Predicting motion sickness

Predicting motion sickness for crew transfer vessels using modelling and data mining techniques

E. Hoogerwerf

Technische Universiteit Delft

TU Delft

A crew transfer vessel approaching an offshore high voltage substation in the wind farm Luchterduinen, near Noordwijk aan Zee.

MSc thesis Applied Mathematics

"Predicting motion sickness for crew transfer vessels using modelling and data mining techniques"

Erwin Hoogerwerf

Technische Universiteit Delft

Supervisor

Prof. Dr. Ir. K.I. Aardal

External supervisor

Ir. G.H. Hulscher

Other committee members

Dr. Ir. G.N.J.C. Bierkens

Thesis defence August 27, 2018 Delft

Abstract

In recent years, the offshore wind industry has grown significantly. The wind turbines are constructed in wind farms, and are serviced with relatively small crew transfer vessels. These vessels transport repair crews from the mainland to the farms and back within a day. One of the big challenges is that these transits can cause motion sickness. If one crew member gets sick, the vessel is legally required to return to the harbour, without performing the repair. It is of interest to predict motion sickness, since both a failed repair and a broken windmill that could have been repaired are costly. This can be done by predicting the motion sickness incidence (MSI), which is an objective measure for motion sickness. The goal of this thesis is to predict MSI using the wave conditions measured in or near the wind farms, by means of an analytical and numerical approach using machine learning. Testing is done on two sites: the Greater Gabbard wind farm and the Westermost wind farm. The analytical approach relates the recorded wave spectrum to the motion spectrum of the vessel, but proves infeasible as a stand-alone method. Gaussian regression, tree ensemble regression and neural networks with Bayesian regularization and backpropagation are the best performing machine learning techniques for MSI prediction. We conclude that the methods from this thesis have sufficient accuracy, by comparing our results with those from Ørsted, whose model is already successfully deployed.

Contents

Acknowledgements

For this thesis I would like to thank everybody who has helped me in creation of this thesis. In particular I have had a lot of help from my daily supervisors Damir Vucovic and Gijs Hulscher. I had several fruitful discussions with Damir regarding vessel behaviour and on writing the thesis in general, whilst Gijs showed me the importance of this research for the current maritime industry. I also want to thank my supervisor Karen Aardal for providing extensive remarks on the thesis. Finally, I am grateful that Ørsted provided the results from their model, to get a qualitative indication of the performance of the model resulting from this thesis.

1 Introduction

For centuries mankind has sailed the seas and oceans, and for just as long many sailors have suffered from rough seas. Whilst the ocean poses barely any threat to modern vessels, motion sickness still causes trouble on smaller vessels. A big reason for this is that motion sickness can only be partially treated by medicine, and even veteran sailors are still susceptible to this sickness. Motion sickness has been studied for quite some time as a result, but currently available medicine is not applicable to workmen as it makes you drowsy. This is problematic, since they need to perform tasks requiring concentration, and can result in additional safety risks for sailors, since e.g. they could lose their footing. Fortunately for those spending longer shifts at sea, people become immune to the effects of motion sickness over time: motion sickness increases for three hours, and finally recedes after a day. According to research by Bles [2], this is because the body is used to only experiencing gravity, which is a constant force. When entering a ship, the mind still expects constant gravity, but experiences a sinusoidal force at best and a rather arbitrary force at worst. This causes a mismatch between the motions that the mind expects and those the body experiences, causing sickness. As the body gets used to the sea motions over time, the motion sickness diminishes. When a sailor gets back to shore, another motion mismatch occurs, albeit less severe.

When projecting the problem of motion sickness for small vessels in the current offshore industry, we see that this is especially impactful for the offshore wind industry. Contrary to many other industries, the offshore wind industry does not use big mining platforms reachable by helicopter. Instead, they work with windmills that are only reachable via small crew transfer vessels or bigger ships with motion compensated gangways. Smaller crew transfer vessels are used for wind farms near the coast, since these are less expensive to operate. These ships are not only used during construction, but are also used for maintenance. The latter is especially important, since a wind farm needs to be serviced for longer periods of time, and a disabled windmill results in a large loss of revenue: up to several thousands of Euros per day. This means it is pivotal to repair disabled windmills as soon as possible.

The vessels used for crew transport are fast catamarans that can get the crew to the offshore work sites in 60 to 90 minutes. For an offshore operation, a crew can travel to the site, perform the operation, and get back to shore within a day. This short travel time means that the operator can choose on which days to work, and when a vessel should not leave the harbour. Whilst it is usually important to perform the offshore operation as soon is possible, the repair itself is also costly. Therefore it is important to have some idea of the likelihood of success, and make the decision on whether or nor to attempt a repair at any given day and time based on that.

Currently, the decision whether or not to perform an offshore mission is based on an estimation of the mission success rate made by experts. These experts are vital for making this decision, due to the unpredictable nature of the sea, and the difference between workable and totally not workable is usually small. This has some major disadvantages:

- The number of correct decisions depends heavily on the skill of the expert.
- The expert can be temporarily unavailable, due to e.g. sick leave.
- The expert can leave the company, taking all of his expertise with him.
- After an unsuccessful mission the expert has only his reputation to defend his decision to perform the mission. This can lead to too little risk being taken.

Especially the expert leaving the company can significantly hamper productivity. Thus if we can support this decision process with a computer program that can yield comparable results, a major leap in productivity is feasible. Furthermore, anyone who wants to join the crew can base his decision on the expected sea conditions.[1]

Most of the prior research done in the field is focused on testing how many volunteers vomit after being exposed to controlled motions. These results form statistics, and only make sense when averaging over larger groups of people. Ships, on the other hand, usually operate with small crews. Motion sickness largely depends on how strong the stomach of the crew is, and how much they had for breakfast, rather than the actual motions. Consequently, no attempt to predict the sickness directly is made. Rather, we introduce an objective criterion called the motion sickness incidence (MSI) , which does not take the crew into account, but statistically agrees with the likelihood of actually falling ill.

The goal of this thesis is to build a working prototype that can predict motion sickness incidence. This prediction is based on the motions experienced during transit from the harbour to the wind farm, and subject to ship type and the weather forecast.

Efforts to solve this problem are made both analytically, by using machine learning and, more specifically, via neural networks. We try to answer the following subquestions:

1. How do we get an objective measure for motion sickness?
2. Can we predict motion sickness analytically?
3. How do we decide what a good prediction algorithm is, and what what is a good MSI prediction scheme?
4. What is machine learning, how does it work, and which specific techniques are available to solve our problem?
5. What are neural networks, how do they work and how can we apply them to our MSI prediction?
6. Do the results from any of these approaches predict MSI with sufficient accuracy?

In order to answer these questions, this research is done in accordance with BMO measurement solutions. BMO is a company that provides vessel performance data to

[1] Crew selection by the wharf or any other party is legally not allowed at this time.

companies active in the offshore wind industry. This allows their clients to evaluate the performance of vessels that they have on contract. In particular, BMO has several years of vessel performance data available from crew transfer vessels that supply various wind farms. It is this data that is used to train and test the various models for MSI prediction in this thesis. The data used in this thesis mostly comes from vessels serving the Greater Gabbard offshore wind farm, which is owned by SSE. SSE is a large energy company based in the United Kingdom. Additionally, we use energy density spectra as well as several other parameters which are provided by the Centre for Environment, Fisheries and Aquaculture Science (Cefas), also from the United Kingdom. Finally, model comparisons are made with a model currently being developed by Ørsted, which is an energy company based in Denmark. Ørsted owns several offshore wind farms, and already uses their model to improve the uptime of their windmills.

The next section features a more in-depth introduction to motion sickness as well as the wave energy spectrum, which is an essential tool in understanding both what MSI is and what the underlying physics of the problem are. We also formally define how to compute the MSI from a crew transit, and show a more convenient way to compute this directly from the energy spectrum. Section 3 discusses a case study with additional motivation for MSI prediction, by showing that there is ample room to improve vessel utilization. This leads to the formal problem definition in Section 4. In Section 5 an analytical approach to the problem is given, where we consider the sea behaviour, and try to compute expected vessels motions. In theory, knowing these motions, one could compute the MSI perfectly. Unfortunately, the analytical approach does not provide the motions we would measure at sea, and therefore the remainder of this thesis is focused on machine learning techniques to solve this problem. The basics of machine learning are introduced in Section 6. Section 7 introduces dimension reduction techniques called principle component analysis and independent component analysis, which are used for identifying the variables relevant for predicting MSI. Some specific regression models from machine learning that prove useful for MSI predicting are introduced in Section 8, with neural networks explained separately in Section 9. The results from applying the machine learning algorithms as well as the neural networks are shown in Section 10. This is done by evaluating the errors made by either approach, and seeing how they relate. Also, we provide a comparison with the data from Ørsted. Ørsted has prediction data for the Westermost wind park, which is similar to the Greater Gabbard wind farm. By comparing all of the approaches in Section 11, we conclude that the best model is an artificial neural network with an algorithm called Bayesian regularization with backpropagation, and show that the errors made are acceptable for our application. An overview of all abbreviations and the most important symbols used throughout the thesis is given in Appendix C.

2 Preliminaries

This section introduces the concepts that lie at the base of our problem: What is motion sickness, and what prior research exists for predicting motion sickness? To express motion sickness as a number, we first introduce the wave energy spectrum. This gives us a different way of describing motions, which is better at describing statistical properties. Then we show how MSI is defined as well as how to compute MSI using an energy spectrum. This last step is crucial, as later on we will attempt to predict the motion spectrum of a transfer vessel, without knowing the motions themselves.

2.1 Spectral wave information

Whilst on board of vessels, motion sickness appears to be largely affected by vertical motion, whilst forward and sideways motion have only a minor influence [19]. The vertical motion of the vessel is naturally related to the sea motions, and sea motions can be described using one of the most powerful tools in wave analysis: the wave energy spectrum. The energy spectrum is defined as the amount of energy associated with different frequencies at which the waves move, and hence describes the strength of a wave that occurs at a specific frequency. This frequency can be either in the spatial or temporal domain. It is even possible to combine the two, but here we assume to be working over the temporal domain. We start with the discrete spectrum, since a computer can only perform computations on discrete values. It can be interpreted as follows: If we associate a value of 1.44 to the energy spectrum at $f = 0.5$Hz for a 1-dimensional signal S, and assume the spectrum is zero everywhere else, then this corresponds to a wave oscillating between ± 1.2 with a period of two seconds. It can be represented by

$$\eta(x) = 1.2 \sin(\pi t - \psi). \tag{2.1}$$

Here $\psi \in [0, 2\pi]$ represents the phase of the wave. When the spectrum is non-zero for

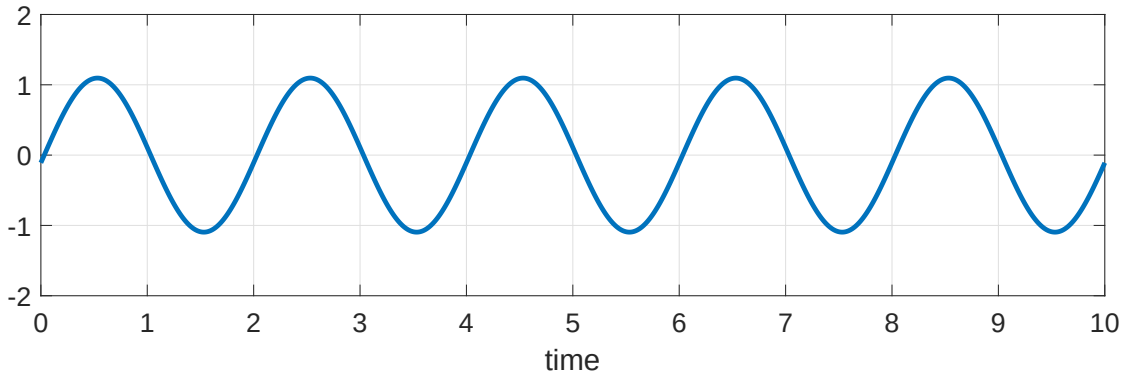


Figure 2.1: The plot associated with equation 2.1 ($\psi = 0$).

multiple frequencies, the resulting wave is a superposition of the waves associated the

individual frequencies, each with their own phase. The resulting waves are usually hard to interpret. An example is given in Figure 2.2. However, nature seldom uses discrete spectra, which is why we usually consider those signals continuous. Since we can only measure the signal discretely, however, we usually sample this continuous spectrum on a discrete set of frequencies. The result of this procedure is similar to the generalization from discrete to continuous probability: we describe the continuous spectrum using the energy spectral density distribution. The main difference for the continuous case is that

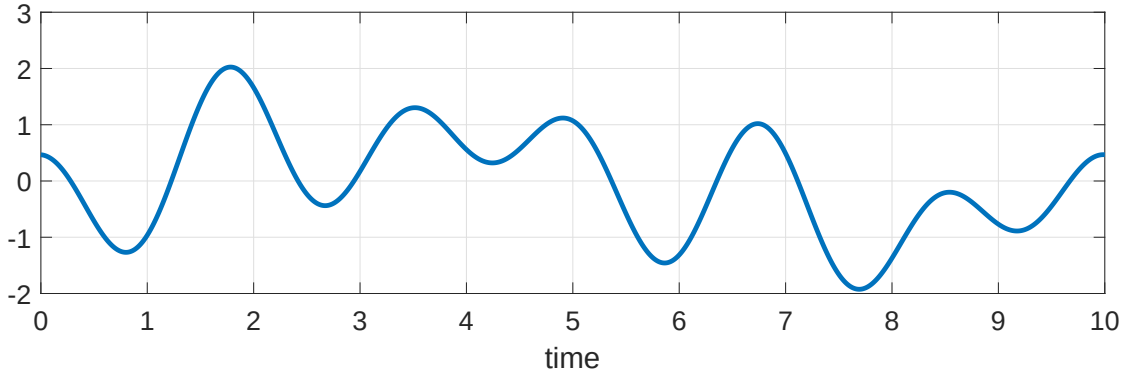


Figure 2.2: A superposition of 3 waves defined by $\boldsymbol{f} = \{0.1, 0.4, 0.6\}$, $\mathbf{E} = \{0.6, 0.4, 0.8\}$ and $\boldsymbol{\psi} = \{0.4, 3, 5\}$.

when we want to retrieve the surface, we need to add another factor $\Delta\omega$. Here ω is the radial frequency, such that $\Delta\omega = 2\pi\Delta f$ with Δf the sample frequency. Assuming we have N different frequencies over which we sample, we end up with the following equation for a wave surface:

$$\eta(t) = \sum_{n=1}^{N} \sqrt{E(\boldsymbol{\omega}_n)\Delta\omega}\cos(\boldsymbol{\omega}_n t + \boldsymbol{\psi}_n). \tag{2.2}$$

Since this work does not contain any further need for discrete spectra, no further distinction between energy spectra or energy density spectrum is made for the remainder of the work. The energy spectrum can be computed using the fast Fourier transform (FFT). An example of an energy spectrum for real data is given in Figure 2.3. The reason for using an energy spectrum rather than a regular spectrum, is that the former represents the actual energy that is stored inside the wave, and is also the convention in the field. The FFT is a fast algorithm that performs the discrete Fourier transform given by

$$\mathcal{F}(\boldsymbol{x})_k = \sum_{n=0}^{N-1} \mathbf{x}_n W_N^{nk} \tag{2.3}$$

$$W_N = e^{-2\pi i/N}. \tag{2.4}$$

Here x_n represents signal strength at time n, and W_N^{nk} the nk'th power of W_N. For a 1D signal, the energy spectrum $S(f)$ or $S(\omega)$ is then computed from the (usually complex)

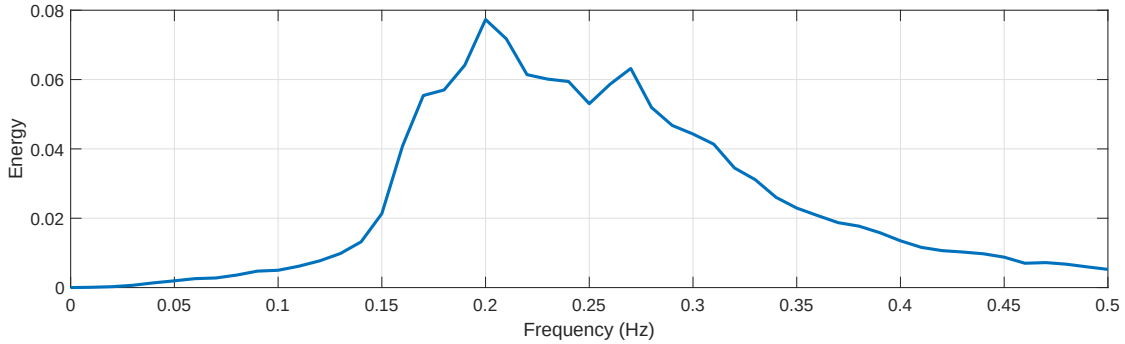


Figure 2.3: Example of an energy spectrum for heave motion.

Fourier transform using constant sampling rate SR via

$$S(f) := \frac{2}{SR} \left(\mathcal{F}(\mathbf{x})_{f/SR}\right)^2 . \tag{2.5}$$

Here we use the Euclidean norm. To obtain $S(\omega)$ recall that $\omega = 2\pi f$, and f/SR is the division of the frequency by the sampling rate to obtain some k from Equation (2.3). In case higher dimensional data is provided, such as wave height in square area, then the FFT yields the amount of energy associated with different directions as well as the frequencies they correspond to. The various frequencies are also referred to as the wave components. Therefore, any such reference concerns the spectral domain. Finally, the temporal domain can also be included to get 3D spectral data. This is the extension of the 1D spectrum to also include spatial data. For our purposes, the 3D spectrum $S(\omega, \theta)$ is defined as the amount of energy associated with a frequency in a certain direction/orientation. This orientation can either be absolute, i.e. with respect to the earth's north, or it can be relative to the vessels heading. If this is not specified explicitly, we always refer to the absolute orientation. Note that the name 3D is a little misleading here, as only two parameters are given. The missing information is the spectral information in the spatial dimension, which is not given here. However, there is no need for such spectral information, as it can be easily computed via the dispersion relation, which relates temporal wave frequency (when fixing space) to the spatial wave frequency (when fixing time). This implies that the knowing the spectrum with respect to time allows one to compute the spectrum with respect to space. This dispersion relation is discussed in detail in the relevant chapters.

2.2 Related work

No related work exists on predicting motion sickness in the offshore industry. There also is no existing work in literature that provides a mathematical model of the process, and only few that even mention the problem. However, there exists some prior work on computing MSI based on the heave motions of a ship, and the two most important works on this topic are treated below.

2.3 MSI computation

For our work we are interested in an objective evaluation of the motion sickness incidence. However, as mentioned earlier, most of the prior research is focused on subjective measures. To distinguish between the two, we refer to the objective measure as motion sickness incidence (MSI) and to the subjective measure as the vomiting incidence (VI). The current standard for objective measurement is the ISO 2631/3 standards for whole-body vibration as presented in [19]. This work is a slightly more sophisticated version of the work by Griffin in [8].

In order to estimate the VI, Griffin performed several experiments, and he noticed that problematic motions mostly occur around 1/6 Hz. Thus he proposes to filter out or dampen motions oscillating at frequencies that are further away from 1/6 Hz. Then the VI is obtained by computing the root mean squared error (RMSE) and scale it to reflect recorded vomiting.

The exact computation goes as follows: As input is taken the raw acceleration $\mathbf{a}$ along the Z-axis (vertical), which is a discrete signal of length N. This signal is then transformed into the spectral domain by means of the FFT, thus obtaining $\mathcal{F}(\mathbf{a})$. This allows us to filter out unwanted frequencies by multiplying each frequency component $\mathcal{F}(\mathbf{a})_f$ by $w(f)$, which is given by

$$w(f) = \begin{cases} \frac{f}{0.125} & \text{for } 0.1 \leq f < 0.125 \\ 1 & \text{for } 0.125 \leq f < 0.25 \\ \left(\frac{0.25}{f}\right)^2 & \text{for } 0.25 \leq f \leq 0.5 \\ 0 & \text{otherwise.} \end{cases} \tag{2.6}$$

We can now use the inverse FFT to return the signal $\mathbf{a}' = \mathcal{F}^{-1}(\mathbf{w}\mathcal{F}(\mathbf{a}))$ to the temporal domain. We now compute the VI as

$$VI = \frac{K}{SR}||\mathbf{a}'||_2 = \frac{K}{SR}\left(\sum_{i=1}^{N}(a_i')^2\right)^{\frac{1}{2}}. \tag{2.7}$$

Here, $K = 0.33$ is a scaling indicating the susceptibility of a group to motion sickness. Furthermore, Griffin claims that when looking at men or women separately $K = 0.25$ and $K = 0.4$ should be taken instead.

This is however contradicted by [19], which states that the difference in susceptibility between men and woman is at most 5%. Furthermore, they also provide an updated weighting curve, which is shown in Figure 2.6, and a different constant $K = 1/1.4$. Otherwise, the computation is perfectly identical to the work of Griffin. This way we get the following definition for MSI:

$$MSI = \frac{1}{1.4\sqrt{SR}}\sqrt{\sum_{i=1}^{N}(\mathbf{a}_i')^2}. \tag{2.8}$$

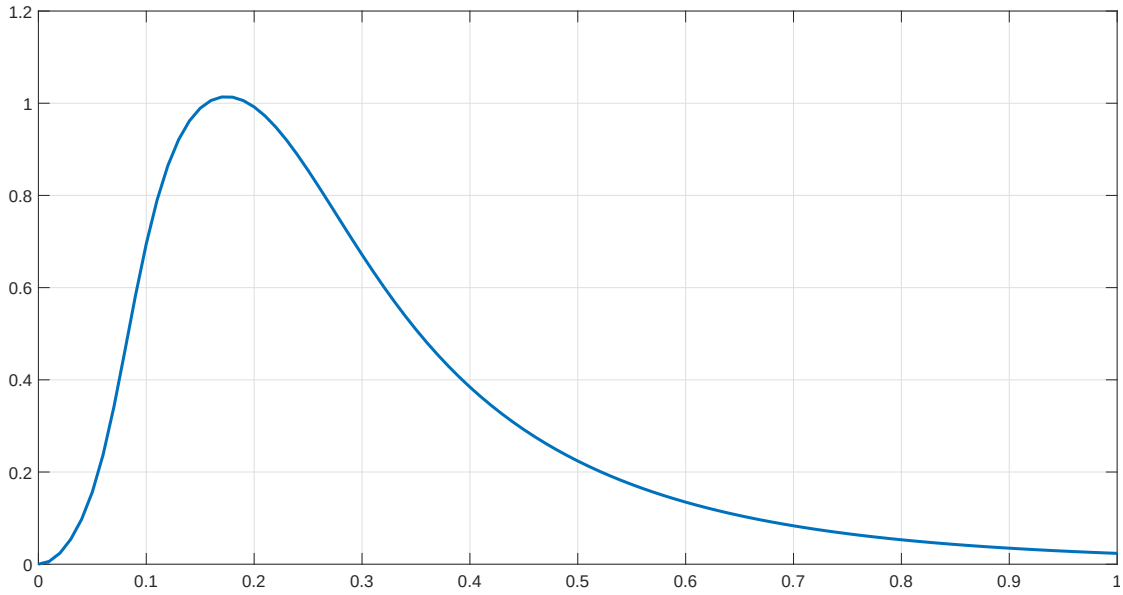


Figure 2.4: Weighting curve $\mathbf{w}$ as a function of frequency f.

The new weighting curve can be found in Figure 2.4. This curve is described by analytical formulas, which means that in order to get the coefficients on the desired frequency interval we can simply sample this distribution. For the formulas themselves I refer the reader to the original work, where they are explicitly stated [19]. The MSI computed by this standard forms the definition in the remainder of this work.

2.4 Computing MSI directly from the spectrum

We have thus far found a formula that allows to compute the MSI, should the original time trace be known. This is, however, not sufficient. Ideally, we want to compute the MSI without the phase information. This is desirable, since the spectra provided by wave radars or buoys do not provide such information either, as such information does not have any statistical meaning. Suppose that the definition of the FFT as in Equation (2.3) is followed, and that the spectrum is in the form of Equation (2.5). Then the MSI (ignoring the constant K) is computed as:

$$MSI^2 = \frac{1}{SR} \sum_{j=0}^{N-1} \left| \frac{1}{N} \sum_{i=0}^{N-1} \mathbf{w}_i \mathbf{Y}_i W_N^{ij} \right|^2 . \tag{2.9}$$

In this definition, we use $Y_i := \mathcal{F}(\mathbf{a})_i$. It is not immediately clear how to compute this, but fortunately Parseval's theorem [14] comes to the rescue:

Theorem 1 (Parseval)**.** *Let* $\mathbf{X} := \mathcal{F}(\mathbf{x})$ *denote the discrete Fourier transform of a signal* $\mathbf{x} \in \mathbb{C}^N$*. Then*

$$\sum_{n=0}^{N-1} |\mathbf{x}_n|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |\mathbf{X}_k|^2. \tag{2.10}$$

A proof is provided in Appendix A.1. We apply this theorem using $\mathbf{x}_i$ (from the theorem) defined as $\frac{1}{N} \sum_{j=0}^{N-1} w_j Y_j W_N^{ij}$. As this equals the inverse Fourier transform of $\mathbf{wY}$, this implies that

$$MSI^2 = \sum_{i=0}^{N-1} \left| \frac{1}{N} \sum_{j=0}^{N-1} \mathbf{w}_j \mathbf{Y}_j W_N^{ij} \right|^2 \tag{2.11}$$

$$= \sum_{i=0}^{N-1} \left| \mathcal{F}^{-1}(\mathbf{wY})_i \right|^2 \tag{2.12}$$

$$= \frac{1}{N} \sum_{j=0}^{N-1} \left| \mathbf{w}_j \mathbf{Y}_j \right|^2. \tag{2.13}$$

Here we use that $\mathcal{F} \circ \mathcal{F}^{-1}$ equals the identity operator. Taking roots on both sides of the Equation, we get

$$MSI = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} |\mathbf{w}_j \mathbf{Y}_j|^2}. \tag{2.14}$$

Note that this definition of MSI is not bounded, but in practice MSI values exceeding even 80% are extremely rare. Yet we have a new problem: If we consider signals with different lengths, they can not be compared. In particular, it is not possible to relate the MSI scores computed via the spectrum with those computed using the signal itself if we do not take the length of the signal into account. Assume the signal spans t hours. Then we define the normalized wave spectrum as

$$MSI_{60} = \frac{MSI}{\sqrt{t}}. \tag{2.15}$$

At this point we have a good understanding of what MSI is, and can therefore formally define the research question of this thesis. However, first we give some further motivation for the need for MSI prediction, as the lack of literature might suggest that MSI prediction is not relevant at all. This is done by answering the all-important question: Is there anything to gain from MSI prediction?

3 Preliminary case study

As mentioned in Section 2 there exists little work on MSI prediction, desirable to get additional motivation as to this topics relevance. We present a case study investigating the deployment of catamarans by SSE from Lowestoft to the Greater Gabbard offshore wind farm in the UK to provide show this relevance. This site is also the main focus for the remainder of the work, and is discussed in more detail in the next section. Two different sea craft are used for deployment at the Greater Gabbard site:

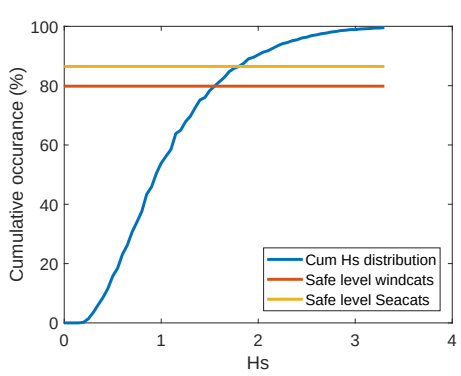
- Windcat, a smaller catamaran (18 meter)
- Seacat, a larger catamaran (26 meter)

This is an interesting case study, as the Windcats were operational for 54 hours, whilst the Seacats operated for 104 hours per month on average. Deploying a Seacat is more costly than deploying a Windcat, which means that we want to deploy the smallest vessel possible. We anticipate that the Windcats are deployed too conservatively, even though the Seacats can operate in rougher circumstances than the Windcats. In order to test this hypothesis, we compare the maximum possible uptime for both vessels. Workability depends on whether the significant wave height H_s exceeds a certain threshold: 1.50 meter for the Windcats or 1.75 for the Seacats.

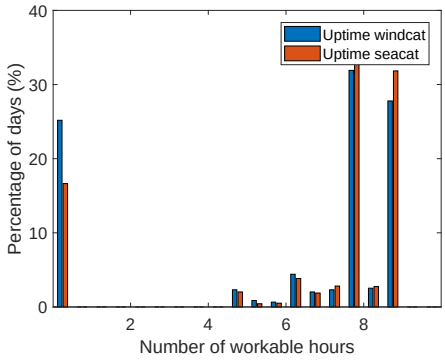
The method to perform this test is as follows: first, we compute the cumulative distribution of the significant wave height over time. This is obtained by first accumulating the significant wave height as measured by a buoy during the a month. The buoy is located in close vicinity to the wind farm. For the purpose of getting a good representation, only workable hours have been taken into account, i.e. from 9.00 till 17.00. This way we obtain a discrete density distribution, which is shown in Figure 3.1a. The thresholds for both vessel types have also been added in this figure. This gives us a first indication of the possible uptime for the vessels. However, it is not a proper indication of when the ships can actually perform work: for this we need a consecutive period of 4 hours in which the workability criteria is matched. If the 4-hour period is interrupted at any point, we consider the entire period unworkable. The reason for rendering the entire period unworkable is because crew transfers need to be save both when leaving the vessel and when boarding it, and the crew should be able to leave at any time. Finding such periods is a windowing problem, and is solved easily. If the longest found workable period is longer than 4 hours, those values are shown instead. The resulting distribution is shown in Figure 3.1b. If the entire day is workable, the workable period is set at 8 hours. The reason we see two peaks is simply because H_s is only recorded every 90 minutes, thus allowing for a different maximum number of workable hours in the computation. The results are shown using 30-minute bins.

In addition to this buoy data, there were also 4 measurement units located in the wind farm itself. They measure the wave height by means of radar. Results vary slightly per radar, but the results in Figure 3.1 are representative for all of these sources.

These results show that the optimal downtime for the Windcat and Seacat are 26 and 17 % respectively (first bin in Figure 3.1b). This implies that the Seacat should be



(a) Discrete cumulative distribution of H_s.



(b) Number of consecutive workable hours per day.

Figure 3.1: Possible uptime of the Windcat and Seacat, which shows that theoretically the vessels could have an uptime of 80% and 88% for the Windcat or Seacat respectively, when they would work every feasible hour. When we consider days where there are at least 4 consecutive workable hours, then we find 65% and 83% days respectively.

deployed 12% more often than the Windcat. As practice shows this to be over 90 %, there is clearly a lot to be gained. If we assume that optimal deployment is 8 hours per day on average (in practice this number has shown to be conservative), the total possible uptime is around 180 hours. Even though we may assume that motion sickness plays a role even under operational limits, it is reasonable to assume that the difference is too large. Hence we conclude there to be a solid case for further investigation on automated decision protocols for deployment of both Windcats as well as Seacats.

These numbers present the total amount of uptime that can be realized. Yet much of these trips are for preventive maintenance, rather than corrective maintenance. The latter is when a turbine is broken and does not produce any power until repaired, and hence is where the real benefit is to be gained. The turbines currently have an uptime of roughly 95%. So let us analyse the possible gain from an improved scheduling algorithm, where we are less likely to stay in the harbour even though it was feasible to go out, and therefore increase the uptime of the turbine. Suppose that a good predictive algorithm can increase the uptime for a turbine by 0.2 percent point on average. Considering there currently is around 5% downtime, this likely is conservative estimate. The Greater Gabbard site produces roughly 1900 GWh annually, which means that a 0.002 increase amounts to some 3.7 GWh per year. This amount of additional energy is equivalent to the consumptions of some 1000 UK households, or 1100 households in the Netherlands [3]. It also results in an additional profit of some €600 000. Thus it is from both an economical as well as societal perspective interesting to apply MSI prediction if this achieves even very minor improvements in the overall turbine uptime. As the Greater Gabbard site forms a little under 10% of the total installed offshore wind capacity in the UK, full-scale deployment can theoretically increase that by another factor ten.

It is my hope that this section is proof that MSI prediction has potential for significant benefit. In the next section we formalize the problem we face in MSI prediction, and more thoroughly introduce the Greater Gabbard site.

4 Formal problem definition

We try to predict motion sickness as is defined in Equation (2.8). We have also seen that reducing the downtime for service vessel can significantly increase the power output of a wind farm. Hence we introduce the the research question of this thesis: How to best predict the motion sickness index, having access only to wave forecasts for any specific vessel using its historical performance in the Greater Gabbard wind field? The restriction to a specific vessel is important as it is common knowledge that the motion sickness is dependent on the vessel characteristics. An easy example is the size of the vessel: large container vessels are only affected by very long waves whilst a rowing boat feels even the smallest of waves. The reason for focussing on the Greater Gabbard wind field is that we have four different sources available for aggregating wave data. This means that it allows for cross-checking the quality of the individual sources (which is already done by Cefas), and means that there are only few moments on which no data is available. Since the data from the four sources is more or less identical for most of the time, this also suggests that the wave conditions are fairly constant, at least throughout the field. Below the Greater Gabbard wind farm is discussed in detail. We conclude by showing that the Greater Gabbard site is representative for a large number of other wind fields, both now and in the foreseeable future.

4.1 Greater Gabbard site

The site used for testing is the Greater Gabbard wind field, which is located roughly 60 km off the English coast. The site is illustrated in Figure 4.1 as the northern patch. The southern patch is the Galloper wind field, and is also used in MSI prediction for transits to the Greater Gabbard site. The main reason for this is that vessels travelling to the Galloper field have to pass the Greater Gabbard site. This means these transits can also be interpreted as transits to the Greater Gabbard site. To normalize, all transits are cut off when they reach any of the outer edges of the Greater Gabbard site. The wave radars that are used to obtain wave spectra for this site are located inside of the Greater Gabbard field. Additionally, a wave buoy is located 1 km north-east of the northern-most tip. This buoy is the only source capable of producing 3D spectra.

In terms of crew transfer vessels that operate on these sites, the Windcat and Seacat are already mentioned in Section 3. Besides these vessels, there are also crew transfer vessels used from Iceni, Malltraeth and Dalby. These are catamarans of respectively 21, 23 and 23 meters long, which makes them the middle ground between the Windcat and Seacat. Upon combining all these transits, we get a dataset totalling 2200 unique transits, although many of these journeys take place on the same days, with only minor differences in sea state. The techniques used in this thesis are not restricted to the Greater Gabbard site, however, as there are ample sites with wave buoys nearby. Figure 4.2a shows the various wave buoys that Cefas operates around the United Kingdom, and a quarter of these buoys also has spectral data available. In Figure 4.2b most of the wind farms near the United Kingdom and the Netherlands are plotted. As a good portion of these fields are all located near the coast, they can be served by the same type

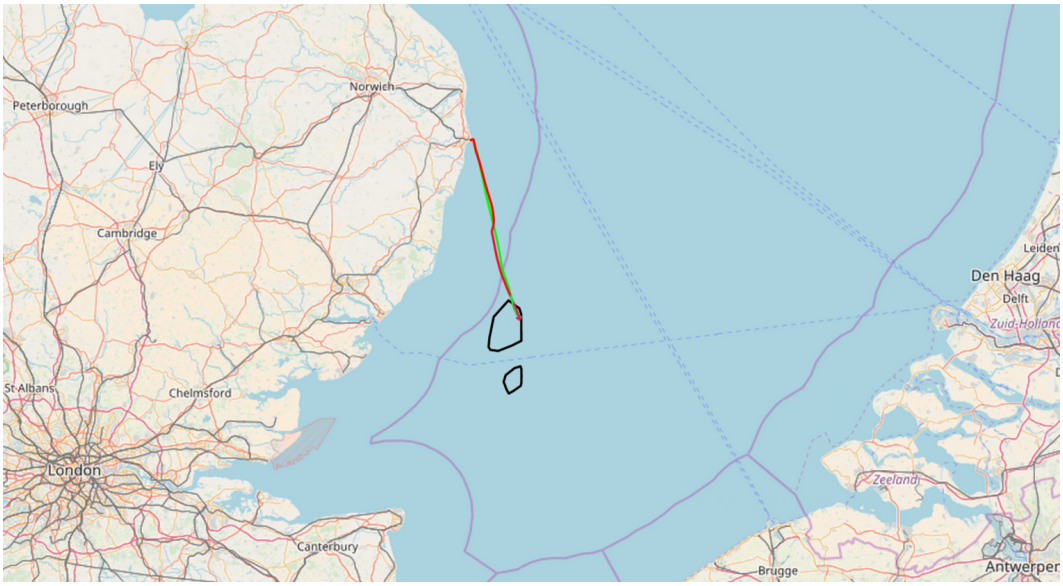
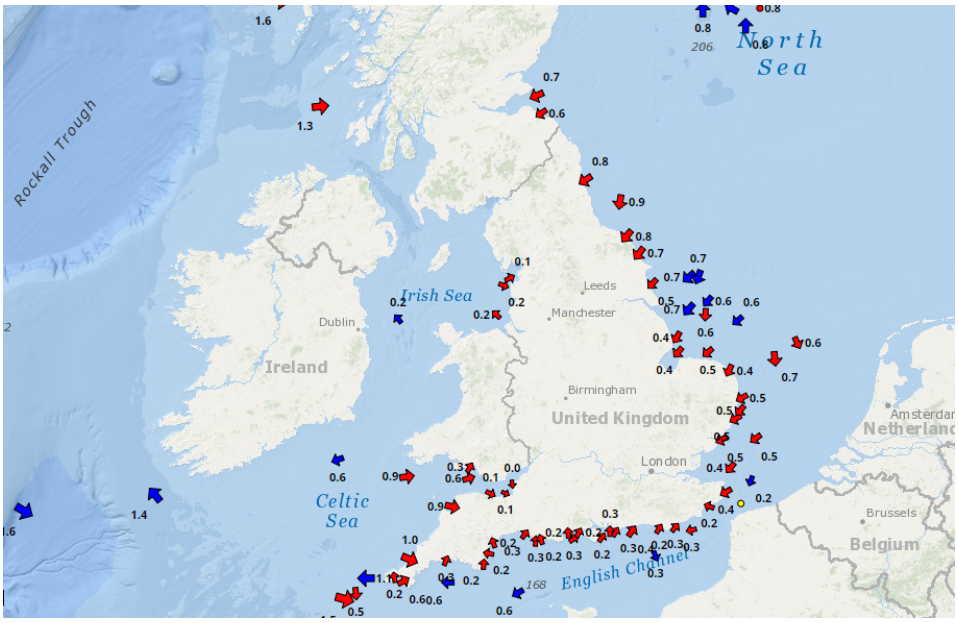
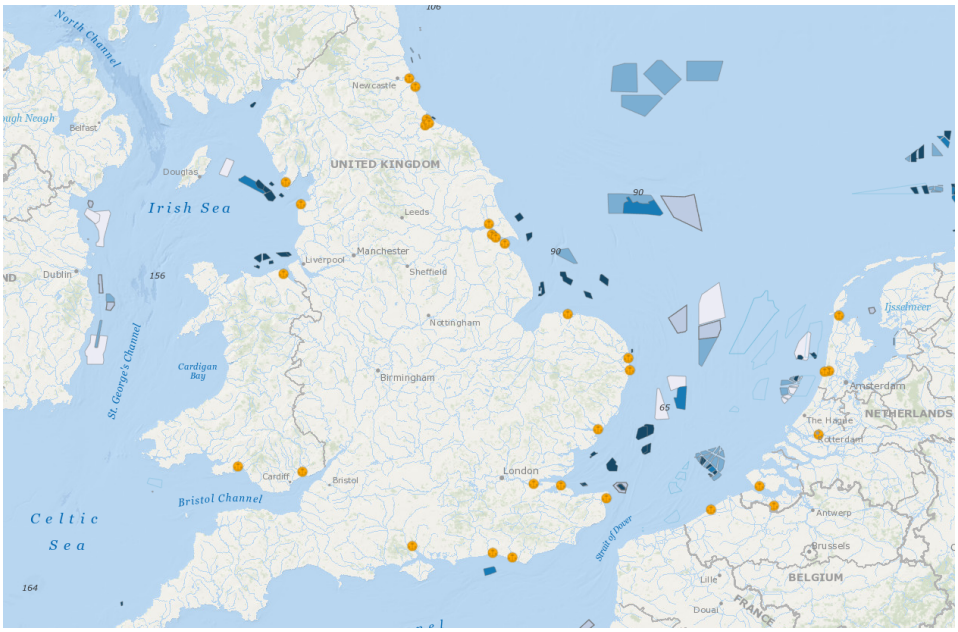


Figure 4.1: A standard transfer from Lowestoft to the Greater Gabbard wind field. The field is painted in black and the journey from start to end in green to red.

of vessels as the Greater Gabbard site. Furthermore, Greater Gabbard also has a size that is representative for the fields that are to be build in the next 7 years [11]. These fields require maintenance for a further twenty years, which means this work should be applicable for at least 25 years on this application alone.



(a) Overview of Cefas wave buoys near the United Kingdom. Blue indicates a buoy with the wind direction while the red arrows indicate buoys in the main wave direction.



(b) Overview of some of the wind fields in the North Sea which are served by crew transfer vessels, as well as the coastal harbours from which they are served. In theory, the methods of this work can be applied to all farms near the shore. [7]

Figure 4.2

5 Theoretical analysis transit

As was explained in Section 2, MSI is computed using the spectral values from the acceleration motion traces. To do so, we first attempt an analytical approach to the problem. Recall from the previous section that while no acceleration data about the sea is known, there is wave data available in the form of heave energy spectra. These spectra describe the energy distribution of the wave surface. They also have the nice property of being easily convertible to acceleration spectra, as will be shown later on, assuming that the vessel in question follows the wave surface. There is one problem, however, as this acceleration spectrum is only valid if the vessel is not moving. In this Section we first show how to retrieve the acceleration spectra that an observer following the wave surface should experience, given that the observer is travelling with a given speed and direction, under some assumptions. Then we explain how to convert these theoretical motions to motions that we expect to observe on a ship. Finally, we test the validity of the assumptions on real data, and conclude that the theoretical approach by itself is not sufficient to solve the problem.

Throughout this section the notation given in Table 5.1 is used.

Table 5.1: Wave parameter explanation

Parameter	Explanation
η	Sea surface elevation from the mean water level.
$\mathrm{E}(\omega)$	Measured energy density as function of ω.
ω	Radial temporal frequency given by $\omega = 2\pi f$, where f is frequency in Hertz.
k	Radial spatial frequency given via $\omega^2 = gk \tanh hk$, where h stands for depth in meters and g the gravitational constant $g = 9.81$.
θ_v	Radial vessel heading.
$\theta_{0,n}$	Radial wave component direction, centred around main wave direction.
ψ_n	Wave component phase, uniformly distributed in $[0, 2\pi]$.
acc_z	Vertical acceleration of the vessel.

5.1 Analysis for a simple wave

Recall that the sea surface elevation η can be interpreted as a superposition of sinusoids with varying amplitudes A_n, directions θ_n and phases ψ_n. To see what is happening, first consider a simple sinusoid with amplitude A, travelling speed s and wave length L. This is also referred to as a simple wave. When considering simple waves it is fine to ignore the phase, as the starting point can be chosen arbitrarily. Let the angle between the wave and ship be given by $\theta \in [0, 2\pi]$, and define the radial wavelength $C := L/S$.

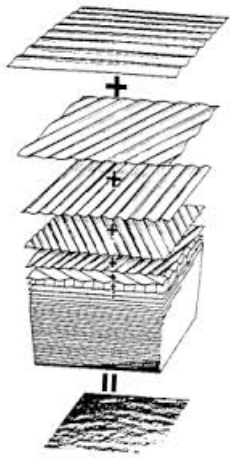


Figure 5.1: A sea can be interpreted as a superposition of sinusoids [15].

As the acceleration is the second derivative of position, this equals:

$$\eta(t) = A\sin(2\pi t/C) \tag{5.1}$$
$$C = L/s; \tag{5.2}$$
$$acc_z(t) = -A(2\pi/C)^2\sin(2\pi t/C). \tag{5.3}$$

Now assume the ship itself also moves with speed s', then this becomes

$$\eta(t) = A\sin(2\pi t/C) \tag{5.4}$$
$$C = L/(s - \cos(\theta)s'); \tag{5.5}$$
$$acc_z(t) = -A(2\pi/C)^2\sin(2\pi t/C). \tag{5.6}$$

Here the limit $s \to \cos(\theta)s'$ corresponds to the case that the speed component of the ship in the wave direction equals the speed of the wave, hence resulting in no further sinusoid behaviour.

On the North sea, default parameters are $L = 150$m, $s = 13$m/s and $s' = 12$m/s. Hence, for $\theta = 0$ we see that $acc_z \approx 0.36H$, while for $\theta = \pi/2$ we see that $acc_z \approx 3.3A$ and for $\theta = \pi$ we can expect peak accelerations of up to $acc_z \approx 5.9H$.

This model clearly is not representative for the experience to be had at sea, as ships will usually have a dampening effect. Additionally, a real sea is usually a superposition of such waves, which results in cancelling terms. How each type of wave affects a ship exactly depends on the response amplitude operator (RAO), which is unique to each ship. Mathematically, the RAO is a function that assigns weights per frequency to describe how much the vessel is affected by a wave with a certain frequency. An example is plotted in Figure 5.2. In general, the RAO is 1 for small frequencies, then increases near its critical frequency, after which it quickly drops to zero. The critical frequency is the frequency where the wavelength equals the vessel length and hence resonance occurs. However, these operators are generally not known. A theoretical derivation on how to

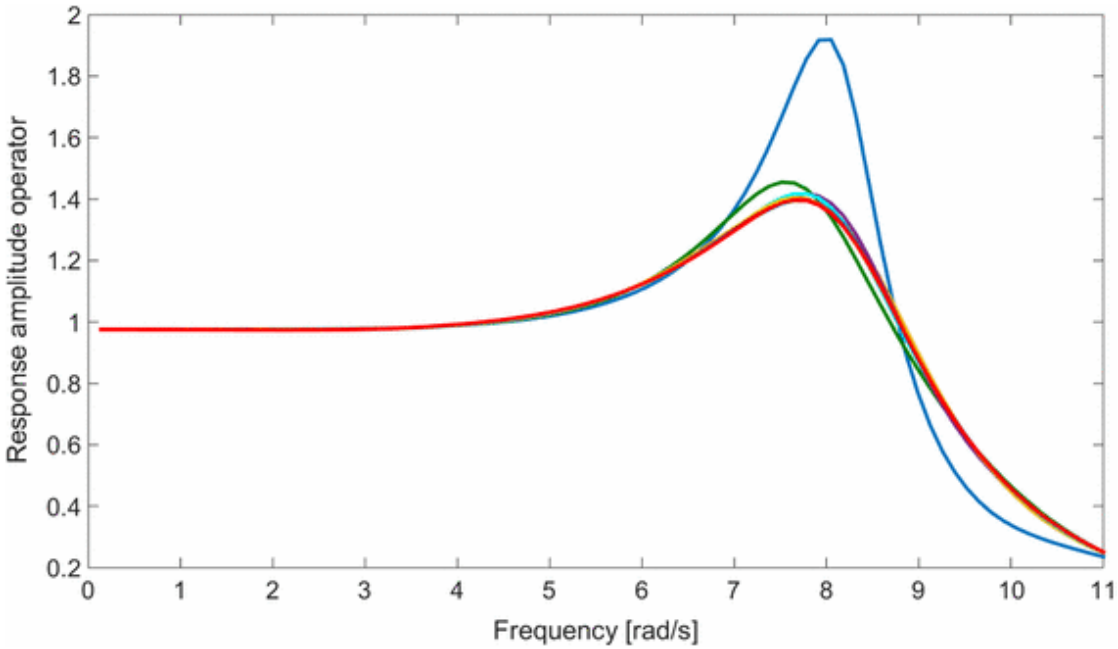


Figure 5.2: Example of several response amplitude operators [6].

compute the RAO's using the ships dimensions falls outside the scope of this thesis, but is possible. Still, I believe it to be fair to conclude that the ship's direction relative to the wave can have a huge impact on the resulting accelerations. Hence this warrants further exploration.

First, we provide an analytical analysis of the sea surface by introducing a numeric representation. Then, we introduce a parametrization of the vessel to link the sea surface to the conditions that the vessels experiences during transit. Next, the practical difficulties and limitations of this process are discussed, and finally, by means of numerical tests on real-life measurements we show that these limitations in practice imply this approach not sufficient for MSI prediction.

5.2 Analytical approach

According to [22], we can represent the ocean surface as

$$\eta(r,\theta,t) = \sum_n \sqrt{E(\omega_n)(\Delta\omega)_n}\cos(k_n r\cos(\theta-\theta_{0,n}) + \omega_n t + \psi_n). \tag{5.7}$$

This means that we interpret the sea surface as a superposition of sinusoids, each corresponding to a different frequency, both in terms of spatial and temporal frequency. The amplitude of each wave component is scaled to match the width of each particular frequency band. An illustration of a wave surface created using this function is shown in Figure 5.3.

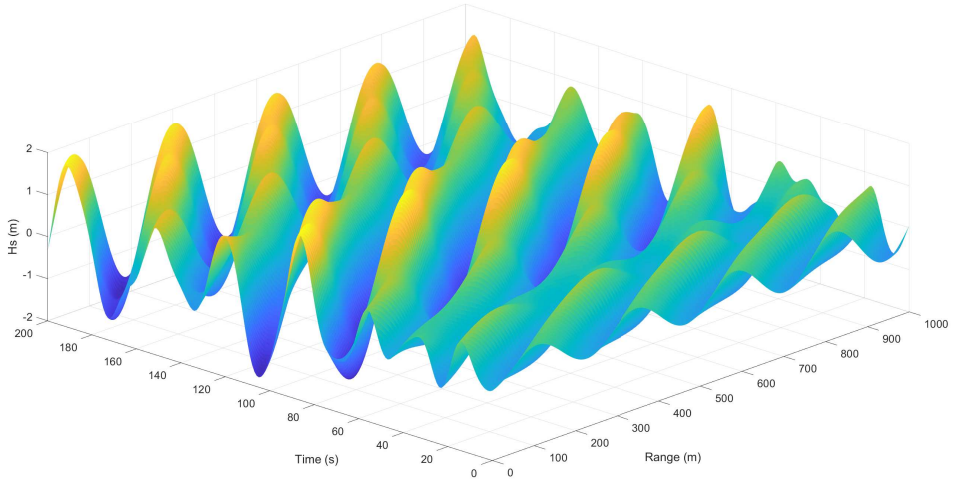


Figure 5.3: Sea surface plot

If we want to simulate a ship passing through this sea with constant speed v_0 and direction $\bar{\theta}$, we can simplify Equation (5.7) as follows:

$$\begin{cases} \eta(r,\theta,t) = \sum_n A_n \cos(k_n r \cos(\theta - \theta_{0,n}) + \omega_n t + \psi_n) \\ A_n = \sqrt{E(\omega_n)(\Delta\omega)_n} \\ r(t) = v_0 t \\ \theta(t) = \bar{\theta} \end{cases} \tag{5.8}$$

$$\Rightarrow \begin{cases} \eta(t) = \sum_n A_n \cos(B_n t + \omega_n t + \psi_n) \\ B_n = k_n \cos(\bar{\theta} - \theta_{0,n}) v_0. \end{cases} \tag{5.9}$$

The step above is valid since θ can be considered constant as we assume the ship not to change its heading. We can also define the position of the ship at $t = 0$ to be equal to $r = 0$. As the vertical acceleration is the second derivative with respect to time, we can now explicitly compute its derivative as

$$acc_z(t) = \frac{\Delta^2 \eta}{\Delta t^2}(t) = -\sum_n A_n (B_n + \omega_n)^2 \cos(B_n t + \omega_n t + \psi_n). \tag{5.10}$$

Thus, the vertical acceleration equals the surface height with each wave component amplified with a factor

$$amp_n := -(B_n + \omega_n)^2 = -(k_n \cos(\bar{\theta} - \theta_{0,n}) v_0 + \omega_n)^2. \tag{5.11}$$

Keeping in mind the dispersion relation for water $\omega^2 = gk \tanh hk$ for depth h, we can

further reduce this to

$$amp_n = -(k_n \cos(\bar{\theta} - \theta_{0,n})v_0 + \sqrt{gk_n \tanh hk})^2 \tag{5.12}$$

$$\approx -(k_n \cos(\bar{\theta} - \theta_{0,n})v_0 + \sqrt{gk_n})^2 \qquad \text{in deep water} \tag{5.13}$$

$$= -k_n(\sqrt{k_n} \cos(\bar{\theta} - \theta_{0,n})v_0 + \sqrt{g})^2 \tag{5.14}$$

$$= -\frac{\omega_n^2}{g^2}(\omega_n \cos(\bar{\theta} - \theta_{0,n})v_0 + g)^2. \tag{5.15}$$

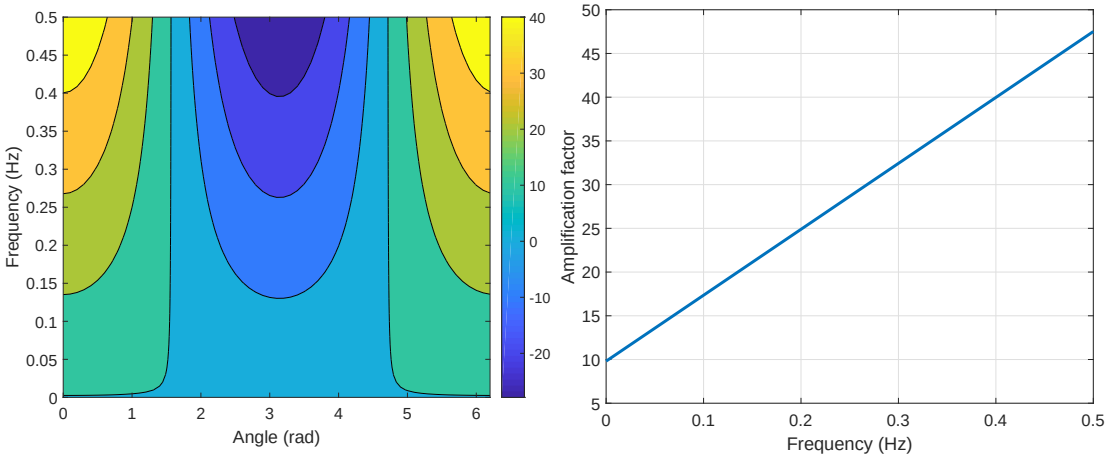
The last step is done because ω is more convenient to work with. From this equation we note that it may be worthwhile to consider the quantity $\cos(\bar{\theta} - \theta_{0,n})v_0\omega + g$. A plot of this quantity is given in Figure 5.4. In the upper left (Figure 5.4a) we see the amplification factor for various angles and omega. To better illustrate what we are seeing there, there are two more figures focussing on each of these parameters individually. We can see that this quantity takes values in the range [-20,50], and scales linearly with frequency. The extreme value only occurs when all the energy of the wave component is focussed in one specific direction. This does not occur in nature, where spectra tend to span at least 30 degrees. We can also see in Figure 5.4c that the lines all cross one another in two points: this is because there the vessel is perpendicular to the waves, and thus the amplification equals exactly Earth's gravitational constant, implying that the amplification there is independent of the vessel speed and heading.

In the computation above, the assumption of deep water was made. When considering naval charts along the route, I observed that the journey starts in shallow water (several meters deep), but that the vast majority of the route takes place in water with depths in the range of 20 to 25 meters, although no exact data is available, as the chart only allows usage through a web application. Still, using these values we note that $\tanh(kh) > 0.9$ for any $k > 0.08$ and depth $h > 20$ meters. This corresponds to $\omega \approx 0.27$ rad/s or 0.08 Hz. Since 0.08 Hz is well below the range of frequencies that affect motion sickness, we conclude that the assumption for deep water is justified. However, we use the original value during computations whenever possible.

5.3 Application of analysis to real data

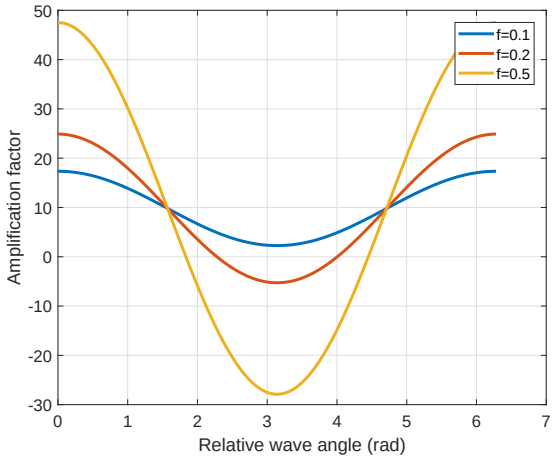
To get a more realistic idea of the impact of the amplification effects, we can compute Equation (5.12) for some real data. For this we use 2D spectral data, which allows us to take the directional spreading into account. A 2D spectral image contains values $S = S(\omega)$, where $\omega = (\omega_x, \omega_y)$. In order to be useful to us, we first convert this spectrum into a $(\omega_r, \omega_\theta)$ domain via a standard polar transformation.

We would like to test the theory developed above in practice, and see if measured ship accelerations match the anticipated response. Clearly we cannot do this directly, but we can compute motion sickness scores. Since these scores are based on the amount of energy that is associated with each frequency, they should be comparable to those computed based on the spectral data. As was mentioned earlier, we have not yet taken



(a) Contour plot of quantity as a function of both frequency and relative angle.

(b) Quantity as a function of frequency for $\bar{\theta} = 0$.



(c) Quantity as a function of the relative angle for various fixed frequencies.

Figure 5.4: Plot of the quantity $\cos(\bar{\theta} - \theta_0)v_0\omega + g$ using $v_0 = 12$ m/s and $\theta_0 = 0$.

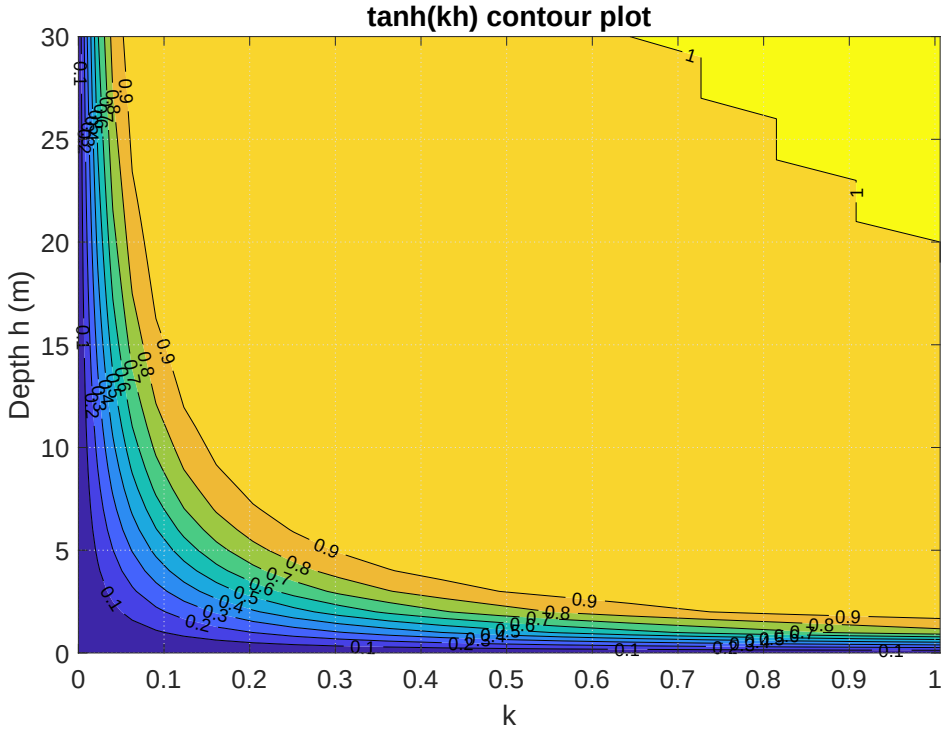


Figure 5.5: Contour plot to show the effect of tanh(kh).

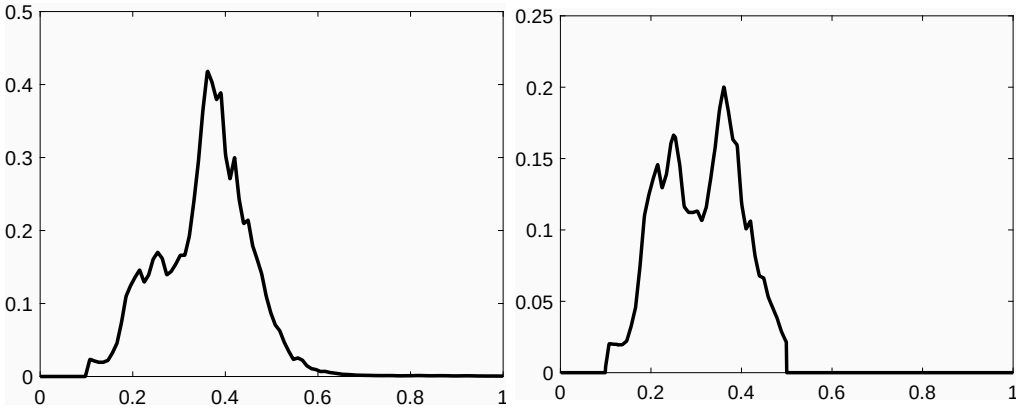


Figure 5.6: Energy spectra as recorded (left) and after applying the amplification factor (right).

the RAO's into account. Unfortunately, for all the ships considered in this research these operators are not known, but in theory they can be computed easily by comparing the recorded spectrum with the predicted spectrum, assuming the RAO equals the identity operator. This approach is illustrated in Appendix B. Unfortunately, it turns out that this approach does not work, but this will become clear. For now, however, we choose to be oblivious regarding these operators and assume that the RAO equals the identity operator. This implies we can compute the MSI as

$$MSI^2 = \sum_{(r,\theta)} \left(amp(r,\theta) RAO(\omega_r) w(\omega_r) \sqrt{E(\omega_{r,\theta}) \Delta\omega_{r,\theta}} \right)^2 . \tag{5.16}$$

The reason why this is true is because according to (5.10) $amp\sqrt{E\Delta\omega}$ are exactly the spectral components that the ship experiences. Then the RAO is needed to go from sea accelerations to vessel accelerations.

There is a second benefit to this procedure. If we can use the spectral data from the radar to predict motion response behaviour on board of these vessels, we can also use the spectral data to compute MSI scores on days when no ship is actually sailing. This implies that we can potentially relate weather to MSI scores 24 hours per day. In turn, this means that we will actually be able to extrapolate beyond the available data, and potentially even for different ships should their RAO's be known. Additionally, it eliminates the issue where only a very limited amount of data is available during very bad weather events, which is where regression models mostly fail.

The investigation thus far takes into account most effects, but there is still one major component missing: sea waves never all travel in the same direction. Thus even if we manage to perfectly know the spectrum, RAO's and relative angle with the main wave direction, the predictions will still fall short. In the perfect setup this is solved by using full 3D spectral images, which include the amount of energy corresponding to each of the wave directions per frequency. As mentioned in Section 4, we do not have access to the full 3D spectral images, as those cannot be recorded via wave radars or buoys. We do, however, know the wave spreading, which is defined as the standard deviation of the energy away from the main wave direction per wave component. Thus we circumvent the problem of not having a full 3D spectrum as follows: we generate the 3D energy distribution from the most popular wave spreading formula as proposed by Mitsuyasu in [13]. According to Mitsuyasu, the directional spreading follows the density model given by

$$D(\theta) = \begin{cases} A\cos^{2s}(\theta - \theta_0) & \text{for } |\theta - \theta_0| \leq \pi/2, \\ 0, & \text{otherwise.} \end{cases} \tag{5.17}$$

for some $s \in \mathbb{N}$ around main wave direction θ_0, and A a scalar to ensure that the integral over $[0, 2\pi]$ equals 1. This s is an indicator of how wide the spreading is. For our purposes, we choose s to be the smallest positive integer such that the integral over the spreading contains 1 standard deviation:

$$\int_{\theta_0-spread}^{\theta_0+spread} D(\theta)\ d\theta > 0.68. \tag{5.18}$$

This assumption is validated as the directional spreading, since the recorded directional spreading is defined as the standard deviation of wave direction. By consequently resampling this distribution on the desired grid we get a complete 3D spectrum on the desired grid. For testing we have used a grid with a bin size of 10 degrees. This size is somewhat arbitrary, but some quick tests shows that the system is not very sensitive to the number of bins. During this resampling we take the mean value of the spreading in the interval multiplied by the spectral value associated with that wave component. This ensures that integration with respect to θ yields the original 1D spectrum.

5.4 Aggregating various measurements

Up to now we have been computing the impact of speed and heading as a function of waves represented by a single frequency and heading. However, in practice the sea is a superposition of such waves. Therefore we need to investigate exactly how to compute the amplification factor over said superposition. As mentioned before, the very fact that the vessel is moving, results in a change of the relative frequency in which the vessel encounters waves. This implies that we have to make a choice on which reference frame to use for future comparisons of vessel spectra with wave spectra for the wave buoys. The chosen reference is the vessel, since motion sickness is a result of the vessels motions rather than the sea state. This does present a problem, however, as the output frequencies are no longer fixed. This is resolved by resampling the final result on fixed frequencies.

Aggregating distinct frequencies is usually pretty simple. Given some frequency ω and corresponding k, the relative frequency is given by

$$\omega_{rel} = |k\cos(\bar{\theta} - \theta_0)v_0 + \omega|. \tag{5.19}$$

However, problems arise when we find two distinct ω that correspond to the same relative frequency but with opposite directions. For example, this corresponds to the case where we consider two waves travelling at 10 and 20m/s respectively, and the vessel moving at 15m/s (all in the same direction). This way the vessel experiences one forward wave at 5m/s and one backward wave. Assuming that the waves in both directions have amplitude A and B, phase 0 and ϕ_0 respectively at $t = 0$, we can describe the result as

$$\begin{aligned} A\sin(\omega t) + B\sin(\phi_0 - \omega t) &= A\sin(\omega t) - B\sin(\omega t - \phi_0) && (5.20)\\ &= \mathrm{Im}\left(Ae^{\omega t i} - Be^{(\omega t - \phi_0)i}\right) && (5.21)\\ &= \mathrm{Im}\left(Ae^{\omega t i} - Be^{\phi_0 i}e^{\omega t i}\right) && (5.22)\\ &= \mathrm{Im}\left((A - Be^{\phi_0 i})e^{\omega t i}\right). && (5.23) \end{aligned}$$

This implies that the result is a sinusoid with frequency ω and an amplitude of

$$||A - Be^{\phi_0 i}|| = \sqrt{(A - B\cos(\phi_0))^2 + B^2\sin^2(\phi_0)} = \sqrt{A^2 + B^2 - 2AB\cos\phi_0}. \quad (5.24)$$

Note that the initial choice of phase differential describes the general case, since we can always reduce to this case via a translation along the time axis. This implies that depending on the initial phase, the superposition of these wave components can change drastically, and can vary in amplitude between $A - B$ and $A + B$. It is also worth noting that a direct implication of this result is that when $B > A$, the direction of the wave component ω_{rel} relative to the ship flips direction around $B\cos(\phi_0) = A$. In our example, this occurs exactly when the slow wave has more energy than the fast wave.

Clearly, it is not desirable to have this phase influence the response of the vessel, even though it may play a significant role. Note, though, that enforcing a constant speed and heading is impossible. Furthermore, wave conditions can have minor variations over time. Therefore, I believe it justified to assume a uniform distribution over the various possible phase angles over the transit period, and then take the mean as the amplitude of the resulting amplitude:

$$amp_{\omega_{rel}} = \mathbb{E}(||A - Be^{\phi_0}||). \quad (5.25)$$

This is a difficult expression to compute analytically, and is therefore numerically approximated using 1000 uniformly distributed choices of ϕ_0 for each pair of matching ω.

Yet we are still not done, since we are never going to find perfectly matching pairs of ω, as we are using a discrete set of values. To compensate for this effect, we have chosen to resample the set of frequencies with $k\cos(\bar{\theta} - \theta_0)v_0 + \omega < 0$ to match the frequencies where $k\cos(\bar{\theta} - \theta_0)v_0 + \omega \geq 0$. This means that we have a proper representation of the resulting spectrum.

5.5 Numerical results

Whilst the theory behaves nicely, practice proved otherwise. Results for this procedure are displayed in Figure 5.7. Contrary to Figure 5.6, the general spectrum obtained via the procedure above looks like the energy spectrum shown in Figure 5.8. There, the Cefas raw heave spectrum, the predicted vessel spectrum and the recorded acceleration spectrum have been plotted for one instance. We can see that the initial energy peak is translated to a different part of the spectra, and that the higher frequency range contains significantly more energy than we would expect. Albeit less common, examples to the contrary also exist: see Figure 5.9. This implies that something is going seriously wrong in these computations, and may imply that some of the assumptions made earlier do not hold. Most notably, I believe that the assumption of the vessel following the wave surface is extremely wrong. Whilst this can partially explain the results, there must be more problems to the procedure, but this is left for further research. As the analytical methods do not provide sufficient accuracy, we now resort to machine learning methods and machine learning. The main advantage of such methods is that they do not require a theoretical framework, but are still capable of benefiting from the results of this section.

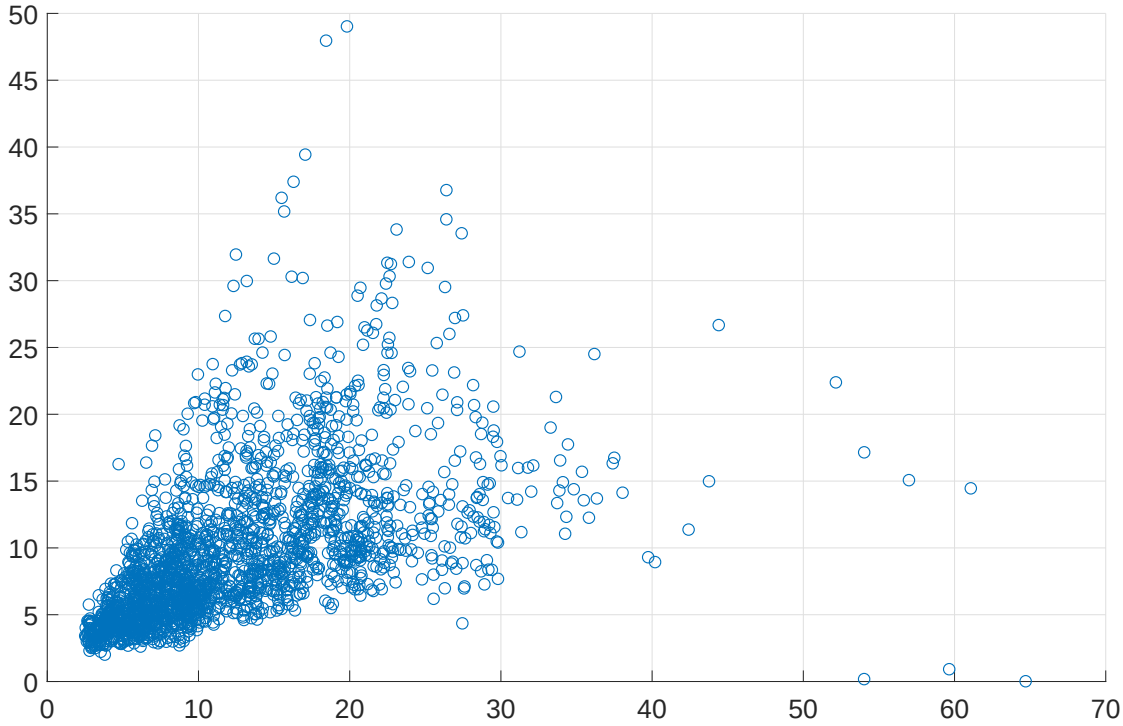


Figure 5.7: Scatterplot showing predicted MSI versus recorded MSI for all vessels in the Greater Gabbard site.

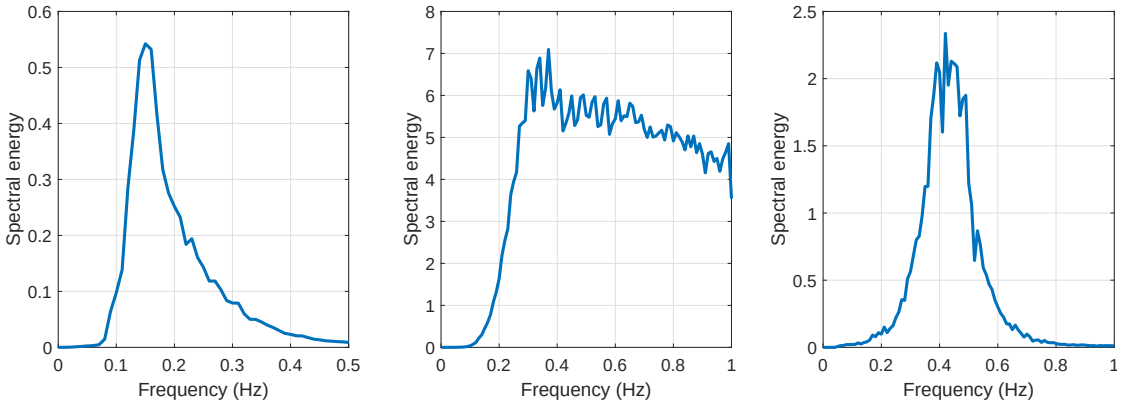


Figure 5.8: Raw heave energy spectrum (left), the expected parametrized acceleration energy spectrum (middle) and the recorded vessel spectrum for the Dalby Aire, 16 May 2016. These images are representative for the mismatch between predicted and recorded vessel spectra.

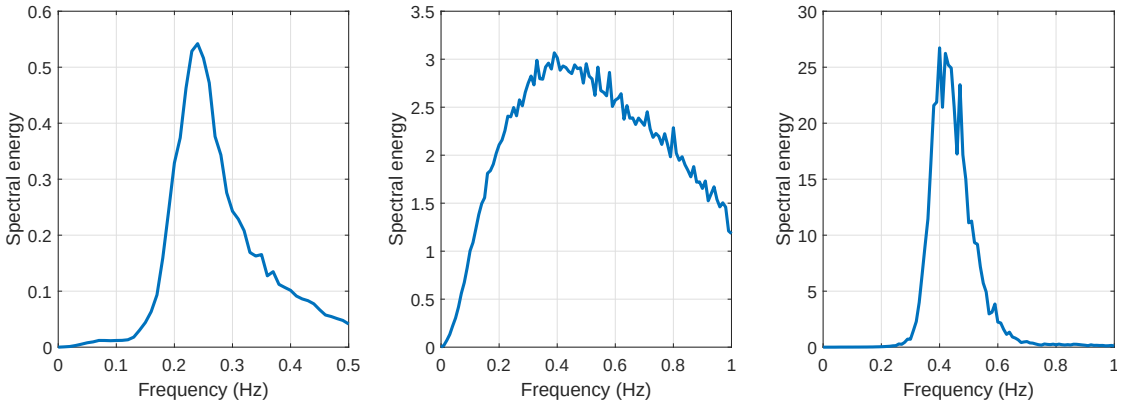


Figure 5.9: Raw heave energy spectrum (left), the expected parametrized acceleration energy spectrum (middle) and the recorded vessel spectrum for the Dalby Aire, 27 June 2016.

6 Introduction machine learning

We have tried to compute the MSI using the laws of physics. It is, however, also possible to predict MSI using machine learning. In layman's terms, machine learning is the art of letting the computer create the model for us, such that the machine learning algorithm functions as the 'brain' from Figure 6.1: as a generic input-output solver without a need for context (given enough data), but which can benefit from context. But how does this brain work? Despite its name, and whilst machine learning can definitely be a great asset, it is far from an automated process, and requires a lot of manual assistance. A more precise description of machine learning is therefore to let the computer find the best approximation of the (generally unknown) target function using a set of base functions and optimization criteria specified by the user, based on a series of past measurements. The simplest example of machine learning is linear regression, where you try to fit the best line through some data points in a usually high-dimensional space.

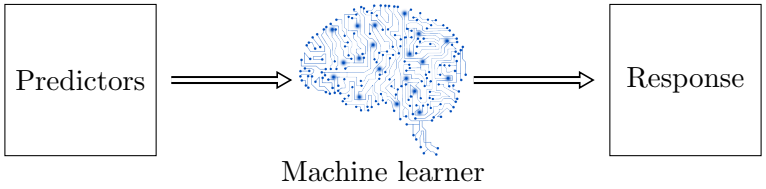


Figure 6.1: Most simple representation of a black-box machine learner.

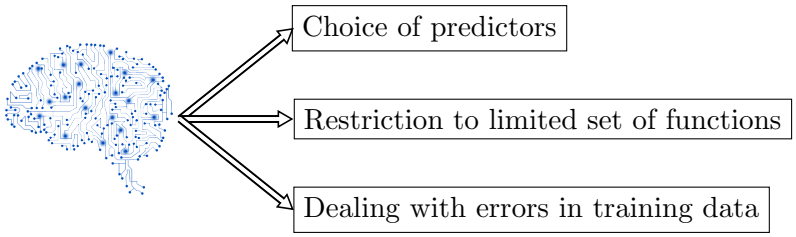


Figure 6.2: Main aspects for machine learning.

Algorithms for machine learning use known data sets with a set of predictors P to find a function f that best explains an observable O. The latter is also referred to as the response. The goal for a machine learning algorithm is to estimate our unknown function f by minimizing an error. While a variety of norms can be used to describe such an error, the most common criteria is the l_2 norm. For a set of predictors $x \in P \subset \mathbb{R}^m$ and a set of observables $y \in O \subset \mathbb{R}$ corresponding to P, this means we want to optimize over all functions $f : \mathbb{R}^m \to \mathbb{R}$ such that

$$E := \sum_{i=1}^{N} \epsilon_i^2 := \sum_{i=1}^{N} (y_i - f(x_i))^2. \tag{6.1}$$

is minimal. Clearly, this set of functions is uncountably infinite, and hence we need to restrict the set of functions. In practice this is done by assuming that a function has a certain form with one or several parameters that can be tailored to the relevant dataset. This is also where the largest difference with regular modelling takes place, as no assumptions are made regarding the exact form of the function, whilst in modelling one usually has a physical reasoning to assume a certain type of model. The simplest example is by finding the best linear approximation of a dataset $(P, O) \in \mathbb{R} \times \mathbb{R}$. Intuitively, this equals drawing the best line through the dataset. A mathematical derivation of this procedure is given in Section 8.1. For the remainder of this section, we introduce the general techniques used in machine learning. In the next section, two additional techniques often used in data science are explained, which will help us better determine how to use the machine learning algorithms. Then, in Sections 8 and 9 the specific models used in this thesis are introduced.

6.1 Discrete vs. continuous data

Learner models are available for fitting both continuous and discrete observables. These are respectively known as regression learning and classification learning. When applying the learner to a discrete set of observables, we also assume this set to be finite. Examples of classification learners are object recognition or decision making, in which the algorithm returns one out of a finite set of predefined options. On the contrary, regression models can also be used for interpolation between and extrapolation beyond known data points, albeit at the cost of accuracy. Especially extrapolation quickly becomes unreliable. In most cases the choice between the two is pretty easy: the problem itself is either continuous or discrete.

6.2 Validating models

Once a model is trained, the model needs to be validated. This is needed since a machine learning model suffers from a number of weaknesses. These weaknesses are illustrated in Figure 6.3. These images show that the best linear approximation does not always imply a good model. In the upper-left, a properly trained model shows. However, the linear approximation is poor when the fitted data is non-linear (upper-right), when it contains significant outliers (lower-left) or when data is not correlated (lower-right). These weaknesses can be applied to any prediction model in general. Finally, validation is necessary to prevent over-fitting. This is especially important when working with errors in the training data, as a model performing well on the training set might not work so well in general. An example of over-fitting is given in Figure 6.4. In practice, we often take the validation step by performing a cross-validation on the available data: we use 90% of the available data to train the model, and then use the final ten percent to test the models performance. We repeat this procedure ten times, such that we use a different set to check each time. This way, we avoid focussing on a single model that appears to perform much better than its alternatives because there are no outliers its the test data.

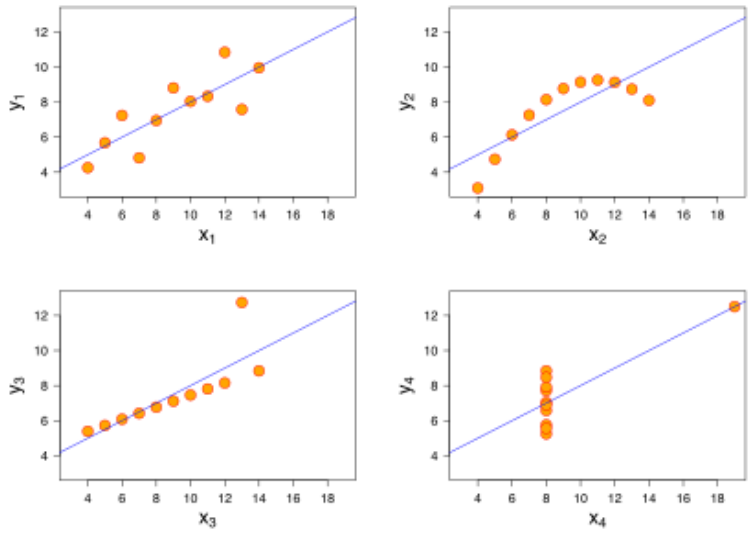


Figure 6.3: Anscombes illustration of modelling weaknesses [1].

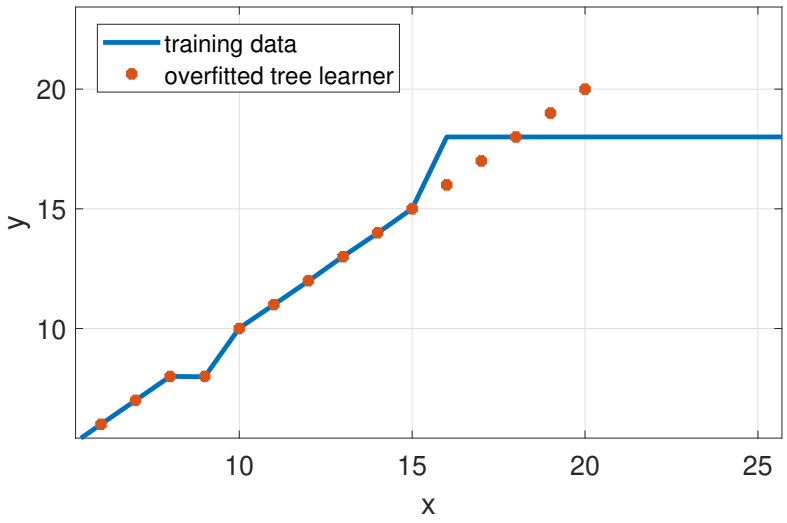


Figure 6.4: Example of an over-fitted tree regression model for function $y(x) = x$ with a single error in the training data at $x = 9$. Also, it shows that certain models have difficulty extrapolating the data beyond the last training point, here at $x = 20$.

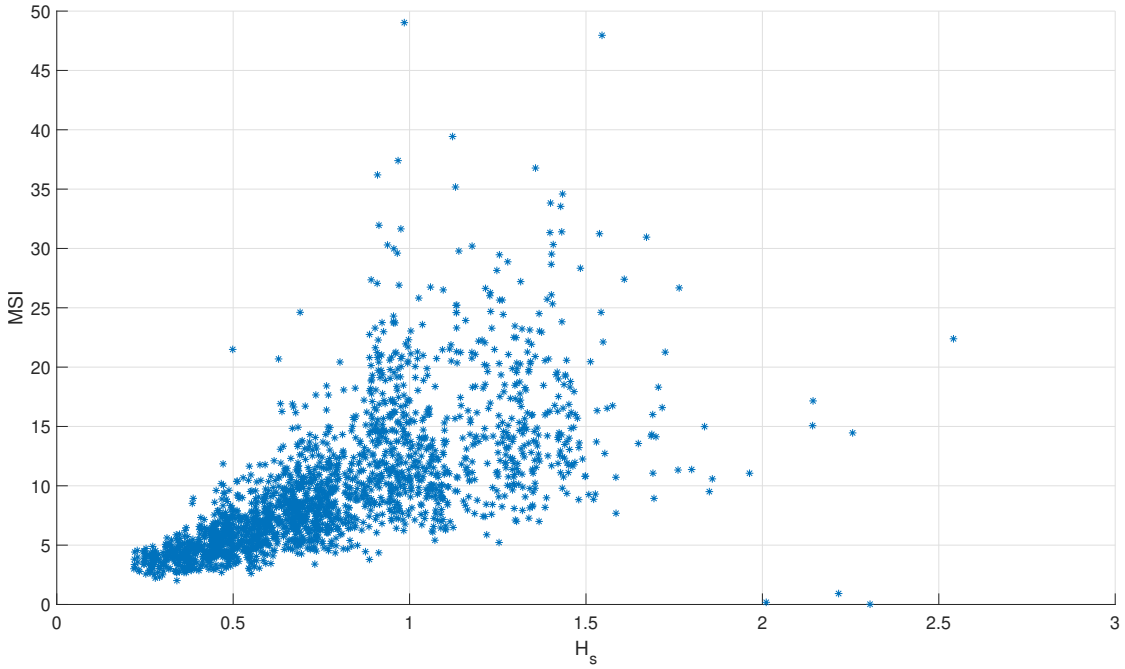


Figure 6.5: Scatter plot showing that the MSI behaves fairly linear for smaller wave heights, but wavers out when values exceed 1 meter.

6.3 Extreme values

A more interesting case of the non-linear data case occurs when fitting the MSI scores as a function of H_s. Whilst the data behaves mostly linear for $H_s < 1$, the MSI scores vary wildly as H_s increases. As we are most interested in accurate predictions when H_s is high, we need to make sure that the systems is particularly accurate during such events. This becomes more problematic as most of the available data is for lower values of H_s.

One way to enforce better behaviour is by defining a cost function in (6.1). If we want to increase the focus on correctly fitting extreme values we could e.g. scale using (a constant times) H_s, in which case our optimization problem becomes

$$\min_f E' := \sum_{i=1}^{N} c_i (y_i - f(x_i))^2 \tag{6.2}$$

$$c_i = H_s(x_i) \qquad (i = 1 \dots N). \tag{6.3}$$

This cost function can be changed to be application-specific, and there is no optimal way on how to choose this cost function.

6.4 Feature

Features are the input variables used in the model. It is common practice to first try fitting with the input parameters that are available. However, these parameters

are usually insufficient to fully explain the output data. For example, suppose the we attempt to fit a linear model on a number of samples of the 1-dimensional function $f(x) = x^2$. Clearly this is doomed to fail. Yet if we were to use a different feature, by attempting to fit x and x^2 to our model instead of just x, we should get a perfect fit. This process of applying functions to the input data before fitting is called feature selection, and hence one of the most important steps in machine learning. It is also where understanding the physics behind the problem can be beneficial, as these may give an indication for what types of input data transformations to use. This step can sometimes be automated by the use of smart algorithms. The best examples of such smart algorithms are found in image processing, specifically facial recognition.

In the next section two methods are introduced, called principle component analysis and independent component analysis. These are standard methods in data science that are useful in choosing which predictors to use, as well as choosing features.

7 Dimensional reduction techniques

There exist two main dimensional reduction techniques for machine learning: principle component analysis (PCA) and independent component analysis (ICA). PCA is an analytic tool that is useful to distinguish the most influential parameters in a process, whilst ICA tries to distinguish independence between various components. The two methods are closely related, and the results from PCA can even be used in computing ICA. We introduce both models below and show the advantages and shortcomings of both methods.

7.1 Principle component analysis

Consider the orbit of the planet Venus as can be observed from Earth. As was discover by Galileo, Venus revolves around the sun rather than Earth. At the time this was a big observation, as it overthrowed the model of Ptolemy, which assumed that all celestial bodies orbit around Earth. The model of Ptolemy was also capable of describing the orbit of Venus, albeit in a much more complex manner: he added an additional invisible point which rotated around the Earth, and Venus was assumed to be orbiting that invisible point. Consequently, the mathematics involved to predict the orbit of Venus were much harder than those needed by Galileo. This is exactly the type of problem PCA tries to solve, by trying to simplify the model by reducing the amount of components involved.

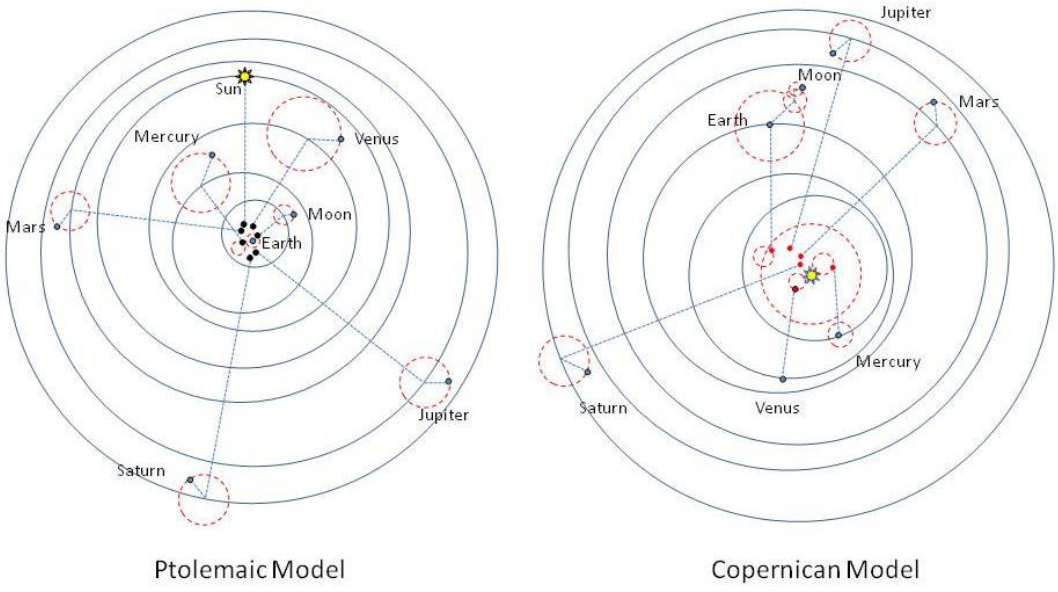


Figure 7.1: The Ptolemaic model (everything orbits Earth) and the Copernican model (everything orbits the sun) are both capable of describing the orbit of other celestial bodies such as Venus [20].

Mathematically PCA tries to find the vectors that explain most of the variance within the data, the so-called principle components. The first principle component indicates the best explanatory vector, the second the second-best explanatory vector etc. It does this by performing a change of basis. In the example this would correspond to translating the origin from Earth to the sun. The various principle components would then correspond to the new orbits of the various bodies as seen from the sun[2]. A more detailed description of PCA in general is available in [18]. Before we can properly explain PCA, however, we first introduce the covariance matrices on which the analysis is based. Whilst sharing the name with the covariance matrices mentioned in Section 8, they are not the same, and hence this definition is used only in this section.

7.2 Covariance matrix

Suppose we have two random variables X, Y with mean μ_x, μ_y respectively. Then the covariance between X and Y is defined as $cov(X,Y) := \mathrm{E}[(X-\mu_x)(Y-\mu_y)] = \mathrm{E}[XY] - \mu_x\mu_y$, where $E[\cdot]$ denotes expectation. The last equality holds since the expectation is linear and $\mathrm{E}[X\mu_y] = \mu_y\mathrm{E}[X] = \mu_x\mu_y = E[Y\mu_x]$. Once again, this definition of covariance is restricted to this section! Note that if X and Y are independent random variables, then $\mathrm{E}[XY] = \mathrm{E}[X]\mathrm{E}[Y]$, and therefore $cov(X,Y) = 0$. If, on the other hand, X and Y are maximally correlated, it follows that $cov(X,Y) = cov(X,X) = var(X) = \sigma_x^2$. If X is normalized, $\sigma_x = 1$, and the covariance becomes a value between 0 and 1 measuring how much the random variables X and Y are correlated.

Now suppose we have some vector

$$\mathbf{X} = \begin{bmatrix} X_1 \\ \vdots \\ X_m \end{bmatrix},$$

where X_i is some random variable with mean μ_i. We define $\Sigma_{ij} := cov(X_i, X_j)$. Then the covariance matrix of X is given by

$$\Sigma = [\Sigma_{ij}] \tag{7.1}$$

This is, however, not a quantity that we can generally compute: usually the exact distribution of a random variable is not known. Therefore, assuming $\mathbf{x} \sim X$ and $\mathbf{y} \sim Y$ to be n-dimensional samples, we compute $cov(X,Y) = \frac{1}{n-1}\mathbf{x}\mathbf{y}^\mathsf{T}$. Now we want to generalize this result to covariance matrices. Suppose that for our vector $\mathbf{x}$ we also have n samples per random variable. Let S be the $m \times n$ matrix with the samples for each random variable along the rows. Then the columns form the recordings of a single measurement. In our case, an example would be a record of H_s, the wave direction and the recorded MSI from a single (and the same!) trip. Then we can estimate Σ from S as follows:

$$\Sigma = \frac{1}{n-1} SS^\mathsf{T} \tag{7.2}$$

[2]Strictly speaking do basis transformations not allow for translations of the origin, but, as we see later, we do not care about changing constants

7.3 Derivation PCA

It turns out that the principle components are the eigenvectors of $A := SS^{\mathsf{T}}$. Even more so, the corresponding eigenvalues are $\frac{1}{n-1}$ times the variances associated with these eigenvectors, and thus allow us to find all of the principle components. When we look at (7.2), we see that A and Σ are closely related, and their eigenvalues only differ by a factor $\frac{1}{n-1}$ whilst sharing the same eigenvectors. Recall that for finding the first principle component PCA attempts to find the normal vector that maximizes the variance within the data, i.e. it tries to find

$$\max_{\mathbf{x}:\ ||\mathbf{x}||=1} ||\Sigma x||. \tag{7.3}$$

This implies we want to find the eigenvector associated with the largest eigenvalue. It is a standard result from linear algebra that the other principle components are the eigenvectors corresponding to the eigenvalues of Σ in decreasing order.

In order to conclude that the second principle component is indeed the eigenvector associated with the second largest eigenvector, note that we can decompose Σ in its eigenvalue decomposition

$$\Sigma = \sum_i \lambda_i \mathbf{v_i} \mathbf{v_i}^{\mathsf{T}} \tag{7.4}$$

Here we assume $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. We can write any vector $\mathbf{b} = \sum_{i=1}^{n} b_i \mathbf{v_i}$ if we assume $\mathbf{v_i}$ forms a complete basis. In practice, this is (almost) always the case due to the presence of noise. In particular we can chose $\mathbf{b}$ to be the vector maximizing the variance. As we work with the restriction that $||\mathbf{b}|| = 1$ it is now straightforward to see that if $\mathbf{b}$ is orthogonal to $\mathbf{v_1}$, then it must be that $\mathbf{b} = \mathbf{v_2}$.

7.4 Interpretation of PCA

Suppose we have computed the PCA for a dataset, then what does this tell us about the data? As mentioned before, PCA tries to explain as much of the variance in the input data as possible. Each of the components show an effect and its relevance in explaining the variance. The hope is that the physical model that we actually want to describe corresponds to the principle component(s) with the highest variance, since this signal best distinguishes itself from uniform random. This also shows that heavily polluted data can no longer be used in this analysis, as such data shows less variance.

Once we have found each of the principle components and their corresponding variances, we can apply dimensional reduction. In practice, this is done by simply considering the components that have low variance, as we assume that these have nothing to do with the process that we attempt to model, and therefore are considered noise. By taking the k components with the highest variance, we end up with only k parameters to use in the model. While this reduces the complexity of the model, a significant downside is that the model becomes much harder to interpret, as a principle component is nothing but a weighted sum over the input, and usually does not represent a physical quantity.

7.5 Restrictions to using PCA

PCA is a very powerful tool, but one needs to be very careful in choosing how to apply it to the data. Since PCA tries to maximize the variance in its input variables, we need to normalize the data first. An easy example here would be using the wave direction and the significant wave height. Since H_s rarely exceeds 2, its variance will be very limited. On the other hand, it does not even make sense to define a mean for the wave direction, as the data is circular. Under the assumption that the wave direction is uniformly distributed we would find a mean of 180 and a standard deviation of a little over 100. Thus it makes sense to normalize the data, i.e. by subtracting the mean for each of the variable and then divide by the standard deviation. Note that subtraction of the mean does not affect the PCA at all.

Another restriction is the assumptions of linearity in the sample data. This also is why PCA does not work for circular data such as wave direction. Also, if there are variables in the data that are, say, quadratically related, then the variance found using PCA is going to be less than one might expect. Considering our belief that the MSI is not linear, this assumption may later prove not to hold. Still there is a widespread belief that PCA may still point in the right direction even if the assumption is that linearity does not hold. Assuming linearity goes hand-in-hand with the assumption that the principle components are linearly independent. Clearly, that is not always the case, and hence the principle components found must contain noise to some degree.

7.6 PCA vs. SVD

Very closely related to PCA is the singular value decomposition (SVD, not to be confused with SVM). The result of using SVD is the same as PCA, except that it uses a different methodology to compute the eigenvectors of Σ. Whilst being harder to interpret, using SVD has the advantage of having an algorithm that is faster and numerically more stable. This last property is important since the covariance of independent components is expected to quickly drop with the number of computed independent components. A thorough explanation of SVD is given in the explanation of ICA below.

7.7 Independent component analysis

Thus far we introduced principle component analysis. As mentioned, PCA is mostly used for dimensional reduction in order to distinguish the strongest correlated signals. The goal of independent component analysis (ICA), however, is to decompose a multivariate signal into independent signals. ICA does this in the same setting as PCA, i.e. the signal under investigation is assumed to be a linear superposition of the underlying signals. The most common use cases for ICA are the cocktail party problem and image blurring. In the first case we attempt to separate one conversation from the background, in the other we attempt to split the actual image from the blurring that is caused by motion. In both these settings we cannot use PCA, since the noise is one of the vector responsible for the variance. This is where ICA is useful, since ICA strives for independence of the

signals rather than the best explanatory vector of noise. Below we first provide some background, and then provide a derivation for ICA. Finally, the interpretation of ICA as well as its restrictions are discussed.

7.8 Mathematical assumptions and derivation ICA

Similarly to PCA, ICA tries to find some useful linear transformation A. In PCA, the goal was to get orthonormal vectors that attempt to maximize the variance. In ICA, the goal is to find a linear transformation A such that the elements of $s = Ax$ are statistically independent. The only assumption other than linearity of the individual components is the independence between the different signals that we try to distinguish. In the case of the cocktail problem this makes sense, as the conversation is not going to be affected by the background noise. However, when the process that is being investigated is not well understood, this may not be the case. If ICA returns an unexpected signal from the data, then this likely means the signal you are looking for is intertwined with another one. Mathematically, this assumption is represented as $\mathrm{E}[s_i s_j] = \mathrm{E}[s_i]\mathrm{E}[s_j]\ \forall i, j$. Another way to describe this is that $C_s = cov(\mathbf{s}) = I_{m\times m}$. Before we can properly explain ICA, the singular value decomposition (SVD) is introduced. This is a generalization of the eigenvalue decomposition, and a standard concept from linear algebra.

Theorem 2 (Singular value decomposition)**.** *Let A be a real $m \times n$ matrix. Then there exists a singular value decomposition $A = U\Sigma V^{\mathsf{T}}$ such that*

1. *$U \in \mathbb{R}^{m\times m}$ is an orthogonal matrix.*
2. *$\Sigma \in \mathbb{R}^{m\times n}$ a rectangular diagonal matrix with only non-negative (real) entries on the diagonal.*
3. *$V \in \mathbb{R}^{n\times n}$ an orthogonal matrix.*

A proof is provided in Appendix A.2. The singular value decomposition equals the eigenvalue decomposition if A is normal and positive semi-definite. While the theorem here is stated for real valued matrices, it also holds over the complex numbers after replacing orthogonal with unitary. As mentioned earlier, we want to find a linear transformation A such that $\mathbf{s} = A\mathbf{x}$. As these linear transformations are real, A has a singular value decomposition

$$A = U\Sigma V^{\mathsf{T}}. \tag{7.5}$$

This implies that for finding $\mathbf{s}$ we need to recover $A^{-1} = V\Sigma^{-1}U^{\mathsf{T}}$. Note that $C_x = C_{As} = AC_sA^{\mathsf{T}}$ due to linearity of the expectation. However, since C_s is the identity, we can further simplify this to

$$C_x = U\Sigma V^{\mathsf{T}}V\Sigma U^{\mathsf{T}} = U\Sigma^2U^{\mathsf{T}}. \tag{7.6}$$

Alternatively, we can also represent C_x using the eigenvalue distribution since C_x is a symmetric matrix:

$$C_x = EDE^{\mathsf{T}}. \tag{7.7}$$

Here we use the fact that E can be chosen orthogonally. Hence we can now rewrite (7.5) as $A^{-1} = V\sqrt{D}E$, where D and E are known. So the problem of finding an arbitrary linear transformation A has been reduced to finding a rotation matrix V. While it may not immediately be obvious here, is that in doing this we have effectively solved PCA to obtain this factorization. A full explanation on how to find V from this point on goes beyond the scope of this work, as the solution cannot be found analytically. However, it involves solving the optimization problem

$$\arg\min_{V} \sum_{i=1}^{n} H[(V\mathbf{x}_w)_i], \tag{7.8}$$

where $H[\cdot]$ is the entropy of a distribution and $\mathbf{x}_w = \sqrt{D}E\mathbf{x}$, using the method of gradient descent. Informally, the solution of this optimization problem solves V since the $\mathbf{s}_i$ are independent if and only if their shared mutual information (for which entropy is a measure) is as small as possible. A more detailed explanation involving this as well as ICA in general is provided in [17].

7.9 Other properties ICA

As mentioned before, ICA splits the different signals. However, it does not always succeed in doing so. One of the main reasons for failure is an incorrect estimation of $H[(V\mathbf{x}_w)_i]$ when using a limited set of data. This can be helped by adding more (diverse) data, and should therefore not be relevant when working with large datasets. There also exists a fast ICA algorithm, which converges cubic (usually, worst-case is quadratic, see [10]) rather than the standard algorithm described here, which converges only linearly. This means that ICA can be efficiently implemented, and hence be applied to larger datasets, which is a prerequisite when doing machine learning.

Thus far we have discussed the basics of data science. In the next two sections, we give some concrete examples of machine learning algorithms.

8 Regression models

There exist many machine learning algorithms, but most of these algorithms are variants of five basic regression models: polynomial regression, regression trees, support vector machines, Gaussian process regression and neural networks. Below, we introduce the first four algorithm and provide some examples to get an idea of their performance by a one-dimensional test case. The last algorithm is introduced separately in Section 9, as it is significantly different from the first four.

8.1 Polynomial regression

Linear regression, and more generally, polynomial regression, are the simplest and most commonly used forms of regression. They work well when there is a clear relation between the input and output variables, and polynomial regression models are relatively easy to understand. Polynomial regression of degree n assumes that the response can be described by a polynomial of degree n in the predictors. Surprisingly, the algorithm for polynomial regression of degree $n > 1$ is the same as that of degree 1 (=linear regression). This can be seen as follows: suppose that we have predictors $\mathbf{x}_1, \ldots, \mathbf{x}_m$. Then, define $\mathbf{y}_{jk} = \mathbf{x}_j^k$. for $j = 1, \ldots, m$ and $k = 1, \ldots, n$. Then an optimal solution for polynomial regression corresponds directly to the optimal solution for linear regression when using $(\mathbf{y}_{jk})$ as predictors. Thus it is sufficient to consider linear regression. To understand how linear regression works, a deduction of linear regression using only one predictor is given below. The general case is slightly more involved, but follows a similar reasoning. Hence let us assume that $m = 1$ and $f = ax + b$, which means we want to minimize

$$\sum_{i=1}^{N} \epsilon_i^2 = \sum_{i=1}^{N} (\mathbf{y}_i - a\mathbf{x}_i - b)^2. \tag{8.1}$$

Here $(\mathbf{x}_i, \mathbf{y}_i)$ for $i = 1, \ldots, N$ denotes the training data of size N. In order to minimize Expression (8.1), note that

$$\begin{aligned}
\sum_{i=1}^{N} \epsilon_i^2 &= \sum_{i=1}^{N} (a\mathbf{x}_i + b - \mathbf{y}_i)^2 && (8.2)\\
&= \sum_{i=1}^{N} \mathbf{x}_i^2 a^2 + 2(b\mathbf{x}_i - \mathbf{y}_i\mathbf{x}_i)a + (b - \mathbf{y}_i)^2 && (8.3)\\
&= \left(\sum_{i=1}^{N} \mathbf{x}_i^2\right) a^2 + \left(2\sum_{i=1}^{N} (b\mathbf{x}_i - \mathbf{y}_i\mathbf{x}_i)\right) a + \sum_{i=1}^{N} (b - \mathbf{y}_i)^2 && (8.4)\\
&= Nb^2 + \left(2\sum_{i=1}^{N} (a\mathbf{x}_i - \mathbf{y}_i)\right) b + \sum_{i=1}^{N} (a\mathbf{x}_i - \mathbf{y}_i)^2. && (8.5)
\end{aligned}$$

Note that the last two expressions are simple quadratic functions in a and b, and are therefore minimized at

$$a = \frac{-2\sum_{i=1}^{N}(b\mathbf{x}_i - \mathbf{y}_i\mathbf{x}_i)}{2\sum_{i=1}^{N}\mathbf{x}_i^2} \tag{8.6}$$

$$= \frac{\sum_{i=1}^{N}(\mathbf{y}_i\mathbf{x}_i) - Nb\bar{x}}{\sum_{i=1}^{N}\mathbf{x}_i^2} \tag{8.7}$$

$$b = \frac{-2\sum_{i=1}^{N}(a\mathbf{x}_i - \mathbf{y}_i)}{2N} \tag{8.8}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i - \frac{a}{N}\sum_{i=1}^{N}\mathbf{x}_i \tag{8.9}$$

$$= \bar{y} - a\bar{x}. \tag{8.10}$$

Here we write $\bar{x}$ and $\bar{y}$ to represent the mean of P and O respectively. Substituting (8.10) in (8.7) yields

$$a = \frac{\sum_{i=1}^{N}(\mathbf{y}_i\mathbf{x}_i) - N(\bar{y} - a\bar{x})\bar{x}}{\sum_{i=1}^{N}\mathbf{x}_i^2} \tag{8.11}$$

$$\Leftrightarrow a = \frac{\sum_{i=1}^{N}(\mathbf{y}_i\mathbf{x}_i) - N\bar{y}\bar{x}}{\sum_{i=1}^{N}\mathbf{x}_i^2 - N\bar{x}^2} \tag{8.12}$$

$$= \frac{\sum_{i=1}^{N}(\mathbf{y}_i - \bar{y})\mathbf{x}_i}{\sum_{i=1}^{N}(\mathbf{x}_i - \bar{x})\mathbf{x}_i} \tag{8.13}$$

$$= \frac{\sum_{i=1}^{N}(\mathbf{y}_i - \bar{y})(\mathbf{x}_i - \bar{x})}{\sum_{i=1}^{N}(\mathbf{x}_i - \bar{x})^2}. \tag{8.14}$$

Here the last step follows as $\sum_{i=1}^{N}(\mathbf{x}_i - \bar{x})\bar{x} = 0 = \sum_{i=1}^{N}(\mathbf{y}_i - \bar{y})\bar{x}$ as a direct consequence of the definition of $\bar{x}$ and $\bar{y}$.

Now let us consider the case with multiple predictors. If we want to find an optimal approximation using m predictors, we want to find $(a, b) \in \mathbb{R}^m \times \mathbb{R}$ that minimizes

$$E = \sum_{i=1}^{N}(\mathbf{y}_i - \mathbf{a}^\mathsf{T}\mathbf{x}_i - b). \tag{8.15}$$

The solution to the multivariate minimization problem is given by

$$(\mathbf{a}, b) = \left(X^\mathsf{T}X\right)^{-1}X^\mathsf{T}Y. \tag{8.16}$$

Here $X = (\mathbf{x}_i, b) \in \mathbb{R}^{m+1\times n}$ and $Y = (\mathbf{y}_i) \in \mathbb{R}^n$. Note that in the case of one predictor this formula simplifies to (8.14).

Whilst linear regression is a very powerful tool, it has some limitations:

- First and foremost, the model only works on assumption that the data is linear. This directly means that outliers, with a relatively large error, can significantly harm the estimate. Also, for processes that are not linear, one typically sees large errors among the extremes of the dataset.
- While linear regression is capable of handling errors in the observables, it assumes there to be no errors in the predictors. When using time as a predictor, this usually holds, but if the predictor is a measurement, or even worse, a forecast, this assumption this might be far from the truth. There exist variants on linear regression called robust linear regression that can deal with such errors, but this comes at the cost of model performance.

8.2 Regression trees

Regression trees are conceptually just a series of if-then statements. In contrast to many of the other regression methods, its output is not continuous but instead a finite discrete subset of the observable space. For example, if we want to estimate the selling price for a house (which is more or less continuous), we can round the selling prices to multiples of 10.000, and then use rules based on variables such as distance to city ($>$ 10km or not) to best fit the house to a price category. Mathematically, this means that regression trees transform the continuous problem into a classification problem, where the categories are subsets of the observable space. Then, the problem is solved using a solver for classification learning, in this case classification trees. A graphical illustration is given in Figure 8.2. By design, classification trees usually have no fitting error amongst the training data. A direct consequence of the non-existence of such a fitting error is that the model is almost always over-fitted. Furthermore, regression trees are very poor at prediction for data outside the training range: the model can only predict as much as the most extreme training point.

A special type of regression tree is the tree ensemble. A tree ensemble is a weighted superposition of various trees, each of which fits the data using different parameters. This vastly reduces the over-fitting, whilst maintaining the low fitting error amongst the training data. However, a significant downside of ensembles is that they are hard to interpret and rarely describe a physical process. Also, predicting extremes stays problematic, as the model still cannot extrapolate. Finally, keep in mind that neither model is continuous.

8.3 Support vector machines

A support vector machine (SVM) is a clustering algorithm that attempts to best split the data using a higher-dimensional hyperplane. While originally designed as a classification algorithm, there is an easy adjustment that changes SVM into a regression learner. The latter is referred to as support vector regression (SVR), but in this work no distinction is made. SVM is a broadly used algorithm, but it failed to deliver any significant results for predicting motion sickness, and therefore a detailed analysis is omitted. Still, this

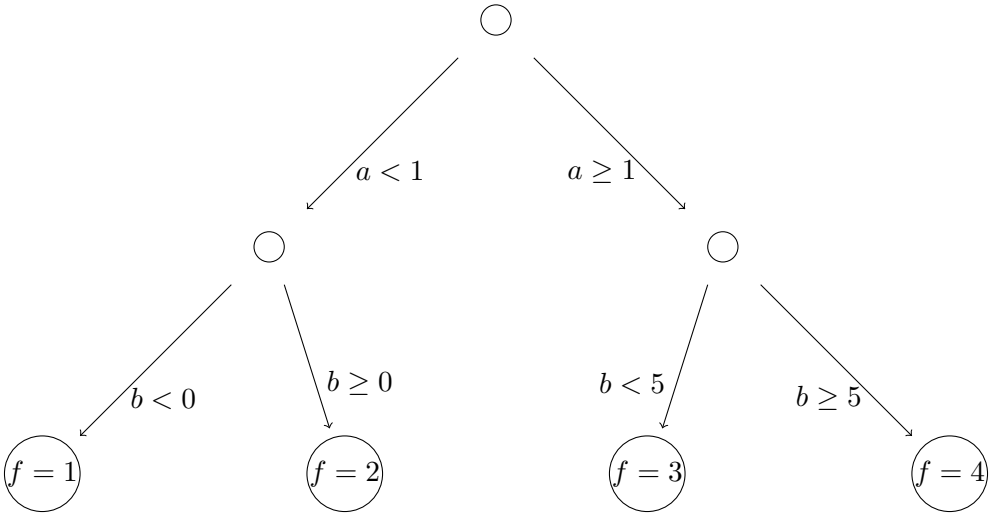


Figure 8.1: Schematic concept of a regression tree of depth 2 using predictors a, b to estimate some function f.

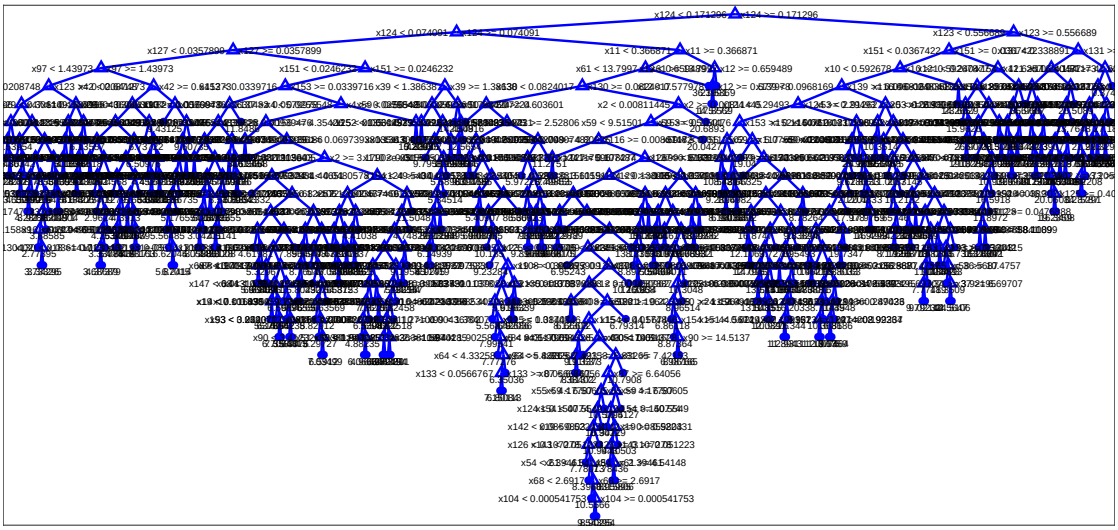


Figure 8.2: Example of what a realistic regression tree looks like.

does not immediately disqualify the algorithm even for prediction of motion sickness. The clue is finding a good transformation which transforms the prediction data into a hyperspace that is suitable for SVM. The investigation to find such a transformation is left to the reader.

8.4 Gaussian processes

The final machine learning algorithm is called Gaussian process regression, also known as kriging. This method aims to give the best linear and unbiased prediction of the intermediate values. It achieves this by computing a weighted average of the known function values in the neighbourhood of the point under investigation. The following analysis is based on [16].

Before looking deeper into the regression itself, we first need to build up a basic understanding of the relevant mathematics.

Definition 1 (Gaussian process)**.** *A Gaussian process (GP) is a collection of random variables, such that taking a finite subset of such random variables have a joint Gaussian distribution.*

An example of such a process is the height of the water surface as a function of time (or space), since we assume that the wave height at any point is presumed normally distributed around the mean water level. Gaussian process regression works on the principle of starting with a prior distribution over functions, to obtain a new distribution that has a very small error around known data points, but that has a larger deviation further away. This corresponds to the case where we know the behaviour of the function near measurements, but where we face increasing uncertainty the further away we get from these points.

First, let us assume that we live in a world where the training set X does not contain any errors, and that therefore the function value $f(\mathbf{x}_i)$ for $\mathbf{x}_i \in X$ is perfectly known. Let us also assume a prior of $f \equiv 0$. Let $K(X, X') \in \mathbb{R}^{|X|\times|X'|}$ be the covariance matrix, which we at this point will treat as a black box that shows how well the different points from X are correlated with points from a (different) set X'. In the literature, this matrix is sometimes also referred to as the covariance kernel. Then we know that for new data X' and $f' := f(X')$, their joint distribution is given by

$$\begin{bmatrix} f \\ f' \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X,X) & K(X,X') \\ K(X',X) & K(X',X') \end{bmatrix}\right). \tag{8.17}$$

Note that $K(X', X)$ and $K(X, X')$ need not be square matrices, but $K(X, X)$ and $K(X', X')$ are. Up to this point we have not done anything interesting: we merely acted as if we had a larger training set. However, we can now condition the predicted mean and errors of the new data X' on the older data. This way we effectively change the prior based on the recorded data. Another way to look at this step is that the initial assumption of allowing any function with mean zero is now being changed to only allow

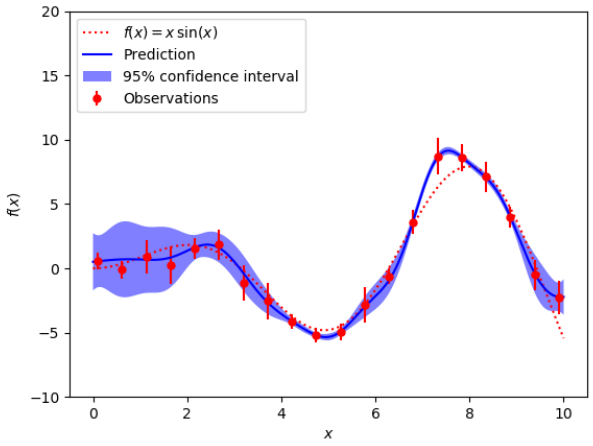


Figure 8.3: Example of Gaussian regression for $f(x) = x\sin(x)$, with errors in the training data. Note that the confidence interval becomes smaller if the training set agrees well with the prediction line (low variance), and larger if they do not [5].

functions that agree with the observed data from the training set. Mathematically, this step is given by

$$f'|X', X, f \sim N\left(K(X', X)K(X, X)^{-1}f, K(X', X') - K(X', X)K(X, X)^{-1}K(X, X')\right). \tag{8.18}$$

The actual function values f' can now be obtained by sampling the distribution above. Keep in mind that the distribution above is a vector, and each entry a function of the test data as well as the training data. This model can be improved to incorporate noise in the input data. Assuming that this noise is also Gaussian with zero mean and variance σ^2, we can replace $K(X, X)$ with $K(X, X) + \sigma^2$.

So what about this magical covariance matrix? The covariance function determines the smoothness of the estimated curve by putting different weights on the various types of errors. There are several ways to define the covariance function. A stronger covariance kernel results in curves that are less smooth since they put more emphasis on getting close to the training data. However, this comes at the cost of a greater risk of over-fitting. The most common covariance kernel is the squared exponential kernel and is defined as

$$K(X, X')_{ij} := k(\mathbf{x}_i, \mathbf{x}_j|\theta) \tag{8.19}$$

$$k(\mathbf{x}_i, \mathbf{x}_j|\theta) := \sigma_f^2 \exp\left(\frac{(\mathbf{x}_i - \mathbf{x}_j)^\mathsf{T}(\mathbf{x}_i - \mathbf{x}_j)}{2\sigma_l^2}\right), \tag{8.20}$$

where σ_l and σ_f can be computed as

$$\theta_1 = \log(\sigma_l) \qquad \theta_2 = \log(\sigma_f). \tag{8.21}$$

This is also the kernel that I am using when applying Gaussian regression unless specified otherwise.

The observant reader may have noticed that we are using two parameters which I am yet to formally define: θ and σ^2. In general there is also a third parameter β, which is the coefficient vector for the basis functions that describe the initial state if the prior is not assumed to be zero. The estimation of these parameters is done in such a way that we maximize the log likelihood on the training data:

$$\hat{\beta}, \hat{\theta}, \hat{\sigma}^2 = \arg\max_{\beta,\theta,\sigma} \log P(y|X, \beta, \theta, \sigma). \tag{8.22}$$

How this is computed exactly falls outside the scope of this work, but is standard included in any machine learning software package.

Finally, there are several issues regarding the computation worth noting. Firstly, the process above requires inverting $K(X, X)$. Matrix inversion is generally expensive, and results in a computational complexity of $\mathcal{O}(n^3)$ for a training set of size n. Furthermore, there are significant memory requirements for the scheme above $\mathcal{O}(n^2)$. For the specifics on these matters (and much more) we refer to [16].

8.5 Examples machine learning

Let us illustrate all this theory with some examples. Let

$$f(x) = \begin{cases} x \text{ for } x < 5 \\ 5 \text{ otherwise} \end{cases} \tag{8.23}$$

be the function that we want to approximate. Figure 8.4 illustrates how each of the models performs, when trying to predict f without noise. The linear estimate and SVM prediction are almost the same, since both try to establish some linear relation if the predictor is 1-dimensional. We are not trying to predict a linear function, and thus two models perform relatively poorly. We can see that the ensemble performs significantly better than the tree predictor. This makes sense, since there are no errors in the input data, meaning that the model cannot be over-fitted. This example also nicely illustrates the step pattern resulting from using regression trees. Finally, note that the Gaussian predictor estimates the function almost perfectly, with the exception being around $x = 5$, as a consequence of its assumption that f was smooth. The errors that are included in the legend are computed by taking the mean squared error (MSE) over the predicted values for $x \in [0, 10]$, by sampling this interval every 0.01.

Now suppose we also allow for noise, by adding a normally distributed error vector ($\sigma = 0.5$) to the function values for the training data. The results are given in Figures 8.5 and 8.6. The latter has been plotted using $f = x\sin(x)$ and also features a gap in the training data.

In the next section we introduce the final machine learning technique, artificial neural networks. The results from all these techniques are shown and compared in Section 10.

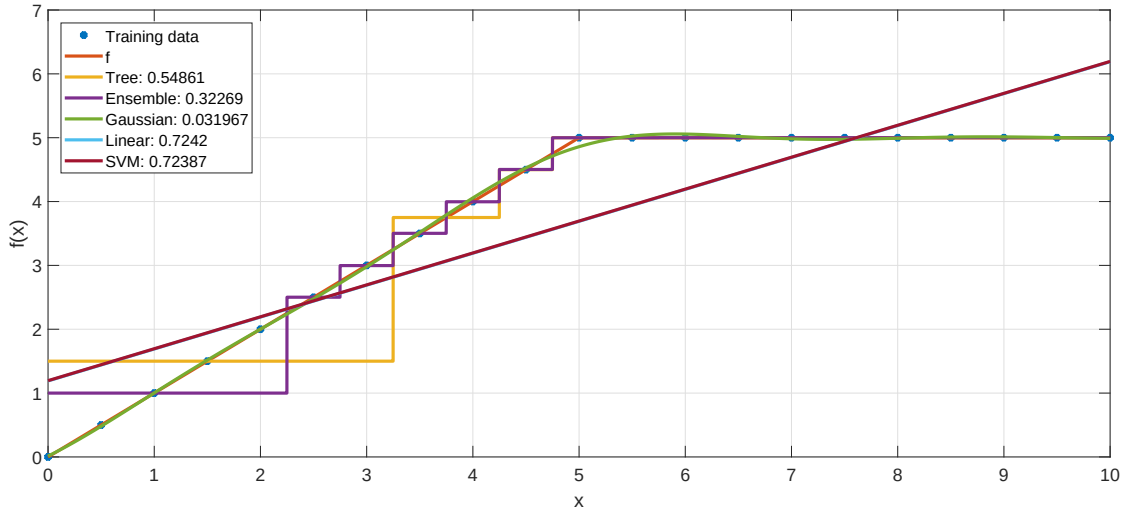


Figure 8.4: Example of how the various machine learning models perform in 1 dimension without noise. The linear approach and support vector machine result in the same prediction. Validation errors are provided in the legend.

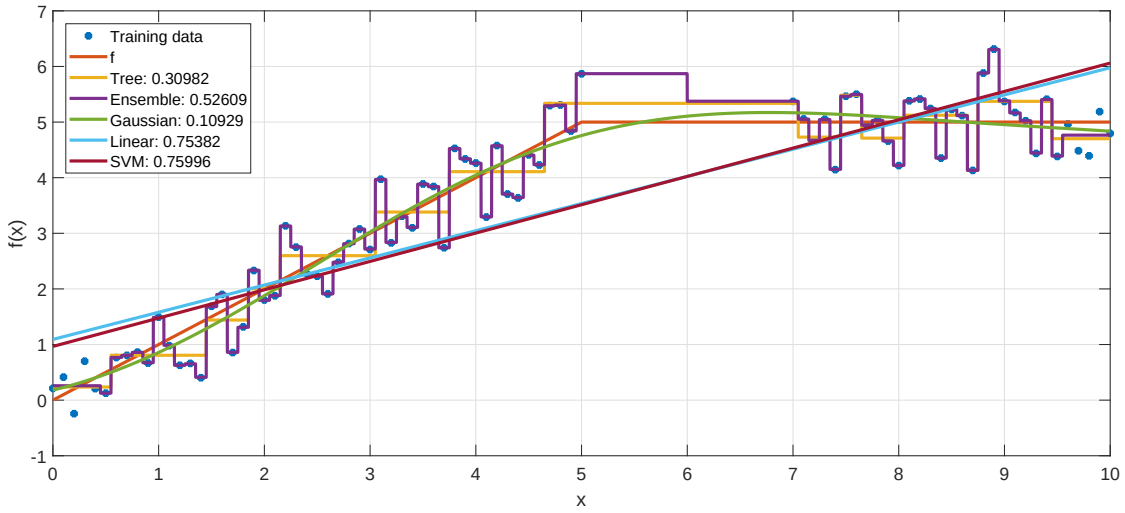


Figure 8.5: Example of how the various machine learning models perform in 1 dimension with normally distributed noise. The linear approach and support vector machine result in the same prediction. Validation errors are provided in the legend.

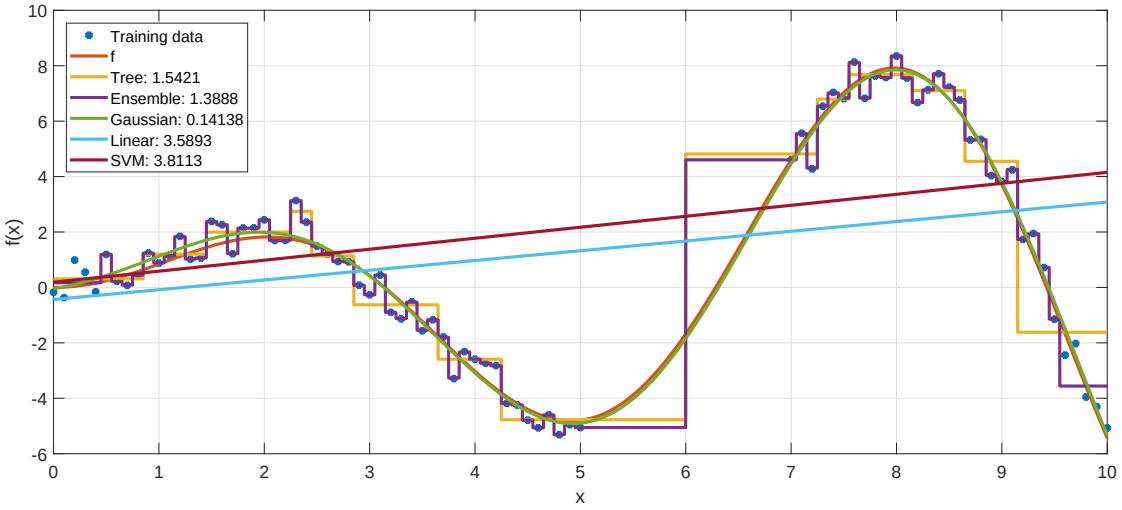


Figure 8.6: Example of the weakness of the tree and tree ensemble, using $f(x) = x\sin(x)$ with noise. Note that the ensemble performs worse than the tree since it is over-plotted near the noise. Also, near the gap in training data both tree model clearly have very significant errors.

9 Neural networks

Thus far we have introduced most of the frequently used regression models, as well as some techniques associated with it. There is one final candidate that is yet to be discussed, and more precisely, this concerns a class of regression learners: the artificial neural network (ANN). Similarly to the regression models, ANN's can be approached as black box models, but they are far more useful if we understand why these models work. The idea behind neural networks is that they work similarly to how our brain processes information: it receives a bunch of inputs from our senses, and from this information manages to extract useful knowledge about the world around us. The key aspect here is that in doing so, our brain does not need to have a complete understanding of the processing underlying its observations, and is capable of doing so using neurons that can only be on or off. The goal of neural networking is to simulate this process, by creating artificial neurons that do the same, but in a more efficient manner, since we do not have trillions of neurons available. In theory, ANN's should perform really well, as the universal approximation theorem for ANN's tells us that the space covered by ANN's lies dense in the space of continuous functions $\mathbb{R}^n \to \mathbb{R}^m$. Practice shows, however, that neural networks require immense amounts of data to obtain said accuracy. In the remainder of this section we introduce the basic concept of artificial neural networks, then we show how to complement this with backpropagation. Finally some concrete algorithms for ANN's from literature are discussed.

9.1 The basic concept

As said above, an artificial neural network tries to mimic the way our brain works. It takes some inputs, and tries to approximate the output we are interested in. The simplest neural network looks like the diagram below in Figure 9.1, and consists of the inputs, the input layer, the hidden layer(s), the output layer and finally the output itself.

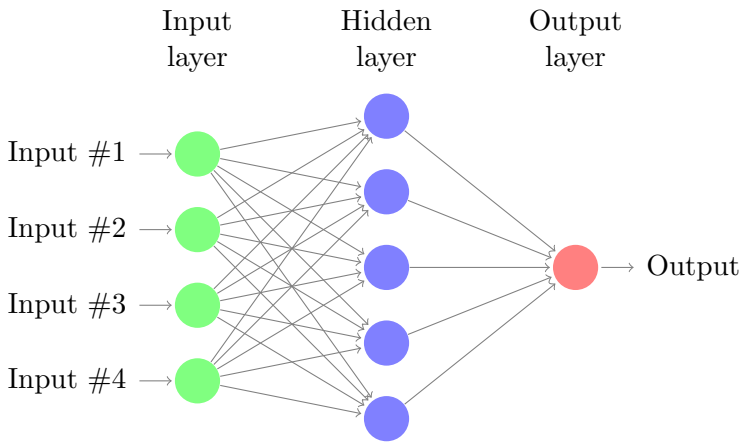


Figure 9.1: Basic neural network structure [21].

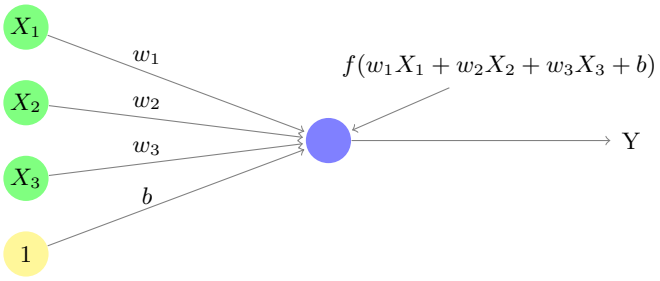


Figure 9.2: Sketch of the inputs and output of a single neuron, which is connected to three other neurons X_i and attains value Y. The w_i represent the weights on the various edges, and b the bias (which is not a neuron).

Whilst this diagram only features a single hidden layer, there can be many (say, a dozen) in general. We distinguish between 4 distinct aspects within this diagram:

1. The input and output layers, in which the predictor and response variables are represented.
2. The hidden layer, which for now we consider to be a black box.
3. The neurons, which form the basis of each of the layers.
4. The connections between the various layers, which form a complete bipartite graph between consecutive layers.

The input and output layer are already familiar: these are the same as for any other machine learning model. However, neurons usually parse their output to a finite interval, say [0,1], by means of an activation function f. More on this function comes later. This way we avoid that the network becomes linear. Then we get one, or usually several, hidden layers, which consist of a number of neurons. Both the number of neurons per layer and the number of layers can be chosen as input to neural network creation. The hidden layer usually has no meaningful interpretation, and in general cannot be interpreted at all (alas the name). Between each two adjacent layers, a complete bipartite tree is drawn, after which each edge is assigned some weight. This is illustrated in Figure 9.2. Then the value of a single neuron can be computed as a function of the weighted sum of the neuron values from the previous layer plus some bias. Note that all these weights and biases are input to the neural network.

The activation function $f : \mathbb{R} \to \mathbb{R}$ is used to determine what value a neuron attains, and is used primarily to introduce non-linearity into the model. This is important, as otherwise we might as well use linear regression. Popular choices for this function are the Sigmoid function $f = (1 + \exp(-x))^{-1}$, the hyperbolic tangent, or the rectified linear unit $f = x_{x>0}$. These result in values in $[0, 1]$, $[-1, 1]$ or positive values respectively.

As the algorithm runs, all of the weights and biases change. How we do this exactly is called the learning rule, and what rule is used depends on the algorithm. Still, the general concept is the same: we want to minimize the error between the various outputs

and the predicted outputs, by means of a cost function. Whilst this is once more subject to choice, usually the mean squared error is used. Given a training set (X, Y) of size N, this means we want to solve the minimization problem

$$\min_{w,b} C(w,b) := \min_{w,b} \sum_{i=1}^{N} \left(\mathrm{ANN}_{w,b}(X_i) - Y_i\right)^2. \tag{9.1}$$

This is not an easy problem to solve. Suppose we have 150 input variables, 10 hidden layers of 10 neurons and 1 output, then we have 150*10+9*10*10+101=2501 variables to optimize over (number of edges plus biases). However, when using neural networks for image recognition this can easily extend to tens of thousands of variables. One way to solve this problem is by means of gradient descent, but there exist various other solution methods for this problem. Gradient descent was originally proposed by Cauchy and assumes that we can find the global solution by searching for a local minimum. This can be done by looking for the direction of steepest descent, which can be found by computing the gradient. Then a small step is taken in the direction of this gradient, after which this procedure is repeated. We stop when the solution stops improving, i.e. the derivative is very small in all directions. One can think of this procedure as a ball rolling down a mountain in our hyperplane in the direction where the slope is steepest, and we stop moving when the slope becomes (almost) horizontal. We assume that this means we have found a local minimum. This gradient search is repeated several times, so that we find a multiple local minima. We assume that the best local minimum that we have find this way either is the global minima, or has a function value very close to the global minimum, and is returned as the tentative optimal solution. A more detailed explanation of gradient descent, which in particular focusses on how to choose the step size, is given in [23]. The main problem left for machine learning is how to compute the gradient. As we apply a non-linear function at each neuron, an analytical approach is not feasible. A numerical approach, on the other hand, is also not desirable, since this requires evaluating the neural net in the direction of each of the variables, of which we have thousands. This is why we use backpropagation, which avoids this problem altogether.

9.2 Backpropagation

So really, the algorithm tries to solve some abstract optimization problem in a very high-dimensional space. It must therefore not come as a surprise that the solution in the hidden layers cannot be interpreted very easily.

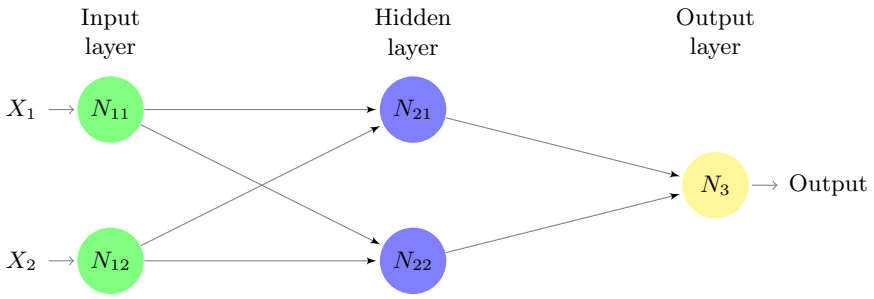


Figure 9.3: Example neural network with 2 inputs, 1 hidden layer with two neurons, and 1 output.

One way to improve this is called backpropagation or backwards propagation of errors. For the sake of explanation, suppose we use no activation function, and have binary inputs and outputs. Now consider the example from Figure 9.3. Suppose that the initial state is given by Figure 9.4.

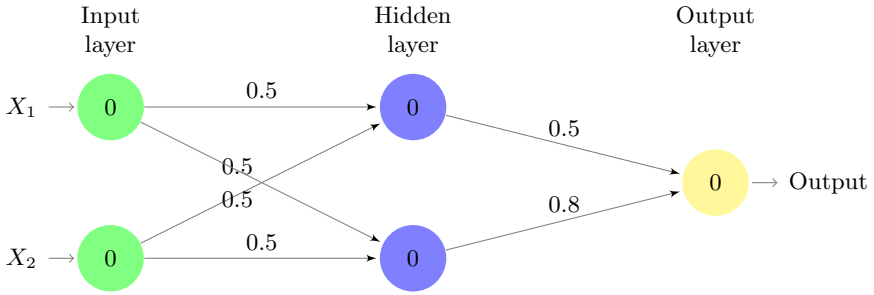


Figure 9.4: Example neural network where the values in the nodes give the respective biases, and the values along the edges their respective weights.

Now suppose that we have an observation $X = (1, 0)$ with $Y = 1$ as training element. If we follow the network, we can use the current network to get a prediction as is shown in Figure 9.5.

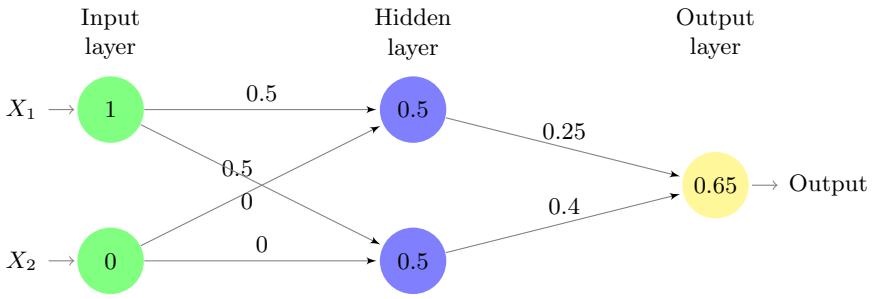


Figure 9.5: Values of prediction scheme using the training data $X = (1, 0)$. Along the edges we show the relative contribution of its source node to its recipient. Thus the network currently predicts the response to be 0.65.

The neural network currently predicts a value of 0.65, and therefore makes an error of 0.35. Since we want to do better, we consider the weights w_{31} and w_{32}. If we consider the impact of each of the weights, we note that changing their weights results in equal rise of function value, and this occurs with rate 0.175 (as both hidden neurons have value 0.5, and the output neuron needs to change 0.35). On the other hand, increasing the bias of the last neuron results in an increase rate of 0.35. Finally, increasing the values of the hidden neurons results in a change of $0.5 \cdot 0.35 = 0.175$ and $0.8 \cdot 0.35 = 0.28$ respectively (the weighs of the respective edges). Now we repeat this process for each of the hidden nodes: we compute the rate at which of changing the values of single weight or bias changes the final value. For this we need to sum the impact a neuron has when it affects multiple neurons in the next layer: changing the first impact layer can result in a change for both of the hidden neurons. In general this sum can take both positive and negative values. Therefore, this computation is generally done per layer, after which aggregation can be done immediately. This way we obtain the following state:

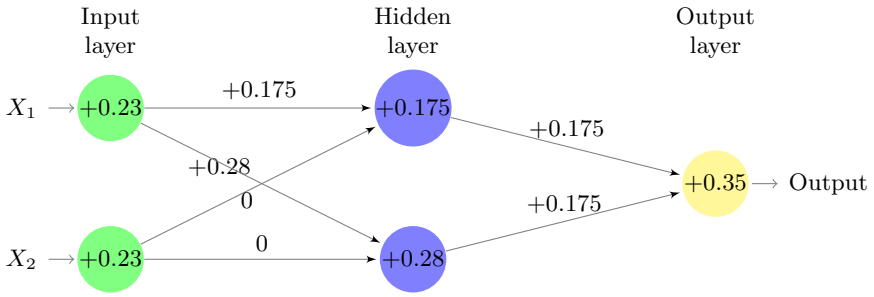


Figure 9.6: Contribution to the backpropagation using the training data $X = (1, 0)$ with $Y = 1$. In the nodes we show the relative change in bias, whilst in the edges the relative change in weight is given.

Note that this leaves us with a rate for each of the parameters. It can be proven [9] that the rates we end up with, only need to be normalized to get the gradient. However, in general we have more than one training example, and then the final result is the

average over all of the gradients that are computed in this way. In practice, this is not done, as usually the set of training data is extremely large. Then, only a portion of the data (batch) is used to compute the gradient at that step, and a different batch is used in each iteration. This improves the training time, without losing a lot of accuracy.

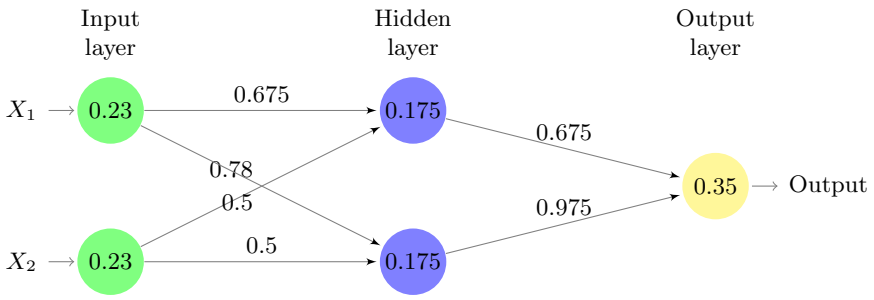


Figure 9.7: The new biases and weights for the same network after learning with example $X = (1, 0)$ and $Y = 1$ via gradient descent without normalizing and with step size 1.

Now consider the updated network for Figure 9.7. Does this network actually perform any better? The short answer is no: upon re-evaluating the network for the same training example we find that the network currently predicts 2.31. The long answer, however, is that we did not normalize and the step size is too large. This meant that all of these changes combined were too much, and we should have chosen a (much) smaller step size. As was mentioned earlier, more details on how to choose the step size are given in [23].

9.3 Specific models

The algorithm for learning an artificial neural network presented thus far has many different variants. For example, how we distribute the training data over the various batches is currently not discussed. One can simply divide the set of training data and then use each of the batches once, or use each one several times. In the latter case, there is yet more choice, as we can use the various batches in a cyclic manner, or pick one at random. We can even reshuffle the batches in between iterations.

Let us mention one specific model that stands out when used for MSI prediction, called Bayesian regularization. Below the general concept of Bayesian regularization is briefly introduced below, but the specific variant that is used is by Foresee and Hagan [4]. Bayesian regularization makes use of the Levenberg-Marquardt algorithm, which is used as a generic least squares solver like gradient descent. Bayesian regularization furthermore makes use of principles seen in Gaussian regression (Section 8.4). The Levenberg-Marquardt involves estimating the Hessian of our target function at each time step, which can be computed via backpropagation similar to how gradient descent uses the gradient.

Regularization implies that rather than trying to minimizing the network errors E_D, we add an additional penalty E_W term for the network. Here E_W is defined as the sum of the squares of the network weights. So the objective function we try to

minimize over, which was E_D, now becomes a minimization problem with objective function $\alpha E_D + \beta E_W$. How to choose these parameters α and β is rather involved, but it is based on a derivation using Bayes' rule, and discussed in detail in [12]. α and β are both updated at each iteration, and therefore the objective function keeps shifting. The algorithm terminates when the objective function is no longer changing. The final solution is then computed via one more step of the Levenberg-Marquardt algorithm.

10 Results

As we saw earlier, the analytical approach did not achieve the desired accuracy. Hence, we now approach the problem using machine learning and ANN's. In this section we present the results from both approaches. To provide a reference frame: a best-fit linear model, which did not provide sufficient accuracy for MSI prediction, achieves a mean squared error of approximately 21. Below, we first justify our choice of machine learning algorithms via some initial testing. Then, we discuss which predictors have been used precisely, and show the results from extensive testing with the various machine learning models. At this point we validate our choice of predictors, and show their relevance in the trained machine learning models. Here we see that not all predictors achieve the relevance that we expected, and try to give an explanation where possible. To show the applicability of a model trained on a location to other sites, we test the trained models on a different (but comparable) site, called the Westermost wind farm. This comparison illustrates that a model trained on one site can not be immediately be used on another, and entirely new models must be used instead: by training the models on data from the Westermost wind farm, the machine learning models achieve the same level of accuracy as the tests on Greater Gabbard. Then, the same is done for neural networks: we explain our choice of specific ANN algorithms as well as the other parameters used, and try to deduce their significance. Contrary to the machine learners, the ANN's perform significantly better on the Westermost wind field. We end this section with a comparison of the results from the theoretical approach with those from machine learning and ANNs, and conclude that the best predictive model can be trained using a well-configured ANN.

10.1 Model choices machine learning

The models for which the results are elevated below are tree regression, tree ensemble regression and finally Gaussian regression. The choice to restrict to these models is made after some preliminary testing with the dataset using support vector machines and linear regression, see Figure 10.1 and 10.2. Figure 10.1 shows that SVM makes errors on the training set that are already significantly higher than the alternatives make on their validation data. In Figure 10.2, the linear model is shown to make errors that are significantly higher than the alternatives. Combined with the low training errors for linear regression, we can only conclude that these models are either over-trained or try to describe a non-linear process using linear functions. Either way, the errors made by SVM and linear regression are approximately twice as large as those made by the tree regression and Gaussian regression, and hence no additional time has been invested into these methods. From now on, we only consider the regression tree, the tree ensemble and Gaussian regression.

10.2 Predictors

As was mentioned earlier in Section 6, the choice of predictors is very important for getting good predictions. Below, each of the predictors used for predicting MSI are

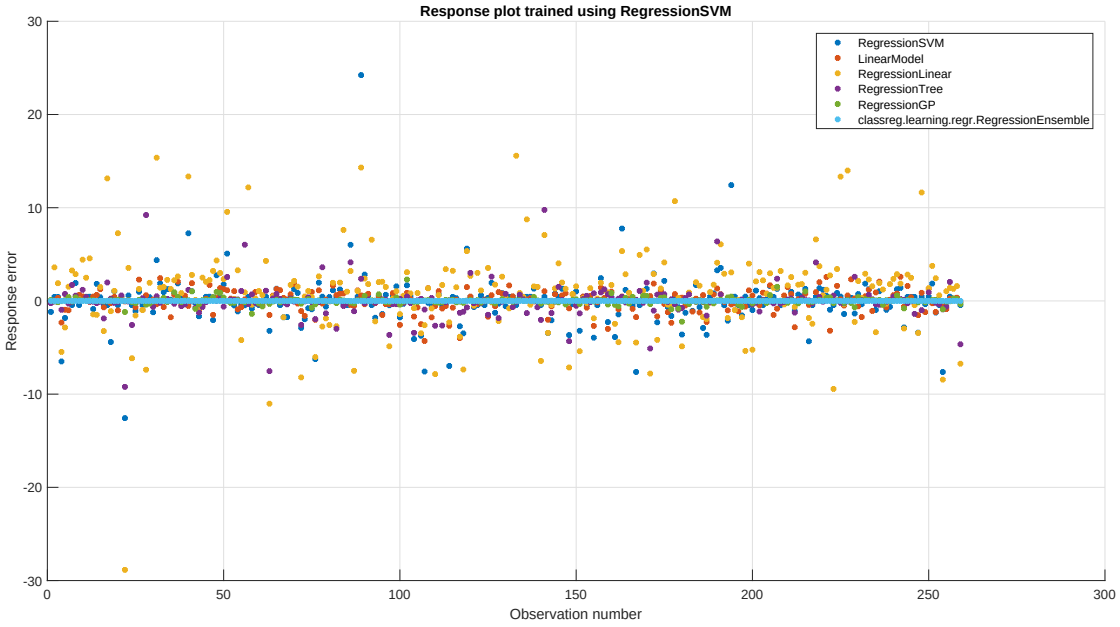


Figure 10.1: Error plot of the Iceni Venture for training data for various models. Errors are limited to ±25 to keep the plot easier to read, but they may in fact be larger. Note that especially the regular linear model (yellow) and the SVM model (darker blue) perform very poorly.

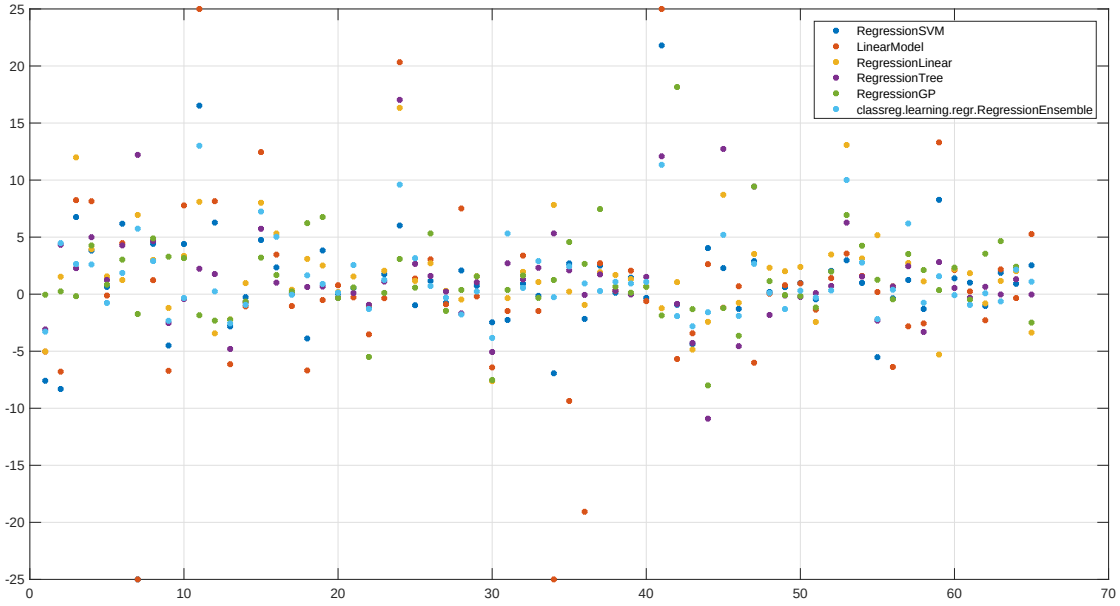


Figure 10.2: Error plot of the Iceni Venture for validation data for various models. Errors are limited to ±25 to keep the plot easier to read, but they may in fact be larger. Note that especially both linear models (yellow and red) perform very poorly (both a regular as well as a robust linear model are tested).

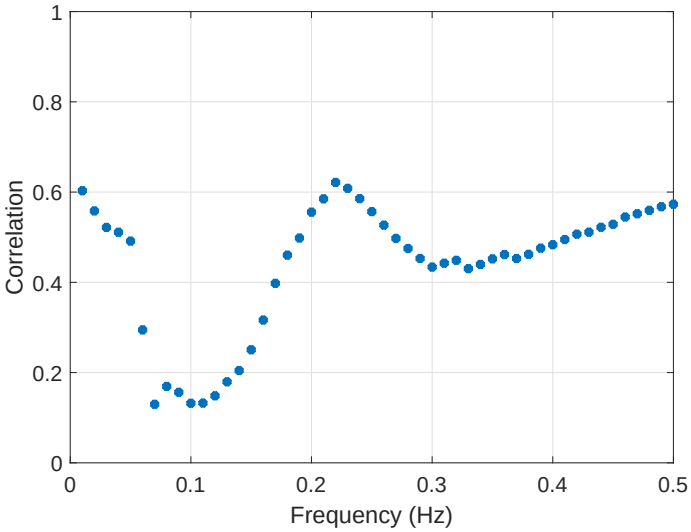


Figure 10.3: Correlation between the wave spectrum and MSI.

introduced together with their importance in modelling MSI. For each of these predictors the correlation between the predictor and the normalized MSI (MSI normalized to a 1-hour period) is given, and an explanation is given where possible.

10.2.1 Wave spectrum

As the first predictor we use the wave spectrum for heave motion, as is recorded by the Cefas buoy near the Greater Gabbard wind farm. This is a significant predictor for motion sickness as experienced by the vessel: in Figure 10.3 we can see that there is a correlation of almost 60 percent between normalized MSI and the wave spectrum. It is also immediately clear that it is not sufficient information to accurately predict MSI using any of these values as individual indicators. An interesting observation is also that the extremely low frequencies appear to have significant correlation with MSI (> 0.5 for frequencies between 0.01 and 0.04). Whilst this might be an indicator for another process, no clear reason as to why this peak exists can be given from a purely physical standpoint.

10.2.2 Significant wave height

The significant wave height H_s is the single most important wave characteristic to describe the sea state, and also was the single best predictor for MSI using just the linear model. It should therefore not come as a surprise that H_s is the parameter that has the strongest correlation with MSI in the set of features that we use, with a correlation of 0.591.

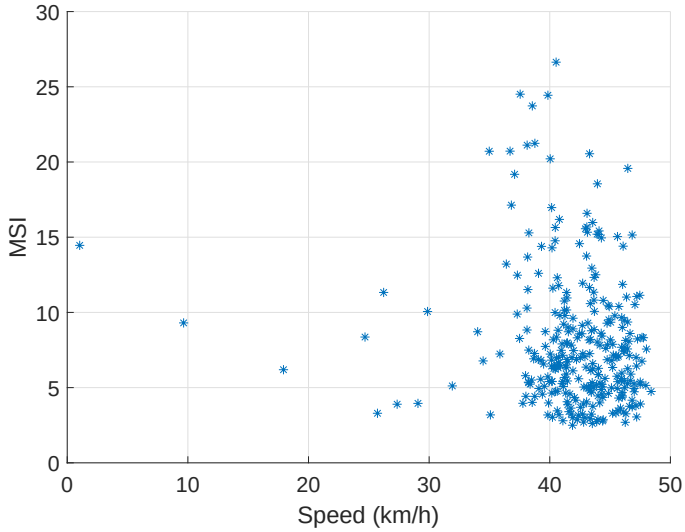


Figure 10.4: Scatter plot of speed versus MSI.

10.2.3 Speed

Speed is defined as the speed over ground (SOG) and computed by looking at the difference in GPS coordinates over time. Surprisingly, there exists a (weakly) negative correlation between the vessel speed and MSI of -0.147. While this may not be a parameter that is known prior to departure, it is one that we can manipulate, and can in theory be changed as to minimize the MSI. A scatter plot of vessel speed versus recorded MSI is given in Figure 10.4. There we see that this correlation is not the result of non-linearity, but rather of speed not being able to be used as a predictor, since no trend is distinguishable by inspection. The only plausible explanation for this phenomenon is that the vessel may be forced to slow down as a result of extreme weather. Under that assumption, we cannot deduce any statistical information from the dataset regarding speed, as the data must be skewed. A second explanation may be that if the vessel is going faster, the assumption of the vessel following the waves is no longer true. This might mean that the vessel skips over a larger portion of the waves, and therefore experiences fewer motions in general.

However, the relative speed of the vessel, here defined as the speed differential between the vessel and the wave corresponding to the wavelength where the most energy is allocated, is weakly positively correlated (0.084). Still, this is not a result of any statistical significance.

10.2.4 Predictor wave spectrum

The expected wave spectrum is obtained via the parametrization of the vessel from the analysis in Section 5. To compute this spectrum we use the speed, relative angle,

standard deviation for the relative angle and wave spectrum from all of the Cefas buoys where available. If we have 3-dimensional data available, we use those values instead of the 1D wave spectra. Where missing, we substitute the relative angle to equal $\frac{\pi}{2}$, i.e. port side waves. The standard deviation defaults at 0.5, which is roughly equal to the average standard deviation. If wave spectra are not available, we discard the dataset for that day entirely. The output is the expected spectrum in the range of [0,1] Hz.

10.2.5 Indicators

Besides all the previous predictors, we also include an indicator parameter that indicates whether the model was trained using recorded wave directions from the 3D wave radar, the 1D wave radar, or the default values (in case no directional data is available). This might specifically benefit tree learning algorithms, as they can easily classify the results based on this criterion. This way, the indicator serves as an additional layer of accuracy.

10.2.6 Vessel length

The length of the various vessels is also included, as this effects which wavelengths the vessel is most affected by. It also provides us with an easy way to distinguish between the various vessels. It has a correlation of -0.14 with MSI, which makes sense: vessels should be deployed up to their respective operational limits, and (at least in theory) these limits correspond to the same type of vessel motions. Thus, for both smaller and larger vessels, the MSI attains low and high values.

10.3 Results machine learning

Using machine learning we obtained results with an MSE of roughly 10 on the global dataset. The results are oblivious to the vessel type and plotted in figures below, where training and validation has been performed with the same datasets. We see that the three prediction methods provide very similar results, and testing this several times shows that no method ever gets a significant advantage over the others. Moreover, prolonged testing showed that each of these methods outperform the others on selected partitions. To compare their results, we performed the Kolmogorov-Smirnov goodness-of-fit hypothesis test, to see if either model performed significantly better. The results used to perform this comparison are 100 cross-validations, which have been performed on each of the models. These result in average MSE errors of 11.20, 11.96 and 11.41 for Gaussian regression, tree regression and the tree ensemble respectively. With these results, we find that the null-hypothesis that the models result in errors that follow the same distribution is rejected, as the probability that all errors follow the same distribution is less than 10^{-10}. This holds for both tree models relative to the Gaussian regression. Since the mean of Gaussian regression is significantly smaller than its alternatives, we conclude that Gaussian regression has the best performance of all the machine learning models. Illustration of the error distribution for all three models are given in Figures 10.5 and 10.6. At this point, however, we still cannot make any claims about the actual

performance or robustness of these models. Therefore, we now investigate the impact of each of the individual parameters on the model.

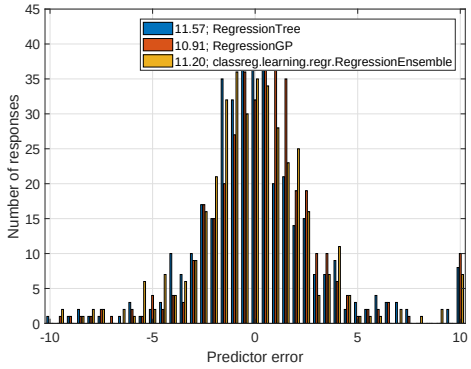


Figure 10.5: Error histogram on global dataset.

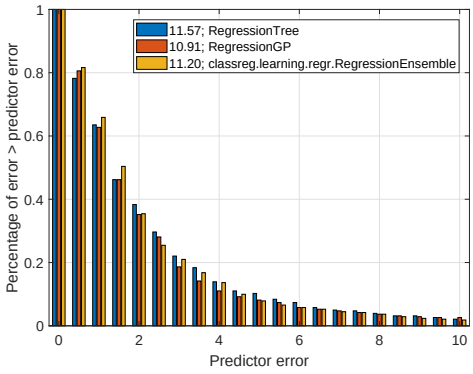


Figure 10.6: Cumulative error histogram on global dataset.

10.4 Influence individual parameters

In order to investigate how the final model behaves, we consider the impact of the individual predictors within their valid range. This is done by training the data on a dataset, i.e. on the combined input of all vessels serving the Greater Gabbard site. The results are shown in Figure 10.7. Since we need to keep the remainder of the parameters constant for this plot, we predict MSI using one instance of the training data, and then vary one particular parameter. The plotted data here are representative instances of the behaviour of that parameter for the choice of the training data. Consequently, the absolute value for the MSI in any of these plots has no significance, but the deviation in the error does.

When considering H_s, there is only minimal impact on the expected MSI. This might come as a surprise: this was the single most important predictor beforehand. However, H_s can be derived from the wave energy spectrum, and therefore be implicitly used in the model. Also, notice that the indicator for whether or not directional data was used had absolutely no impact on the expect MSI.

Speed also poses an interesting question. Whilst the correlation analysis showed that the speed is almost perfectly uncorrelated to MSI, there appears to still be a significant connection to MSI. This probably implies that this parameter should not be considered by itself, but rather by applying some transformation, or in combination with one of the other predictors.

The indicator, which shows whether or not the wave data contains the 3D spectral data, appears to have no impact at all, as all three models are invariant under the indicator. Therefore, we could remove this freely.

For considering the effect of the energy spectra for the raw cefas data on MSI, we have chosen to scale the spectrum by a factor between 0.1 and 2. Whilst the tree learners vary wildly, the Gaussian predictor does not show any change at all. Although the MSI

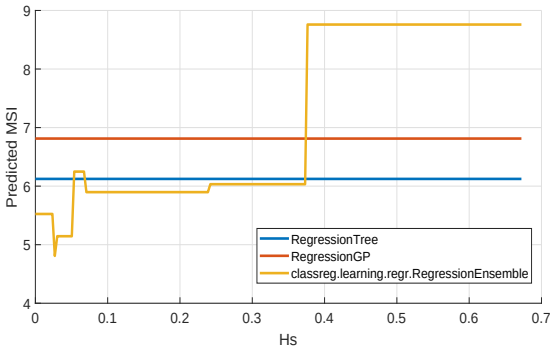
here is limited, and therefore also the net error, it is surprising that the behaviour of the regression ensemble actually shows a decline with increasing energy spectrum. This is in sharp contrast to the results from applying the same procedure to the parametrized energy spectrum, where the Gaussian predictor shows a strong positive trend for almost all of the energy spectra. The prediction from the other models remain more or less constant, although we can often notice large fluctuations.

Let us consider the effect of scaling both energy spectra. Now we notice that all three models follow a positive trend, with the tree models fluctuating around the Gaussian predictor. This holds until we consider the higher energy spectra, and we attempt to extrapolate beyond the data by looking at what happens beyond the dataset. A good example if given in Figure 10.7g. We see that initially the predicted MSI rises as expected, but after the scaling factor approaches 1, we initially see a drop, likely due to sparsity in the training set that results in moderate over-fitting, followed by a rise to roughly its original peak level, and remaining constant afterwards. For the Gaussian predictor we suspect this behaviour to be a result of the lack of data beyond this point. Therefore the Gaussian model assumes nothing about this function,. and the most likely trajectory is a straight line beyond the final training point (since this is the last known tangible point), as we see in the figure.

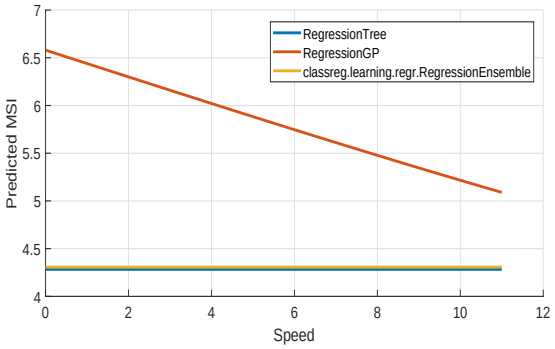
Finally, to see how important each of the predictors is, we have removed each of the parameters to see how well the model performs entirely without them. The results are given in Table 10.1. Note that the values from relatively small sample sizes, but it is clear to see that removing the predicted wave spectrum improves the performance for both the tree regression algorithms, and their performance even supersedes the Gaussian regression using all predictors. At the same time, removing the predicted wave spectrum only worsens the prediction from Gaussian regression. The best explanation we can give for this is that the Gaussian regression is better at using all of the data at the same time, while tree regression only benefits if it uses predictors with more dense information about the response variable.

Table 10.1: Average MSE when omitting one of the predictors. The results are obtained by performing 5 cross-validations, each time with a single predictor removed, and then compute the mean error.

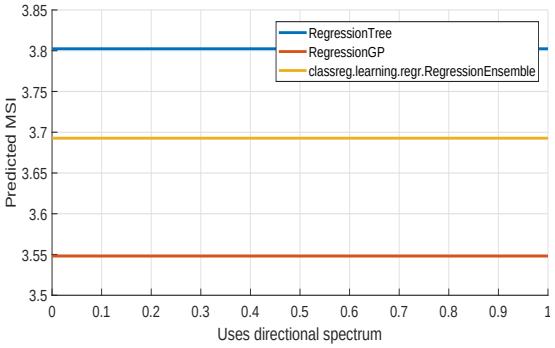
Omitted predictor	Tree regression	Gaussian regression	Tree ensemble
Predicted vessel spectrum	11.41	12.10	10.79
Raw Cefas spectrum	15.11	12.68	15.05
H_s	11.86	11.24	11.19
Speed	12.12	11.74	11.62
Indicator	11.46	11.22	11.58
Vessel length	11.88	11.60	11.11



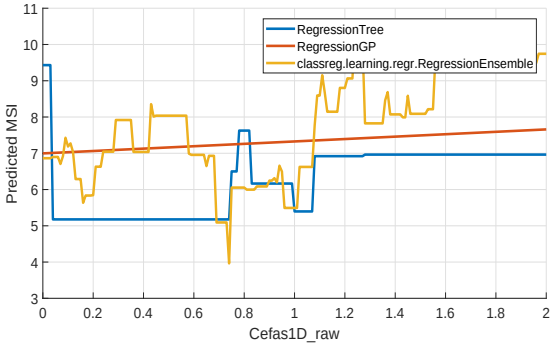
(a) Predictor H_s.



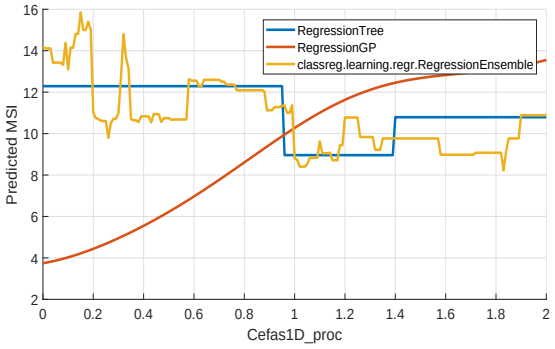
(b) Predictor speed.



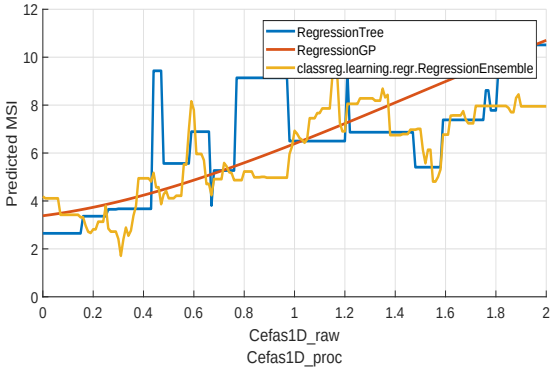
(c) Impact classifier for directional data.



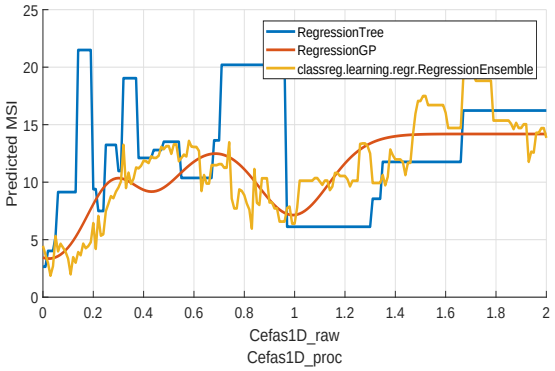
(d) Predictor scaled raw Cefas heave energy spectrum.



(e) Predictor scaled parameterized acceleration energy spectrum.



(f) Predictor scaled both raw and parameterized energy spectrum.



(g) Predictor scaled both raw and parameterized energy spectrum.

Figure 10.7: The effect of the various parameters for each of the three machine learning models. H_s, speed and the indicator are in absolute value, whilst the spectra show expected MSI versus the scaled energy spectra.

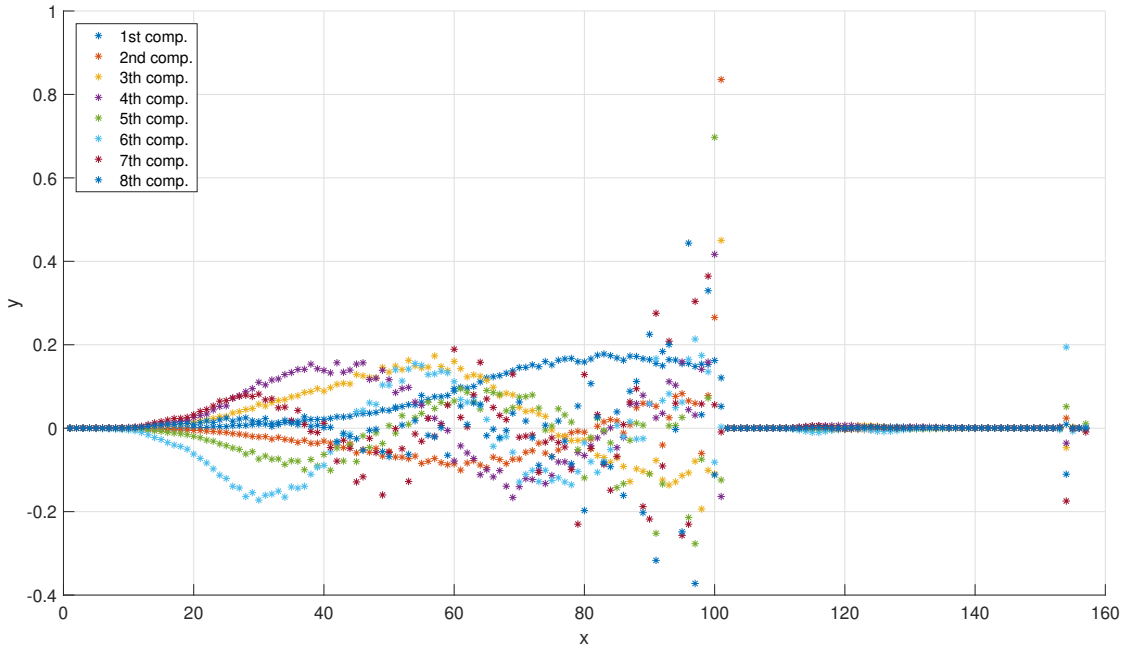


Figure 10.8: First eight principle components found when applying PCA to the predictors to the Greater Gabbard data.

10.5 Results PCA and ICA

To further investigate the influence of the various parameters, we apply the techniques from principle and independent component analysis. In Figure 10.8 we see the first eight principle components found when considering all the predictors. The limitation to eight components is to keep the figures readable. The first 100 entries correspond to the predicted vessel spectrum, entry 101-150 to the raw wave spectrum and the final few to the remaining predictors. The main take-away, however, is that the most influential process in the data is the predicted vessel spectrum. The wave spectrum itself does not have any coefficients exceeding 0.1 until the 72'th component. This may seem as an antithetic result: we have seen regression trees perform better without the predicted vessel spectrum. However, one explanation is that most of the signal resulting from the predicted wave spectrum is the result of noise within the data. Still, Gaussian regression performs better with the predicted wave spectrum, which suggests that at least some of the signal contains useful information that is not linearly related to any of the other predictors. In Figure 10.9, the first four principle components of the predictor space without the predicted wave spectrum are shown. Here a similar result to what we have seen before is displayed, with the significant wave height (2nd component), the vessel length (3th component) and the indicator together with wave spectrum, forming the 5th principle component. We cannot explain the first principle component, which appears to have a peak at 0.01 Hz (this the same peak found in the 2nd principle component with all predictors).

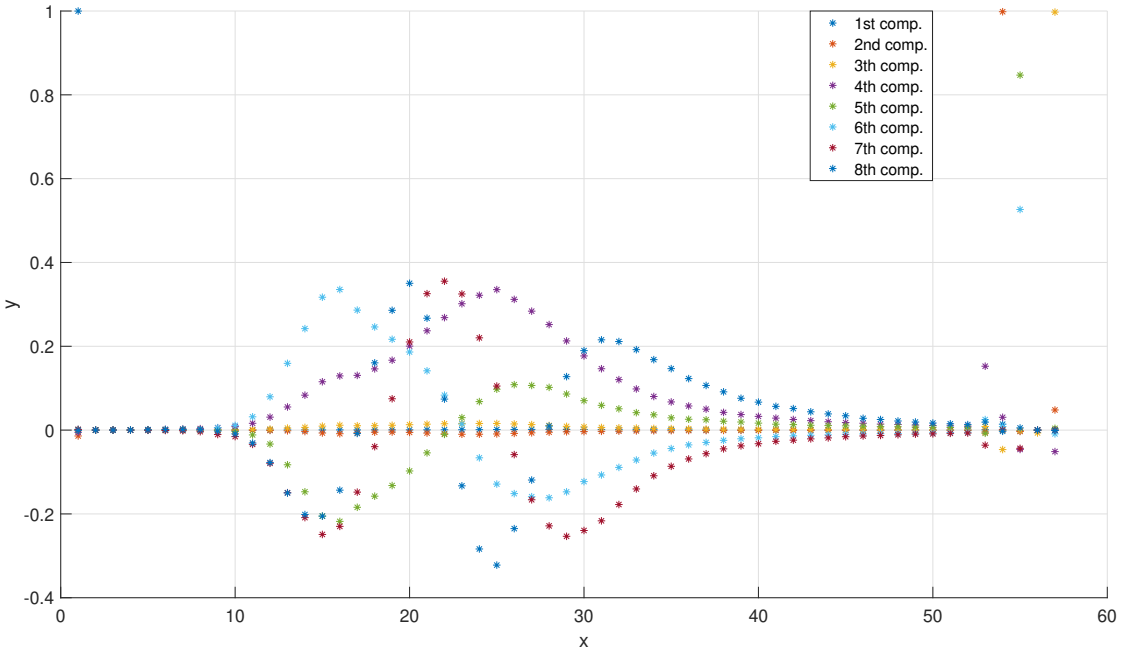


Figure 10.9: First four principle components of the predictor data without the predicted wave spectrum.

The question now becomes: Can we separate the data into a small number of independent signals, one of which might be useful for MSI prediction? The answer turns out to be no. As can be seen in Figure 10.10, there is no distinguishable signal, but rather several standalone peaks in the predicted vessel spectrum, and one outlier corresponding to vessel length. We expect the peaks in the predicted vessel spectrum to be the result of the averaging procedure in the computation of the predicted vessel spectrum. It is our belief no big errors are made, but rather the same error is made every time. This is supported by the observed ripples in Figure 5.9, as well as the observation that the fourth independent component has several peaks all at the same height. Each of these peaks is a multiple of 0.05 Hz apart, which corresponds to the observed ripples. Zooming in further shows that this is also the case for several other components. Removing the predicted vessel spectrum does not yield any new observations.

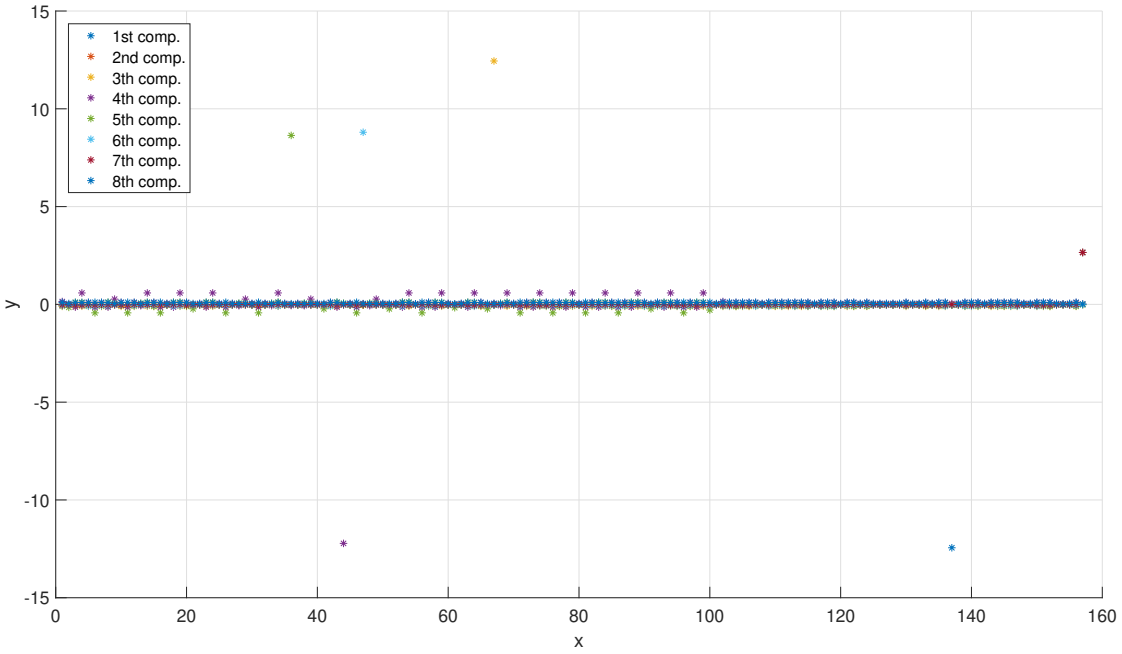


Figure 10.10: The first eight independent components found after performing an ICA with all the predictors.

10.6 Comparison with a different field

In order to get a better idea of how well the trained model works, we validate the model on a second wind field. The field is called the Westermost Rough wind field, and is located some 350 km north-west of the Greater Gabbard wind field. Westermost Rough is still part of the English channel. Therefore we might expect similar wave behaviour to the Greater Gabbard test site, which makes it an ideal test case. An illustration of the route taken by vessels serving this field is given in Figure 10.11. Validation is performed on two sister ships called the Eden Rose and Ginny Louise, which are catamarans with a length of 20 meters, which makes these average to the ship models used on the Greater Gabbard site. For this procedure we use the model that is trained on the various vessels serving the Greater Gabbard wind field, with no training data on the Westermost Rough field at all. The results are displayed in Figure 10.12, and show that the performance is horrible, with MSE over 30.

But what happens if we train on the Westermost site? It turns out to be much better than the performance seen thus far on the Greater Gabbard, as can be seen in Table 10.2. Whilst only 1 type of vessel serves this field, this cannot explain the performance differential: Restricting to one vessel type in the Greater Gabbard site does not show any significant improvements.

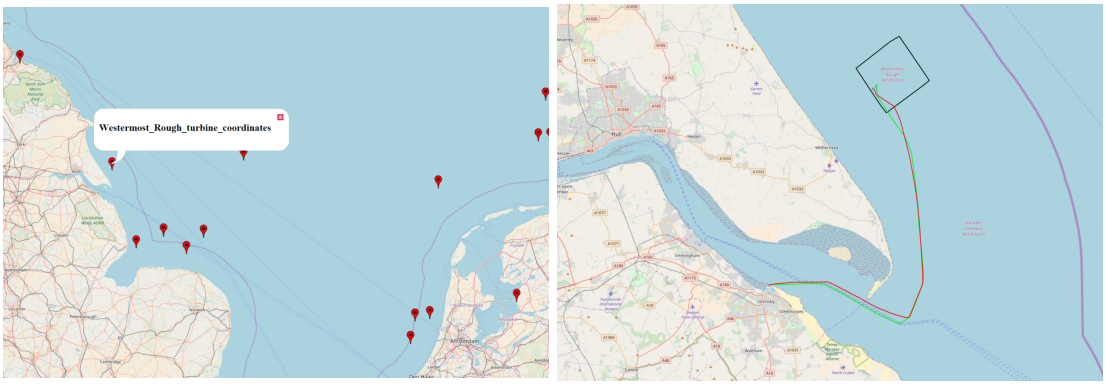
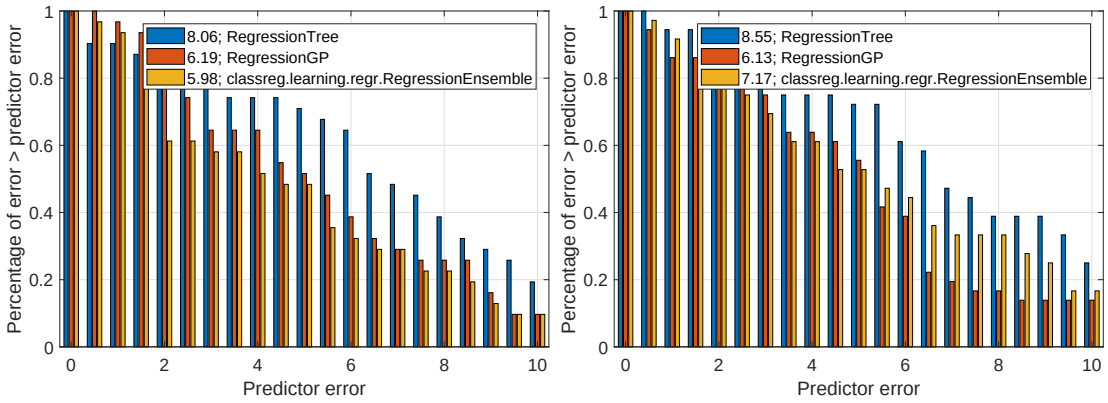


Figure 10.11: Westermost wind field location.



(a) Eden Rose.

(b) Ginny Louise.

Figure 10.12: Cumulative error plot for two vessels on the Westermost wind field.

Table 10.2: Performance regression models at the Westermost. Errors are given as mean squared errors averaged over 100 trainings.

	Gaussian	Tree	Ensemble
All predictors	5.17	7.51	7.77
Without predicted wave spectrum	6.86	7.52	6.77

10.7 Results neural networks

As mentioned before, there exist a lot of algorithms for machine learning. Preliminary testing showed, however, that the performance of the neural networks varied wildly. This variance is two-fold: first some of the methods failed to converge at all. On the other hand, the performance depends heavily on which data is used for training, even much more so than is the case for the regression models. Curiously, which algorithm performed better also seemed to depend on which dataset is used for training: Gaussian regularization performs well on the Greater Gabbard site. On the other hand, using gradient descent whilst training on individual data points which are presented cyclically works better on the Westermost wind field. The best explanation found for this phenomenon is that the networks that may tend to over-fit on the Greater Gabbard site actually provide better accuracy here, since both vessels are identical and only a limited amount of data is available. Alternatively, it may be that different methods are better for learning different specifics for a transit, such as hull (type) or route, and that the method may therefore change per site and vessel. The first hypothesis can be tested by using a larger dataset, and see if the methods that work well on the Greater Gabbard site start performing better. The second hypothesis can be verified by testing on more sites whilst using different vessel types.

10.7.1 Choice of model

A wide variety of algorithms for neural networks is available. To limit ourselves to the best candidates, some initial testing was performed with all available algorithms. The comparison is made by training each network 5 times, and compare the results. The results from this testing are provided in Table 10.3. From this, we note that the best performing algorithm is Bayesian regularization with backpropagation. As this is, however, a rather small sample size, it can very well be that some of the other algorithms may perform better. However, as training for the Bayesian regularization already takes a significant amount of time, no other algorithms are considered further.

Table 10.3: Average mean squared error over the test data for the most common neural network trainers. Methods that failed to converge have been omitted.

Name algorithm	Average error	Smallest error
Levenberg-Marquardt backpropagation	14.05	11.30
BFGS quasi-Newton backpropagation	16.57	9.50
Conjugate gradient backpropagation with Fletcher-Reeves updates	12.26	7.37
Conjugate gradient backpropagation with Polak-Ribiére updates	11.95	9.58
Gradient descent backpropagation	512.46	122.15
Gradient descent with adaptive learning rate backpropagation	413.79	20.24
Gradient descent with momentum backpropagation	3562.31	160.17
Grad. desc. with momentum and adaptive learning rate backprop.	740.92	9.71
One-step secant backpropagation	11.49	8.82
Resilient backpropagation	12.01	9.17
Scaled conjugate gradient backpropagation	12.35	9.40
Batch training with weight and bias learning rules	146.51	32.19
Bayesian regularization backpropagation	7.75	5.68
Random order incremental training with learning functions	149.10	25.95
Sequential order incremental training with learning functions	32.31	24.12

10.7.2 Impact number of neurons and predictor choice

As we saw in the results from machine learning, the choice of predictor can have significant impact. There we noticed that the removal of the predicted wave spectrum could improve the MSI prediction. Also, the number neurons used is not yet discussed. Thus, for a varying set of neurons the average MSE prediction error has been computed over 100 trained networks. The error distribution is plotted in Figure 10.13, with the raw errors in Table 10.4. From this we quickly deduce that the inclusion of a second training layer is not beneficial at all. Also, omitting the prediction error only reduces the performance, although requiring significantly less computing time (not shown here). In terms of the optimal choice with respect to the number of neurons in the first and only layer, 10 seems to be the optimum, and this number is therefore used as the standard for the remainder of this work.

Table 10.4: Average errors for the various choices of neurons, after training the Bayesian regularization model 100 times. The indices 1 or 2 indicate whether the predicted wave spectrum was present or omitted respectively.

Network size	4_1	4_2	$[4,4]_1$	$[4,4]_2$	$[6,6]_1$	6_1	7_1	10_1,	10_2	15_1
Average MSE error	9.03	12.16	9.76	9.92	8.80	8.59	8.33	8.06	8.19	8.47
rms MSE error	9.51	13.07	10.47	10.58	9.332	8.97	8.59	8.47	8.47	8.83

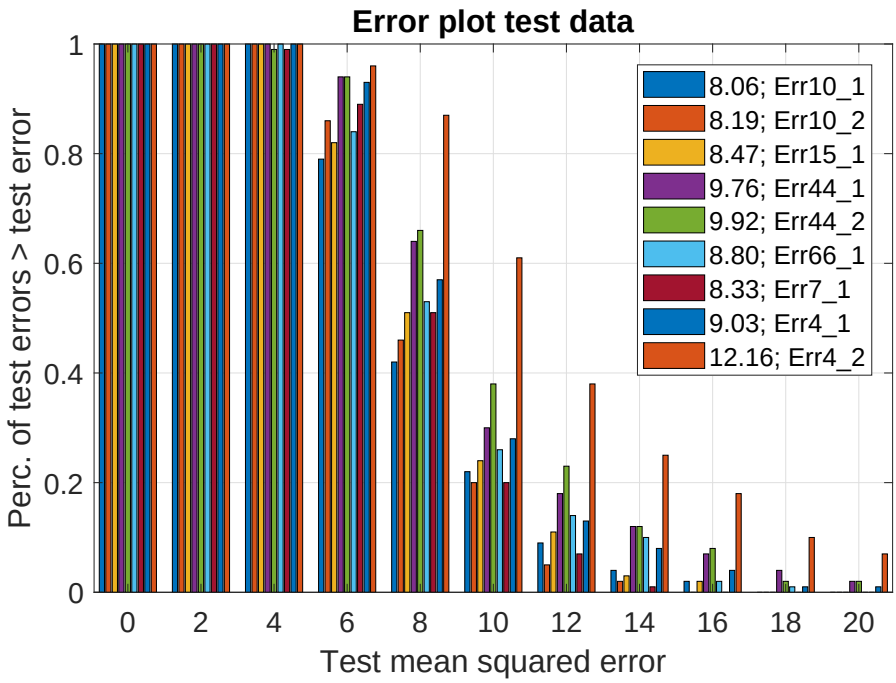


Figure 10.13: Cumulative error plot of the various choices for the number of neurons. The indices 1 or 2 indicate whether the predicted wave spectrum was present or omitted respectively.

10.8 Comparison neural networks Westermost

Like we have done with the machine learning models, the models constructed with the ANN's have been subjected to a comparison with the Westermost wind farm. And similar to the comparison with the machine learning models, the ANN's trained on the Greater Gabbard wind farm perform poorly when predicting MSI for the Westermost site. Also, when training the ANN's on the Westermost site, a good accuracy was achieved. But now a strange phenomenon is observed: when training the different types of neural networks, we get a significantly different picture than the one we got from the Greater Gabbard site. The results are given in Table 10.5.

Table 10.5: Average mean squared error on the Westermost for the same training methods used at the Greater Gabbard site.

Name algorithm	Average error	Smallest error
Levenberg-Marquardt backpropagation	5.00	1.50
BFGS quasi-Newton backpropagation	25.82	6.02
Conjugate gradient backpropagation with Fletcher-Reeves updates	7.06	4.07
Conjugate gradient backpropagation with Polak-Ribiére updates	17.14	2.99
Gradient descent backpropagation	172.34	62.52
Gradient descent with adaptive learning rate backpropagation	12.45	4.73
Gradient descent with momentum backpropagation	299.88	30.48
Grad. desc. with momentum and adaptive learning rate backprop.	10.30	1.90
One-step secant backpropagation	10.81	2.35
Resilient backpropagation	11.15	4.62
Scaled conjugate gradient backpropagation	7.61	4.15
Batch training with weight and bias learning rules	102.26	20.15
Bayesian regularization backpropagation	4.73	2.63
Random order incremental training with learning functions	26.27	12.58
Sequential order incremental training with learning functions	8.67	1.19

10.9 Comparison various models

To give an overview of the relative performance of the best performing methods, we have given an overview in Figure 10.14 for the Greater Gabbard site and in Figure 10.15 for the Westermost site.

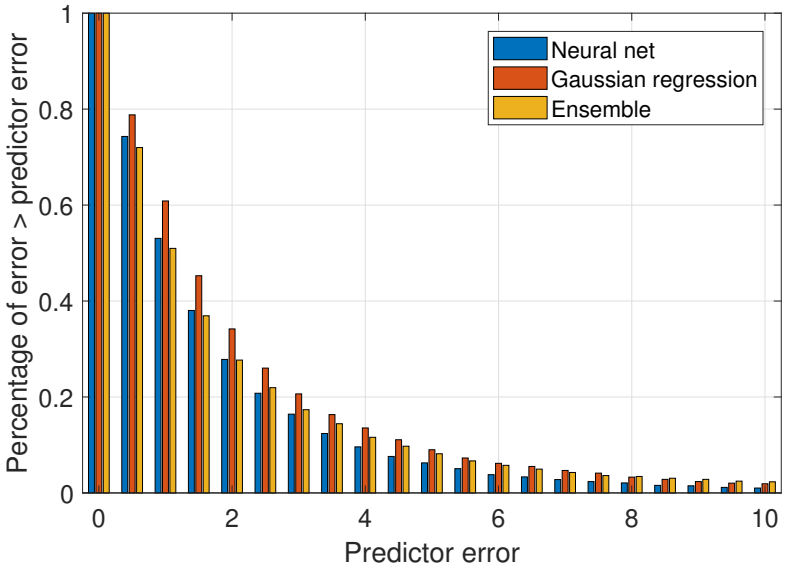
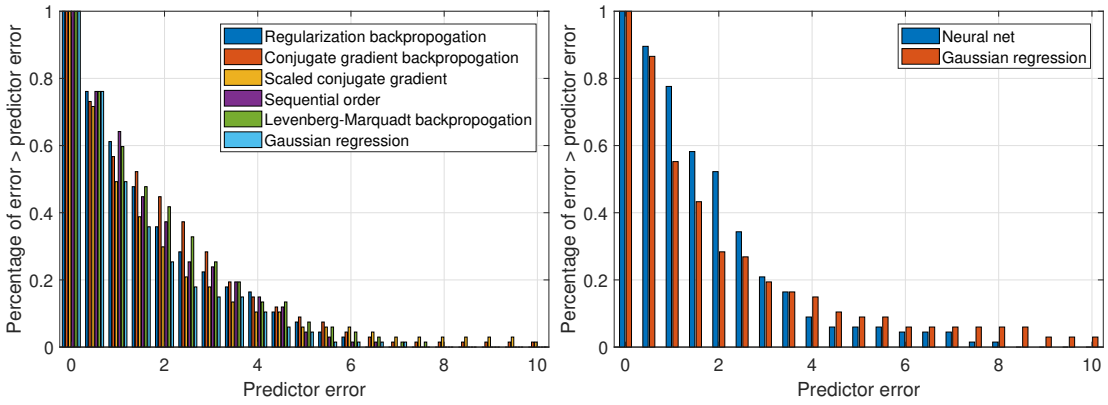


Figure 10.14: Cross-validation for the best performing predictors at the Greater Gabbard site. The tree ensemble does not use the predicted wave spectrum, the Gaussian regression and neural network do.



(a) (b)

Figure 10.15: Two cross-validations for the Westermost site. The overall performance of the neural net is slightly better.

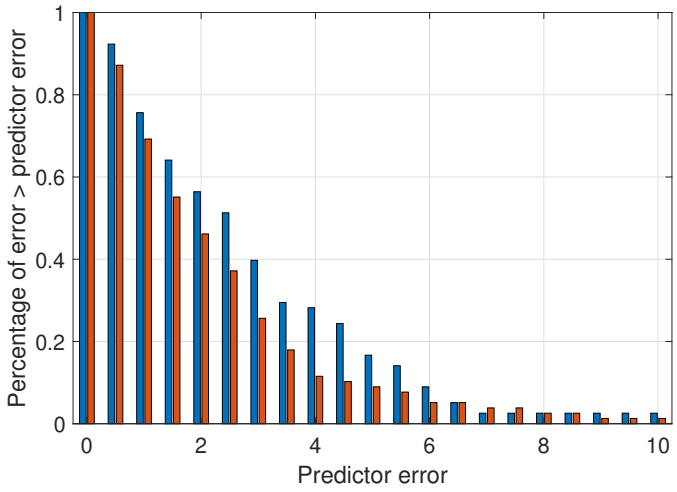
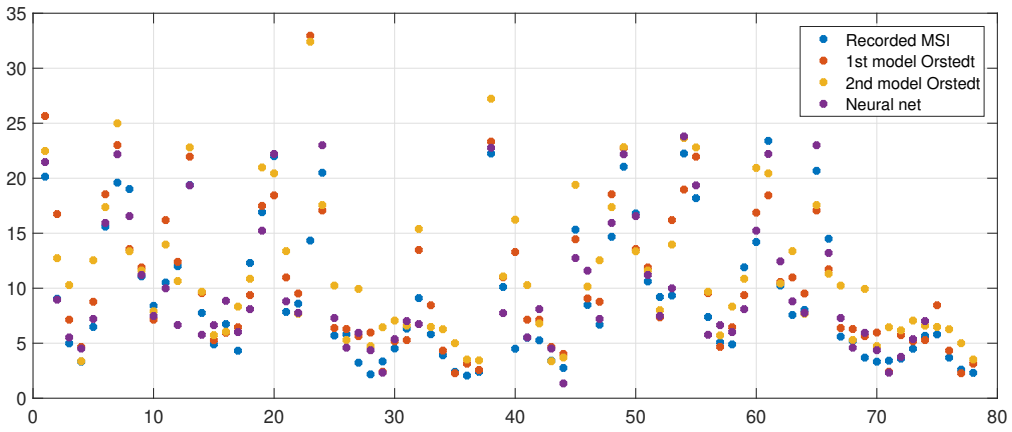


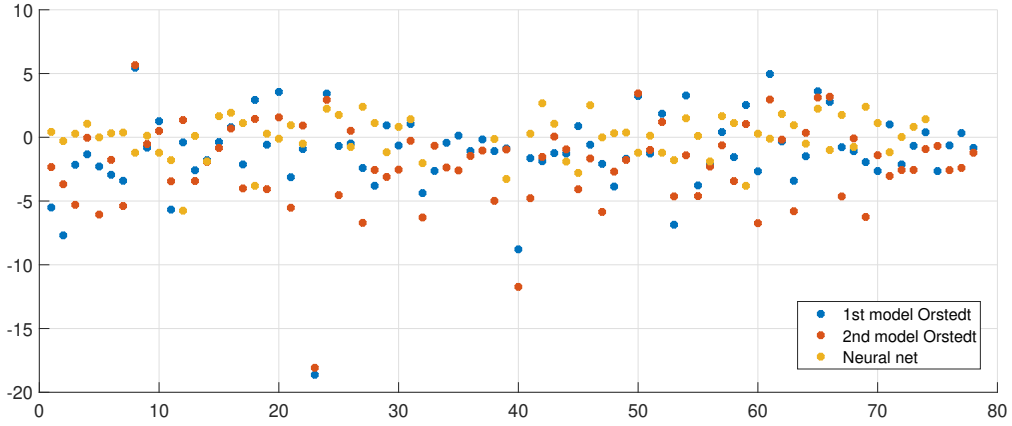
Figure 10.16: Cumulative error histogram of the prediction made by Ørsted.

10.10 Comparison Ørsted

Ørsted has been kind enough to share their prediction values for the Westermost wind farm. The data contains two separate predictions, with an MSE of 12.23 and 16.23 for the two predictions. They do not serve the Greater Gabbard wind field, so no comparison can be made on that field. A cumulative error histogram from their prediction errors has been given in Figure 10.16. This shows that the majority of the error is because they had a single extreme event where the vessel only recorded for 8 minutes due to an aborted transit. If we omit that day from the data, we find MSE of 7.82 and 12.28. This is still significantly larger than the cross-validation errors we found earlier for the Westermost wind farm. Figure 10.17



(a) Predicted values



(b) Errors

Figure 10.17: Comparison with the data provided by Ørsted. The neural net shown for comparison is the cross-validated neural network using Bayesian regularization backpropogation with optimal settings.

11 Conclusion and recommendations

Motion sickness is one of the critical factors in serving wind fields. A small increase in operational efficiency can significantly increase energy output. An objective measure for motion sickness called the motion sickness incidence was introduced, by relating MSI to the vessel motion along the vertical axis, and then show its relation to motion spectra. It is the goal of this thesis to develop a prototype capable of predicting MSI. This prototype needs to have sufficient accuracy to help wind farm operators better deploy their service vessels. The base mark for a basic linear regression model is an MSE of about 21, which is insufficient for prediction.

To do so, we first made an analytical analysis Section 5. We derived the motion spectrum that a vessel should experience during transit, by relating it to the sea spectrum measured by wave buoys. Unfortunately, it turns out this approach does not provide the desired accuracy. Therefore, we deem it impractical as a stand-alone method for MSI prediction.

The alternatives to the analytical approach that are presented are machine learning and, in particular, artificial neural networks. Three candidate regression models were investigated in detail, which showed that the best performing machine learning algorithm is the tree ensemble regression model. It makes an average error of 10.79 on the Greater Gabbard site, and does so without the predicted motion spectrum from the analytical derivation. Neural networks, however, perform significantly better than any of the regression models, with an MSE of 8.07. The best performing method for neural networks on Greater Gabbard is Bayesian regularization with backpropagation, with the alternative algorithms performing worse than the three highlighted regression models. The network chosen contains 1 hidden layer of 10 nodes, after a numerical performance analysis. Noteworthy is that for ANN's the inclusion of the predicted wave spectrum improves performance, regardless of the chosen method or network configuration.

A comparison with the Westermost wind field was made to see the dependency of the model on the site. This site is of interest, since it is comparable to the Greater Gabbard site, and for this site Ørsted also made MSI predictions. None of the models trained on the Greater Gabbard site performed adequately on the Westermost site, which means that the models trained on one site are not immediately applicable to other sites. We conclude that this is because the machine learning models also train the journey rather than just the relation between spectra and MSI. As this route differs per site, we would expect this to result in a different model for every site. This belief is strengthened by the observation that we we train the machine learners on the Westermost site, all the models perform even better than on the Greater Gabbard site. This is either because the transits are easier to predict, or because the data contains fewer errors. A final explanation of the difference in performance is that the vessel on this field are deployed more conservatively, which would result in fewer transits with extreme weather. The fact that only one vessel type is responsible in the smaller errors cannot be the reason, since reducing to single vessels at the Greater Gabbard site does not improve model performance. Neural networks using Bayesian regression once more performed the best

with an error of 4.62. On the Westermost site there was, however, a far wider variety of ANN's with good performance, with the extremes obtained by other models. Thus we conclude that for the Westermost site a variety of models can be used, but Bayesian regression is the most consistent. The regression models performed with an average error of 6.

Regardless of site, the MSE error can be brought down to be below 5 for 90% of the time. As it also performs better than the model of Ørsted, which has already proven itself, we can only conclude that the MSI prediction obtained is sufficiently accurate for deploying crew transfer vessels at both the Greater Gabbard and the Westermost site.

For future work we recommend to test on more sites, in order to reduce the dependency on training a new model for every single site. In general, more (good quality) data always improves the performance of machine learners, but more sites might make it possible to identify the hidden variable(s) behind MSI prediction. If such a variable can be found, an analytical approach similar to the one made in this thesis might be feasible. This could in turn enable more advanced techniques such as route optimization with respect to minimizing MSI.

References

[1] Francis J Anscombe. "Graphs in statistical analysis". In: *The American Statistician* 27.1 (1973), pp. 17–21.

[2] Willem Bles. "Coriolis effects and motion sickness modelling". In: *Brain research bulletin* 47.5 (1998), pp. 543–549.

[3] *Energieverbruik van particuliere huishoudens*. URL: https://www.cbs.nl/nl-nl/achtergrond/2018/14/energieverbruik-van-particuliere-huishoudens.

[4] F Dan Foresee and Martin T Hagan. "Gauss-Newton approximation to Bayesian learning". In: *Neural networks, 1997., international conference on*. Vol. 3. IEEE. 1997, pp. 1930–1935.

[5] *Gaussian Processes regression: basic introductory example*. URL: http://scikit-learn.org/stable/auto_examples/gaussian_process/plot_gpr_noisy_targets.html.

[6] Giuseppe Giorgi and John V. Ringwood. "Implementation of latching control in a numerical wave tank with regular waves". In: *Journal of Ocean Engineering and Marine Energy* 2.2 (2016), pp. 211–226. ISSN: 2198-6452. DOI: 10.1007/s40722-016-0052-8. URL: https://doi.org/10.1007/s40722-016-0052-8.

[7] *Global offshore wind farm map*. URL: https://www.4coffshore.com/offshorewind/.

[8] Michael J Griffin. *Handbook of human vibration*. Academic press, 1990.

[9] *How the backpropagation algorithm works*. URL: http://neuralnetworksanddeeplearning.com/chap2.html#proof_of_the_four_fundamental_equations_(optional).

[10] Aapo Hyvarinen. "The fixed-point algorithm and maximum likelihood estimation for independent component analysis". In: *Neural Processing Letters* 10.1 (1999), pp. 1–5.

[11] *List of offshore wind farms*. URL: https://www.4coffshore.com/windfarms/norfolk-boreas-united-kingdom-uk69.html.

[12] David JC MacKay. "Bayesian interpolation". In: *Neural computation* 4.3 (1992), pp. 415–447.

[13] Hisashi Mitsuyasu et al. "Observations of the directional spectrum of ocean waves using a cloverleaf buoy". In: *Journal of Physical Oceanography* 5.4 (1975), pp. 750–760.

[14] Marc-Antoine Parseval. "Mémoire sur les séries et sur l'intégration complète d'une équation aux différences partielles linéaires du second ordre, à coefficients constants". In: *Mém. prés. par divers savants, Acad. des Sciences, Paris,(1)* 1 (1806), pp. 638–648.

[15] Willard J Pierson Jr, Gerhard Neumann, and Richard W James. *Practical methods for observing and forecasting ocean waves by means of wave spectra and statistics*. Tech. rep. NAVAL OCEANOGRAPHIC OFFICE NSTL STATION MS, 1971.

[16] Carl Edward Rasmussen. “Gaussian processes in machine learning”. In: *Advanced lectures on machine learning*. Springer, 2004, pp. 63–71.

[17] Jonathon Shlens. “A tutorial on independent component analysis”. In: *arXiv preprint arXiv:1404.2986* (2014).

[18] Jonathon Shlens. “A tutorial on principal component analysis”. In: *arXiv preprint arXiv:1404.1100* (2014).

[19] International Organization for Standardization. *Mechanical vibration and shock-Evaluation of human exposure to whole-body vibration-Part 1: General requirements*. The Organization, 1997.

[20] *The Great Ptolemaic Smackdown: Down for the Count*. URL: `http://tofspot.blogspot.com/2013/08/the-great-ptolemaic-smackdown-down-for.html`.

[21] *Tikz example neural network*. URL: `http://www.texample.net/tikz/examples/neural-network/`.

[22] Andreas Parama Wijaya, P Naaijen, E van Groesen, et al. “Reconstruction and future prediction of the sea surface from radar observations”. In: *Ocean engineering* 106 (2015), pp. 261–270.

[23] Ya-xiang Yuan. “Step-sizes for the gradient method”. In: (June 2018).

A Mathematical proofs

Below a selection of the theorems used in this thesis are proven.

A.1 Parseval

Theorem 3 (Parseval)**.** *Let* $\mathbf{X} := \mathcal{F}(\mathbf{x})$ *denote the discrete Fourier transform of a signal* $\mathbf{x} \in \mathbb{C}^N$*. Then*

$$\sum_{n=0}^{N-1} |\mathbf{x}_n|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |\mathbf{X}_k|^2 \tag{A.1}$$

Proof. By definition of $\mathbf{X}$ we get

$$X_k = \sum_{n=0}^{N-1} \mathbf{x}_n W_N^{nk} \tag{A.2}$$

Note that $\mathbf{X} \subseteq \mathbb{C}^N$, and thus $|\mathbf{X}_k|^2 = \mathbf{X}_k \mathbf{X}_k^*$, with $\mathbf{X}_k^*$ denoting the complex conjugate of $\mathbf{X}_k$. When we put this into the right-hand side of Eq. (A.1) we get

$$\frac{1}{N} \sum_{k=0}^{N-1} |\mathbf{X}_k|^2 = \frac{1}{N} \sum_{k=0}^{N-1} \left| \sum_{n=0}^{N-1} \mathbf{x}_n W_N^{nk} \right|^2 \tag{A.3}$$

$$= \frac{1}{N} \sum_{k=0}^{N-1} \sum_{n=0}^{N-1} \mathbf{x}_n W_N^{nk} \sum_{m=0}^{N-1} \mathbf{x}_m^* W_N^{-mk} \tag{A.4}$$

$$= \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \mathbf{x}_n \mathbf{x}_m^* \sum_{k=0}^{N-1} W_N^{(n-m)k} \tag{A.5}$$

Recall that W_N is a complex N'th root of unity, which means that summing over all such m gives

$$\sum_{k=0}^{N-1} W_N^{(n-m)k} = \sum_{k=0}^{N-1} e^{2\pi i (n-m)k/N} = \frac{e^{2\pi i (n-m)} - 1}{e^{2\pi i (n-m)/N} - 1} \tag{A.6}$$

The last step follows since it is the partial power series of W_N^{n-m}, which is valid whenever $W_N^{n-m} \neq 1$, or in other words, if $n \neq m$ (since $||W_N|| = 1$). Since it also holds that $n, m \in \mathbb{Z}$, this last expression evaluates to zero. However, if $n = m$, all terms in this sum are 1 and we find that this sum equals N. This implies that

$$\sum_{k=0}^{N-1} W_N^{(n-m)k} = N 1_{n=m} \tag{A.7}$$

and this implies that

$$\frac{1}{N}\sum_{k=0}^{N-1}|\mathbf{X}_k|^2 = \frac{1}{N}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}\mathbf{x}_n\mathbf{x}_m^* N 1_{n=m} \tag{A.8}$$

$$= \sum_{n=0}^{N-1}\mathbf{x}_n\mathbf{x}_n^* \tag{A.9}$$

$$= \sum_{n=0}^{N-1}|\mathbf{x}_n|^2 \tag{A.10}$$

□

A.2 Singular value decomposition

Theorem 4 (Singular value decomposition)**.** *Let A be a real $m \times n$ matrix. Then there exists a singular value decomposition $A = U\Sigma V^\mathsf{T}$ such that*

1. *$U \in \mathbb{R}^{m\times m}$ is an orthogonal matrix.*
2. *$\Sigma \in \mathbb{R}^{m\times n}$ a rectangular diagonal matrix with only non-negative (real) entries on the diagonal.*
3. *$V \in \mathbb{R}^{n\times n}$ an orthogonal matrix.*

Proof. Note that the matrix $A^\mathsf{T}A$ is real symmetric and therefore has an eigenvalue decomposition. For each positive eigenvalue λ_i and corresponding eigenvector v_i we define

$$q_i := \frac{Av_i}{\sqrt{\lambda_i}}$$

Note that

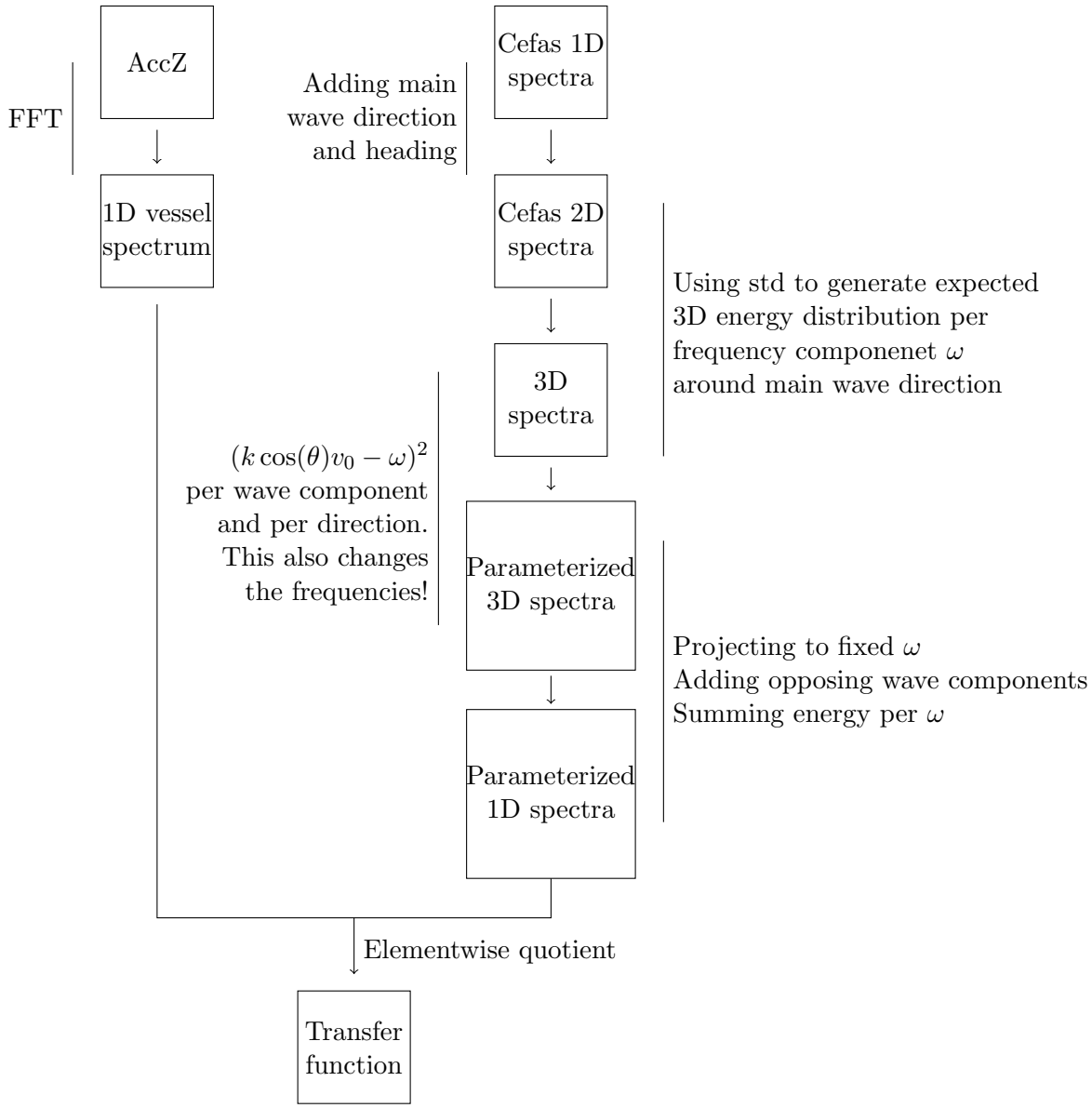
$$q_i^\mathsf{T}q_j = \frac{1}{\lambda_i}v_i^\mathsf{T}A^\mathsf{T}Av_j = \frac{1}{\lambda_i}v_i^\mathsf{T}\lambda_j v_j = \delta_{ij}, \tag{A.11}$$

and hence we can extend (q_i) to a orthonormal basis for $\mathbb{R}^m$. This can always be done since there can be at most $\min(m,n)$ non-zero eigenvalues. Define $U := [q_i]$ and $V := [v_i]$. Then U and V are orthogonal matrices, as all their columns form an orthonormal basis. If we now evaluate $U^\mathsf{T}AV$ we find that

$$(U^\mathsf{T}AV)_{ij} = q_i^\mathsf{T}Av_j = \sqrt{\lambda_j}q_iq_j = \sqrt{\lambda_j}\delta_{ij} \tag{A.12}$$

Thus let the Σ be given by $\Sigma_{ij} := \sqrt{\lambda_j}\delta_{ij}$. Then $\Sigma = U^\mathsf{T}AV$. Clearly Σ is a rectangular matrix with only non-negative elements on its diagonal. Since U and V are orthogonal (and hence invertible), this last equation can now be rewritten in the desired form $A = U\Sigma V^\mathsf{T}$. □

B Spectral processing scheme



Assume that the transfer function T is computed via the scheme above. Then the MSI can be computed as

$$MSI = msi(TS), \tag{B.1}$$

where TS denotes the element-wise product and

msi = motion sickness function for vessel spectra (known)

S = Parameterized 1D wave spectra computed using cefas buoy data via the scheme above.

C Symbols and abbreviations

Variable	Full name	First defined
MSI	Motion sickness incidence	2
FFT	Fast Fourier transform	5
SR	Sampling rate	6
VI	Vomiting incidence	7
RAO	Response amplitude operator	17
ICA	Independent component analysis	33
PCA	Principle component analysis	33
SVD	Singular value decomposition	37
SVM	Support vector machine	41
MSE	Mean squared error	45
ANN	Artificial neural network	48
η	Sea surface elevation	5
$S(f), S(\omega)$	Energy density spectrum	5
f	Wave frequency in Hz	5
ω	Wave frequency in radians	5
H_s	Significant wave height	10
$E[\cdot]$	Expected value from distribution	34