turbines. The amount of historical operational data available is very limited, the implementation of the monitoring system in this thesis deviates from a solution running in real-time with a continuous stream of input data.

In this thesis, two hybrid modelling approaches are implemented. Both hybrid methods contain the same workflows; the main difference is what the target features of the DDM are and how the data flows between the models.

Hybrid model A, shown in Figure 4.9, utilizes the data stream from the wind turbine as inputs in both the PBM and the DDM, and the DDM is trained to predict the deviation from the sensor data and the PBM, the results from both models are combined into the resulting tension prediction.

Figure 4.10 shows the alternative hybrid model implemented. In this setup, the DDM is trained to estimate the line tension with the PBM results as input features and the turbine measurements.

The hybrid methodology and its subsystems are implemented in Python 3.9 (Van Rossum and Drake 2009). The main workflow of the python application is described in Figure 4.11. The data is collected from Ore Catapult, then ingested into our data preprocessing pipeline, it then controls the workflows of the PBM and the DDM, according to the hybrid methodology applied. The module for data-driven

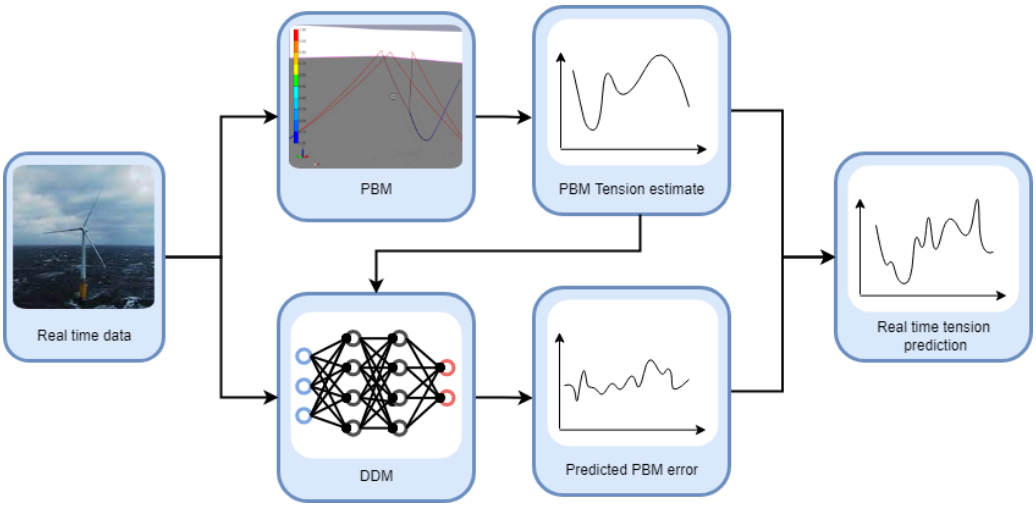


Figure 4.9: Hybrid model A

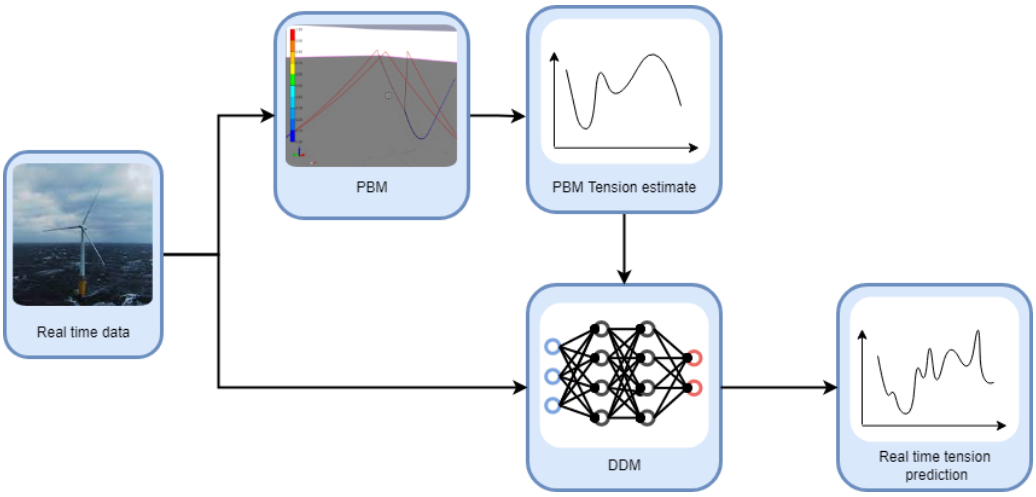


Figure 4.10: Hybrid model B

modelling can be changed from Random Forest Regressor, LSTM, and ANN by altering only one line of code.

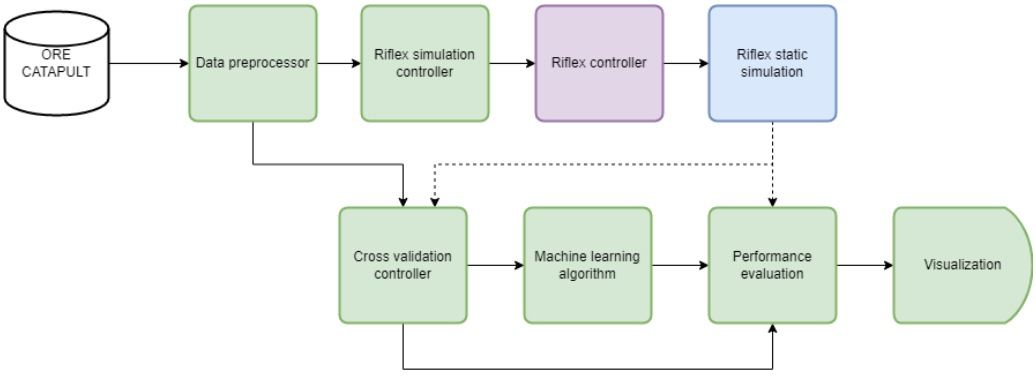


Figure 4.11: Process overview flow chart. Green blocks are python modules, purple are bash scripts, and blue is the Riflex module.

4.4 Data pre-proccesing

4.4.1 Sampling

The operational data available from the sensor package described in subsection 4.1.2 has different sampling frequencies, presented in Table 4.6. Non-matching sampling time creates a range of problems for the data pipeline. As neither the physical models nor the data-driven models used in this thesis are capable of handling data with different sampling frequencies, the data must be resampled.

The advantage of upsampling the data to a higher frequency is the ability to utilize the complete fidelity of the high-frequency sensors. At a sampling frequency of 5Hz, every tension measurement is utilized in the model, providing a higher resolution of the variations in tension. The disadvantages are the higher cost of training and calibration due to more data points included in the model and the fact that all lower frequency data is used multiple times. At 5Hz, every dGPS measurement is used five times. Techniques like interpolation can mitigate these issues, but they assume a linear relationship between the position and tension, which may not be accurate.

By downsampling the data instead, the higher frequency data streams are filtered by aggregating to the mean of the samples within the new sampling period, reducing the noise of the measurement.

Sensor	**Sampling frequency [Hz]**
dGPS	1
Tension	5
Tower MRU	10
Nacelle MRU	10
Windspeed	0.95
Tower yaw angle	5
Metocean data	Constant within case

Table 4.6: Mean sampling frequency for sensors at Hywind Scotland

4.4.2 Merging data sets

The sensor streams are not synchronized to provide measurements simultaneously. This may cause inaccuracies for the model, especially in offsets between load cell data and the GPS/IMU data, which may further result in a tension measurement being mapped to the wrong turbine position. In practice, the delay between the measurements has a negligible effect on this thesis. With the data being downsampled to 1 Hz, the effect of the approximate 100 ms offset in sampling timestamps is small.

4.4.3 Feature engineering

Feature engineering is applying domain knowledge to transform raw data into features for machine learning. Feature engineering attempts to create better features to help machine learning models to better extract relationships and knowledge from the data.

The data from the turbine's anemometer decomposed from wind direction and wind speed into a north / east wind component. This way, the ML model does not have to learn the relationship between wind direction and floater response and remove the jump in directional data between 360 degrees and 0 degrees.

Instead of using latitude and longitude measurements directly from the dGPS in the model, a transformation into a local frame is performed, providing features measuring the offset in meters from the equilibrium position. also known as the field zero point, of the turbine.

The zero offset position in dGPS measurements refers to the intended installation location of the FOWT. If this is not equal to the mooring field zero position, the offset in latitude and longitude need to be calibrated such that the origin of the offset is in the field zero point. We investigated the physical model's sensitivity to moving the zero offset position. To investigate this phenomenon, we utilized methods from the field of design of experiments (DoE). DoE is the planning of systematic variation of controllable, independent variables to induce a response in the dependent variables. The experiments are planned such that the maximum amount of information is gained from the minimum amount of effort. Design Expert 13 (Stat-Ease 2022) is used to build and evaluate the experiment setup shown in Figure 4.12. This design is optimal with 30 experimental runs, and it is optimized to create an accurate response surface.

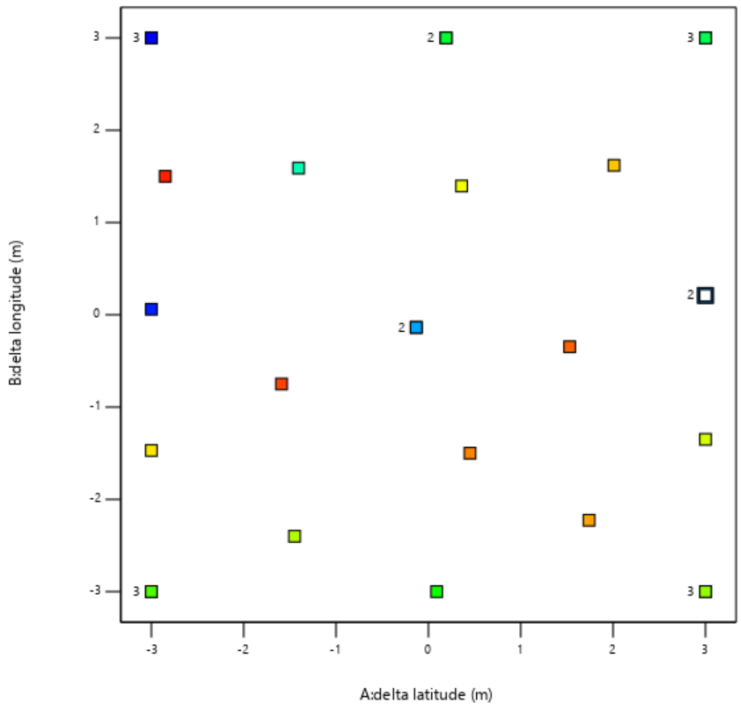


Figure 4.12: Experimental setup to find mooring equilibrium position. The points show experiments where the zero offset position is shifted north/east.

The responses discussed in subsection 3.3.1 (R2, RMSE) is used to evaluate the experiments. A sample of 2000 data points from the data set is selected randomly to perform the experiments. The statistical performance indices are found from the physical model and the tension measurements, meaning we try to find the zero offset position which optimizes the physical model. The number of samples used to optimize the physical model is a trade-off; too few samples and the results are too sensitive to noisy measurements, too many samples included cause very high computational cost and can also risk overfitting the physical model. Ideally, a completely separate set of data should be used to optimize the zero offset position. However, due to the limited amount of data available, we used a small percentage of the data (5%) to optimize the physical model.

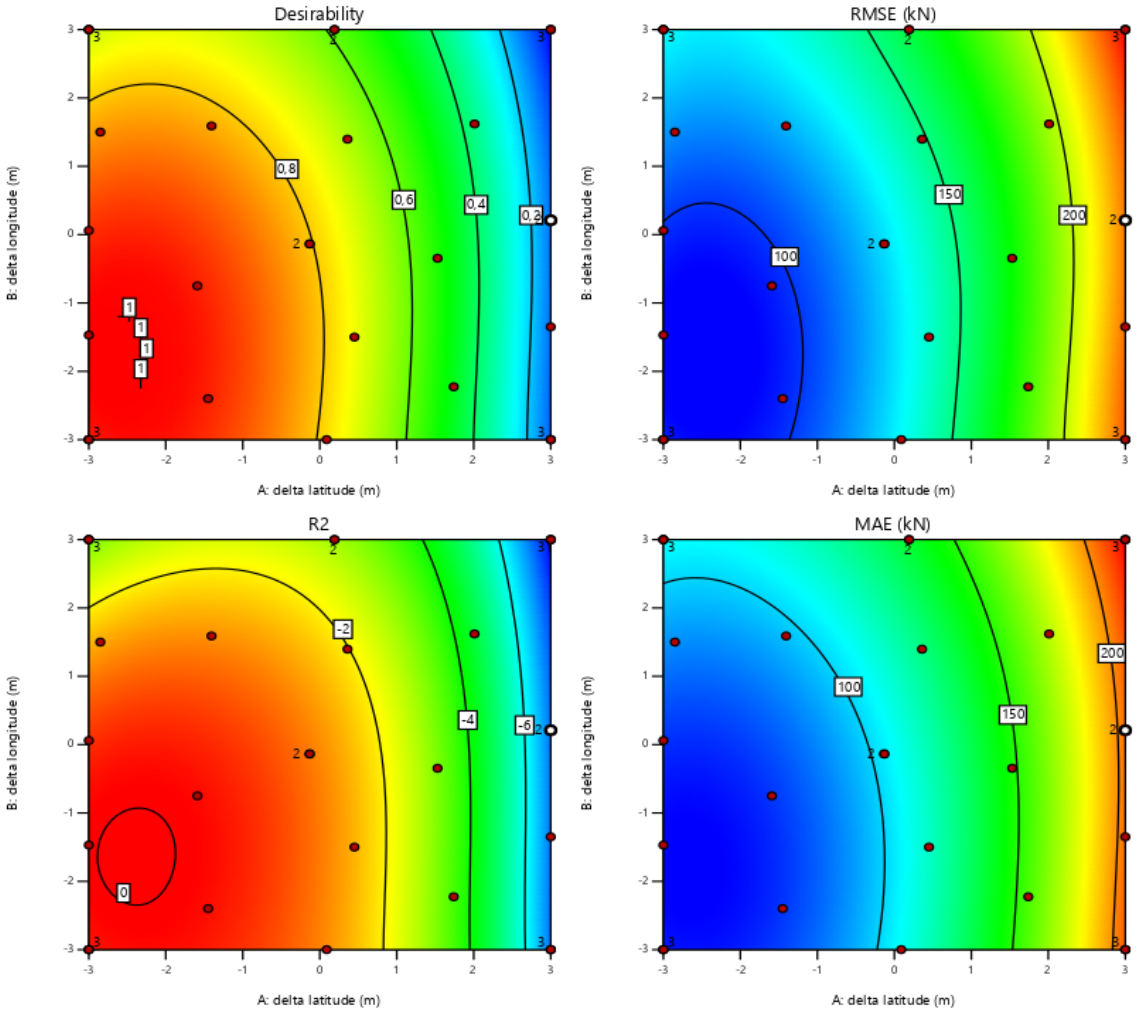


Figure 4.13: Results from design of experiments. Red points are experiments performed. Desirability define the areas where RMSE and MAE is minimized, while R2 is maximized.

A cubic model is fitted to the responses of the experiments, results of this can be seen in Figure 4.13. The best performance is achieved in the area around $(-2, -2.5)$ where the RMSE between the physical model and the tension measurements is minimum, and the coefficient of determination (R2) is maximum. As a result, the dGPS measurements used in this thesis are shifted 2m west and 2.5m south from the original data set.

4.5 Physical modelling

The following section will introduce the concepts and processes needed to produce a physical model of the system. Two main tools will be presented, MoorPy and Riflex, their respective implementations, and a general workflow for the PBM.

4.5.1 RIFLEX

RIFLEX is a tool for the analysis of slender marine structures. This includes static and dynamic analysis of risers, mooring lines, wind turbines, and umbilicals. Both Catenary analysis and fully nonlinear Finite Element Analysis are available for static analysis. For dynamic analysis, there is eigenvalue analysis and Dynamic time domain analysis. RIFLEX may be run as a standalone but is commonly used in the simulation workbench SIMA, providing a graphical user interface. For the application of real-time condition monitoring of mooring systems, computational cost is a limiting factor; therefore, static analysis is focused on in this thesis.

4.5.2 MoorPy

In addition to implementing RIFLEX as the primary physical model, some time was also spent adopting the MoorPy framework from Hall et al. (2021). MoorPy is a python package for quasi-static mooring analysis, utilizing catenary equations for effective computations. The motivation behind implementing a second simulation tool was three-fold: Firstly, the needed input and workflow for MoorPy are almost identical to that of RIFLEX, and as such the extra development time was reasonable. Secondly, vastly differing results would likely mean there was an error in implementing either model. Finally, as MoorPy only solves the catenary equations, it is significantly faster than Riflex. It can be a valuable tool for investigating the different properties of the mooring system. More specifically, MoorPy was an integral part of the tuning of zero offset position as presented in subsection 4.4.3.

Because of the similar implementations for the MoorPy framework and RIFLEX, MoorPy will not be presented in-depth. The framework uses the same node positions file as RIFLEX, and passes the results on identically.

Approach

The goal is to obtain an accurate estimate grounded in physical principles of the tension in the mooring lines for each time-step. For each time-step, the sensor data explained earlier is the available input. The tension in each of the six bridles is the desired output. RIFLEX provides a mapping between the turbine position and line tension.

For physical analyses such as these, some trade-off must be made between performance and precision.

Real-time calculations usually require simplifications of the given system or precalculations. One possible solution for this is precalculating a reasonable number of positions within the working area for the FOWT. The resolution of these positions would then decide the number of calculations needed before running the system, with interpolations covering the rest of the domain. However, given that the MRU and the dGPS together provide displacements in 5 DOF, the precalculated database of tension would be pretty sizable.

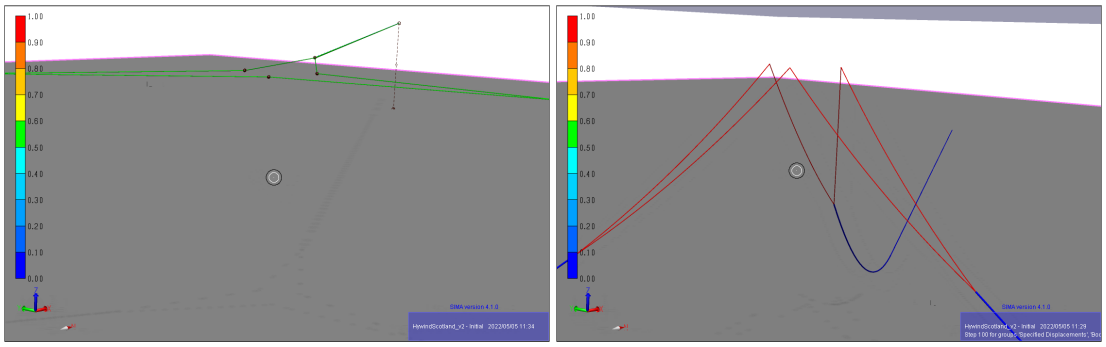
Another option is to do concurrent calculations of each time-step. One standard CPU can estimate tensions for a time-step in 10 seconds, then 10 concurrent units can keep up with real-time data at only a 10 seconds delay. This delay is well within the bounds of reason for service life monitoring. Instead of needing interpolations, this approach would also calculate the tensions at the exact position measured. It is significantly more computationally heavy real-time, but also more robust as this method would not be dependent on inputs within a given area.

For this thesis, the assumption is that the second option is the most feasible. The tensions are calculated separately for every single available position for the mooring system. This is done at one-second intervals, as this is the lowest sampling frequency for the necessary sensors, as given in Table 4.6. The workflow is then to take the measurements of the IMU and dGPS, use these to calculate the position of mooring connection points, and finally calculate the tension in each bridle at that given position. This workflow will be discussed more in-depth below.

4.5.3 Mooring system model

As the node positions at the fairlead of the wind turbine is calculated from measured data, no coupled analysis is needed to figure out the floater response. We assume the environmental conditions are reflected in the sensor measurements and therefore the modelling of the Turbine and its substructure is not needed. Instead, the mooring system can be defined with only the following components: supernodes, lines, cross sections, and seafloor contacts. In addition, the sea surface and seafloor must be defined by the environment.

- **Supernodes:** There are two main types of supernodes, constrained and free. Free nodes are free to be displaced in 6DOF, while constrained nodes can be either constrained in all 6DOF or have free rotation. Translational constrained nodes are used for the supernodes that model the connection between the mooring lines and the FOWT floater, and the connection between the mooring lines and the suction anchors. For the connection point between the bridles, a free node is used. This method investigates the mooring tension for a given position of the floater but without making any assumptions about how the mooring lines would fall in this given position.
- **Lines:** For the modelling of mooring lines, the leading property is their initial line length. Each line has a firm line length, which matches up with the length between the supernodes that the lines connect.



(a) RIFLEX model before simulation. **(b)** RIFLEX model on final step of static simulation.

Figure 4.14: Comparison of the mooring lines before and after static analysis

- **Cross sections:** For the Hywind Scotland mooring system both the bridles and the main lines are stud-less chains. The section from anchor to bridle is of a higher diameter than the section from bridle to substructure, therefore have to define two separate cross-sections for the model. The cross-sections are defined by their respective mass coefficient, cross-sectional area, and axial stiffness.
- **Seafloor contacts:** For the model to interact with the seafloor properly, it needs to define normal, in-plane, and lateral friction forces between the seafloor and the mooring lines.

With the above properties defined, the foundation for the analysis is in place. Several calculation parameters must then be decided, primarily the loading sequence. This sequence decides which loads are calculated and the number of iterations that should be performed for these loads. For the static analysis without any environmental conditions, the applied loads are volume forces, specific displacements, body forces, and bottom friction forces. With the above-explained model, and some desired node positions, RIFLEX can calculate the tension profile of each of the mooring lines and bridles. Figure 4.14 shows the initial RIFLEX model before static analysis and after the analysis at the final step of the simulation.

4.5.4 From sensor to supernodes

The RIFLEX model requires positional input for each supernode to calculate the bridle tensions correctly. There is a difference between the initial and final position for constrained nodes to let the mooring lines iteratively fall into their final position from the initial position. Some of these inputs are constant, like the final position of the suction anchors, while the rest are dependent on the desired position of the floater. The final and initial positions of the floater-connected super nodes are equal, the bridle connected super nodes are free and as such have only initial positions, and so the only nodes with both initial and final positions are the anchor nodes. Both the initial and final positions are illustrated in Figure 4.15, as well as some of the denotations that will be used below.

The desire is for the RIFLEX model to, as closely as possible, replicate the behavior of the actual floater and the tension in its mooring lines. This is dependent on a mapping between the measured sensor

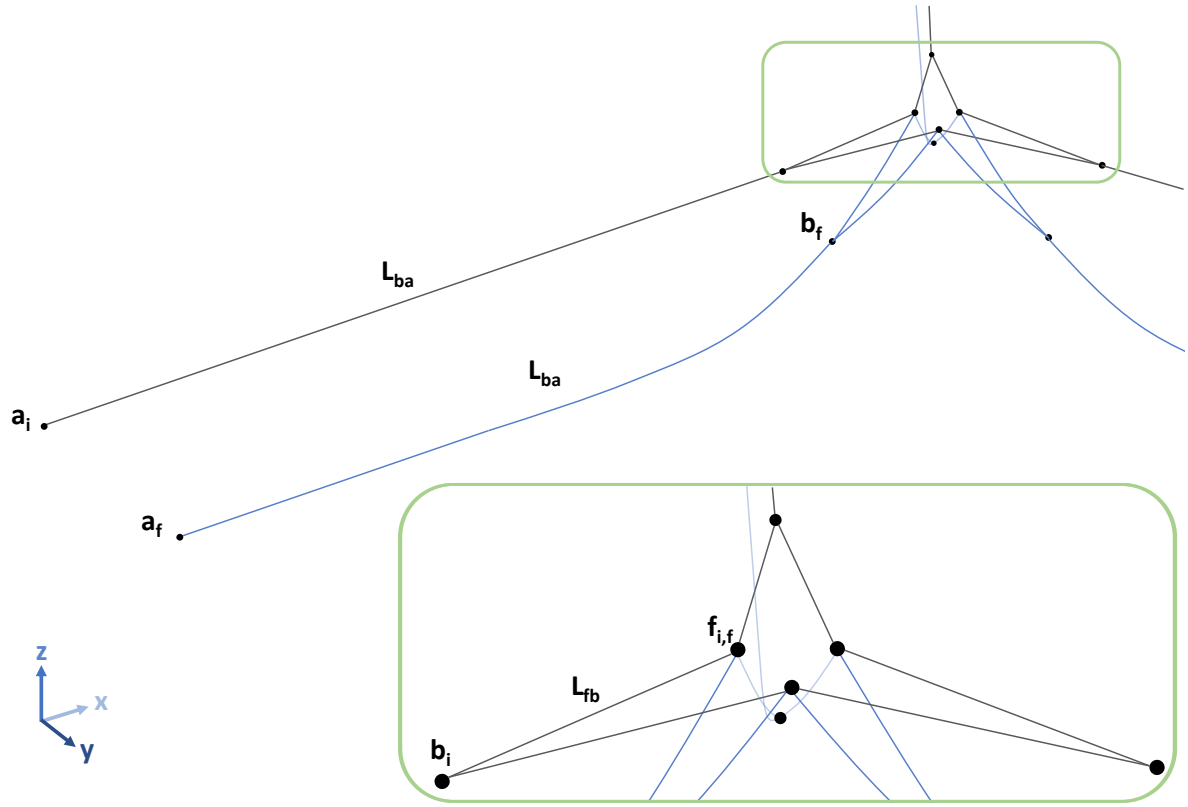


Figure 4.15: Diagram showing the geometrics of the mooring system. The blue lines are the final positions of the system, while the black lines are the initial positions. The supernodes a, b, and f, are denoted with i if it is in the initial position and f if it is the final position. Supernode f is denoted i, j as it has an equal initial and final position. The lines and nodes are iterated from the intial position into the final position, with $f_{i,j}$ and a_f being known, and the rest of the supernodes unknown or calculated in subsection 4.5.4.

data and the position of all the supernodes. The correlation between sensors and nodes is illustrated in Figure 4.3. There is a set distance between the mooring center and the dGPS sensor. This distance, in combination with the measured pitch and roll from the MRU, makes it possible to estimate the desired mapping, as shown in Equation 4.1.

$$p_c = \begin{bmatrix} x_s - L_{cs}sin(\Phi) \\ y_s + L_{cs}sin(\theta) \\ D_c \end{bmatrix} \tag{4.1}$$

where symbols denoted with subscript c are in the mooring center, symbols denoted with subscript s is in the dGPS sensor position, θ is rotation in the pitch direction, Φ is rotation in the roll direction, L_{cs} is the length between the mooring center and the dGPS position, and D_c is the constant debt of the mooring center. The debt of the mooring center is considered constant as there are no measurements in heave, z, direction. When the mooring center is found, the next step is to calculate the position of the three supernodes connecting the bridles to the floater. For each of these supernodes, f_i, their respective positions compared to some origo is known, p_{0i}. By rotating these positions with the measured rotations and placing them relative to the mooring center, all the connected supernode positions can be calculated

as shown in Equation 4.2

$$f_i = R_z(\psi)R_y(\Phi)R_x(\theta)p_{0i} + p_c \tag{4.2}$$

The placement is more arbitrary for the anchor nodes and the bridle nodes, as they will be iterated from their initial position into the final position. This final position is, as mentioned, known for the anchor node and irrelevant for the bridle node. However, there are two main constraints for the nodes: Firstly, The Euclidian distance between two nodes must be equal to the length of the line segment modeled between the nodes. This means that between a floater and bridle supernode, the Euclidian distance must be equal to the length of a bridle and likewise between other nodes. Secondly, the placement of the nodes should be such that, when RIFLEX iteratively lets nodes "fall" between their initial and final position, the movement should primarily be vertical. The easiest way of doing this is constraining connected bridle and anchor supernodes to lay on a line parallel to the seafloor.

Mathematically, the position is calcated as follows. For two floater nodes fi and fj, calculate the vector between them:

$$\vec{IJ} = f_j - f_i \tag{4.3}$$

Then, we assume that the bridle node, b_i connected to both these floater nodes, has a position in the z-direction equal to the average of its two nearest floaters for simplicity. This is reasonable as the position in z is primarily arbitrary and has little impact on the final calculations. The bridle node can then be viewed as the final corner of a bilateral triangle, with the floater nodes making up the two other corners. The length of the sides in the triangle is known and denoted L_{ff} for the distance between two connected floater nodes and L_{fb} for the distance between a connected floater and bridle node. The position of the bridle node is then simple geometrics:

$$b_i = \frac{\vec{IJ} \times \hat{e}_z}{\|\vec{IJ} \times \hat{e}_z\|}\sqrt{L_{fb}^2 - (L_{ff}/2)^2} + \frac{1}{2}(f_j + f_i)$$

Where $\hat{e}_z$ is the unit vector in z-direction. The above equation calculates the midpoint of $\vec{IJ}$ and then calculates a vector out from the plane defined by $\vec{IJ}$ and $\hat{e}_z$, i.e., a vector normal both to $\vec{IJ}$ and parallel with the xy plane. This vector is then placed in the midpoint, and stretched to be the height of the aforementioned triangle.

Finally, the anchor nodes detonated a. The final positions for the anchors at the seafloor are known and are used to find the initial anchor node positions. We can define a vector between the two points by defining a point directly above the final anchor node position, a_i0, along the xy plane with the same

depth as the respective bridle node. This vector is placed in b_i, and set to the constant length of mooring lines L_{ba}, as seen in the following equation:

$$a_i = \frac{a_{i0} - b_i}{\|a_{i0} - b_i\|} L_{ba} + b_i$$

With the above step, the node placement procedure is complete. It is important to note that the method is only valid with the assumption that the tension in lines placed in some position is the same as when the lines are pulled out from equilibrium position. The above method lets the lines iterate into position from the new position of the nodes, when in actuallity they are displaced from their equilibrium position. This assumption was necessary to make due to limitations in the physical modelling frameworks. To achieve the alternative, a floater would need to be modelled, and then the floater would need to be displaced with a force exactly strong enough to place the nodes in their desired position. This would require several analysis per data point to iterate the correct forces, and subsequently be magnitudes more computationally heavy than the current approach.

4.5.5 Pipeline for static analysis

Above, the conceptual mathematics and logic behind the physical modelling have been explained. These steps need to be reproduced for every available positioning data point before the same data points can be input into the RIFLEX static analysis.

Figure 4.16 shows the flowchart for the physical modelling, starting after the sensor data has been imported and reprocessed and ending after the static analysis is completed and post-processed for each data point. The process utilizes python to interface with RIFLEX, which allows for complete end-to-end integration. This integration also allows for easier tuning and troubleshooting of the system dynamics and utilizes different files to checkpoint the data as it is processed.

Several measures can be taken to keep the run-time reasonable when performing an analysis of this magnitude, i.e. thousands of computationally heavy simulations. Improving hardware is the least time-consuming if better hardware is readily available. Profiling the code and identifying bottlenecks is also advisable. Both of these techniques were used to speed up the process of the physical analysis.

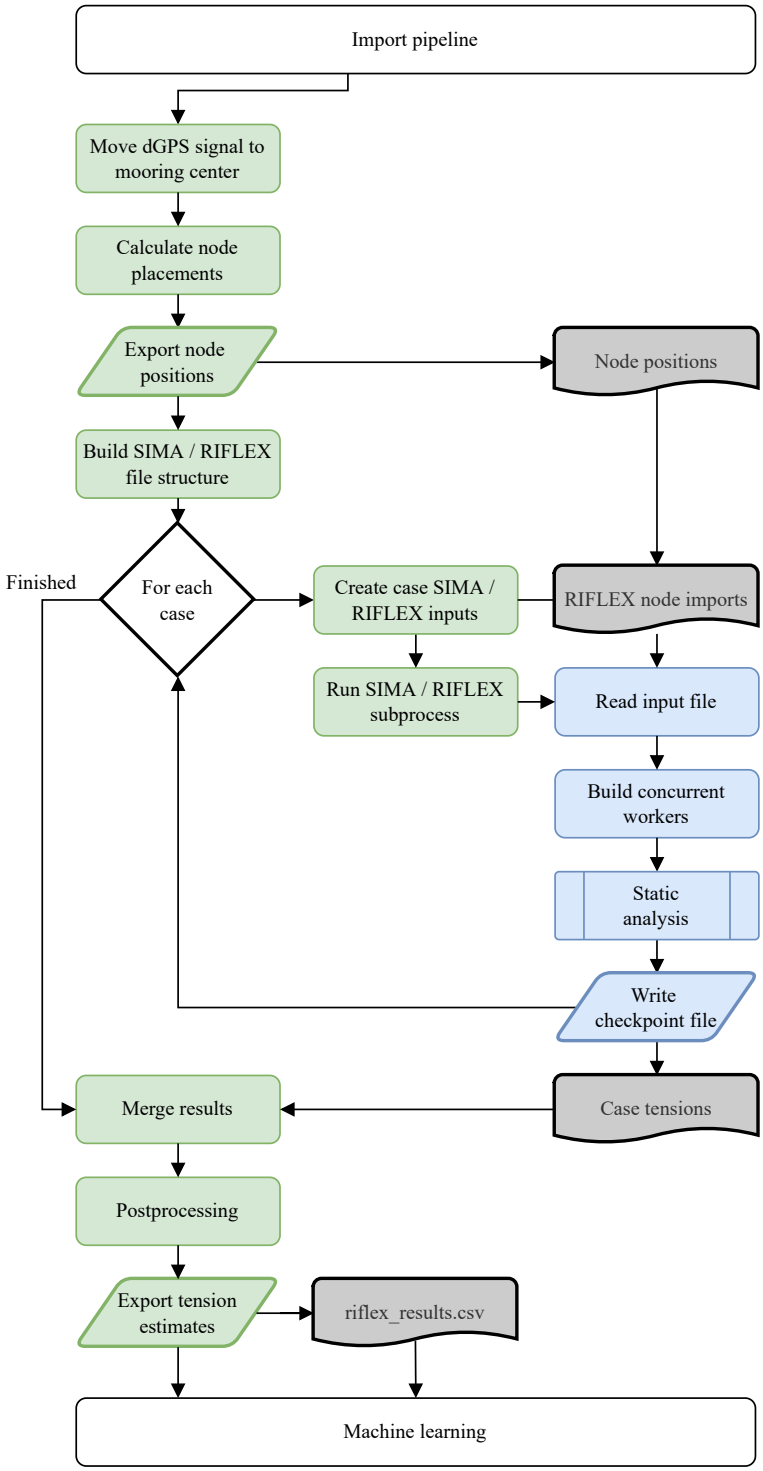


Figure 4.16: Flowchart showing the method for physical modelling using Riflex. Here, blue denotes Riflex, green denotes python, and gray denotes files.

4.6 Data driven modelling

In the following section, the different implementations of the DDM are presented. In total, three different DDMs are implemented and compared: Random Forest Regressor, ANN, and LSTM. For each of these, the tuning and hyperparameters will be discussed. Finally, the pipeline for the DDM is introduced.

Input	Target
Latitude driftoff	Measured tension L1B1
Longitude driftoff	Measured tension L1B2
Tower roll angle	Measured tension L2B1
Tower pitch angle	Measured tension L2B2
Tower yaw angle	Measured tension L3B1
Wind speed in x direction	Measured tension L3B2
Wind speed in y direction	
Metocean data (optional)	

Table 4.7: Input and target features for data driven modelling

4.6.1 Random Forest regression

The Random Forest regressor used is based on the implementation of Random Forest regressor from (Pedregosa et al. 2011). The advantage of Random Forest regressor is the minimal amount of tuning needed. Only one parameter needs tuning in the Scikit-learns implementation, which is the number of decision trees in the forest. Figure 4.17 shows the development of the statistical performance indexes as the number of trees grows. For both RMSE and MAE, decreasing values mean increasing performance. Diminishing returns are seen from 30 trees and upwards. A conservative number of 100 trees were used for this thesis.

As Random Forest trees are invariant to the scale of the features, no normalization of the features are needed for the Random Forest implementation. This reduces the pre-and post-processing of the data.

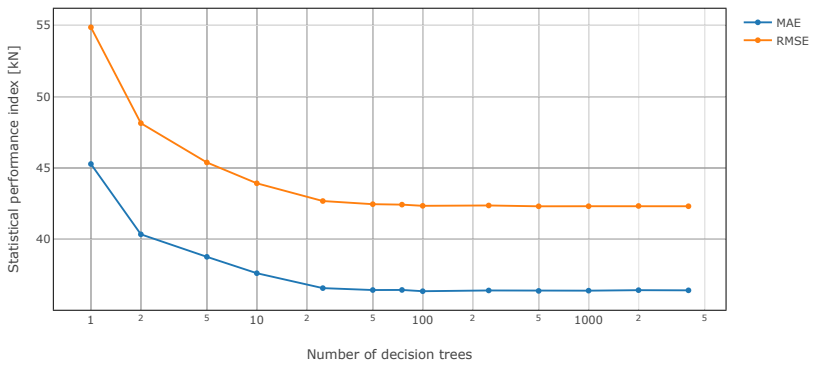


Figure 4.17: Effect of increasing amount of decision trees in Random Forest model

4.6.2 Neural Networks

Using NNs for experimental setups requires several steps. The construction of the architecture, and training of the models, are make-or-break for the experimental results. Since the NNs use layers as building blocks for the network, these need to be defined first. For the general purpose of ANN, deeper networks tend to need fewer units per layer and generalize better to the test set but are often harder to optimize. For this thesis, given that the purpose is to investigate the hybrid approach, an architecture that generalizes well is preferred. This means less time can be used to optimize each of the models.

Feature scaling

Machine learning methods that map a smooth function between input and output features are sensitive to the input scale. This can be seen from the perceptron introduced subsection 3.4.2, where all inputs are weighted and summarized before passing through the activation function. Although the weights can be tuned to encapsulate the difference in scale, this often causes longer convergence times for the models (Zheng and Casari 2018). For these models and components, it is a good idea to scale the features so that the output is within some desired range.

The machine learning data set is scaled using standardisation scaling, also known as Z-score normalisation. Standardisation leverages the mean values of each feature, as well as the variance, to map each feature to a similar scale. Equation 4.4 shows how the Z-score is calculated. Each point x is placed in accordance to the mean value, and then scaled down based on the variance. The scaled points are then placed around $x = 0$, with a variance of 1. Z-score feature scaling is quite robust with regards to outliers, but does this by assuming that data in each feature is normalised. When this is not the case, and the data is skewed, the distribution of the new scale is not equal. The majority of the sensor data is either normalised, or sufficiently normalised for z-score normalisation to be considered an tolerable approach in this case.

$$x' = \frac{x - \bar{x}}{\sigma} \tag{4.4}$$

$$x = \bar{x} + x'\sigma \tag{4.5}$$

Keras & Tensorflow

There are numerous different tools, frameworks, and applications developed over the last decades to simplify the development of NNs. The frameworks allow for faster development, more optimized training, and flexibility for developers. For this project, Keras was deemed most fit (Chollet et al. 2021). The Keras framework interfaces with TensorFlow, and allows for quick deployment and experimentation for

NNs.

Hyperparameters and grid search

When comparing different machine learning methods, using hyperparameter search allows for a more fair compression. Tuning the model parameters can lead to subconscious favoritism towards the desired goal. To avoid this, a standard grid search is performed on the parameters than do not have a recommended value for general purpose. Grid search can often be time-consuming and computationally heavy, so each parameter should only be tested for a small number of variations. The input variables for grid search is presented in Table 4.8. The final hyperparameters, and the reasoning behind the given value, is presented in Table 4.9.

Hyperparameter	Parameters
Batch size	$[128, 256]$
Learning rate	$[0.01, 0.05, 0.1, 0.3]$
Sequence length	$[1, 3, 5, 10]$
Dropout	$[0.1, 0.3, 0.5]$

Table 4.8: Grid search input variables

Hyperparameter	Parameters	Reasoning
Epoch	NA	The number of epochs should be large enough for early stopping to stop the process.
Batch size	NA	The batch size was decided through hyperparameter search
Learning rate	NA	The learning rate was decided through hyperparameter search
Sequence length	NA	The sequence length was decided through hyperparameter search
Dropout	NA	The dropout decided through hyperparameter search
Neurons	NA	The number of neurons was decided through hyperparameter search
Optimizer	ADAM	As mentioned in subsection 3.4.2, Adam has a fast running time, low memory requirements, and requires only tuning for a single parameter. This means it is well fit for general purpose
Output activation function	Linear	For the output layer of regression models, a linear activation function is recommended (Szandała 2020).
Loss function	RMSE	RMSE was chosen due to the intrinsic nature of the prediction: larger deviations from the truth are always more important than smaller ones. In addition, RMSE is computationally less complex than other options.
Early stopping patience	10	A low patience was set to save on computational time.

Table 4.9: The common NN hyperparameters.

ANN

For the ANN, the only specifics that must be defined is the network structure. By implementing a somewhat deep network architecture, varying the amount of nodes should allow the network to fit to most general purposes. Four hidden layers are used, with dropout on each, and a sigmoid activation function on each them.

LSTM

For the LSTM, two stacked layers will be implemented with equal numbers of neurons. The stacking of layers should allow for learning complex temporal connections given enough neurons and input data. The first layer of the LSTM will be many-to-many, as described in subsection 3.4.3, while the second will be many-to-one such that only the final time-step of the tensions is predicted. This structure allows for forward and backward interactions in the temporal space and avoids too computationally heavy training. Also note that the LSTM was trained on the grid search, but not for a sequence length of 1 as presented in Table 4.8.

4.7 Training, Validation, and Testing

To validate the model's predictive performance, a k-fold cross validation scheme was implemented. To limit data leakage, we used one whole case from the operational data as a test case in each cross-validation fold. This ensures that the model has not been trained on identical environmental conditions as seen in the test case. For internal validation during each training epoch, 20% of the training data is randomly selected.

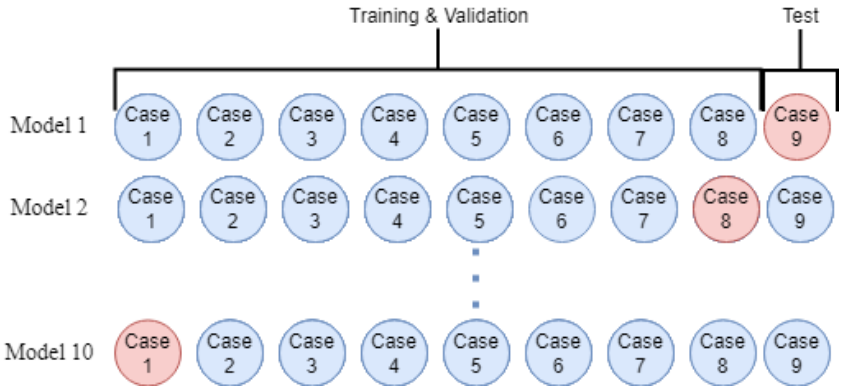


Figure 4.18: Cross-validation scheme. Each case (red) is used once for validation for a model trained on the remaining data (blue).

There are a variety of different approaches to training for machine learning, with different strengths and weaknesses, as explained in section 3.3. CV can maximize the training set, with the trade-off that training time is significantly increased. Within the options for cross-validation, one must also consider if the data

should be shuffled or not:

If the input data is over several time steps, then the number of data points would have to be decreased by a factor of s time-steps to avoid leakage into the test set. This is due to overlapping points also necessarily being overlapping in the input set. Furthermore, no meta-data could be input as it would be a direct way to differentiate each of the different input cases. Most importantly, there would be no test of the robustness of the model. If each of the k-folds in CV has different environmental conditions, it requires the model to generalize instead of overfitting. In short, a shuffled data set for input would improve metrics but not the actual model instead of giving a misleadingly positive result.

4.7.1 Loss function

The pros and cons of different loss functions have been discussed in subsection 3.3.1. When predicting tension measurements, larger deviations are more significant than smaller ones, and minimizing outliers is a priority. Likewise, it is desirable for the loss function to have an easily calculable gradient, as presented in section 3.4.2. This leaves RMSE as the desirable choice in this case.

4.7.2 Fatigue damage estimation

To evaluate the methods performance at fatigue damage analysis, the measured tension from tension sensors and the prediction results from the hybrid model was analyzed with the methodology described in section 2.4. QATS is a python implementation of the rainflow algorithm STM E1049-85 (2011) developed by DNV (Voie and Lone 2021). It was used to quantify accumulated fatigue damage from several cycles in the time series. The S-N curve of the mooring lines was collected from DNV's standard for position mooring, DNV OS E301 (DNV 2021).

4.7.3 Machine learning pipeline

Training and implementation can often be challenging when working with different machine learning frameworks and models. Each of the frameworks needs different inputs, configurations, and different types of handling. There needs to be sufficient abstraction for the pipeline to accommodate all of these. One way to achieve this is through object orientation. By defining a parent class for the models and adapting this to each specific model type, all models can be passed through the pipeline. We also define a database object that handles adjusting the training data to the specific model and ensures there is no data leakage between the training set and the test set.

Figure 4.19 illustrates the abstracted machine learning pipeline. Initially, the import and physical analysis need to complete to finish the input for the models. This is, as mentioned, stored in the database object to ensure accurate configuration. The database object is then called sequentially for each step in the ML

pipeline, passing through the correct data and asserting that no data is leaked between the training and test phase.

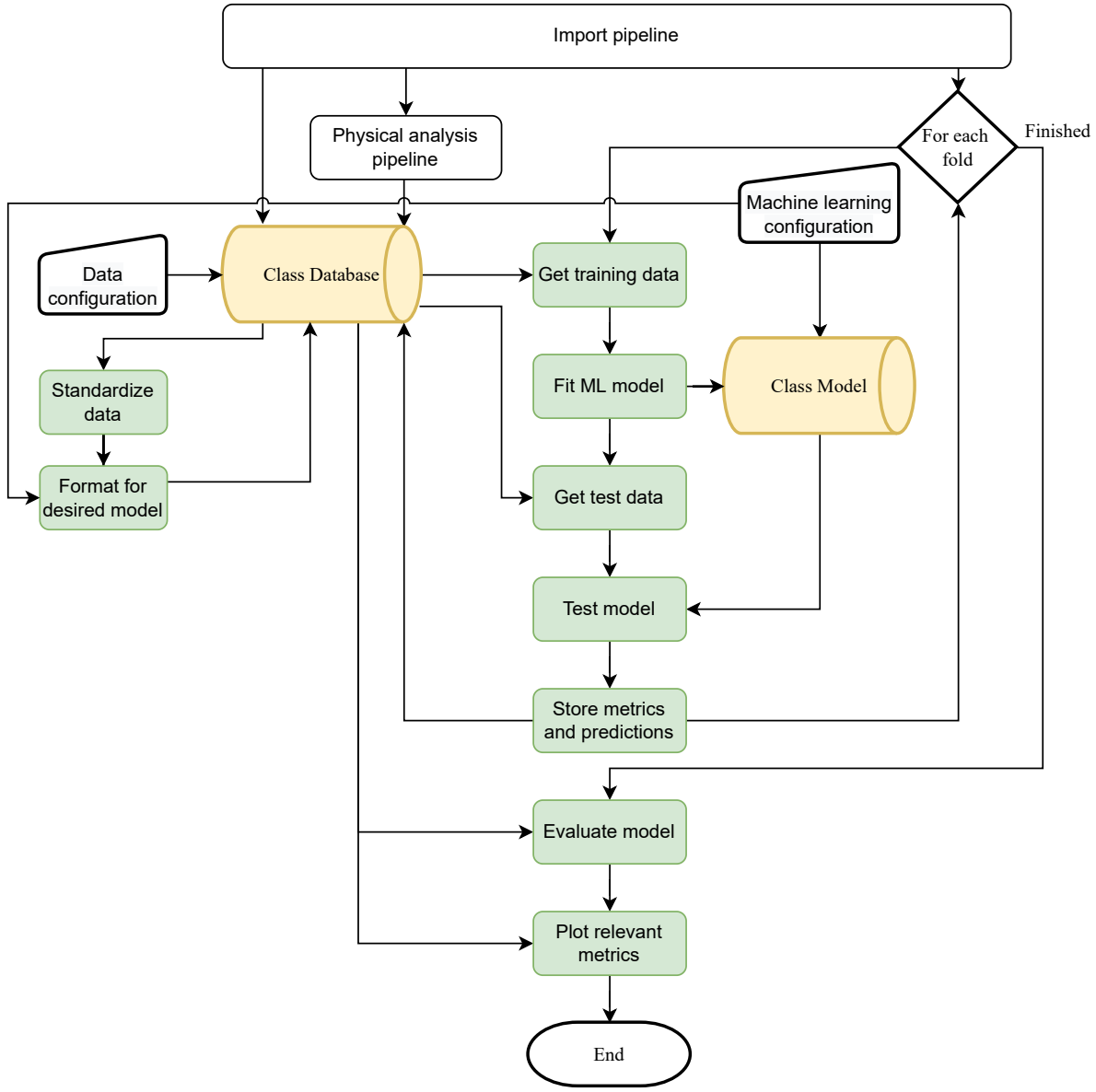


Figure 4.19: Flowchart showing the method for data driven and hybrid modelling. Here, green denotes python, yellow denotes objects, and the white trapezoids denote predefined configurations.

Chapter 5

Results and discussion

5.1 Physics-based model results

The results from the two physics-based models are summarised in Table 5.1 and illustrated in Figure 5.1. For RIFLEX PBM, both the original zero-offset position and the shifted mooring equilibrium position are reported. MoorPy performed similarly to Riflex in the original mooring equilibrium position, and so only the shifted MoorPy results are presented. On average, for all cases and mooring lines, an RMSE of 78.6 kN is achieved for MoorPy and 94.3 kN for Riflex in the shifted position. A significant improvement is seen between Riflex in the original and shifted position, i.e. after re-calibrating the dGPS data. The improvement is valid for all lines except Line 3, bridle 2, where the PBM prediction performance was degraded.

Behind the statistical performance indices in Table 5.1, there is a greater range in the predictive performance of the PBMs. For some lines and cases, the prediction follows the measurements closely with little mean offset, but the deviations are more considerable for other lines. Figure 5.2 shows a well-performing case where the PBM predicts the line tension well and a less optimal case for the prediction

	RIFLEX		**MoorPy**
	Original	**Shifted**	**Shifted**
Line	**RMSE [kN]**	**RMSE [kN]**	**RMSE [kN]**
L1B1	98.511	68.227	47.056
L1B2	62.073	45.141	36.800
L2B1	220.610	120.762	105.522
L2B2	216.286	94.848	86.585
L3B1	89.165	54.725	49.820
L3B2	94.255	182.296	145.617
Avg	130.150	94.333	78.567

Table 5.1: PBM results

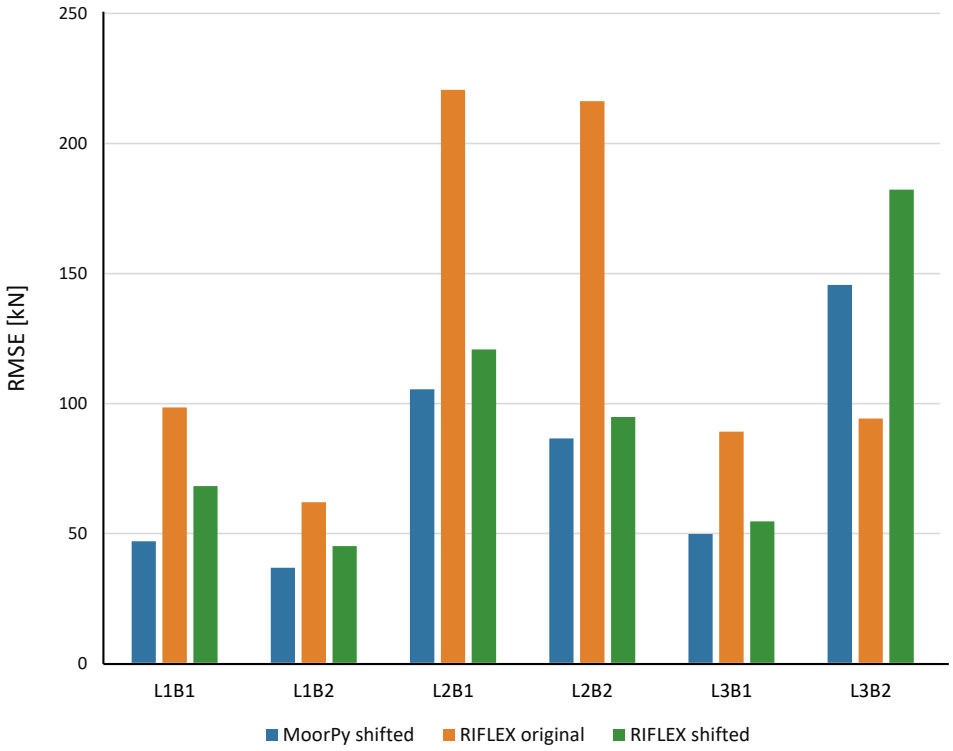


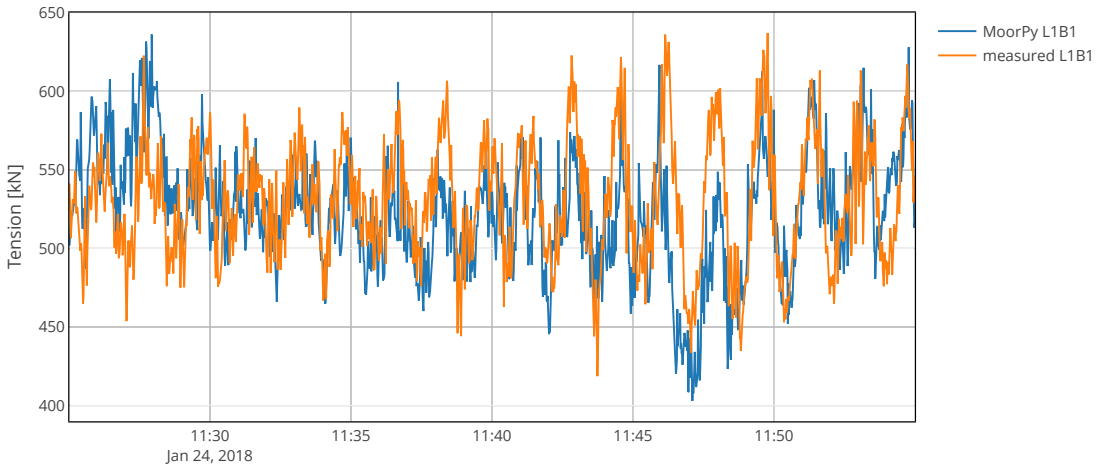
Figure 5.1: PBM results

from MoorPy. For some of the lines, the results had a constant offset, such as (b), and others were more aligned, such as (a).

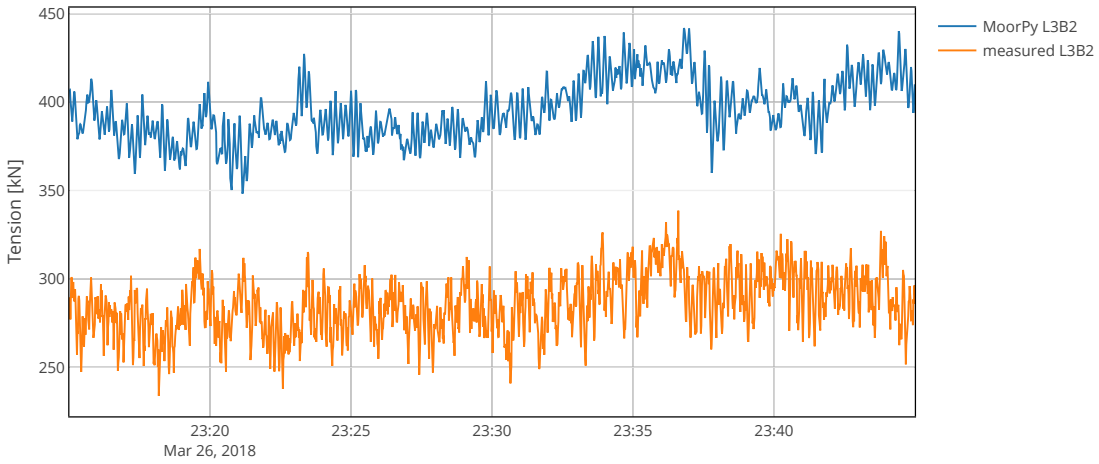
In theory, the data-driven part of the hybrid model should be capable of learning the mean offset and then correcting the prediction. The aim of the PBM is not directly to predict the tension as accurately as possible but to generate a robust mapping between the turbine position and the corresponding mooring line tensions. It would be favourable with an accurate mapping, but part of the motivation behind the hybrid model is the enable the data-driven part to learn from and correct for deviations in the mapping.

As mentioned in subsection 4.4.3 the zero-offset position of the dGPS measurements was optimised using the physical model from MoorPy. The MoorPy model may have an advantage over Riflex since the shifted position is optimised on MoorPy. The results could have been better for Riflex if the results were optimised with Riflex. Figure 5.3 shows the distribution of deviations between tension measurements and physical models predictions for the same sample. MoorPy with shifted position shows a clear improvement in shorter tails of the distribution of errors and improved mean for all lines except L3B2, the only line where the original position provided better predictions. Comparing the original equilibrium position and the new shifted position for Riflex shows improvements in the error distribution and mean deviation for all lines except L3B2.

Due to the optimisation being performed with a limited amount of experiments and samples, L3B2 may be underrepresented in the random selection of samples, causing poorer performance in the L3B2



(a) Example of promising results for the PBM.



(b) Example of less desirable results for the PBM.

Figure 5.2: Predicted tension and measured tension over time for both line 1, bridle 1 and line 3, bridle 2

prediction. Another factor is that the optimisation may primarily optimise the worst performing lines at line 2 and get stuck in a local minimum. It may be that shifting the position corrects underlying errors in the physical model and does not bring the PBM closer to the ”as-installed” condition. The yaw measurement was not correctly calibrated as discussed in the data exploration. Other causes, such as more issues with the mapping between sensors and modelling, can also give complex errors, and the shifting of positions might correlate with other error sources. Nevertheless, the shift did give better results not only for the PBM but also for the subsequent hybrid models.

Another point of contention is what could be regarded as validation data leakage when shifting the equilibrium position of the mooring system. In theory, by optimising the PBMs equilibrium position with all available data, one could argue that we are to some extent optimising on both training and validation data. When running the same optimisation on a case by case basis, the cases all gave similar

optimal position, and as such running the PBM shift through CV as well was deemed unnecessary.

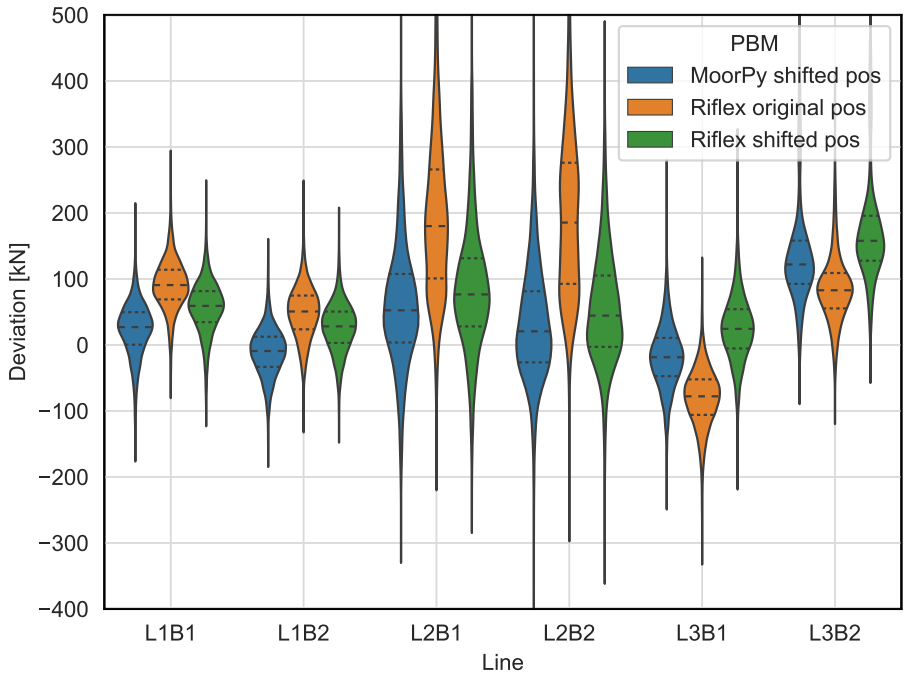
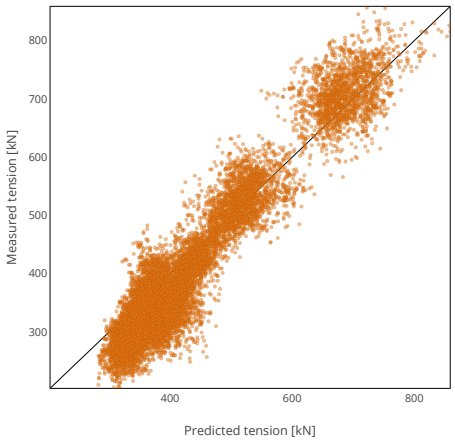


Figure 5.3: Violin plot of the deviation between measured data and the proposed physics-based models. The middle dotted line is the mean of the samples and the outer lines enclose 50% of all samples.

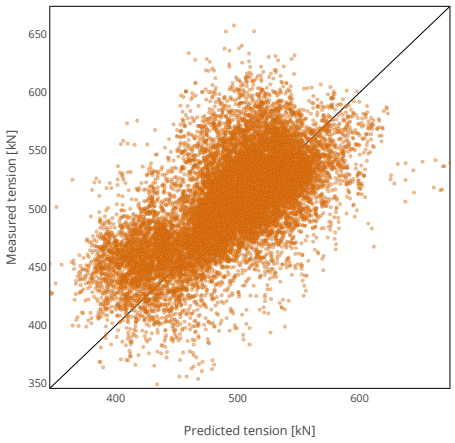
Figure 5.4 shows scatter plots of the predicted tensions from the PBM plotted against the measured tension from the load cells for every line. Ideally, the predictions would be identical to the measured results, placing all the sample points along the line $x = y$. As summarised in Table 5.1, mooring lines L1B1, L1B2 and L3B1 are predicted significantly better by the PBM compared to the remaining lines. In (c) and (d), the predictions for the bridles from Line 2 (L2B1, L2B2) show an excellent fit to the measured tension for lower ranges of tension. The discrepancy between measurements and predictions grows from 700-800 kN and upwards. The predicted tension shows values up to 1200 kN, while the maximum measured tension values are capped at around 900 kN. The PBM show overall poor predictive performance for L3B2, but the same trend as for Line 2 of increased deviation for higher tensions is also seen on this line.

Although no causation for the degradation of results for higher tensions was found, there is a clear correlation with the offset from the mooring equilibrium position. Figure 5.5 illustrates this correlation, where darker red points indicate large offsets from equilibrium, while blue points are close to the mooring equilibrium. There is a clear pattern for the least fitted points with the most significant offset. In the lower-left corner of both (a) and (b), there is a patch of points with a large offset that is still not poorly fit. This is due to the one case from January 6th with northern winds. Offsetting the turbine towards the south results in less tension in the line 2 bridles and indicates a trend where the PBM is less accurate for higher tension in the mooring lines.

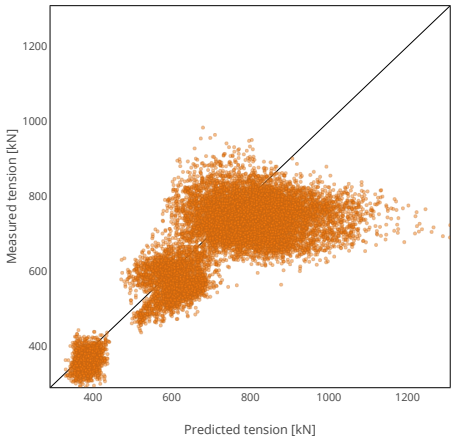
The PBM results are not as close to the measured data as hoped. For L3B2, there non-constant deviation between measured data and the PBM. This deviation is also present for other lines in some validation cases. For the hybrid model to be as efficient as possible, there needs to be a strong and valid foundation in the PBM. If this strong foundation is lacking, then the hybrid model will essentially end up as a machine learning method with unnecessary extra steps. Still, there are likely modelled correlations between the different lines that the DDM would struggle to infer from the data, which should, in theory, improve the model and its robustness.



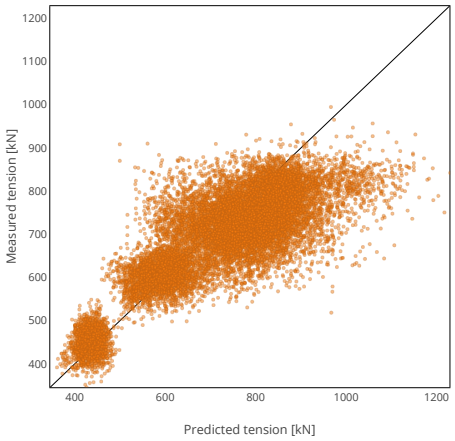
(a) L1B1



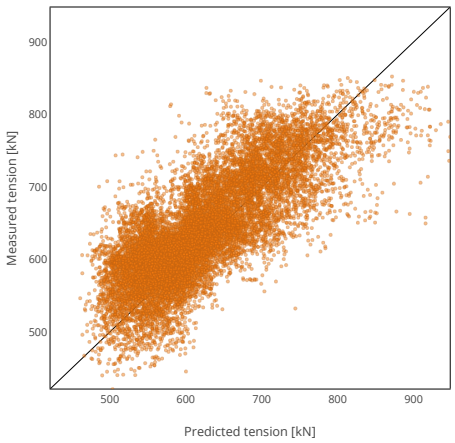
(b) L1B2



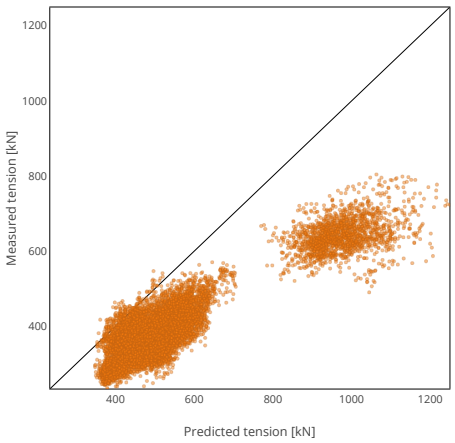
(c) L2B1



(d) L2B2

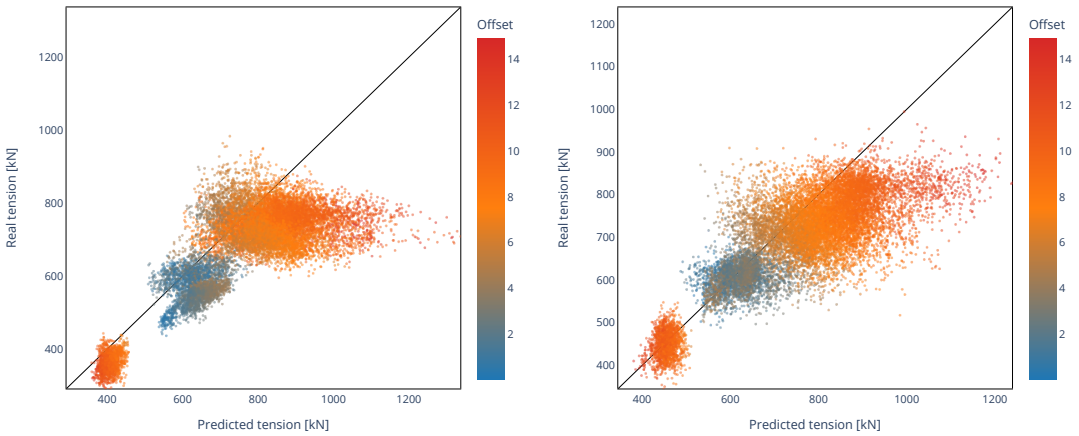


(e) L3B1



(f) L3B2

Figure 5.4: PBM predicted tension vs measured tension for all lines



(a) L2B1 **(b)** L2B2

Figure 5.5: Predicted tension vs measured tension for line 2. The color represents the total offset in meters from mooring equilibrium position.

5.2 Data-driven model results

To investigate the merits of a hybrid modelling approach the addition of a physical model must improve the data-driven models predictive performance. In this section, the three implemented data-driven models are tested without any input from the physical model, to create a foundation for comparison of the Hybrid models.

5.2.1 Random Forest

Results from the Random Forest regressor are summarised in Table 5.2. The data-driven approach was tested both with and without metocean data. An average of 71.8 kN in RMSE was achieved for all lines when the training set included metocean data, and an 11% improvement in RMSE was achieved when excluding the metocean features from the training set. Since the metocean data is constant within each case, and relatively few cases are used in the training set, the metocean features get a low variance. This low variance makes it challenging for data-driven methods to learn relationships between the metocean data and desired target values. Due to the decrease in predictive performance, the metocean features were not included in the Hybrid implementations.

There are significant individual differences in predictive performance on the mooring lines. In the best cases for Line 1, Bridle 2 and Line 3, Bridle 1, the average RMSE is between 40-50% lower than the other lines. While line 2, bridle 1 performs 50% worse than average. No clear trend could be found in the differences between the predictive performance of the different lines for the random forest DDM.

In general, the Random Forest regressor performed poorly on its own as a DDM, and the other methods presented showed significantly better results. A focus is therefore placed on discussing the results from other methods, and the detailed plots for Random Forest DDM are found in Appendix B

	Random Forest	
Line	**With metocean data [kN]**	**Without metocean data [kN]**
L1B1	89.39	73.74
L1B2	31.74	33.66
L2B1	92.01	94.88
L2B2	89.59	72.82
L3B1	54.61	42.12
L3B2	73.25	65.79
Avg	71.76	63.84

Table 5.2: DDM Random Forest regressor RMSE results with and without metocean data

	ANN	LSTM
Line	**RMSE [kN]**	**RMSE [kN]**
L1B1	57.770	58.986
L1B2	38.418	25.439
L2B1	55.516	61.907
L2B2	44.784	48.765
L3B1	34.134	31.120
L3B2	37.775	38.972
Avg	45.627	44.198

Table 5.3: DDM results by line

5.2.2 Neural networks

The results for the LSTM and ANN are presented in Table 5.3. For the neural networks, metocean data did not improve the model for any run in the grid search, so these results will not be presented. With a sufficient dropout on the input layer, the network could ignore the metocean, but this did not warrant further investigation. The averaged results for the ANN DDM were an RMSE of 45.6 kN, a reduction of 42% compared to the best performing PBM model, and 44.2 kN for the LSTM, a reduction of 44 % from the PBM model. When evaluating NNs, it is essential to remember that this number is a lower bound of the performance. Tuning NN hyperparameters is a complex and time-consuming task. The tuning was primarily solved using an extensive grid search for this thesis, one which did not include changes to the layering of the network. Given more time and more attempts at specific tuning, all results containing NNs would likely improve. However, the grid search gives insight into how easy a given model is to train. For both the ANN and LSTM, reaching results within 1 kN of the presented results proved manageable, and there were numerous combinations of parameters that gave these results.

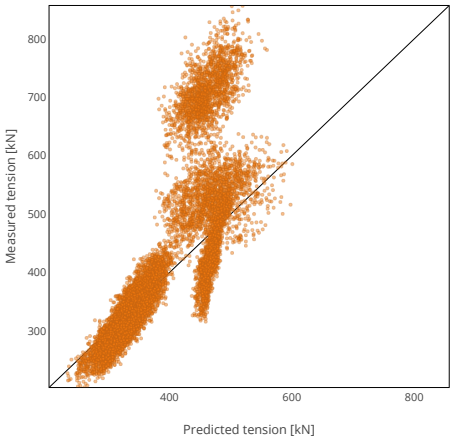
Figure 5.6 shows the predicted tension against the measured tension for each of the lines with an ANN network. The clustering, which can be seen in most subplots, is caused by the different cases and their conditions. The lowest performing lines are L1B1 and L2B1, and for the former, this is primarily due to a single case. As seen in Figure 5.6a, there is a single cluster for L1B1 with a significant offset from the others. All this data is from a single case with an average RMSE of 252.3 kN. Compared to the other cases, which average 34.4 kN, this is an increase of 740%. Although the same case performs worse in general for all lines aggregated, as shown in Table 5.4, it is unclear why it performs so much worse for L1B1 specifically. For the PBM, L1B1 performed better than the average for the same case. The main issue when using pure DDM models is that they struggle to generalise enough to work for unseen conditions. For a DDM model, significant deviations between measured and predicted might occur whenever the model attempts to predict previously unseen conditions. This lack of generalisation is unfortunate for the marine environment, where 100 or 1000 year waves would need to be observed before they occur to model the response accurately.

The above mentioned point regarding the issue with DDM for unseen conditions is clear when comparing Table 5.4, Figure 4.5 and Figure 4.8. Case 1 is the worst performer, and this is also the case which has

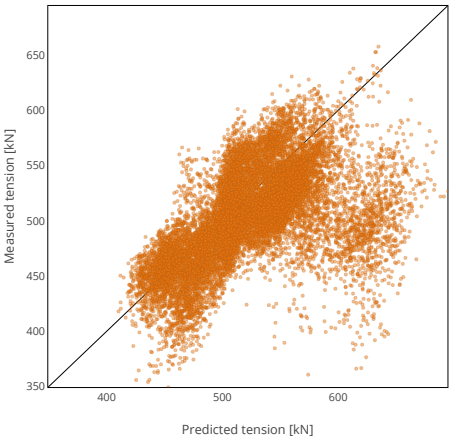
Case	Date	RMSE [kN]
1	06.01.18	93.256
2	09.01.18	34.277
3	14.01.18	33.994
4	13.02.18	74.405
5	24.02.18	28.874
6	26.03.18	23.924
7	14.04.18	35.452
8	02.05.18	49.307
9	29.07.18	37.157
Avg		45.627

Table 5.4: ANN DDM results by case

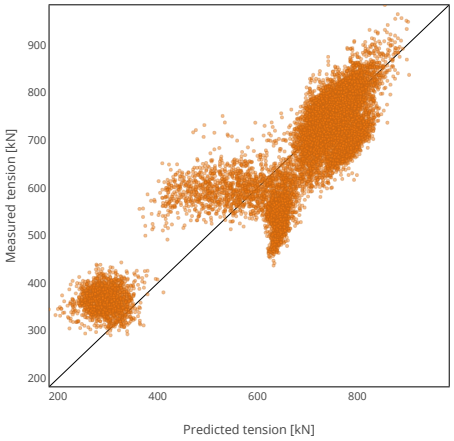
wind coming from the opposite direction as the rest of the cases, as shown in Figure 4.5, resulting in the outlier cluster of longitude-latitude offset as shown in Figure 4.8.



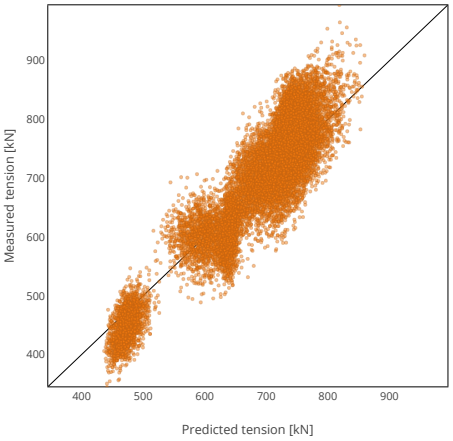
(a) L1B1



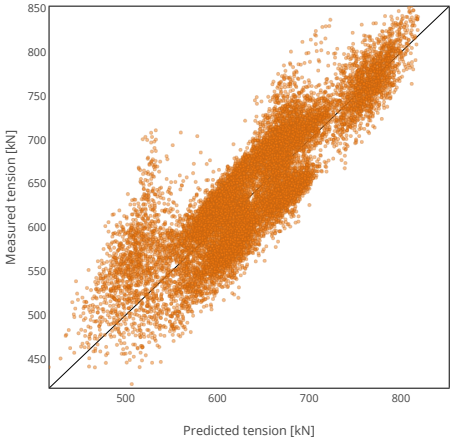
(b) L1B2



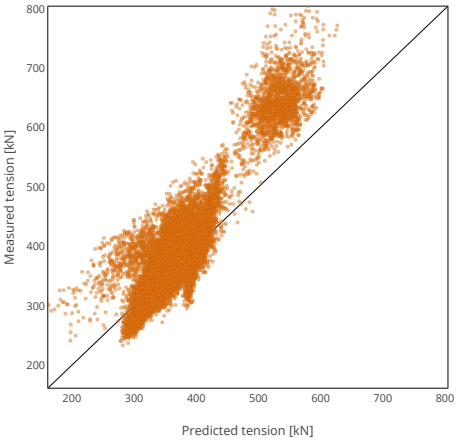
(c) L2B1



(d) L2B2



(e) L3B1



(f) L3B2

Figure 5.6: DDM with an ANN - Predicted tension vs measured tension for all lines

5.3 Hybrid model results

In the proposed hybrid model A the DDM predicts the deviation between the predicted tension from PBM and the measured tension. Figure 5.7 shows how the predictions from PBM and DDM in model A are combined to provide the hybrid model prediction. The initial PBM results are pretty inaccurate, as presented earlier, and so the model needs to be improved upon. Given some patterns from the IMU and dGPS, a DDM model can learn which data there is likely to be a larger or smaller deviation between the PBM and the measured data. The middle plot for Figure 5.7 shows precisely this process, while the final plot shows how much the prediction has improved.

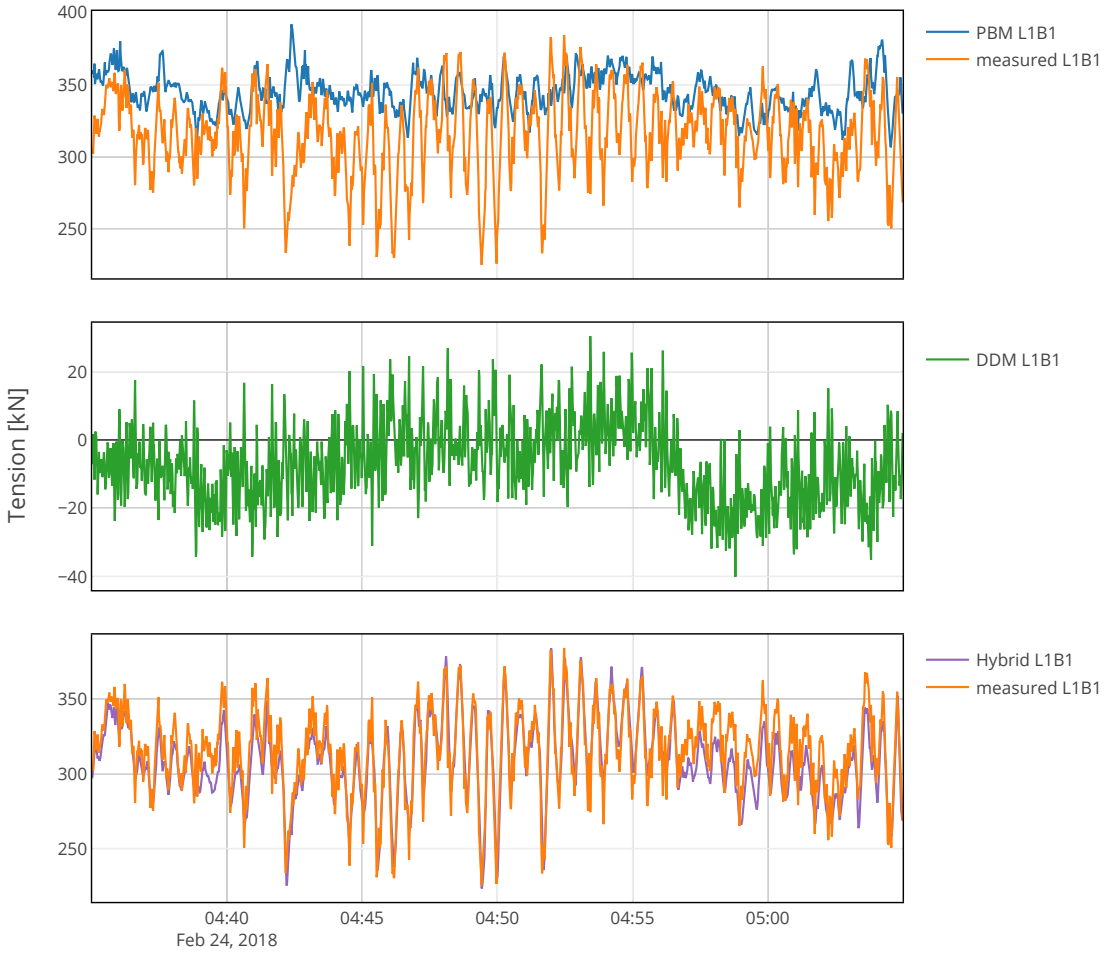


Figure 5.7: Step-by-step results of how hybrid model A combines PBM and DDM predictions. The top plot shows the PBM prediction vs the measured tension, the middle plot shows the DDM predicted deviation between PBM and measured tension, and the bottom plot shows the combination of DDM and PBM compared to measured tension.

5.3.1 Random Forest

Table 5.5 summarises the results from hybrid modelling using model A and model B, with Random Forest algorithm. Model A shows better predictive performance for all lines and gets a 36% lower average RMSE than model B. The largest improvement occurs on the northern lines (see Figure 4.2) L1B1 and L3B2, where the average RMSE is improved by 67% and 44% respectively.

Hybrid model B performs marginally better than DDM alone for the random forest algorithm. This indicates that PBM features may not be weighted enough to improve the hybrid model B results. Both hybrid models prove significantly better predictive performance than the physical-based models presented in Table 5.1.

	Hybrid model A	**Hybrid model B**
Line	**RMSE [kN]**	**RMSE [kN]**
L1B1	27.61	74.88
L1B2	27.39	33.09
L2B1	63.55	92.50
L2B2	50.23	70.48
L3B1	34.66	42.26
L3B2	43.35	60.51
Avg	41.13	62.29

Table 5.5: Random Forest hybrid model results

5.3.2 Neural networks

	Hybrid model A	**Hybrid model B**
Line	**RMSE [kN]**	**RMSE [kN]**
L1B1	25.562	64.666
L1B2	29.337	29.237
L2B1	54.688	49.674
L2B2	47.132	46.476
L3B1	38.598	33.226
L3B2	40.005	38.707
Avg	39.220	43.665

Table 5.6: ANN hybrid model results

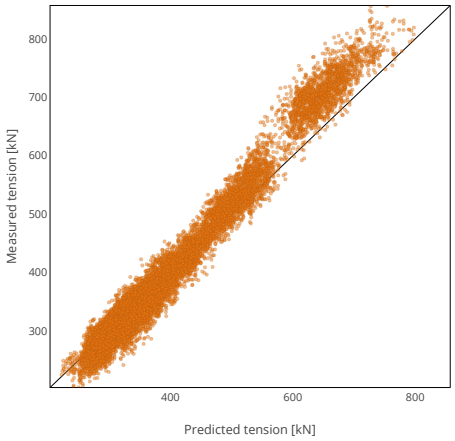
Figure 5.8 shows the predicted tension from ANN hybrid model A, vs. the measured tensions. Compared to the equivalent Figure 5.4 for PBM, there is an overall improvement for all lines. L1B1 (a) shows the most promising results, with a good fit for lower tension measurements and a slight under-prediction for higher tension measurements. The increased deviations seen for L2B1 and L2B2 at higher tensions in the PBM is corrected for by the hybrid method, and the predictions are improved across the entire tension range. The worst performing line in the PBM, L3B2, has a 73% decrease in RMSE for hybrid model A, but still predicts too high tensions for the upper ranges of measured tension.

The metrics of the ANN are presented in Table 5.6 for both methods A and B. The best performing hybrid model was ANN hybrid model A, with an RMSE of 39.2 kN. The ANN likely performed the best due to the less complex tuning process than the LSTM. The grid search did not find an equally performing set of hyperparameters for model B. Although this might be because model A is easier to train, it could also be caused by the grid search not covering optimal parameters for model B. In general, the same information is contained in models A and B, so it should, in theory, be able to perform as well as model A when disregarding the time until convergence during training.

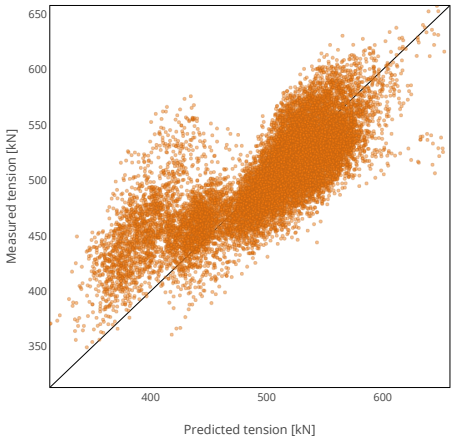
	Hybrid model A	**Hybrid model B**
Line	**RMSE [kN]**	**RMSE [kN]**
L1B1	28.213	55.531
L1B2	30.965	30.530
L2B1	68.439	62.091
L2B2	47.894	45.014
L3B1	41.357	34.750
L3B2	42.013	32.889
Avg	43.147	43.467

Table 5.7: LSTM results for hybrid model A and B

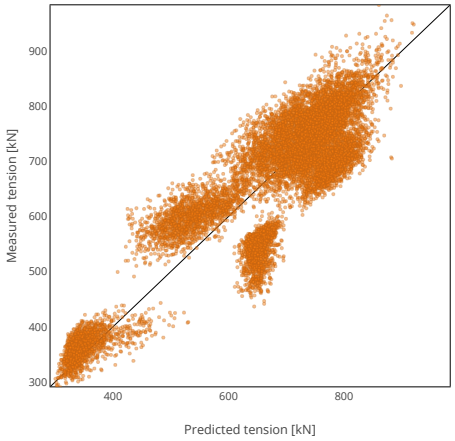
For the LSTM network, the results for both methods is presented in Table 5.7. Both model A and model B underperformed compared to the best results for the Random Forest and for the ANN. In addition to this, the LSTM was significantly more time consuming to train, with the grid search being very computationally heavy. There are several reasons why the LSTM might have underperformed. Firstly, the historic sequences might not significantly impact the tension for each time step. Initially, we thought that the previous time steps would improve the model. However, for both the ANN and the LSTM, the models' accuracy decreased as the sequence length increased. Secondly, the LSTM is undoubtedly the most complex to tune. The grid search might not have been a sufficient tuning method, and more work could potentially improve the performance. Nevertheless, within the confines of this thesis, the LSTM proved to be the least successful hybrid method.



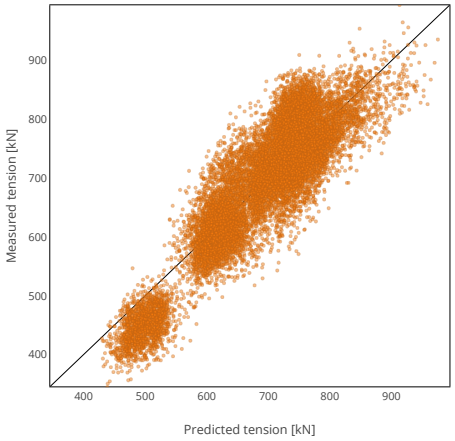
(a) L1B1



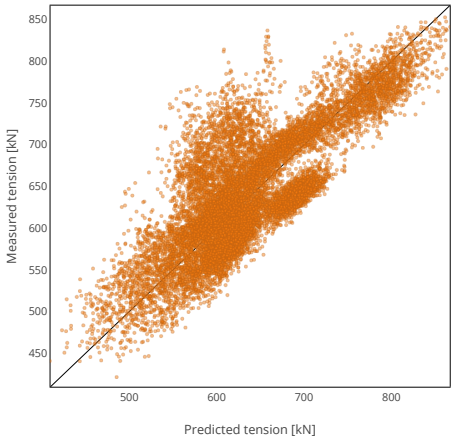
(b) L1B2



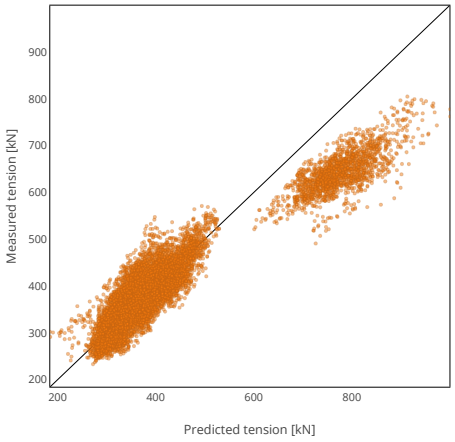
(c) L2B1



(d) L2B2



(e) L3B1



(f) L3B2

Figure 5.8: ANN, hybrid model A, predicted tension vs measured tension for all lines

5.4 Fatigue life estimation

The estimated accumulated fatigue damage is calculated using DNV developed QATS library (Voie and Lone 2021) as described in subsection 4.7.2. The accumulated fatigue damage of the 9 cases was estimated for the measured tension, simulated tension from best performing PBM (MoorPy), and the best performing hybrid models. The percentage-wise deviation in accumulated fatigue damage between measured and estimated tension for each line and method is presented in Table 5.8.

Unfortunately, none of the models presented provides an adequately accurate estimate of fatigue damage in the given cases. As shown in Figure 5.9 the PBM underestimates the fatigue on the north-west facing lines (L1B1 and L1B2) while overestimating the fatigue for the remaining lines. The south facing lines, L2B1 and L2B2, get the highest deviation, with a 2-6 times higher estimated fatigue than the measurements. The total percentage deviation is misleading as the fatigue on L1B1, L1B2, L2B2 and L3B1 are underestimated by all hybrid models. At the same time, L2B1 and L3B2 are overestimated by the Random Forest and LSTM hybrid model causing the contributions to total deviation to mostly cancel out.

Line	PBM (MoorPy)	Random Forest (Hybrid model A)	ANN (Hybrid model A)	LSTM (Hybrid model A)
L1B1	-68,09 %	-56,84 %	-58,31 %	-61,47 %
L1B2	-54,96 %	-53,57 %	-62,97 %	-68,19 %
L2B1	607,22 %	52,76 %	10,98 %	36,33 %
L2B2	193,93 %	-6,06 %	-50,75 %	-29,05 %
L3B1	26,85 %	-21,85 %	-43,54 %	-29,02 %
L3B2	53,28 %	18,88 %	-9,03 %	2,29 %
Total	159,70 %	-5,64 %	-34,13 %	-20,72 %

Table 5.8: Deviations from estimated fatigue life from measured tensions

Even though ANN hybrid model A perform best with the performance metrics set (RMSE) for tuning the machine learning models, the Random Forest implementation achieves better predictions of the accumulated fatigue damage. More favourable fatigue estimation performance may be achieved if a custom error measure, including the fatigue estimation performance, was implemented in the training of the DDM.

As the 10 cases included in this case study only cover 5 hours over six months, the value of looking at estimated fatigue may be limited. Given that the fatigue damage calculated in each 30min case is relatively small, slight deviations in the estimated tension cause quite large deviations in the fatigue results. Fatigue damage accumulates over more extended periods, and so to evaluate the true potential for fatigue monitoring, a longer time frame of data is needed.

In Figure 5.10, the biggest challenge with predicting tension measurements for fatigue estimation is presented. The predictive model needs to predict the larger drifts accurately and differentiate between small wave motions and sensor noise. None of the above-presented methods manages to do so in any meaningful way, which is likely the reason for the large deviations in estimated fatigue. The models try to minimise the mean error and, in doing so, cannot capture the smaller cycles, some of which are noise

and some of which are changes caused by wave motions.

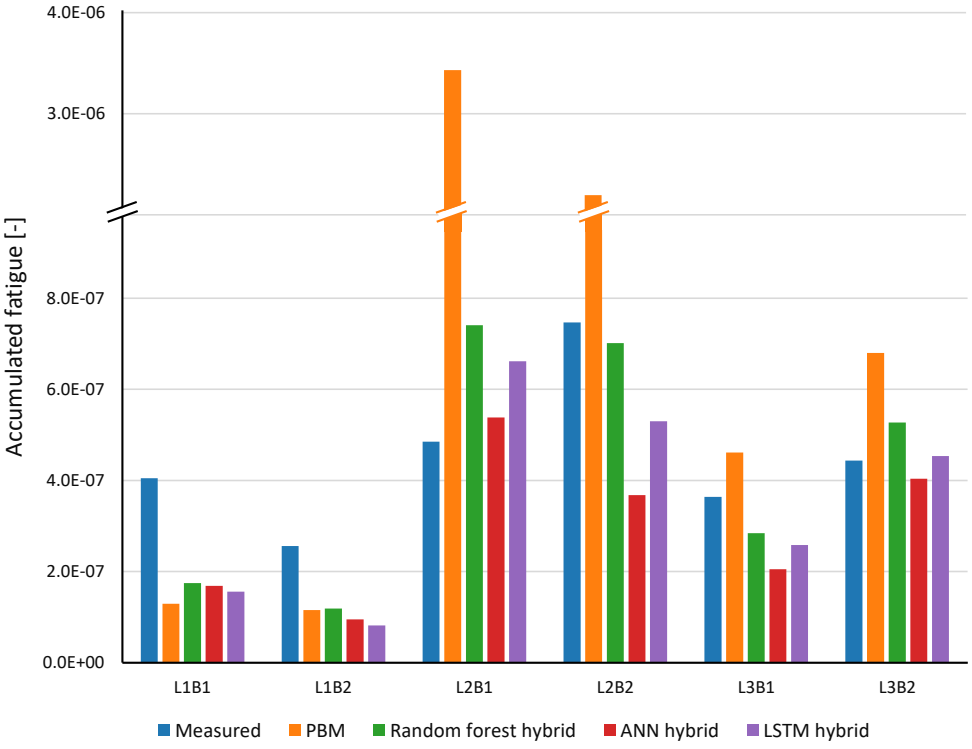


Figure 5.9: Estimated accumulated fatigue damage. Note the axis break, and the difference in scale after the break. The accumulated fatigue is on a scale from 0 to 1, where 1 means complete fatigue of the line.

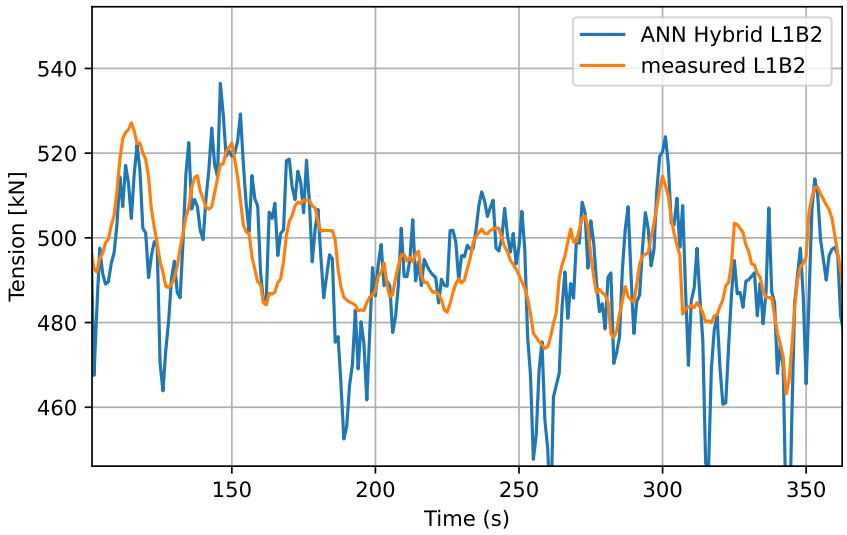


Figure 5.10: Zoomed in example of challenges for fatigue estimation

Line	RMSE [kN]
L1B1	14.903
L1B2	15.441
L2B1	24.522
L2B2	28.824
L3B1	16.912
L3B2	16.435
Avg	19.506

Table 5.9: ANN hybrid A results with random sampling CV.

5.5 Discussion around validation scheme

Previously, we introduced the CV scheme, split case by case. Each fold in the CV is validated on a previously unseen half hour of continuous data. This approach allows for a more thorough comparison of the robustness added from the PBM. The result was less important than the comparison between methods for this thesis, so the used scheme was deemed appropriate. However, there are some downsides to this approach that need to be addressed. A FOWT would likely spend almost all of its lifetime in a condition that it has previously experienced. In other words, the cases in which a FOWT experiences new conditions are rare after startup and become increasingly rarer throughout its lifetime. With this in mind, the results presented previously in this chapter are not comparable to a model trained in a larger dataset with the most potential conditions.

In subsection 1.2.11, we briefly discussed how the results from Walker et al. (2021) are not directly comparable to the results in this thesis. However, training on a randomly sampled set of points has its merits: it is a good indication of how a model would behave in more known conditions, i.e. with more available data for the domain it operates in. Table 5.9 shows the results for the ANN hybrid model A, using grid search for parameters and random sampling CV as a validation scheme. The results are substantially better than the ones presented previously.

Chapter 6

Conclusion

The aim of this master thesis was to develop a digital twin framework for mooring system monitoring of floating wind turbines. The objective was to predict the top tension of the mooring lines using positional sensor data from the wind turbine. Utilising a novel hybrid modelling approach, the predictions of a physics-based model were combined with a data-driven model to improve the predictions robustness and accuracy of data-driven models alone.

A physics-based model of a floating wind mooring system was modelled in MoorPy, a python framework for solving catenary equations, and Riflex, a FEM solver for slender marine structures. In total, three different data-driven models, a Random Forest regressor, an LSTM, and an ANN were implemented to predict the top tension in lines both as a stand-alone model and in a hybrid approach in combination with the physics-based model. Two methods for combining the PBM and DDM predictions were implemented, referred to as model A and model B.

The proposed framework was tested on operational data from a wind turbine at Hywind Scotland. For the PBM an average RMSE of 79 kN was achieved with Moorpy. Out of the three DDMs the LSTM performed best achieving an average RMSE of 44 kN, a significant improvement over the PBM. For the hybrid models, the Random Forest implementation performed significantly better with model A compared to model B. The neural net methods proved more robust to the choice of hybrid approach, but still model A achieved slightly lower average RMSE. ANN Hybrid model A achieved the best average RMSE out of all implemented models with 39 kN RMSE, an 11% improvement compared to the best performing data-driven model.

Overall the proposed digital twin framework achieved a good approximation of tension measurement. However, it is not suitable for fatigue estimation at current state. The method may prove more valuable for fatigue estimation given more training data. More promising results from the randomly sampled data points, indicating the potential performance of the hybrid model trained on a larger training set.

6.1 Further work

One of the most important steps for improved overall performance is further tuning of the PBM. There are several deviations that we simply could not account for, and at times the measured tension and the movement currently appears not to be coupled at all. There is several steps that can be taken to accomplish this, starting with more in-depth analysis of how the data couples with the tension. Next, improved accuracy of the sensors, the resolution for the dGPS in particular, could also have a significant impact on the model. Other options, like a synthetic upsampling, could give some of the same benefits. In general, a more thorough end-to-end physical modelling should give both increased robustness and accuracy of the hybrid model.

Another issue with the PBM the approach for placing the lines and supernodes, as described in subsection 4.5.4. Investigation into a more physically accurate course of action, i.e. displacing the lines using forces rather than coordinates, could significantly improve the PBM, although we have not made enough progress in this area to be certain.

For further work on hybrid modelling for mooring system monitoring more data should be gathered to build a wider foundation of environmental conditions and floater responses. By collecting longer time-series the performance for changing weather conditions may be quantified. This will of course improve the DDM part of the hybrid model, as more training data is always better within ML. In addition to this, more data for more sea conditions might also make the issues with the PBM more apparent.

The work in itself is more valuable if it can help reduce the necessity for direct tension measurement sensors. Investigations should therefore be conducted into how well the hybrid model framework generalises for FOWTs in the same floating wind farms. Is it possible to only equip a subset of the FOWTs with tension sensors, train a hybrid model on these, and then accurately predict the tension using the dGPS and IMU sensor data from FOWTs with tension sensors?

A coupled analysis model for extreme conditions could be applied to train the model on severe but rare storm conditions such as environmental conditions with a 10-year or higher return period.

For further work to enable real time monitoring the physical model must be converted to create a look-up table, A methodology for continuous training of the DDM should be implemented to continuously increase foundation of data the model is trained upon. A method weighting recent measurements above old ones would be valuable to be able to adapt to changes in the mooring system over time, such as drift-off or marine growth.

Bibliography

Archer, Cristina L. and Mark Z. Jacobson (2005). "Evaluation of global wind power". In: *Journal of Geophysical Research: Atmospheres* 110.D12.

Arlot, Sylvain and Alain Celisse (2010). "A survey of cross-validation procedures for model selection". In: *Statistics surveys* 4, pp. 40–79.

Berrar, Daniel (2019). *Cross-Validation.*

Borg, Michael, Maurizio Collu, and Athanasios Kolios (2014). "Offshore floating vertical axis wind turbines, dynamics modelling state of the art. Part II: Mooring line and structural dynamics". In: *Renewable and Sustainable Energy Reviews* 39, pp. 1226–1234. ISSN: 1364-0321.

Bosch, Jonathan, Iain Staffell, and Adam D. Hawkes (2018). "Temporally explicit and spatially resolved global offshore wind energy potentials". In: *Energy* 163, pp. 766–781. ISSN: 0360-5442.

Bottou, Léon (2012). "Practical Recommendations for Gradient-Based Training of Deep Architectures". In: *Neural Networks: Tricks of the Trade: Second Edition*. Ed. by Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 442–448. ISBN: 978-3-642-35289-8.

Cerqueira, Vitor, Luis Torgo, and Igor Mozetič (2020). "Evaluating time series forecasting models: An empirical study on performance estimation methods". In: *Machine Learning* 109.11, pp. 1997–2028.

Chai, T. and R. R. Draxler (2014). "Root mean square error (RMSE) or mean absolute error (MAE)? - Arguments against avoiding RMSE in the literature". In: *Geoscientific Model Development* 7.3, pp. 1247–1250.

Chakrabarti, Subrata (2005). *Handbook of Offshore Engineering (2-Volume Set)*. eng. St. Louis: Elsevier Science & Technology. ISBN: 9780080443812.

Chang, Chih-Wei and Nam Dinh (2016). "A study of physics-informed deep learning for system fluid dynamics closures". In: *American Nuclear Society Winter Meeting, Anaheim, CA*, pp. 1785–1788.

Chollet, François et al. (2021). *Keras*. `https://keras.io`.

Efficient Mooring Line Fatigue Analysis Using a Hybrid Method Time Domain Simulation Scheme (June 2013). Vol. Volume 1: Offshore Technology. International Conference on Offshore Mechanics and Arctic Engineering. V001T01A035.

Ciuriuc, Alexandra et al. (2022). "Digital tools for floating offshore wind turbines (FOWT): A state of the art". In: *Energy Reports* 8, pp. 1207–1228. ISSN: 2352-4847.

Clarke, L. et al. (2022). *In IPCC, 2022: Climate Change 2022: Mitigation of Climate Change. Contribution of Working Group III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*.

Cobra Group (2021). *Kincardine Offshore Floating Wind Farm*.

Daily, Jim and Jeff Peterson (2017). "Predictive Maintenance: How Big Data Analysis Can Improve Maintenance". In: *Supply Chain Integration Challenges in Commercial Aerospace: A Comprehensive Perspective on the Aviation Value Chain*. Ed. by Klaus Richter and Johannes Walther. Cham: Springer International Publishing, pp. 267–278. ISBN: 978-3-319-46155-7.

Dhatt, Gouri, Emmanuel Lefrançois, and Gilbert Touzot (2012). *Finite element method*. John Wiley & Sons.

DNV (July 2021). *DNV-OS-E301*. Standard. DNV.

DNV (Jan. 2018). *DNVGL-RU-OU-0300*. Standard. DNV.

Došilović, Filip Karlo, Mario Brčić, and Nikica Hlupić (2018). "Explainable artificial intelligence: A survey". In: *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 0210–0215.

Educative.io (2013). *Overfitting and underfitting*. [Online; accessed May 20, 2022].

Elman, Priscilla et al. (Oct. 2013). "Reducing Uncertainty Through the Use of Mooring Line Monitoring". In: Offshore Technology Conference Brasil. OTC-24388-MS.

Energias de Portugal (2020). *Windfloat Atlantic Project*.

Equinor (2022a). *Flytende Havvind*.

Equinor (2022b). *Hywind Tampen*.

Faltinsen, O. M. (1999). "Stationkeeping". In: *Sea loads on ships and offshore structures*. Cambridge University Press, pp. 257–270.

Feng, Weijiang et al. (May 2017). "Audio visual speech recognition with multimodal recurrent neural networks". In: pp. 681–688.

Floating Offshore Wind Farms (2016). eng. Cham.

Fossen, Thor I. (2021). *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, Ltd.

Fuchs, Franz G et al. (2016). “Wake modeling in complex terrain using a hybrid Eulerian-Lagrangian Split Solver”. In: *Journal of Physics: Conference Series*. Vol. 753. 8. IOP Publishing, p. 082031.

Gao, Kaifeng et al. (May 2020). “Julia Language in Machine Learning: Algorithms, Applications, and Open Issues”. In: *Computer Science Review*, pp. 100254–100259.

Ge, Zhiqiang (2017). “Review on data-driven modeling and monitoring for plant-wide industrial processes”. In: *Chemometrics and Intelligent Laboratory Systems* 171, pp. 16–25. ISSN: 0169-7439.

Gençay, Ramazan and Min Qi (Aug. 2001). “Pricing and hedging derivative securities with neural networks: Bayesian regularization, early stopping, and bagging”. In: *Neural Networks, IEEE Transactions on* 12, pp. 726–734.

Gordon, Robert B., Martin G. Brown, and Eric M. Allen (May 2014). “Mooring Integrity Management: A State-of-the-Art Review”. In: vol. Day 4 Thu, May 08, 2014. OTC Offshore Technology Conference. D041S047R006.

Grossi, Enzo and Massimo Buscema (Jan. 2008). “Introduction to artificial neural networks”. In: *European journal of gastroenterology & hepatology* 19, pp. 1046–54.

Integrity Management of Mooring Systems (May 2019). Vol. Day 1 Mon, May 06, 2019. OTC Offshore Technology Conference. D011S003R003.

Hall, Mathew et al. (July 2021). *MoorPy (Quasi-Static Mooring Analysis in Python)*. [Computer Software] `https://doi.org/10.11578/dc.20210726.1`.

Hasan, Mohammad Saidee (June 2015). “Fatigue Analysis and Design of Mooring Systems. Assessment and comparison of different methods.” PhD thesis.

Ho, Tin Kam (1995). “Random decision forests”. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1, 278–282 vol.1.

Hochreiter, Sepp (Apr. 1998). “The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6, pp. 107–116.

Hochreiter, Sepp and Jürgen Schmidhuber (Dec. 1997). “Long Short-term Memory”. In: *Neural computation* 9, pp. 1735–80.

Ibrion, Michaela, Nicola Paltrinieri, and Amir R. Nejad (2020). “Learning from failures: Accidents of marine structures on Norwegian continental shelf over 40 years time period”. In: *Engineering Failure Analysis* 111, p. 104487. ISSN: 1350-6307.

IEA (2019). *Offshore Wind Outlook 2019*. Tech. rep.

IEA (2021). *World Energy Outlook 2021*. Tech. rep.

Jimenez, Veronica Jaramillo, Noureddine Bouhmala, and Anne Haugen Gausdal (2020). “Developing a predictive maintenance model for vessel machinery”. In: *Journal of Ocean Engineering and Science* 5.4, pp. 358–386. ISSN: 2468-0133.

Kai-Tung, Ma et al. (2019). *Mooring System Engineering for Offshore Structures*. Gulf Professional Publishing. ISBN: 9780128185513.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Kotsiantis, S. B. (2013). *Decision trees: a recent overview*.

Lillestøl, Dag Børre, Odd Torbjørn Kårvand, and Are Torstensen (Aug. 2021a). "Monitoring and Calibration of Long Term Mooring System Design Assumptions Through Use of AIS and Weather Data". In: vol. Day 3 Wed, August 18, 2021. OTC Offshore Technology Conference. D032S085R001.

Monitoring and Calibration of Long Term Mooring System Design Assumptions Through Use of AIS and Weather Data (Aug. 2021b). Vol. Day 3 Wed, August 18, 2021. OTC Offshore Technology Conference. D032S085R001.

M. Learch, J.A. Norbeck and P.A. Berthelsen (2019). "Qualification of innovative floating substructures for 10MW wind turbines and water depths greater than 50m". In.

Ma, Kai-tung et al. (May 2013). "A Historical Review on Integrity Issues of Permanent Mooring Systems". In: vol. All Days. OTC Offshore Technology Conference. OTC-24025-MS.

Martens, Harald (2021). "Interpretable machine learning with an eye for the physics: Hyperspectral Vis/NIR "video" of Drying Wood analyzed by hybrid subspace modeling". In: *NIR news* 32.7-8, pp. 24–32.

Martínez, Abel and G. Iglesias (Feb. 2022). "Mapping of the levelised cost of energy for floating offshore wind in the European Atlantic". In: *Renewable and Sustainable Energy Reviews* 154, p. 111889.

Mathiesen, Martin, Anja K. Meyer, and Børge Kvingedal (May 2014). *Hywind Buchan Deep Metocean Design Basis*. Tech. rep. RE2014-002. Statoil.

Minnebo, Joerik, Pieter Aalberts, and Arun Duggal (June 2014). "Mooring System Monitoring Using DGPS". In.

Moghadam, F.K., G.F.d.S. Rebouças, and A.R. Nejad (2021). "Digital twin modeling for predictive maintenance of gearboxes in floating offshore wind turbine drivetrains". In: *Forsch Ingenieurwes 85*.

Nwankpa, Chigozie et al. (2018). "Activation functions: Comparison of trends in practice and research for deep learning". In: *arXiv preprint arXiv:1811.03378*.

Olje- og energidepartementet (May 2022). *Kraftfull satsing på havvind*.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pham, Hong-Duc et al. (2019). "Methodology for modeling and service life monitoring of mooring lines of floating wind turbines". In: *Ocean Engineering* 193, p. 106603. ISSN: 0029-8018.

Popper, Jens et al. (Oct. 2018). *Artificial intelligence across industries - IEC Whitepaper*. Tech. rep.

Psichogios, Dimitris C. and Lyle H. Ungar (1992). “A hybrid neural network-first principles approach to process modeling”. In: *AIChE Journal* 38.10, pp. 1499–1511.

Rai, Rahul and Chandan K. Sahu (2020). “Driven by Data or Derived Through Physics? A Review of Hybrid Physics Guided Machine Learning Techniques With Cyber-Physical System (CPS) Focus”. In: *IEEE Access* 8, pp. 71050–71073.

Ren, Zhengru et al. (2019). “Integrated GNSS/IMU hub motion estimator for offshore wind turbine blade installation”. In: *Mechanical Systems and Signal Processing* 123, pp. 222–243. ISSN: 0888-3270.

Riemer-Sørensen, Signe et al. (2019). “Data-Driven Prediction of Vortex-Induced Vibration Response of Marine Risers Subjected to Three-Dimensional Current”. In: *Nordic Artificial Intelligence Research and Development*. Ed. by Kerstin Bach and Massimiliano Ruocco. Cham: Springer International Publishing, pp. 78–89. ISBN: 978-3-030-35664-4.

Rosenblatt, Frank (1958). “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological review* 65.6, p. 386.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1986). “Learning representations by back-propagating errors”. In: *nature* 323.6088, pp. 533–536.

Russell, Stuart and Peter Norvig (Jan. 2003). *Artificial Intelligence: A Modern Approach*. Vol. 82.

Samek, Wojciech and Klaus-Robert Müller (2019). “Towards Explainable Artificial Intelligence”. In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Ed. by Wojciech Samek et al. Cham: Springer International Publishing, pp. 5–22. ISBN: 978-3-030-28954-6.

Sansana, Joel et al. (2021). “Recent trends on hybrid modeling for Industry 4.0”. In: *Computers & Chemical Engineering* 151, p. 107365. ISSN: 0098-1354.

Scheu, Matti et al. (Oct. 2018). “Human exposure to motion during maintenance on floating offshore wind turbines”. In: *Ocean Engineering* 165, pp. 293–306.

SINTEF Ocean (Nov. 2019). *RIFLEX 4.16.2 Theory Manual*.

Srivastava, Nitish et al. (2014). “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1, pp. 1929–1958.

Stat-Ease (Apr. 2022). *Design Expert 13*. Version 13.0.10.0.

Szandała, Tomasz (Oct. 2020). *Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks*.

The World Bank (2019). *Going Global: Expanding Offshore Wind To Emerging Markets*. Tech. rep.

Trinomics (2020). *European Commision - Energy costs, taxes and the impact of government interventions on investments*. Tech. rep.

Van Houdt, Greg, Carlos Mosquera, and Gonzalo Nápoles (Dec. 2020). “A review on the long short-term memory model”. English. In: *Artificial intelligence review: An international survey and tutorial journal* 53.8, pp. 5929–5955.

Van Rossum, Guido and Fred L. Drake (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace. ISBN: 1441412697.

Vapnik, Vladimir (1991). "Principles of risk minimization for learning theory". In: *Advances in neural information processing systems* 4.

Viola, Jairo and YangQuan Chen (2020). "Digital Twin Enabled Smart Control Engineering as an Industrial AI: A New Framework and Case Study". In: *2020 2nd International Conference on Industrial Artificial Intelligence (IAI)*, pp. 1–6.

Voie, P. and E. Lone (2021). *QATS*. `https://github.com/dnvgl/qats`. Version 4.9.2.

Walker, Jake et al. (2021). "Digital twins of the mooring line tension for floating offshore wind turbines to improve monitoring, lifespan, and safety". In: *Journal of Ocean Engineering and Marine Energy*.

Wu, Tim et al. (2014). "Emulating facial biomechanics using multivariate partial least squares surrogate models". In: *International Journal for Numerical Methods in Biomedical Engineering* 30.11, pp. 1103–1120.

Ying, Xue (Feb. 2019). "An Overview of Overfitting and its Solutions". In: *Journal of Physics: Conference Series* 1168, p. 022022.

Zheng, Alice and Amanda Casari (2018). *Feature engineering for machine learning: principles and techniques for data scientists*. " O'Reilly Media, Inc.".

Zidan, Jasmine et al. (2021). "GNSS Vulnerabilities and Existing Solutions: A Review of the Literature". In: *IEEE Access* 9, pp. 153960–153976.

Appendix

A Data exploration

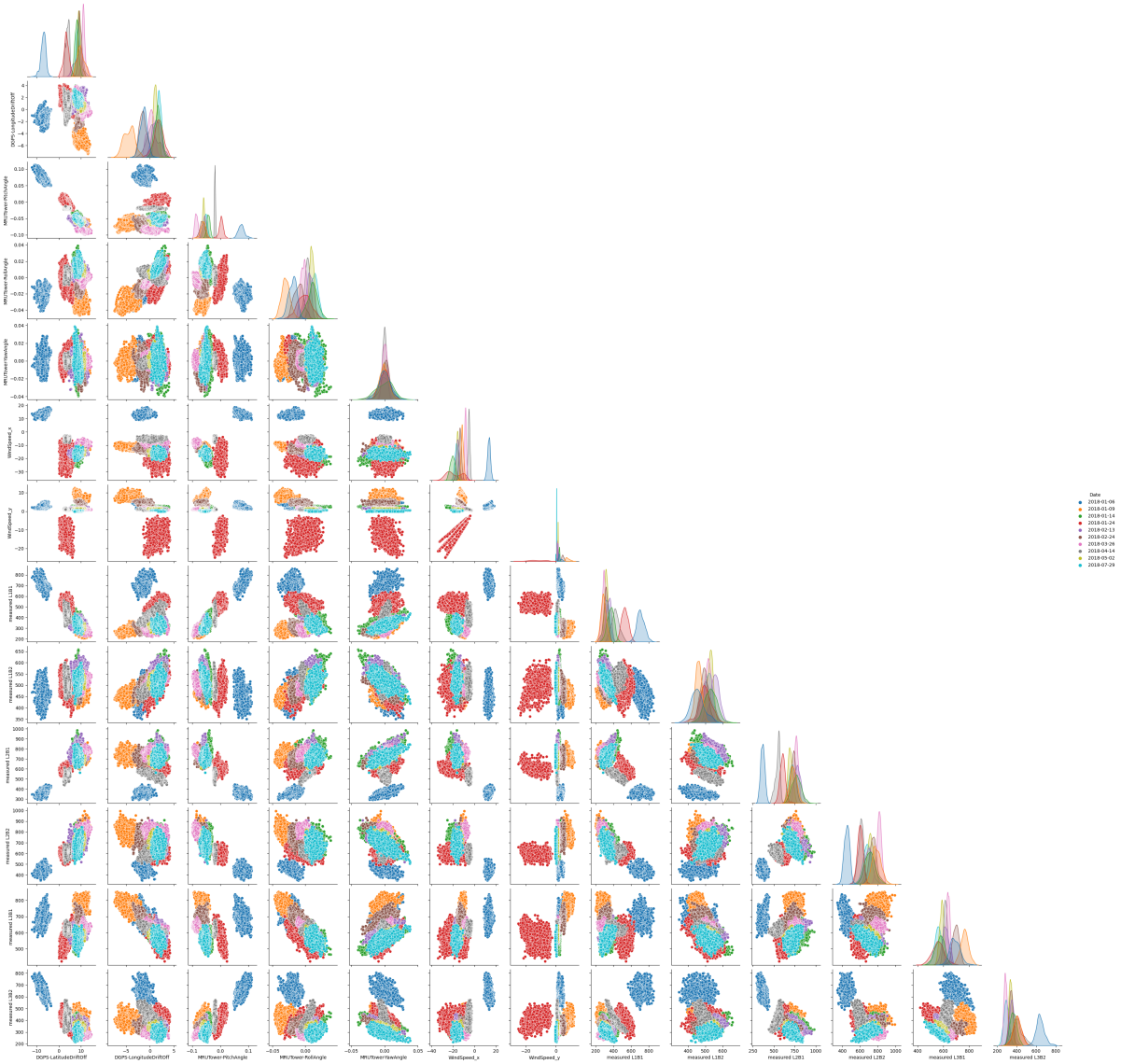
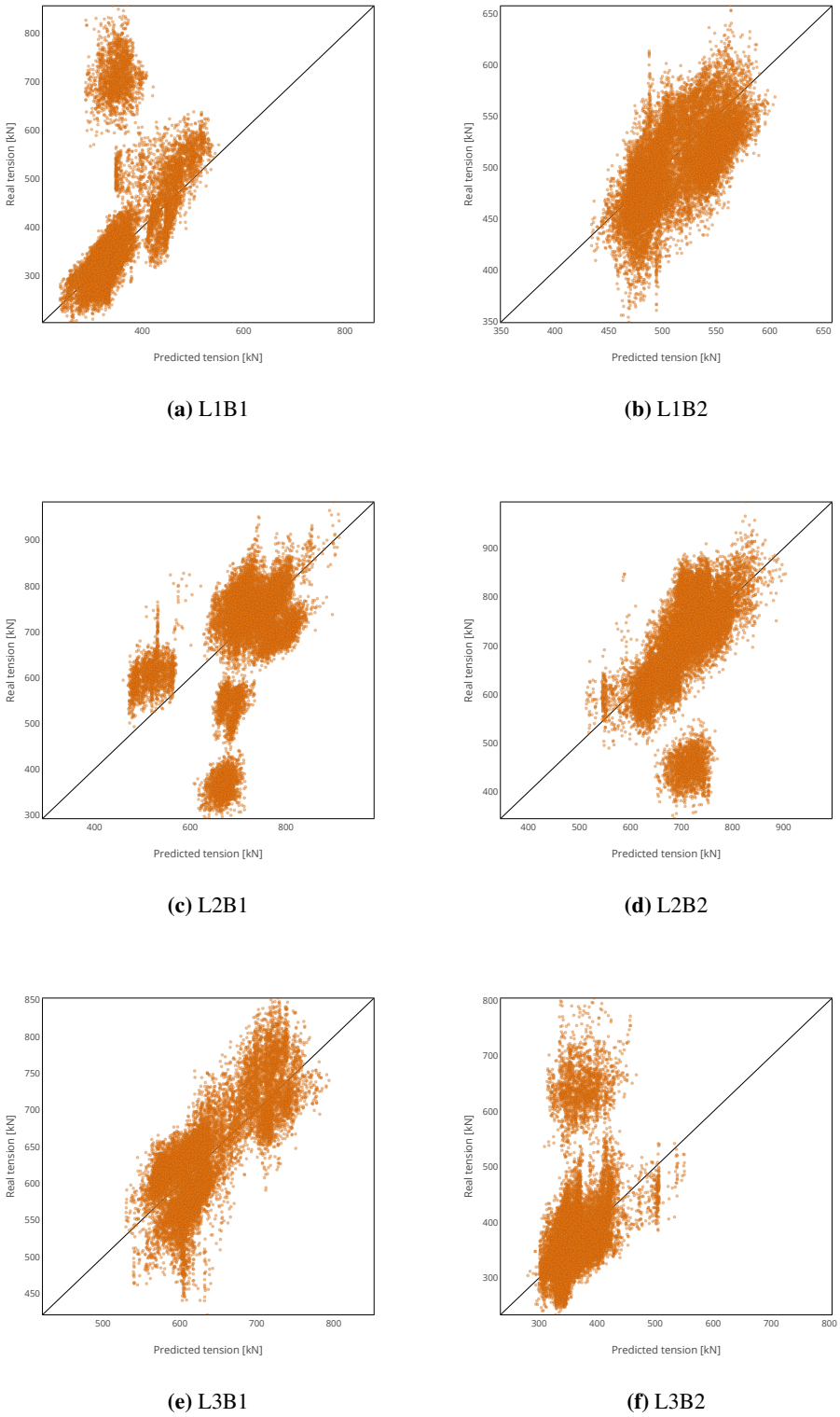


Figure 1: Distribution and correlation plot. For higher resolution image see attached zip file.

B Additional results

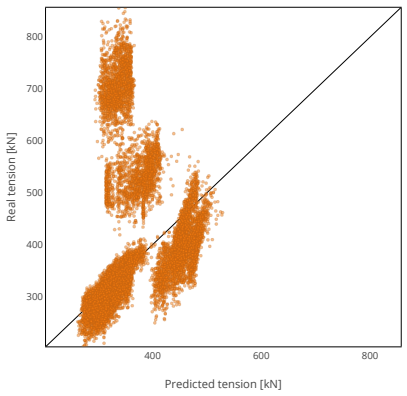


(a) L1B1 **(b)** L1B2

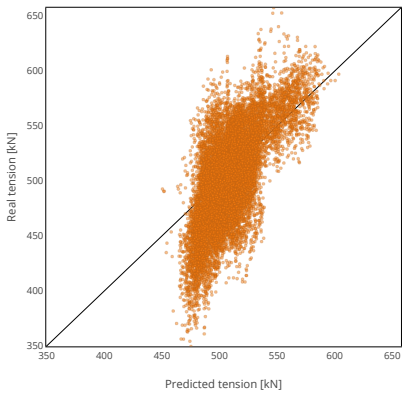
(c) L2B1 **(d)** L2B2

(e) L3B1 **(f)** L3B2

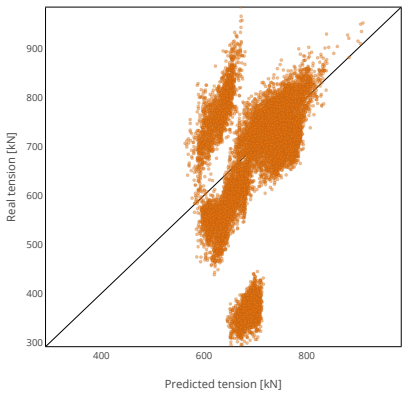
Figure 2: Random Forest, DDM, predicted tension vs measured tension for all lines



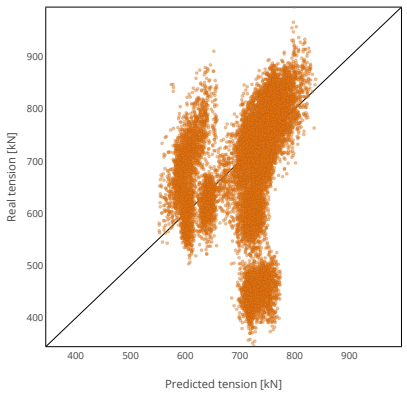
(a) L1B1



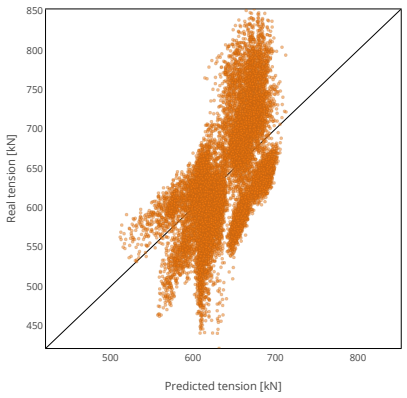
(b) L1B2



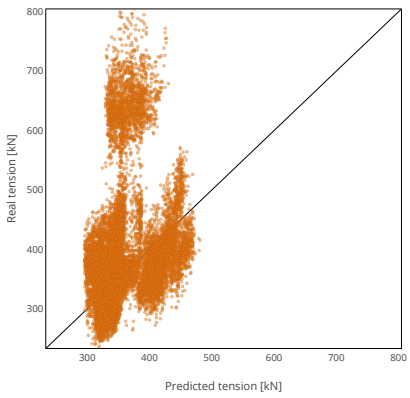
(c) L2B1



(d) L2B2

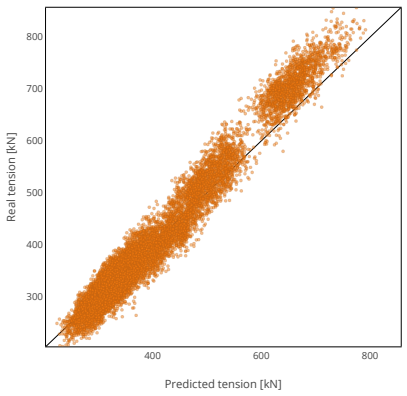


(e) L3B1

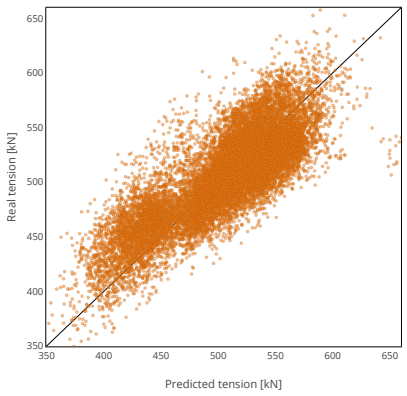


(f) L3B2

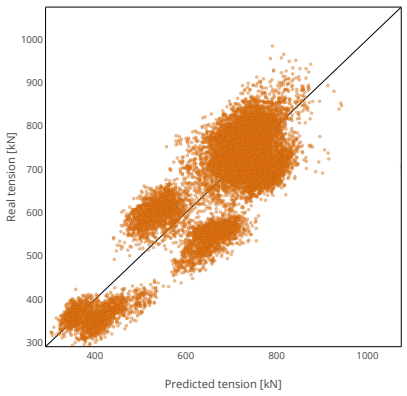
Figure 3: Random Forest, DDM with meta data, predicted tension vs measured tension for all lines



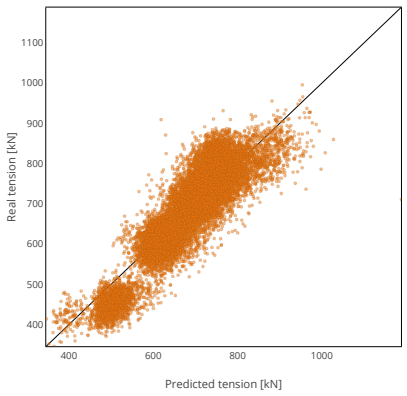
(a) L1B1



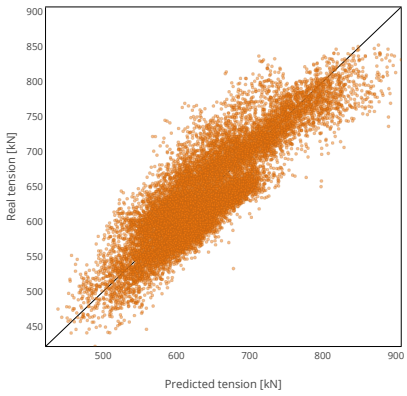
(b) L1B2



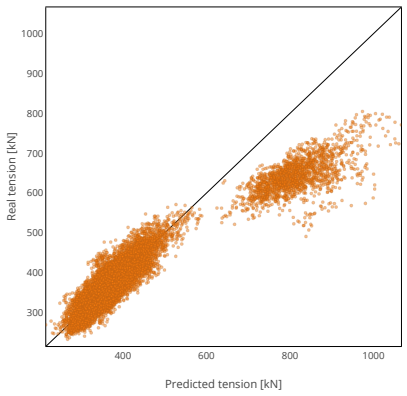
(c) L2B1



(d) L2B2

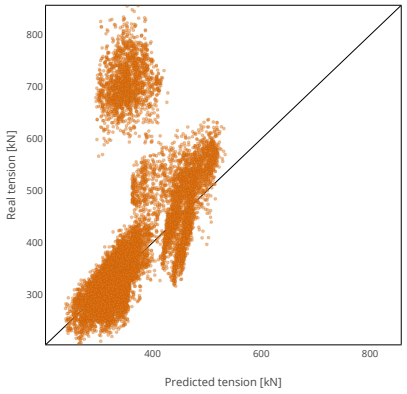


(e) L3B1

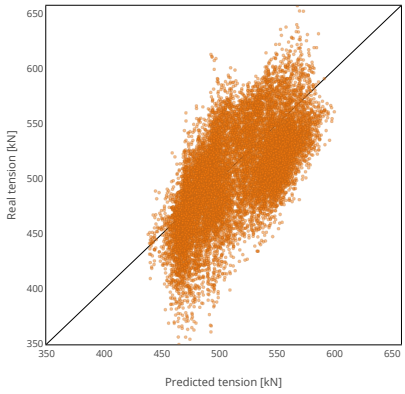


(f) L3B2

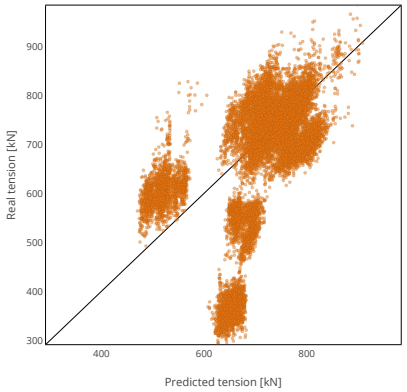
Figure 4: Random Forest, hybrid method A, predicted tension vs measured tension for all lines



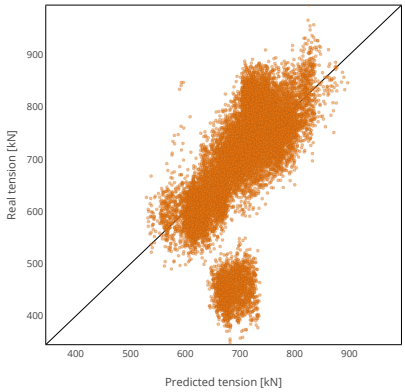
(a) L1B1



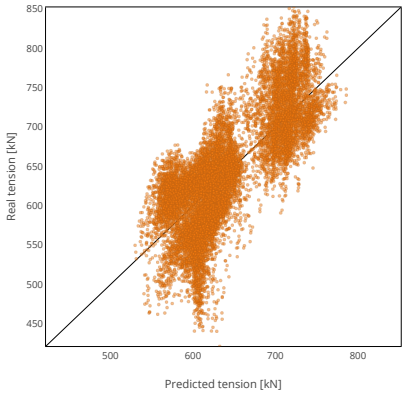
(b) L1B2



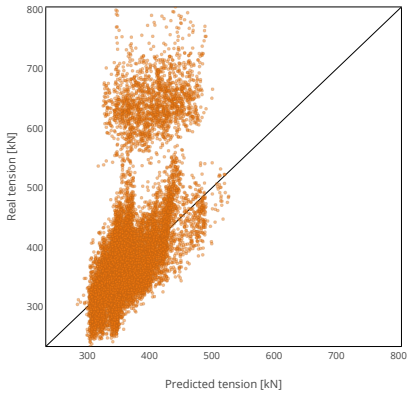
(c) L2B1



(d) L2B2

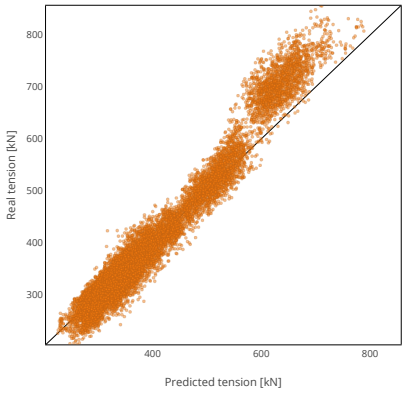


(e) L3B1

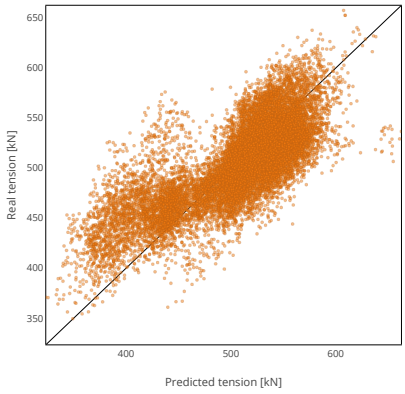


(f) L3B2

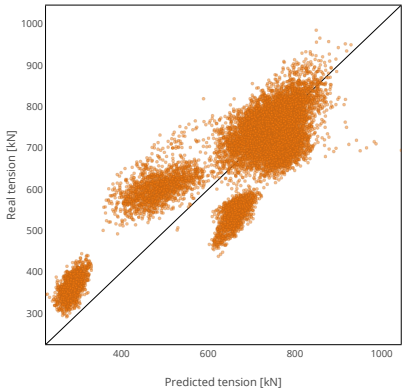
Figure 5: Random Forest, hybrid method B, predicted tension vs measured tension for all lines



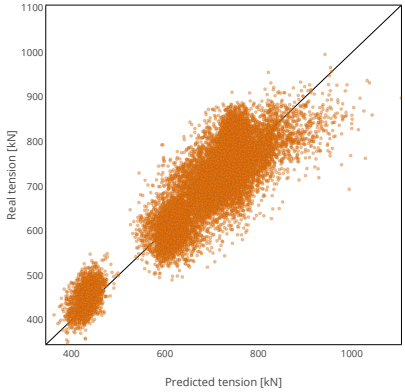
(a) L1B1



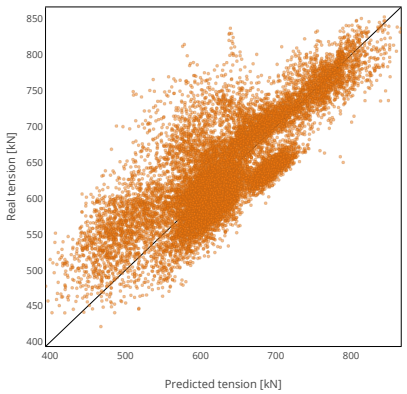
(b) L1B2



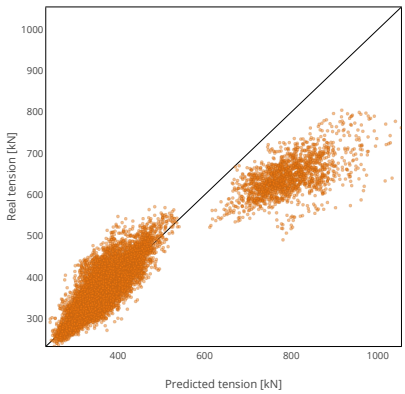
(c) L2B1



(d) L2B2



(e) L3B1



(f) L3B2

Figure 6: LSTM, hybrid method A, predicted tension vs measured tension for all lines

Haakon Lyngstad & Jørgen Engelsen

Hybrid modelling for mooring system monitoring

NTNU
Kunnskap for en bedre verden