Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Emil Johannesen Haugstvedt

On the Potential of Utilizing Laboratory-Scale Experimental Setup as Proxy For Real-Life Applications: Time Series Analysis and Prediction for Hole Cleaning

Master's thesis in Cybernetics and Robotics
Supervisor: Professor Adil Rasheed
June 2023

NTNU
Norwegian University of
Science and Technology

Emil Johannesen Haugstvedt

On the Potential of Utilizing Laboratory-Scale Experimental Setup as Proxy For Real-Life Applications: Time Series Analysis and Prediction for Hole Cleaning

Master's thesis in Cybernetics and Robotics
Supervisor: Professor Adil Rasheed
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Abstract

Hole cleaning is a critical process in oil drilling that involves removing the cuttings from the wellbore using a drilling fluid. The performance of hole cleaning can be measured by the equivalent circulating density (ECD), which is affected by various factors and control parameters. Predicting ECD accurately, can help to optimize hole cleaning and prevent problems such as mechanical pipe sticking, premature bit wear, slow drilling rate, formation fracturing, and loss of circulation, all which may lead to increased non-productive time. However, collecting and processing real oil drilling data is costly and cleaning the data is time consuming. This thesis propose to use cheap lab scale data from a greenhouse as a proxy for oil drilling data, and to evaluate and compare different time series analysis and forecasting methods using this cheap laboratory-scale data. Principal component analysis (PCA), empirical mode decomposition (EMD), ensemble empirical mode decomposition (EEMD) and fast Fourier transform (FFT) are used to analyse the dynamics of the greenhouse data, and PCA, EMD and EEMD are used to analyse the dynamics of the oil drilling data. Feed-forward neural network (FFNN) and long short-term memory (LSTM) based neural network are used to predict temperature and humidity for greenhouse data and the best performing model are used for ECD prediction in oil drilling. The results shows that the greenhouse data can serve as a proxy for the oil drilling data, and that the methods and techniques that provides useful insights and accurate predictions in the greenhouse data, also performs good on the drilling data. Lastly, directions of future work are suggested, such as creating laboratory-scale setups that better mimics the dynamics of oil drilling and to investigate LSTM with continuous learning to make models that better generalizes on different pattern in the data.

Sammendrag

Hullrensing er en viktig del av oljeboring. Hullresning innebærer å fjerne borekaks fra brønnen ved hjelp av en borevæske. Ytelsen til hullresingsprosessen kan måles ved sirkulasjonstrykket (ECD), som påvirkes av forskjellige faktorer og kontrollparametre. Å predikere ECD nøyaktig kan bidra til å optimalisere hullrensing og unngå problemer, som at borerørene setter seg fast, for tidlig slitasje på boret, lav borehastighet, formasjonsbrudd og tap av sirkulasjon, som alle kan føre til økt nedetid. Men, å samle inn og prosessere data fra oljeboring er dyrt og datarensing er tidkrevende. Denne oppgaven foreslår å bruke billig labskala data fra et drivhus som erstatning for data fra oljeboring og å evaluere og sammenligne forskjellige tidsserieanalysemetoder og -teknikker for analyse og prediksjon ved å bruke denne billige dataen. Hovedkomponentanalyse (PCA), empirisk modusdekomponering (EMD), ensemble empirisk modusdekomponering (EEMD) og rask Fourier-transformasjon (FFT) blir brukt til å analysere dynamikken i drivhusdataen, og PCA blir brukt for å analysere dynamikken i oljeboredataen. Et fremovermatet nevrealt nettverk (FFNN) og to langt korttidsminne-nettverksbaserte nettverk predikerer temperaturen og luftfuktigheten i drivhuset og det nettverket med best prediksjon blir brukt til å predikere ECD i oljeboredataen. Resultatene viser at drivhusdataen can brukes som en erstatning for boredata og at metodene og teknikkene som fungerer bra på drivhusdataen og fungerer bra på oljeboringdataen. Til slutt foreslås retninger for fremtidig arbeid som å lage en labskala modell som genererer data som bedre representerer dynamikken i oljedrillingdataen. Å bruke kontinuerlig læring sammen med LSTM blir også foreslått som en metode som kan bidra til modeller som generaliserer bedre og dermed håndterer forskjellige datamønstre bedre.

Preface

This thesis concludes a five-year integrated Master of Science degree in Cybernetics and Robotics at the Norwegian University of Science and Technology (NTNU). It shows how a laboratory-scale greenhouse can be used to test and validate methods for time series analysis and prediction and how these model performs on more complex oil data. Consequently, suggesting how cheap data is valuable for validation of analysis and prediction methods.

I want to thank SINTEF, and especially Senior Research Scientist Philippe Nivlet, for including me in his research group "KPN Hole Cleaning Monitoring in drilling with distributed sensors and hybrid methods". I was included in their weekly meetings, where I presented my work along the way and got useful feedback. I also got to attend technical meetings and a workshop with industry partners, Aker BP, Vår Energi, and TDE, were I presented my work. This was valuable experience. A special thanks to Aker BP for providing me with oil drilling data.

I also want to thank Phd candidate Mehmet Cagri Altindal, who has a background as an oil drilling operator, for his useful introduction, discussions, and feedback on the oil drilling related theory and topics.

Lastly, I want to thank my supervisor, Professor Adil Rasheed, for his guidance and heavy involvement in my thesis, with almost daily meetings in some periods. I also want to thank him for including me in the SINTEF group and giving me the possibility of presenting my work for actual researchers and industry partners. As a final note, I want to thank him for introducing me to time series analysis in the context of dynamical systems and cybernetics in a course two years ago.

I also want to mention that parts of the theory section about neural networks and how they can be used for prediction on time series data are heavily inspired from my own specialization project: "Improving the Accuracy and Robustness of Neural Network Simulations of Dynamical Systems by Inducing Sparsity".

Trondheim, June 2023
Emil Johannesen Haugstvedt

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

Abbreviations

FFNN	Fully-Forward Neural Network
LSTM	Long Short-Term Memory
ECD	Equivalent Circulation Density
WOB	Weight On Bit
RPM	Revolutions Per Minute
FLI	Flowrate In
DMI	Density Mud In
ROP	Rate Of Penetration
RMSE	Root Mean Square Error
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
NPT	Non-Productive Time

1. Introduction

1.1. Background and Motivation

Drilling oil wells is a vital activity for the global energy supply, as it is essential for extraction of oil from underground reservoirs. Oil drilling involves drilling a drill bore into the soil, and pumping drilling fluid through the drill string to lubricate and cool down the bit, remove drill cuttings, and maintain the well bore stability. Oil drilling is a complex, costly and risky operation that requires careful planning, design, execution and monitoring [33].

One of the main challenges when drilling is hole cleaning. This refers to the ability of the drilling fluid to transport and suspend the drilled cuttings from the well bore to the surface. Bad hole cleaning can lead to several expensive problems, such as: mechanical pipe sticking, premature bit wear, slow drilling rate, formation fracturing, and loss of circulation, all of which may lead to increased nonproductive time (NPT) [1]. Cleaning is especially challenging in horizontal and deviated wells, where the cuttings tend to accumulate at the low side of the hole due to the gravity [13].

The hole cleaning performance depends on various factors, such as flow rate, drilling fluid density, cuttings size, and rate of penetration (ROP) [5]. During drilling operations, data is collected from different sensors, both at the platform and along the drill string. These measurements collected at regular time steps are called time series data. To optimize the hole cleaning performance and avoid problems during drilling, it is essential to understand this data and how different factors interact and influence the cuttings transport. Understanding this requires building and applying suitable time series analysis and forecasting techniques and models for analysis and prediction of the data.

Time series analysis and forecasting techniques are a set of tools for extracting meaningful information and making predictions from temporal data. Time series analysis involves developing models to understand underlying patterns in the data to gain deeper insight into the data, while time series forecasting uses these models, and knowledge from these models, to project future values based on historical data. Time series analysis and forecasting have many applications in various domains, such as resource management, traffic flow forecasting, weather forecasting, and disease transmission forecasting [69].

One of the challenges with time series analysis and forecasting is the choice of appropriate models and techniques on different data and problems. There are many methods and techniques for time series analysis and forecasting, such as Principal Component Analysis (PCA), Empirical Mode Decomposition (EMD), Ensemble Empirical Mode Decomposition (EEMD), fast Fourier Transform (FFT), time series regression, time series decomposition, exponential smoothing, ARIMA models, Feed-Forward Neural Networks (FFNN), Long Short-Term Memory (LSTM) based neural networks and others [40, 52]. These methods and techniques have strengths and weaknesses, and their performance will vary for data sets and scenarios. Therefore, to get good results when analysing and predicting on time series data, it is essential to evaluate and compare the performance of the methods and techniques.

Developing and testing such a model on data from real oil drilling is expensive and

impractical, as it involves accessing confidential data from oil companies, dealing with noisy and incomplete data, and facing ethical and environmental issues. Therefore, alternative data sources are needed that can mimic the characteristics of oil drilling data and provide cheap and convenient ways to evaluate and compare different models and techniques for hole cleaning analysis and prediction. Some examples of alternative data sources are miniature models of fluids through pipes, using a small boat in a wave pool to mimic real wave interactions and data simulations of physical processes.

This thesis proposes a novel approach of evaluating and comparing different analysis and forecasting methods using cheap data from a laboratory-scale greenhouse. The greenhouse data consists of measurements of temperature, humidity, light intensity, carbon dioxide concentration, and moisture at regular intervals over several months. The greenhouse also has the possibility of manipulating the measurements with a heater, inlet fan, outlet fan, lighting and a watering system. The greenhouse data is used as a proxy for more expensive oil drilling data, which has similar characteristics of high dimensionality, nonlinearity, nonstationarity, noise, and seasonality. Various methods and techniques for analysis and prediction are applied to the greenhouse data and the results are then validated by application on oil drilling data. The goal is to identify the best methods for analysing the oil drilling data using the greenhouse data. As the data is measurements of dynamical system processes, there is a special focus on how the models generalize on measurements from similar process that are exhibiting other patterns.

1.2. Contribution, Research Objectives and Research Questions

1.2.1. Contribution

This thesis makes a novel contribution to the field of time series analysis and forecasting by showing how cheap data can be used as a proxy for expensive data for evaluating and comparing different time series analysis and forecasting methods. It shows how having a laboratory-scale setup, with labeled measurements and the possibility for manipulation of internal states using controls, can generate data of such complexity that it can serve as a proxy for a complex field scale setup.

1.2.2. Objectives

The main research goal of this thesis is to show how it is possible to evaluate and compare different time series analysis and forecasting methods applied for hole cleaning in oil drilling using cheap data from a laboratory-scale greenhouse.

Primary Objective: The primary objective is to show how cheap data from a laboratory-scale greenhouse can be used to develop, test and validate different time series methods and techniques for analysis and forecasting on hole cleaning data from oil drilling.

Secondary Objectives: In addition to the primary objective five secondary objectives are formulated:

- To collect and pre-process greenhouse data and oil drilling data for time series analysis and forecasting.
- To apply various time series analysis methods, such as PCA, EMD, EEMD and FFT to the greenhouse data and the oil drilling data to get a better understanding of the underlying dynamics.

- To identify the best methods for analysing and predicting hole cleaning performance in oil drilling using the greenhouse data as a proxy.
- To apply time series forecasting methods based on FFNN and LSTM based neural network on the greenhouse data, and further apply the best ones on oil drilling data.
- Investigate how prediction models performs on new measurements from the systems, exhibiting other patterns than the ones in the training data.

1.2.3. Research Questions

The main research question for this thesis is:

- How can time series analysis and forecasting methods be evaluated and compared using cheap data from a laboratory-scale greenhouse as a proxy for expensive oil drilling data?

Other research questions are:

- How to pre-process data from greenhouse and oil drilling to make it suitable for analysis and prediction?
- How can time series analysis methods, such as PCA, EMD, EEMD, and FFT, be used to understand the underlying dynamics of the greenhouse data and oil drilling data?
- How can knowledge about performance of time series prediction models on laboratory-scale greenhouse data be transferred to field scale oil drilling data?

1.3. Structure of the Thesis

The thesis is organized as follows: Chapter 2 provides an overview of methods and techniques used. Chapter 3 describes the models and datasets used and how the experiments are conducted. Chapter 4 presents the results of applying the techniques and methods the the greenhouse data and discusses their performance. It also presents how some of the methods performing best on the greenhouse data, performs on the drilling data and discusses how they perform. Lastly, Chapter 5 concludes the thesis and suggests directions for future work.

2. Theory

The theory section introduces and discusses the theoretical perspectives of the study. The study includes various methods and techniques for time series analysis and forecasting, such as Principal Component Analysis (PCA), Empirical Mode Decomposition (EMD), Ensemble Empirical Mode Decomposition (EEMD), and fast Fourier Transform (FFT) for analysis, and Feed-Forward Neural Networks (FFNN), and Long Short-Term Memory (LSTM) based neural networks for prediction. These methods and techniques are all presented in this section.

The chapter is organized as follows: First, it presents the concept of time series analysis and how PCA, EMD, EEMD and FFT can be applied in this context. Second, it gives an introduction to neural networks. Third, it shows how neural networks can be used for prediction in the context of time series analysis. Finally, it gives an introduction to laboratory-scale greenhouse setup and the hole cleaning operation within oil drilling. It introduces the concepts and presents the dynamics of the different systems and explains their similarities.

2.1. Time Series Analysis

Time series analysis is a range of methods used to analyse and extract trends from time-dependent data, such as: temperature measurements, number of airline passengers and car sales. It is all about understanding the patterns in the data and the underlying causes of the past events. This knowledge can then be applied to improve our understanding of the systems producing the data and predict how this system will act in the future. These techniques ranges from simple statistical methods like autocorrelation [65], to more complex deep learning based methods [18].

A subset of these techniques does what is called a decomposition of the time series. The idea of time series decomposition is old and was used to calculate planetary orbits as early as in the 17-th century [7], but was first explicitly stated by Persons [43]. Decomposition of time series is to separate the time series into a set of non-observable (latent) components that can be associated to different temporal components [7]. In other words, time series decomposition is to extract components that are hidden in the time series data such as seasonal patterns or dominating frequencies.

Traditionally, Persons [43] looked at decomposition from an economic stand point. He thought of the decomposition as separating the time series into a long-term trend, a more frequent cyclical movements superimposed on top of the long-term trend, a seasonal trend and residual variations.

$$\mathbf{X}_t = \mathbf{T}_t + \mathbf{C}_t + \mathbf{S}_t + \mathbf{I}_t. \tag{2.1}$$

In the above equation, the time series data $\mathbf{X}_t$ is described as a summation of the trend, $\mathbf{T}_t$, a cyclical component $\mathbf{C}_t$, the seasonal trend, $\mathbf{S}_t$, and the irregular residual variations, $\mathbf{I}_t$. But the components do not need to be added, they can also be a product of each other:

$$\mathbf{X}_t = \mathbf{T}_t\mathbf{C}_t\mathbf{S}_t\mathbf{I}_t, \tag{2.2}$$

which is called a multiplicative model. A multiplicative model is more suitable when the components are dependent and interact with each other, or when the amplitude of the seasonal or cyclical variations changes over time.

Of course, the components of the decomposed time series can consist of many different component of different frequencies that do not necessarily match the component names presented in equation (2.1) and equation (2.2). For example in engineering, the time series data often describes dynamical systems that has different components ranging over several frequencies. But the point holds, the goal is to separate the data into different components, where some of them (probably most of them) contains noise, or irregular residuals, while some of them contain useful information about the data.

There exists a wide range of different decomposition techniques. In this thesis the focus is on four of the more common ones, Principal Component Analysis (PCA), Empirical Mode Decomposition (EMD), and Fast Fourier Transform (FFT) [30, 63]. The goal of all different techniques is to get a deeper insight into the data and find patterns and relationships that are useful for understanding the data properly.

2.1.1. Principal Component Analysis (PCA)

Introduced independently by Pearson [17] and Hotelling [23] in 1901 and 1933, respectively, Principal Component Analysis (PCA) is a method for reducing the dimensionality of data. PCA finds a set of uncorrelated variables, Principal Components (PCs), such that the first PC captures more of the variance in the data than the second, and so on. Mathematically, PCA can be formulated by letting $\mathbf{X}$ be a real valued matrix of dimension $n \times p$, where the n number of rows represents the number observations and the p number of columns corresponds to measurements of those observations. Each column in $\mathbf{X}$ are normalized such that the matrix has a column-wise empirical mean of zero. The PCs of $\mathbf{X}$, $\mathbf{Z}$, are then calculated via a linear combination

$$\mathbf{Z} = \mathbf{X}\mathbf{A}, \tag{2.3}$$

where $\mathbf{Z}$ is a matrix containing each principal component $\mathbf{z}_i$ such that $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_p] \in \mathbb{R}^{n \times p}$ where the variance captured of each components is decreasing with p. $\mathbf{X} \in \mathbb{R}^{n \times p}$ is the matrix with observations and measurements and $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_p]$ is an orthonormal matrix that rotates the data into a new space. The columns of $\mathbf{A}$ are often called modes or loadings.

The optimization problem of finding the $\mathbf{A}$ that maximize the variance captured by each component, in decreasing order with p, can be formulated as a least-square problem i.e. by minimzing the sum of squared residual error between the input data and the projected data

$$\begin{aligned} \min_{\mathbf{A}} \quad & ||\mathbf{X} - \mathbf{X}\mathbf{A}\mathbf{A}^T||_{\mathrm{F}}^2 \\ \text{s.t.} \quad & \mathbf{A}^T\mathbf{A} = \mathbf{I}. \end{aligned} \tag{2.4}$$

Here $||\cdot||_{\mathrm{F}}^2$ is the squared Frobenius norm. Here PCA differs from normal least-squares by imposing orthogonality constraints on $\mathbf{A}$.

To solve the optimization problem in equation (2.4) one can utilize the Singular Value Decomposition (SVD). SVD is a decomposition methods that decompose the input data matrix $\mathbf{X}$ into

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2.5}$$

where it turns out that the minimizer of equation (2.4) is given by the right singular vectors, $\mathbf{v}$. Thus, setting $\mathbf{A} = \mathbf{v}$ yields the PCA loading matrix. Further, SVD also

provides the PCs as the scaled left singular vectors $\mathbf{Z} = \mathbf{U}\mathbf{\Sigma}$, where the diagonal entries of $\mathbf{\Sigma}$ are the singular values.

The way PCA can be used to reduce the dimensionality of the input data matrix $\mathbf{X}$ is by removing all but the first k dominant PCs. As the PCs accounts for a smaller and smaller amount of the variance with increasing p, the first $k << p$ component will in many cases account for a significant amount of the variance of the input data. This results in a reduced matrix of dimensionality $n \times k$. The optimal value of k depends on the desired level of accuracy and complexity of the analysis. A common criterion for choosing k is to retain a certain percentage of the total variance in the data, such as 95% or 99%. However, choosing a smaller value of k may also result in losing some important information or features in the data.

PCA can also be used to find correlations in the data by analyzing the loadings or coefficients of the principal components [70]. The loadings indicate how much each variable contributes to each principal component, and they can be interpreted as the correlations between the variables and the principal components. By examining the loadings, one can identify which variables are strongly or weakly correlated with each other, and which variables are responsible for most of the variance in the data. For example, if two variables have high loadings on the same principal component, it means that they are positively correlated with each other and with that principal component. Conversely, if two variables have opposite signs of loadings on the same principal component, it means that they are negatively correlated with each other and with that principal component. Furthermore, if a variable has a low loading on all principal components, it means that it is weakly correlated with any other variable and with the overall structure of the data.

2.1.2. Fast Fourier Transform (FFT)

The Fourier transform, or Fourier analysis, is a decomposition technique which decompose a signal into different frequency components [10]. The Fourier analysis is to extract sinusoidal components that sums up to form the original signal or time series. By doing so, the Fourier analysis can reveal the periodic patterns or cycles in the data, as well as the amplitude and phase of each frequency component. The Fourier analysis can also be used to filter out noise or unwanted frequencies from the data, or to compress the data by retaining only the most significant frequencies [45, 68]. The Fourier analysis is widely used in various fields of science and engineering, such as signal processing, image processing, data compression, spectral analysis, and time series analysis.

There are different types of Fourier transforms for different types of signals and situations. The most basic one is the continuous Fourier transform, which applies to continuous signals in the time domain. It produces a continuous signal in the frequency domain, which shows how much each frequency contributes to the original signal.

The continuous Fourier transform is defined as

$$f(v) = \mathcal{F}_t[f(t)](v) = \int_{-\infty}^{\infty} f(t)e^{-2\pi ivt}dt \tag{2.6}$$

where $f(v)$ is the frequency domain representation of the signal as a function of frequency, v. $\mathcal{F}_t$ is the Fourier operator transforming the time domain signal, $f(t)$, from time domain to frequency domain and i is the imaginary unit, $i = \sqrt{-1}$. Noting that

$$Ae^{2\pi ivt} = A\cos(2\pi vt) + iA\sin(2\pi vt),$$

the connection to sinusoidal components is clear.

But, as we are working on computers and our time series data are discrete, the discrete Fourier transform (DFT) is most useful. Consider a generalization of a discrete signal $f(t) \to f(t_k)$ by letting $f_k \equiv f(t_k)$ where $t_k \equiv k\Delta$, with $k = 0, ..., N-1$. Here Δ refers to the time between to discrete time points, t_k and t_{k+1}, and N is the number of time points in the signal from the first to last sample, zero indexed [71].

Switching the integral with a summation, the discrete Fourier transform becomes

$$F_n = \mathcal{F}_k[\{f_k\}_{k=0}^{N-1}](n) = \frac{1}{N}\sum_{n=0}^{N-1} f_k e^{-2\pi i n/N} \tag{2.7}$$

This transform takes a discrete time-domain signal and transforms it into a discrete frequency spectrum. This spectrum contains as many points as there are samples in the time domain signal. These points are complex numbers, each calculated as a function of some frequency and their absolute value represents the energy contained in that frequency.

The DFT makes it possible to get an insight into which frequencies are presents in the data and how much each of them contribute to the total energy in the signal. This is useful when you have a signal that seems to be noisy and contain no information and you want to find whether it contains any cyclical pattern or not.

But the DFT is not perfect. There are two major drawbacks with the approach. Firstly, it requires the signal to be stationary over time. This means that the properties of the signal should not change at any time. Secondly, it can only detect individual periods of cyclical components and not their position in time [58]. These drawbacks will in many cases render FFT less efficient. Although, the results from FFT on non-stationary signals in many cases are not trustworthy, it can be worth trying in many cases.

There are different methods that are designed to cope with its drawback. An example is the Short-Time Fourier Transform, which splits the time domain signal into windows before conducting the transform. The windowing keeps the temporal information about the frequency components and, in many cases, the signal in each window will be somewhat stationary. Here, the window shape and size must be adapted to the properties of the signal as it limits the time-frequency resolution. Sejdić et al. [51] provides a more detailed overview of different time-frequency representation with their advantages and drawbacks.

One last property of the data that may cause problems for DFT is non-linearity. By looking at equation (2.7), the transformation aims to find a sum of frequency components that, combined, sums up to the original signal. A system is said to be nonlinear if it does not satisfy the superposition principle or its output is not directly proportional to its input [48].

DFT is a powerful tool for analyzing signals, but it can be computationally expensive to calculate. The number of operations required to compute the DFT of a signal of length N is proportional to N^2, which means that the computation time grows very quickly as N increases. For example, if $N = 1024$, then we need about one million operations to perform the DFT [60].

Fortunately, there is a way to speed up the calculation of the DFT by exploiting some symmetries and redundancies in the formula [32]. This method is called the Fast Fourier Transform (FFT), and it reduces the number of operations required to compute the DFT from N^2 to $N \log N$. This means that the computation time grows much more slowly as N increases. For example, if = 1024, then we only need about 10 thousand operations to perform the FFT.

2.1.3. Empirical Mode Decomposition (EMD)

Empirical Mode Decomposition (EMD) is a method for decomposing time series data into a finite and often small number of components. The components are called Intrinsic Mode Functions (IMFs), which are zero-mean oscillating signals that are calculated analytically through the Hilbert-Huang transform. The IMFs aims to represents some physically meaningful part of the signal. The EMD extracts the IMFs in a sequential manner based on the energy associated with various intrinsic time scales of the signal, starting high frequencies on a fine temporal scale and ending with low frequencies on a coarser temporal scale. An important advantage with this method, compared to other decomposition methods, is that it is fully data-driven and adaptive with its basis signals adapted from the data automatically. Consequently, there is no need for determining any basis functions or other parameters beforehand [37].

As mentioned, EMD aims to decompose the data into IMFs. By definition, an IMF is a function that satisfies two conditions: the number of extremas and zero-crossings must be either equal or differ by one through the whole data set and, at any point, the mean of the envelope defined by the local maxima and the envelope defined by the local minima is zero. The name "intrinsic mode function" describes how each component contains one oscillation mode within the data. These conditions guarantees a well-behaved Hilbert transform of the IMF [24].

The process of extracting the IMFs from the original time series data consists of 5 steps [19]:

1. Obtain the local minimas and local maximas of the signal $X(t)$.

2. Create the upper envelope by using cubic splines to fit a function to the local maxima. Do the same for the local minima to create the lower envelope.

3. Calculate the mean of $X(t)$, m_1, by averaging the upper and lower envelope at individual points. Let the difference between m_1 and the original data be defined as: $h_1 = X(t) - m_1$.

4. If h_1 satisfies the requirements of an IMF, h_1 is the first IMF of the data. If it it does not satisfy the requirements, which is often the case, swap $X(t)$ with h_1 and repeat step 1-3 until conditions are satisfied.

5. Repeat step 1-4 until the desired number of modes is extracted from the signal or the residual becomes a monotonic function.

This process will result in an finite number of IMFs and a residual. The residual will be a constant, monotonic function or a function with only one maxima and minima from which no IMF can be extracted.

Although being a versatile decomposition method that is applicable to many different signals, EMD has some drawbacks. First, EMD suffers from mode mixing. Mode mixing is when one IMF either consists of signals of widely different scales, or a signal of similar scale being present in different IMFs [74]. Huang et al. [24] presents intermittency as a reason for this mode mixing and also states how this make the physical meaning of IMFs unclear. Intermittency describes irregular or unpredictable behavior of a system where it has periods of activity interrupted by periods of rest. Second, EMD is sensitive to small perturbations in the data or addition of new data. Wu and Huang [74] shows how small perturbations only in the first 10% of the data leads to significantly different

IMFs. This renders EMD quite unstable. Several other drawbacks of EMD are further investigated in [74].

Another use case for EMD is for noise filtering. Boudraa et al. [11] presents an EMD-based noise filtering technique where the signal are decomposed into a set of IMFs, before reconstructing the signal using only a selection of those IMFs. As signals, typically, are corrupted by white noise, they assume that low frequency signal components have a higher signal-to-noise ratio compared to higher frequency components. Based on this, there will be a mode, indexed j, after which the energy distribution of the information carrying parts of the signal will be greater than the high frequency, noise-carrying part. The signal can then be reconstructed using the IMFs with index $\geq j$.

This noise filtering use-case, emphasise EMD's capability of revealing the informational parts of the signal. A frequently used approach is to decompose the signal into IMFs using EMD before further perform analysis on each IMF, often using FFT [9, 28]. This means that each IMF is analysed and either kept or discarded based on its informational components. As noisy IMFs with no information are discarded, using EMD for analysis signals, may be looked on as a noise filtering technique.

In practice the EMD is performed using the PyEMD package in Python [34].

2.1.4. Ensemble Empirical Mode Decomposition (EEMD)

To deal with mode mixing and reduce the effects small perturbations of the signal has on the resulting IMFs of EMD, Wu and Huang [74] presents Ensemble Empirical Mode Decomposition (EEMD). This is an improved version of EMD that seeks to solve the problems of mode mixing and lack of stability of EMD by imposing white noise on the signal.

Generally, all measurement data are a combination of signal and noise,

$$x(t) = s(t) + n(t), \tag{2.8}$$

where $x(t)$ is the measured data, $s(t)$ is the true signal with information and $n(t)$ is some type of noise. Consequently, the goal is to remove as much noise as possible to end up with accurate and information carrying measurements. To do this the ensemble mean is a powerful approach. Here separate measurements of data are obtained, each containing different noise, and the mean of these measurements are used as measurement to improve the signal-to-noise ratio. EEMD utilises this technique by adding white noise to the data,. This makes it as if different measurements of the data were taken in a real life measurement process. The i-th artificial measurement of the signal will then be

$$x_i(t) = x(t) + w_i(t) \tag{2.9}$$

were $x(t)$ is the data and $w_i(t)$ is the i-th unique realization of white noise. Although adding noise will reduce the signal-to-noise ratio, it will provide a relatively uniform reference scale distribution to facilitate EMD. Therefore, the lowered signal-to-noise ratio will not affect the decomposition but enhance it to avoid mode mixing.

This results in the following algorithm to compute IMFs of a signal using EEMD:

1. Add a unique realization of white noise to the data.
2. Decompose the data, with white noise, into IMFs using EMD.
3. Repeat step 1 and 2 several times, using different realizations of white noise each time.

4. Calculate the ensemble mean of corresponding IMFs to the different decompositions as the final IMFs.

As the final IMFs are calculated as the ensemble mean of the corresponding IMFs, the added white noise in each iteration of the algorithm will cancel each other out. Leaving the final IMFs unaffected by the added white noise.

As EEMD share many of its properties with EMD it may also be used for noise filtering in the same way as for EMD. A possible approach utilising this is explained in section 2.1.3.

In practice the EEMD is performed using the PyEMD package in Python [34].

2.2. Neural Networks

Inspired by the structure of the brain, neural networks are a class of machine learning techniques applicable for several different tasks. In the same way as the brain is made up of interconnected neurons, neural networks are made up of artificial interconnected neurons called "units". A network consist of layers, which is a group of neurons grouped together at the same depth. Different units in a network are connected and are able to pass values between each other. When the input values arrives at the neuron, it is processed by an activation function before being sent to the other connected neurons. The importance of these connections between units is encoded in the weight of the connection. That is a number that the connection between two units is multiplied with.

Connections between units can be arranged in different ways. For example, it can be from one unit in one layer to a unit in a different layer, or between two units in the same layer [20]. How units are arranged and how the connections are made is an important part of the "architecture" of a neural network. Different neural network architecture are suited for different types of tasks.

Neural networks are said to be able to learn from data. This refers to their ability to recognise patterns and structures in data. Neural networks are particularly good at learning from complex, high-dimensional data, such as images and natural language [35]. The learning is done by adjusting the weights of the connections between the different units and layers in special ways. In most modern approaches this is done using a gradient-descent based algorithm called backpropagation [49].

Multilayered neural networks are known as universal function approximators, meaning that they can, given the right size and computational power, approximate any function [22]. Consequently, neural networks will only be useful whenever there are a functional relationship between what is known, the input to the network, and what is desired to know, the targets corresponding to these inputs. For example, say the goal is to determine whether an image is of a cat or a dog, then what is known is the image, and the target is cat or dog. Given that the images actually contains a cat or a dog, there will be some property of the image that makes it possible to determine whats in the image. The function relationship the network tries to learn will be these properties, that links the image to a target.

In the context of time series prediction there often exists some function that describes the transition from the values at timestep t and timestep $t+1$, meaning that

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t), \tag{2.10}$$

where $\mathbf{x}_{t+1}$ is the value of the time series in the next time step, $\mathbf{x}_t$ is the value of the time series in the current time step, and $f(\cdot)$ is some function describing their relationship. In real life, this relationship function, f, may be arbitrarily complex.

Will a neural network always be able to predict the time series data, or determine what is in an image, no matter the complexity of the function? No, the theorem of universal approximation also states that any lack of success in approximating any function must arise from inadequate learning, insufficient numbers of hidden units or lack of deterministic relationship between input and target [22]. First, as the complexity in the relationship between historical data and future data increases, so does the amount of hidden units required to approximate this relationship. Adding more units results in an increased demand for memory and computational power during the learning process, which may result in inadequate learning. Second, noise is an issue in many real life applications. The proposed relationship between historical and future data may be polluted by measurement error or random events affecting the data, making the relationship between the historical input data and future target data non-deterministic. In other words, a NN model requires good input data quality to perform. Therefore it will, in real life applications, often be difficult to approximate the relationship between historical and future data, although NNs are universal approximates in theory.

Fortunately, this does not render the problem of predicting time series, or classifying images, infeasible. The solution is to create NN models that generalize such that the most important relationships between historical and future data, or the most important properties of an image, are learned. To achieve this, hyperparameters such as network architecture, activation functions, distribution of weights, and regularization techniques during training needs to be carefully considered.

During training of the networks used in this thesis, different techniques, to improve generalization and reduce overfitting of the network, are applied. This is L1-regularization, dropout and early stopping. Tian and Zhang [66] provides a detailed overview of different regularization techniques, including L1-regularization, Cheng et al. [14] shows how dropout can improve LSTM performance, and Prechelt [44] presents early stopping to avoid overfitting. The optimization is performed using the Adam optimizer [31].

The rest of this section will cover details about different neural network architectures and how they can be used for prediction of time series. Details about backpropagation and training of neural networks will not be covered, but Nielsen [39] and Goodfellow et al. [20] provides well-written and detailed explanations.

Feed-Forward Neural Network

A fully connected Feed-Forward Neural Networks (FFNN) is a type of a fully connected network. These are regarded as one of the simpler types of neural networks, and some type of fully connected FFNNs are present in most neural network architectures. Fully connected FFNNs ranges from simple, one-layer, perceptrons to multilayer perceptrons with numerous layers. What distinguishes fully connected FFNNs from other networks is that all units in each layer are connected to all units in the next layer and that the data are fed through the network in one direction, from input to output, with no recurrent connections.

As all units in one layers has a connection to all units in the next layer, each unit gets as many inputs as there are units in the previous layer. Each unit takes the outputs from the previous layer, which now has been multiplied with the weights corresponding to the connections, adds them together and processes them with its activation function. After training, the output values from the units represents some aspect of the pattern of the input data that the units has learned to recognize.Consequently, by gathering these units into layers, and gathering the layers into networks, it is possible to learn complex patterns from the simple aspects that each unit aims to learn. In this way, a neural

network can be thought of as a long and complicated function that is built up from many weighted sums of nonlinear functions of weighted sums of nonlinear functions, and so on, as many as there are hidden layers hidden layers in the network.

Typically, a FFNN consists of an input layer, an output layer, and a number of hidden layers. The data, such as an image, time series measurements or a portion of a speech signal, are fed into the input layer. From there it is fed to the first of the hidden layers. The hidden layers then apply their weights and activation functions to the data, transforming and augmenting it in an attempt to extract important information. Lastly, the data is fed into the output layer where it is further transformed and nonlinearized. The output from this layer serves as the networks prediction on the solution to the task it tries to learn.

Let the output of a FFNN be denoted as $\hat{\boldsymbol{y}}$, and let the neural network itself be denoted as $f(\boldsymbol{x};\boldsymbol{\theta})$. That is,

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x};\boldsymbol{\theta}), \tag{2.11}$$

where $\hat{\boldsymbol{y}} \in \mathbb{R}^s$ is the output of the network with length s. $\boldsymbol{x} \in \mathbb{R}^d$ is the input to the network with dimension d and $\boldsymbol{\theta} \in \mathbb{R}^p$ is the parameters of the network with dimension p.

The process of feeding data through a network can be described mathematically, here showing how the output of each layer can be calculated. Let j denote the layer number and $\boldsymbol{Z}^j \in \mathbb{R}^{L_j}$ denote the output from the last layer of length L_j. The output from the next layer, $j+1$, can be written as:

$$\boldsymbol{Z^{j+1}} = \sigma(\boldsymbol{W}^{j+1}\boldsymbol{Z}^j + \boldsymbol{b}^{j+1}). \tag{2.12}$$

$\boldsymbol{Z}^{j+1} \in \mathbb{R}^{L_{j+1}}$ is the output vector from layer $j+1$ of length L_{j+1}. $\boldsymbol{W}^{j+1} \in \mathbb{R}^{L_{j+1}\times L_j}$ is the weight matrix of the layer and $\boldsymbol{b}^{j+1} \in \mathbb{R}^{L_{j+1}}$ is the bias vector. $\boldsymbol{W}^{j+1}$ together with $\boldsymbol{b}^{j+1}$ are the parameters of the the layer: $\boldsymbol{\theta}^{j+1} = \{\boldsymbol{W}^{j+1}, \boldsymbol{b}^{j+1}\}$. Lastly, σ is a, often nonlinear, activation function. σ is applied to the output vector element-wise, that is

$$\sigma(\boldsymbol{W}^{j+1}\boldsymbol{Z}^j + \boldsymbol{b}^{j+1}) = \begin{bmatrix} \sigma(W_1^{j+1}\boldsymbol{Z}^j + b_1^{j+1}) \\ \vdots \\ \sigma(W_i^{j+1}\boldsymbol{Z}^j + b_i^{j+1}) \\ \vdots \\ \sigma(W_{L_{j+1}}^{j+1}\boldsymbol{Z}^j + b_{L_{j+1}}^{j+1}) \end{bmatrix}, \tag{2.13}$$

where $i = 1, .., L_{j+1}$ are the indices of all layers in the network. Thus, W_i^{j+1} are the i-th row vector of the weight matrix $\boldsymbol{W}_{j+1}$, and b_i^{j+1} are the i-th entry of the bias vector. In words, the output of each layer in the network is a nonlinear function of the weighted sum of the prior layer in the network plus a bias. Each neuron takes the whole vector from the prior layer as input and puts a value, or weight, to the importance of each output, the more important the output, the higher magnitude of the corresponding weight.

The activation function σ can be any continuous and differentiable mathematical function. However, in order to learn nonlinear and highly complex data, the activation function should be nonlinear. Some popular choices include the sigmoid function, hyperbolic tangent, and ReLU [56].

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) is a type of neural networks designed for pattern detection in sequence data [50]. Sequence data are data where the order of its value is important and each value has some relationship with earlier values. Examples of such data includes handwriting, genes, text or time series data, for example financial data and weather data. The difference between RNNs and FFNN is how the information gets passed through the network. In a FFNN the information are only passed in one direction, from input to output, without any cycles, while, in a RNN, the network has cycle and may pass data back to itself. This introduces the possibility of taking previous inputs into account, instead of just feeding the current input through the network. Consequently, the network will have memory of the earlier inputs and their value may affect the output of the current input.

Denote the memory, or hidden unit, and input at time step t respectively as $\mathbf{H}_t \in \mathbb{R}^{n\times h}$ and $\mathbf{X}_t \in \mathbb{R}^{n\times d}$, where h is the number of hidden units, n is the number of samples and d is dimension of each sample. Further, let $\mathbf{W}_{xh} \in \mathbb{R}^{d\times h}$ be the weight matrix of each layer, $\mathbf{W}_{hh} \in \mathbb{R}^{h\times h}$ be the hidden-state-to-hidden-state matrix, and $\mathbf{b}_h \in \mathbb{R}^{1\times h}$ be the bias. Lastly, each layer has an activation function, ϕ, typically logistic sigmoid or tanh. Combining this results in the following output equations for each layer

$$\begin{aligned} \mathbf{H}_t &= \phi_h(\mathbf{X}_t\mathbf{W}_{xh} + \mathbf{H}_{t-1}\mathbf{W}_{hh} + \mathbf{b}_h) \\ \mathbf{O}_t &= \phi_o(\mathbf{H}_t\mathbf{W}_{ho} + \mathbf{b}_o) \end{aligned} \tag{2.14}$$

Equation (2.14) shows the update equation for the hidden layer and the output equation, respectively. As you can see, the output is calculated using the hidden units and is therefore a result of not only the input, but also prior inputs.

As the RNN are quite different from feed-forward neural networks training RNNs requires modification to the classic backpropagation algorithm. The modified algorithm is called the Backpropagation Through Time (BPTT) algorithm. This algorithm basically unfolds the RNN in a way that makes it possible to apply the regular backpropagation algorithm to it. The inner workings of this algorithm will not be covered in this thesis, but, for the curious reader, Schmidt [50] provides a detailed walk through of this algorithm.

The BPTT involves repetitive multiplication of the hidden states weights, $\mathbf{W}_{hh}$, to calculate the loss function. This can make the overall loss function very large and render the problem infeasible. This is due to eigenvalues less than 1 vanishes and eigenvalues grater than 1 diverging when the network learns from longer time intervals [8]. In practice, the vanishing eigenvalues stops the contribution of far earlier time steps to the current time step. Conversely, exploding eigenvalues results in the network valuing each weight too much and it changes it heavily. This is a key problem for RNNs.

A partial solution to this problem is the truncated BPTT [72]. The problem of vanishing and exploding gradients are avoided by establishing an upper bound for the number of timesteps a gradient can flow back. As the BPTT, in some way, unfolds the RNN to treat it as a feed-forward neural network the truncated BPTT can be seen on as establishing an upper bound for the number of hidden layers in this unfolded RNN. Any time steps before this cut off will not be considered therefore the truncated BPTT will limit the performance of the RNN.

Long Short-Term Memory Networks

Long Short-Term Memory (LSTM) networks were introduced to properly handle the RNN´s problem of vanishing gradients. To solve the problem they store information in gated cells, which are outside the traditional neural network flow. This can make LSTMs able to learn from many more time steps, way over 1000, than traditional RNNs [38].

The gated memory cell is a memory unit outside the flow of the network from which information can be stored in or read from. Whether the content of the cell can be changed, accessed or erased is determined by the input gate, output gate and forget gate, respectively. The input gate controls if the an input should be written to the cell, the output gate controls if what is saved in the cell should be read or not, and the forget gate controls if what is stored in the cell should be erased or not. The cells consists of a gate-wise weight matrix and a sigmoid function, and the value at the gate after multiplying the input with the weight matrix and feed it through the sigmoid function determines if the gate remains closed or not. As these gates are analog, they are also differentiable and can be included in backpropagation based learning algorithms.

Let the input gate be $\mathbf{I}_t$, the output gate be $\mathbf{O}_t$ and the forget gate be $\mathbf{F}_t$. For a time step, t, the computations for each of these gates will be

$$\begin{aligned} \mathbf{O}_t &= \sigma(\mathbf{X}_t\mathbf{W}_{xo} + \mathbf{H}_{t-1}\mathbf{W}_{ho} + \mathbf{b}_o) \\ \mathbf{I}_t &= \sigma(\mathbf{X}_t\mathbf{W}_{xi} + \mathbf{H}_{t-1}\mathbf{W}_{hi} + \mathbf{b}_i) \\ \mathbf{F}_t &= \sigma(\mathbf{X}_t\mathbf{W}_{xf} + \mathbf{H}_{t-1}\mathbf{W}_{hf} + \mathbf{b}_f) \end{aligned} \tag{2.15}$$

In the above equations, $\mathbf{W}_{xo}, \mathbf{W}_{xi}, \mathbf{W}_{xf} \in \mathbb{R}^{d\times h}$ are the weight matrices controlling how the current input affects the cell gates. $\mathbf{W}_{ho}, \mathbf{W}_{hi}, \mathbf{W}_{hf} \in \mathbb{R}^{h\times h}$ are weights controlling how previous time steps, or the hidden states affects the gates. Lastly, $\mathbf{b}_o, \mathbf{b}_i, \mathbf{b}_f \in \mathbb{R}^{1\times h}$ are the gates biases.

In addition to the above gates, there is one more computation taking place, called the candidate memory. This calculation decides how the internal memory state of the gated cell are updated if the input gates decides that it are to be updated. The calculation of this candidate memory, called $\tilde{\mathbf{C}}_t \in \mathbb{R}^{n\times h}$, is done in the exact same way as for the gates, but with its own set of weights and tanh as activation function

$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{X}_t\mathbf{W}_{xc} + \mathbf{H}_{t-1}\mathbf{W}_{hc} + \mathbf{b}_c), \tag{2.16}$$

where $\mathbf{W}_{xc} \in \mathbb{R}^{d\times h}$ is the input weight, $\mathbf{W}_{hc} \in \mathbb{R}^{h\times h}$ is the hidden states weights and $\mathbf{b}_c \in \mathbb{R}^{1\times h}$ is the bias.

To tie it all together, the new state of the memory cell, $\mathbf{C}_t$, is updated in the following way

$$\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \tilde{\mathbf{C}}_t \tag{2.17}$$

where $\odot$ denotes element-wise multiplication.

Lastly, the computation of the hidden state is done by

$$\mathbf{H}_t = \mathbf{O}_t \odot \tanh(\mathbf{C}_t). \tag{2.18}$$

The gated memory cell given by the equations and functionality described above gives the LSTM the possibility of learning important features for many time steps. The memory of the cell is gated by the input gate and input is only allowed to enter and change the memory when the cell gate allows. Also, the cell learns when to let the network read from the internal memory of the cell making it possible to include and

ignore information at different time steps. The network can also totally remove all memory in a cell if the forget gate are triggered.

Nicholson [38] and Schmidt [50] provides more details about the LSTM architecture and how they learn.

2.3. Prediction of Time Series Data

Prediction of time series data is about making predictions about future events in a time series based on historical data. Time series data are typically historical observations of events gathered over time, usually with regular intervals. These observations can be of a wide range of phenomena, for example stock prices, customer behavior or, in this case, temperature measurements inside a small greenhouse placed in the office of a professor and measurements along an oil drilling pipe.

The process of prediction is to use statistics or machine learning to build models that can make predictions about future values of a time series, based on historical data. The goal is to build models that manages to exploit patterns, trends and relationships within the data to make accurate predictions. It is therefore important to have knowledge about the characteristics of the data and know which models that are suitable for such characteristics.

There are several different approaches to make accurate time series prediction models, and which is the best depends on the task. The complexity of models ranges from simple statistical models, such as ARIMA models [21], to simple machine learning models based on feed-forward neural networks [36], to more complex machine learning models such as LSTM [57] and transformers [67]. Not only are there different machine learning models that can be used for predictions, there are also several different approaches to how these models may be applied. Legaard et al. [36] provides an overview of many such approaches, especially in the context of time series predictions of dynamical systems.

2.3.1. Fully Connected Feed-Forward Neural Network for Time Series Prediction

Fully Connected Feed-Forward Neural Networks are NNs where all neurons in one layer is connected to all neurons in the next layer and there are no recurrent connections. There are several possible approaches for using fully connected feed-forward networks for time series predictions. The FFNN based model used in this thesis is what is called a *Euler time-stepper model*. Time-stepper models are similar to traditional numerical solvers as the derivative of the model is approximated at each time step, t_k. This values is then used to integrate the system through time. A benefit with such models is that they can make use of knowledge about and be combined with classical integration schemes.

Specifically, the *Euler time-stepper* tries to learn the derivative of the system at a timestep, t_k, given the states of the system. Starting of with a given set of initial conditions, the method estimates the derivative at each timestep given the internal states and inputs to the system. This derivative is then multiplied by the length of the time step and added to the current value. Mathematically, it integrates through time by

$$\begin{aligned} \dot{\tilde{\mathbf{x}}}_t &= f(\tilde{\mathbf{x}}_t; \theta) \\ \tilde{\mathbf{x}}_{t+1} &= \tilde{\mathbf{x}}_t + \Delta T \cdot \dot{\tilde{\mathbf{x}}}_t, \end{aligned} \tag{2.19}$$

where $\dot{\tilde{\mathbf{x}}}_t$ is the estimate of the derivative of the system at the current timestep. $f(\tilde{\mathbf{x}}_t; \theta)$ is a neural network taking prior measurements or estimates of the states at the current

time step, $\tilde{\mathbf{x}}_t$ as input, and has parameters θ. $\tilde{\mathbf{x}}_{t+1}$ is the estimated value of the states at the next timestep, and it is found by taking the current state value and adding the estimate of the derivative times the size of the timestep, ΔT. It is worthy to note that the actual inputs to a system may be known and their exact value can be incorporated into $\tilde{\mathbf{x}}_t$.

How to use the Euler time-stepper to estimate the values of the dynamical system through the time horizon is shown in algorithm 1.

Algorithm 1: Neural network based Euler time-stepper

Require: Initial conditions, $\mathbf{x}_0$, size of timestep ΔT, prediction horizon, T, trained neural network, f, with parameters θ.
Initialize prediction vector, $\mathbf{x}_{\text{pred}}$
Insert initial conditions as first input in prediction vector: $\mathbf{x}_{\text{pred}}[0] \leftarrow \mathbf{x}_0$
Add prior information to $\mathbf{x}_{\text{pred}}$, such as controllable inputs to the system.
for t = 1, ..., T **do**
  Predict derivative: $\dot{\mathbf{x}} \leftarrow f(\mathbf{x}_{\text{pred}}[t]; \theta)$
  Update next prediction step, $t+1$: $\mathbf{x}_{\text{pred}}[t+1] \leftarrow \mathbf{x}_{\text{pred}}[t] + \Delta T \cdot \dot{\mathbf{x}}$
end for

2.3.2. Long Short-Term Memory Networks for Time Series Prediction

As already mentioned in section 2.2, LSTMs are great for learning connections between far apart samples in sequence data. It has been applied to various domains, such as: Covid-19 predictions and language models [55, 62]. However, there are many different ways to use LSTMs for prediction. This section will explain two of the many possible approaches to use LSTM for time series prediction.

Based on different input and output sequences, LSTMs can be classified into four types [59]:

- One-to-one: The LSTM takes one input and gives back one output. It is suitable for simple tasks such as classification or regression.
- One-to-many: The LSTM takes one input and produces several outputs. It is suitable for tasks such as image captioning or music generation.
- Many-to-one: The LSTM takes a sequence of inputs and generates one output. It is suitable for tasks such as time series prediction.
- Many-to-many: The LSTM takes a sequence of inputs and generates a sequence of outputs. It is suitable for tasks such as speech recognition.

For the time series prediction case, the last two types of LSTMs are most suitable as prior time steps are provided.

This thesis focuses on two different approaches based on many-to-one LSTMs. The first approach is a direct time-stepper model [36], which predicts the value in the next time step based on a given number of prior time steps. Let this number of prior time steps be called lookback and denoted as l. Further, let the LSTM-based neural network be denoted by $\boldsymbol{f}$ with parameters $\boldsymbol{\theta}$. Then a prediction of the next time step, $\mathbf{x}_t$ will be

$$\tilde{\mathbf{x}}_{t+1} = \boldsymbol{f}(\mathbf{x}[t-l:t]; \boldsymbol{\theta}) \tag{2.20}$$

Algorithm 2: LSTM based direct time-stepper

Require: Initial conditions $\mathbf{x}_0$, size of timestep ΔT, prediction horizon T, lookback l trained LSTM based NN f, with parameters θ.
Initialize prediction vector, $\mathbf{x}_{\text{pred}}$
Insert initial conditions as first input in prediction vector: $\mathbf{x}_{\text{pred}}[0] \leftarrow \mathbf{x}_0$
Add prior information to $\mathbf{x}_{\text{pred}}$ such as controllable inputs to the system.
for t = l, ..., T **do**
  Predict value of next timestep: $\mathbf{x} \leftarrow f(\mathbf{x}_{\text{pred}}[t-l:t];\theta)$
  Update next prediction step, $i+1$: $\mathbf{x}_{\text{pred}}[t+1] \leftarrow \mathbf{x}$
end for

where $\mathbf{x}[t-l:t]$ are all values of timesteps within the lookback horizon.

The second approach is similar to the one presented in section 2.3.1, except that the derivative is predicted using a LSTM, instead of an FFNN. Again, let the number of prior time steps used for prediction be denoted l, let the LSTM based neural network be denoted by $\mathbf{N}$. Then a prediction of the derivative of the current time step, $\dot{\mathbf{x}}_t$

$$\dot{\tilde{\mathbf{x}}}_{t+1} = \mathbf{N}(\mathbf{x}[t-l:t]) \tag{2.21}$$

A prediction of the value at time $t+1$ can then be made by integration

$$\tilde{\mathbf{x}}_{t+1} \approx \mathbf{x}_t + \Delta T \cdot \dot{\mathbf{x}}_t \tag{2.22}$$

Algorithm 3: LSTM based Euler time-stepper

Require: Initial conditions $\mathbf{x}_0$, size of timestep ΔT, prediction horizon T, lookback l trained LSTM based NN f, with parameters θ.
Initialize prediction vector, $\mathbf{x}_{\text{pred}}$
Insert initial conditions as first input in prediction vector: $\mathbf{x}_{\text{pred}}[0] \leftarrow \mathbf{x}_0$
Add prior information to $\mathbf{x}_{\text{pred}}$ such as controllable inputs to the system.
for t = l, ..., T **do**
  Predict derivative: $\dot{\mathbf{x}} \leftarrow f(\mathbf{x}_{\text{pred}}[t-l:t];\theta)$
  Update next prediction step, $i+1$: $\mathbf{x}_{\text{pred}}[t+1] \leftarrow \mathbf{x}_{\text{pred}}[t] + \Delta T \cdot \dot{\mathbf{x}}$
end for

2.4. Laboratory-Scale: Greenhouse

A central part of this thesis is to show how Laboratory-scale data can be used to verify methods for analysis and prediction, before applying the same method to field scale data. The Laboratory-scale data, in this case, is a greenhouse placed in to office of Professor Adil Rasheed. The greenhouse is $50 \times 50 \times 60$ cm with clear polycarbonate walls. A plant is placed in the center of the greenhouse. Bruaset [12] built and designed the greenhouse setup. A picture of the setup are shown in figure 2.1.

The greenhouse is packed with sensors and control units that affect the conditions within the greenhouse. The sensors are one humidity and temperature sensor, one moisture sensor in the soil of the plant, one light sensor and 13 temperature and humidity sensors distributed in various locations. For technical sensor information see table A.1.

Figure 2.1.: Picture of greenhouse setup.

The controls are one heater hanging from the ceiling, one inlet fan, one outlet fan, lighting and watering valve.

This combination of sensors and controls creates the possibility of generating high quality data from a real life dynamical system. Instead of using data from a complex field-scale system where measurements are expensive and unlabeled, this system can generate data with similar patterns and complexity. This data can further be used to test methods for analysis and prediction on cheap data.

The greenhouse is a miniature of a bigger, industrial greenhouse. The dynamics of such greenhouses are influenced by interactions between physical components of the greenhouse, such as the cover, the soil, the plants, and external factors, such as solar radiation, wind, and humidity. The interaction between these factors results in heat and mass transfer that results in changes in energy and water balance. These dynamics can be described mathematically as a set of differential equations that relate the different variables, such as temperature, humidity, and CO_2 concentration [46].

One of the main challenges in modeling the greenhouse dynamics is to account for the spatial variability of the state variables within the greenhouse volume and the nonlinear dynamics of the radiation and heat transfer [6, 47]. Fortunately, as the laboratory-scale greenhouse is both a miniature and placed inside an office, its dynamics are easier to explain.

The temperature in the greenhouse will be a function of the temperature in the office and the heater in the greenhouse. The outside temperature will affect the inside temperature in two ways: when the fans are blowing air into the greenhouse, and through heat transfer. The heater will, when having a higher temperature than the inside temperature of the greenhouse, result in an increase in the temperature. Temperature is the main driving force for the internal states in the greenhouse, as it affects the other measurements, such as humidity and moisture in different ways.

Humidity is affected by temperature as the saturation water vapour pressure is as function of temperature. This is the pressure at which water vapour is in thermodynamic equilibrium with its condensed state. If the pressure gets higher than this vapour pressure, water will condense, and if the pressure gets lower, water will evaporate. Saturation water vapour pressure is linked with humidity through the formula for relative humidity:

$$\text{Relative humidity } (\%) = \frac{\text{Water vapour pressure}}{\text{Saturation water vapour pressure}} \times 100 \tag{2.23}$$

The saturation water vapour pressure increases with temperature because higher temperature means higher kinetic energy of the water molecules, which makes them more likely to escape the liquid phase and enter the gas phase [41]. Based on this, the relative humidity will decrease when the temperature increases. Also, when the fans are turned on, air from the office is sucked in by the inlet fan and blown out by the outlet fan resulting in a air flow through the greenhouse. Depending on the humidity in the surrounding environment, the humidity inside the greenhouse will be affected. Sucking in air with lower humidity will make the humidity drop, while air with higher humidity will make it increase [54, 73].

Çetin and Sevik [75] shows that how much a plant affects the CO_2 concentration in a room is directly proportional to their size. As the only plants placed in the greenhouse are small sprouts, the effect of their photosynthesis can be neglected. Humans are the main contributor to indoor CO_2 levels in non-industrial indoor environments [4]. Therefore, human presence in the office, or close to the greenhouse, and human interaction with the greenhouse will be the main contributor to change in CO_2 level in the greenhouse.

The walls of the greenhouse are clear polycarbonate sheets. As they are clear, the outside light is let in and, since it is placed in a office with windows, the light sensors will capture the outside light cycle. There are also lighting in the greenhouse that can be turned on and off. The measurements of light will then be a function of the daily light cycle, combined with the light from the greenhouse lighting.

Lastly, we have measurements of moisture. These are measurements of the moisture in the soil of the plant. Seneviratne et al. [53] shows how moisture and temperature are coupled and affects each other with different positive and negative feedback loops. Therefore, the exact relationship between moisture and temperature is complex and, depending on factors, such as soil type, soil condition, and temperature, it may be inverse or proportional.

2.5. Field Scale: Oil Drilling Hole Cleaning Process

The field scale data used in this thesis to validate the analysis and prediction methods on data from the oil drilling domain. Specifically, the data used in this study is collected from drilling operations on the Norwegian continental shelf. The well contains multi-lateral systems that include highly deviated and horizontal sections. The end-of-well reports indicates that there were several issues during drilling, such as: poor cuttings transportation, loss of circulation, stuck pipe and problems with well bore stability. Consequently, some portions of the drilled sections were plugged and abandoned. The data consist of measurements of physical parameters, obtained from both surface and down-hole sensors along the drill string. The data is sampled with an average frequency of 1 hertz. The data used in this study also includes tripping in and out, drilling activities in different formations and various well bore configurations with vertical, inclined and horizontal sections.

Hole cleaning is the process of removing cuttings while drilling through the soil, carving a hole where the oil pipe can go to reach the oil reservoir. The drilling is done using a rotating drill string with a bit at the end. This bit drills through the soil resulting in a vast amount of cuttings being left. To remove these cuttings, drilling fluids are pumped through the drill string. When reaching the end, the fluid, combined with the rotation of the string, carries the cuttings out of the hole. Figure 2.2 shows a simple illustration of this hole drilling process.

Hole drilling is a complex process and bad hole cleaning may lead to abortion of drilling session and, in worst case, loss of expensive equipment and increased nonproductive time (NPT). Good cuttings circulation is essential for effective and safe drilling. To monitor this process, there are mounted sensors at different locations along the drill string, as well as sensors on the surface. These sensors are measuring depth, pressure, temperature and torque, to mention some. Measurements from such sensors are denoted Along String Measurements (ASM). How the sensors are placed along the drill string are illustrated in figure 2.2.

The most important indicator for cuttings circulation is Equivalent Circulation Density (ECD) [2]. ECD is a derived quantity from measurements of mud weight (MW), annular pressure loss (APL), and true vertical depth (TVL). ECD is calculated as

$$\text{ECD} = \text{MW} + \frac{\text{APL}}{0.052 \times \text{TVD}}. \tag{2.24}$$

An increase in drilling cuttings in the hole will reduce the fluid flow area and increase the effective drilling fluid density, which consequently leads to an increase in the ECD [2]. This makes ECD a good indicator for the state of the cuttings transportation.

To ensure good cuttings circulation conditions it is therefore desirable to know, in advance, how the ECD will change given current drilling conditions and future control parameters. This also makes it possible to experiment with how different control inputs may affect the future ECD values.

The well boring is controlled by a team of operators that are monitoring the conditions of the drilling process. The drilling operation is constantly monitored by a drilling team. This team of operators can adjust some of the drilling parameters during drilling operations to ensure the safe drilling of the well. Some of the important control parameters are: the density of the mud drilling fluid pumped into the hole (DMIAVG), the flow rate of this drilling fluid (FLIAVG), the revolutions per minute for the drill string and bit (RPMBAVG), and the weight on bit (WOBAVG). An overview of these controls are presented in table 2.1.

Drilling fluid is pumped through the borehole to transport cuttings from the bottom of the well to the surface. This fluid is a mud drilling fluid that is pumped from the surface, through the drill string and out by the bit. This fluid ensures that the cuttings are carried up and out of the hole. It also reduces friction between the bit and rock, clean the bit, maintains well bore stability, and controls pressure [16].

The mud density and the rate at which the mud is pumped into the well, affects the ECD. Increasing the mud density, results in an increase of the weight of the mud exiting the hole. Referring to equation (2.24), this results in an increase in the ECD. The flow rate of the drilling fluid affects the annular pressure loss. Again, referring to equation (2.24), this will affect the ECD. But, as the drill string is rotating and the walls are stationary, the behaviour of the drilling fluids in the well is governed by complex dynamics and the exact relationship between the fluid density, flow rate and ECD is often hard to determine [27]. It is also important to note that, although being controllable, the mud density cannot be changed easily [15].

To drill through solid rock, a great pressure needs to be put on the drill bit and the bit needs to rotate. This is controlled by the parameters: weight on bit and revolutions per minute. Weight on bit is the amount of downward force exerted on the bit during drilling operations. Too low weight on bit during drilling leads to a low rate of penetration and too high weight on bit may result in a damaged bit. The revolutions per minute is how fast the drill string is rotating. These controls, in combination, affects the torque on bit, vibrations in the drill string, bit rotation, as well as the rate of penetration [3]. An increase in the rate of penetration will lead to an increased amount of cuttings, which again will lead to higher ECD, as this affects the mud density, flow of the drilling fluids and the annular pressure loss [29].

A key point to consider is that the controls have various combinations that can either increase or decrease the ECD. For instance, a higher flow in enhances the cuttings transportation, which can lower the ECD. Likewise, a higher RPM can reduce the ECD by improving the hole cleaning, especially in the directional and horizontal well sections. Hence, these parameters have both positive and negative effects on ECD. The safe range of these parameters is essential for a successful drilling operation.

2.6. Similarities Between Greenhouse and Oil Drilling

Both greenhouse dynamics and oil drilling dynamics are complex systems that involve interactions between physical components, external forces and control inputs. They can both be modeled mathematically as a set of differential equations that relate the different variables. However, these models are often both non-linear, uncertain, and are difficult

Table 2.1.: Measurements and controls the drilling operator can inspect and adjust during drilling to ensure good cutting circulation and hole cleaning.

Control:	**Unit:**	**Description:**
ECD	kg/L	Equivalent circulation density (Measurement)
DMIAVG	kg/m^3	Mud density in (Control)
FLIAVG	m^3/min	Flow rate in (Control)
RPMBAVG	RPM	Revolutions per minute total averaged (Control)
WOBAVG	kN	Average weight on bit (Control)

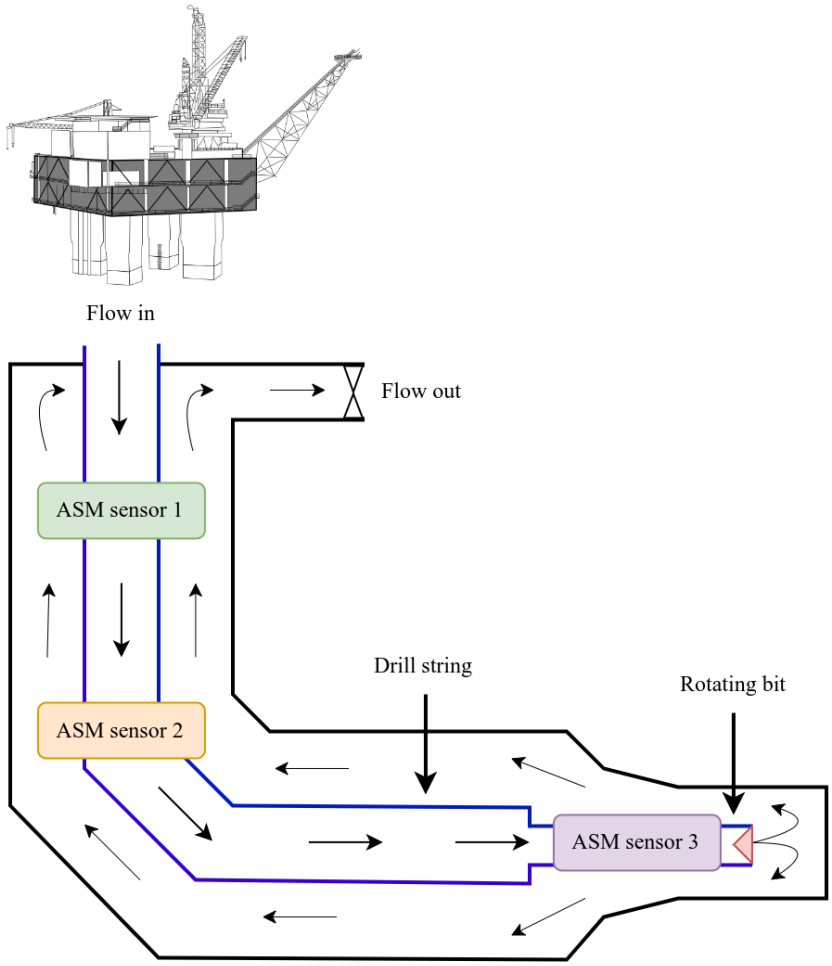


Figure 2.2.: Figure of drilling operation. The rotating drill string with a bit at the end bores through the soil and carves a hole. Throughout the drilling operation, drilling fluid are pumped through the string to create a circulation to remove cuttings from the hole. Along the drill string there are Along String Measurement (ASM) sensors.

to solve analytically or numerically.

Both systems has control inputs that can be adjusted to optimize the systems performance and achieve certain objectives. In the green house these are heater duty cycle, fan controls, lighting and watering valves. These controls can be used to control temperature, humidity, lighting and soil moisture in the greenhouse. In oil drilling, the control inputs are mud density, flow rate in, weight on bit and revolutions per minute. These inputs can be used to manipulate the rate of penetration, the torque on bit, the cuttings circulations, and the ECD in the well.

Another similarity is that both systems have sensors that measures different variables in the system. In the greenhouse these are temperature, humidity, light intensity and soil moisture sensors. In the greenhouse there are several sensors, including ECD, temperature, pressure, mud weight and depth. Measurements of physical quantities are often corrupted by noise. Noise in measurements are the deviation from the measured value of a physical quantity due to various sources of error and uncertainty in the measurement process. There exists numerous methods for trying to remove this noise from the measurements, such as using Kalman filters [42]. But although the noise is removed, the signal may not be perfect. Therefore, both systems show how a model performs when that data are from measurements of physical quantities where there may be noise present.

3. Method

The method section of this thesis consists of four steps: data collection, data pre-processing, model architecture and model evaluation. In the first step, data is collected from the two data sources: the laboratory-scale greenhouse and the field scale oil drilling operation. In the second step that data is processed to fit the analysis and prediction methods. In the third step, the models for analysis and prediction are designed. In the fourth step, different data sets and scenarios are created to evaluate the performance of the models and techniques based on some popular metrics.

3.1. Data Collection

The greenhouse data is collected from several sensors that measures temperature, humidity, light intensity, CO_2 and moisture inside the greenhouse over several months. The oil drilling data is provided by Aker BP through SINTEF and contains 92 different measurements from different oil drilling operations. The measurements include ECD, flow in, and out, weight on bit, revolutions per minute, hook load, depth, and rate of penetration. Some of these measurements are obtained using sensors placed at different depths along the drilling string using, such are called Along String Measurements (ASM).

3.2. Data pre-processing

In the second step the both data sets are pre-processed in different ways. The greenhouse data is temporally aligned, while both data sets are arranged to fit the models.

3.2.1. Laboratory Scale: Greenhouse

A benefit of the greenhouse setup is that the data is clean, cheap, and labeled. However, the distributed sensors and the control inputs have different sampling frequencies. The data from the controls has a sampling frequency of 15 seconds, while the data from the distributed sensors have a sampling frequency of 1 second. To align the data, the control data is re-sampled using linear interpolation before merged with the sensor data. This results in one common data set with sampling frequency of 1 second.

3.2.2. Field Scale: Oil Drilling Hole Cleaning Process

The field scale data from the oil drilling hole cleaning process is heavily pre-processed by Aker BP and SINTEF. The data is from long drilling operations and contains data from different parts of the operation, not only drilling. Therefore, the different drilling sections are identified, thanks to PhD candidate Mehmet Cagri Altindal, based on end-of-well reports. Based on this, the data is split into these different drilling sections. This is the only pre-processing step for the drilling data.

3.2.3. Standardization

Standardizing data before using it in the training process of neural networks can significantly reduce both the prediction error and the time it takes to find an optimal solution [61]. For analysis methods, such as PCA, standardization is essential as the analysis is working with variables often at different scales [26].

To standardize the data, each data value x_{ji} is both centered and divided by the standard deviation s_j of the n observations of variable j:

$$z_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j},$$

The initial data matrix $\boldsymbol{X}$ is then replaced by the standardized data matrix $\boldsymbol{Z}$ filled with standardized measurements.

3.2.4. Windowing for LSTM

Before feeding the data to a LSTM model, the data needs to be arranged in the correct way. Section 2.3.2 presents how the lookback is an important part of the LSTM model. The lookback is how many timesteps the model is allowed to look backwards in time. For example, let the lookback be 10 and the goal is to predict the values at the next time step. Then the model needs to get ten timesteps as input and one target value to evaluate the output. This means that the data needs to be organized in windows, where each window is the same size as the lookback used in the model, and each window has as many target values as time steps to predict. This pre-processing is applied to both greenhouse and oil drilling data.

3.3. Model architecture

The various time series analysis and prediction methods are developed and applied to the greenhouse data, and some of them are applied to oil drilling data. This section provide details about model architecture to make it possible to reproduce the experiments in this thesis.

3.3.1. Methods for Analysis

The methods for analysis, PCA, EMD, EEMD, and FFT does not require any architectural considerations. They are applied straight out of the box.

3.3.2. FFNN model

The simplest model used in this thesis is the FFNN model. It is used to predict the derivative of the time series and integrate through time. The model has an input layer of size 4, which takes in the measured or predicted temperature and humidity in the current time step and the controls, the heater duty cycle and fan, applied in the current time step. It has two hidden layers, each consisting of 32 neurons. The output size of the model is two, which corresponds to the predicted derivative of the temperature and humidity. The network is trained for 750 epochs, using the Adam optimizer. The learning rate was 0.0001 and the network was regularized during training using L1-regularization. An overview of the parameters for the model are presented in table C.1.

3.3.3. LSTM models

The LSTM models used in the thesis vary some in parameter value from task to task, but they all consist of one LSTM layer and three fully connected feed-forward layers at the end.

The models take measurements or predictions of temperature and humidity, and the control inputs, heater duty cycle and fan, as inputs, all with a lookback of 50. Based on this, the model produces two outputs: prediction of temperature and humidity. The LSTM layer of these models has a hidden size of 32 for the direct model and 64 for the Euler time-stepper, and it has one hidden layer for both cases. The fully connected output network has a hidden layer size of 32 for the direct model and 64 for the Euler time stepper. The models are trained for 50 epochs using the Adam optimizer and early stopping. During training, the LSTM using a dropout of 0.4, and the output layers are regularized using L1-regularization. All hyperparameters are presented in table C.2.

The LSTM model used for prediction on drilling data is, in many ways, similar to the greenhouse model. This model takes 5 inputs, measurements or predictions of ECD, and control inputs, mud weight, flow in, revolutions per minute and weight on bit, all with a lookback of 30. As output, the model produces one prediction for ECD. The hidden layer size for the LSTM is 32 and it has one layer, the number of layers in the output network is 3, each with a size of 32. Training is done using the Adam optimizer with a learning rate of 0.0001. During training the LSTM is regularized using dropout with a probability of 0.4 and the output network is regularized using L1-regularization with a value of 0.0001. The model is trained for 20 epochs with early stopping. Table C.3 gives an overview of the model parameters.

3.4. Model Evaluation

In the fourth step, the performance of the developed prediction methods are evaluated and compared based on different popular metrics: root mean squared error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE) [64]. While the analysis methods are presented with their strengths and weaknesses based on how well they manage to extract information from the data. The evaluation is done on both the greenhouse data and the oil drilling data to validate the results and identify the best methods for hole cleaning analysis and prediction. For the methods of analysis, one data set were used for greenhouse and one for oil drilling. But the evaluation also considers how the models generalize on measurements from similar processes that are exhibiting other patterns. For example, how well a model trained on one oil drilling session performs on another oil drilling session. Therefore, different scenarios were developed.

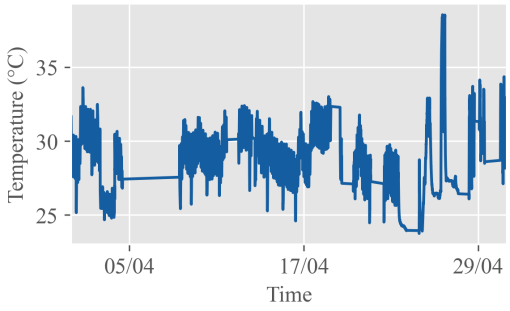
3.4.1. Data for Analysis

Laboratory Scale: Greenhouse

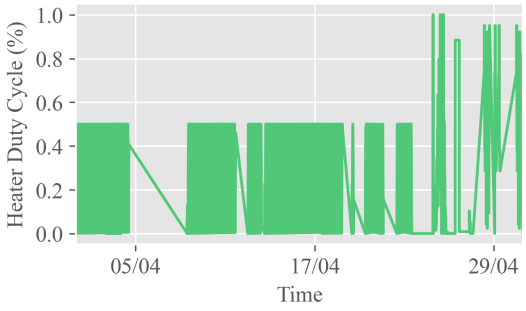
As mentioned, the data used for analysis of greenhouse data is both the measurements from the single sensors and the measurements from the 13 distributed sensors. The positioning of the distributed sensors are shown in table 3.1. The data for the single sensors are collected over a period of 1 month, 01.04.2023-01.05.2023. Plots of the data from the single sensors are shown in figure 3.1 and the temperature measurements for the distributed sensors are shown in figure 3.2.

Table 3.1.: Coordinates of the sensors inside the greenhouse. (25, 25, 0) marks the midpoint of the bottom of the greenhouse, where the plant is placed. The distances from the heater, the inlet fan and the outlet fan to the sensors are shown in the table.

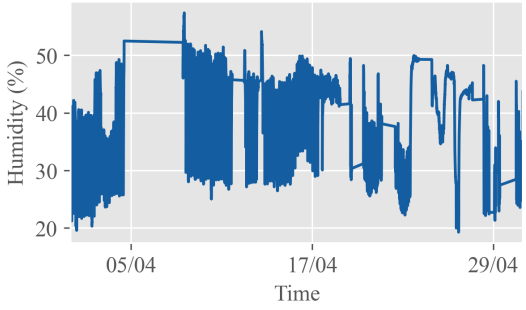
Name	X	Y	Z	Dist. heater	Dist. inlet fan	Dist. outlet fan
Sensor 1	5	24	14	41.19	40.31	51.00
Sensor 2	14	48	0	56.12	67.94	61.61
Sensor 3	40	23	22	31.83	52.01	26.32
Sensor 4	6	48	58	30.89	49.84	78.59
Sensor 5	48	24	0	55.05	70.68	27.86
Sensor 6	48	48	50	32.53	68.00	60.03
Sensor 7	7	42	16	42.06	52.09	60.14
Sensor 8	12	0	34	32.40	16.97	42.94
Sensor 9	20	5	24	33.18	30.15	32.02
Sensor 10	42	24	48	17.15	48.41	42.38
Sensor 11	37	40	11	43.47	64.76	42.17
Sensor 12	120	120	120	151.49	185.14	174.75
Sensor 13	-10	24	14	50.22	41.23	64.62



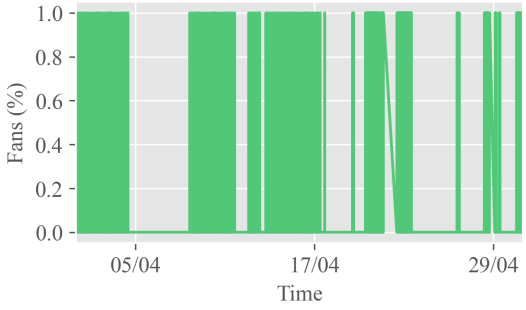
(a) Temperature measurements



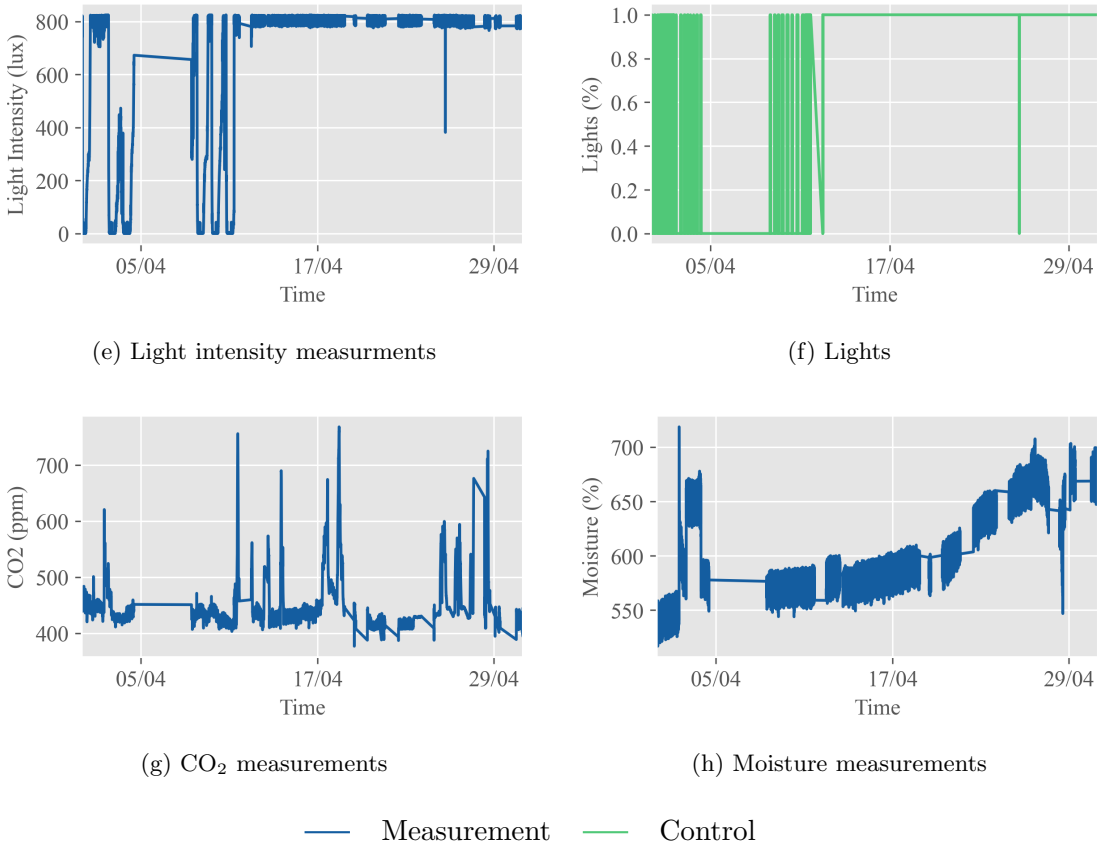
(b) Heater duty cycle



(c) Humidity measurements



(d) Fans



(e) Light intensity measurments

(f) Lights

(g) CO_2 measurements

(h) Moisture measurements

Figure 3.1.: Measurements and controls used when analysing the greenhouse data.

Field Scale: Oil Drilling Hole Cleaning Process

For analysis of the hole cleaning data, the whole data set for one drilling operation in case A are used. The data is shown in figure 3.12.

3.4.2. Predition Evaluation Metrics

The metrics used for evaluating performance of model prediction are: root mean squared error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE).

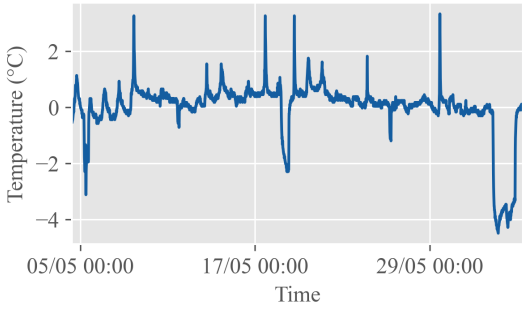
RMSE are calculated as the squared error between actual values and predictions:

$$RMSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \tilde{x}_i)^2. \tag{3.1}$$

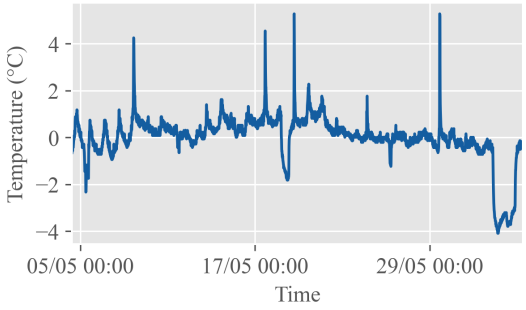
As the error is squared, the RMSE are sensitive to outliers and large errors. Therefore, outlier will affect the RMSE and in many cases show a worse fit than it actually is. Also RMSE has no clear interpretation in terms of the original units of the measurements. This will, in many cases, make it hard to interpret the metric.

MAE are calculated as the mean of the absolute difference between the actual and predicted values:

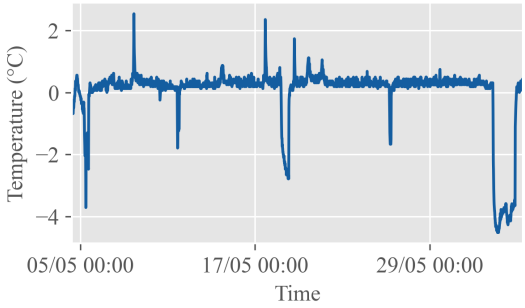
$$MAE = \frac{1}{n}\sum_{i=1}^{n}|x_i - \tilde{x}_i|. \tag{3.2}$$



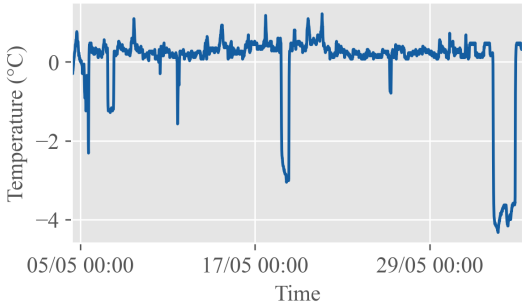
(j) Temperature measurements sensor 1



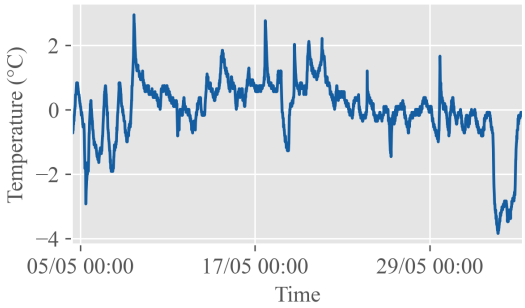
(k) Temperature measurements sensor 2



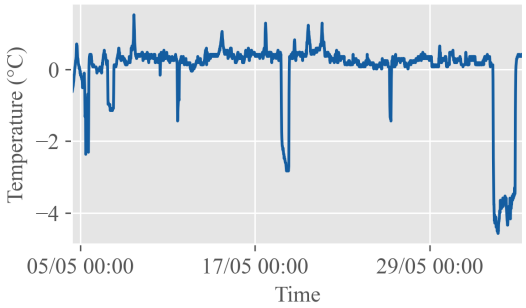
(l) Temperature measurements sensor 3



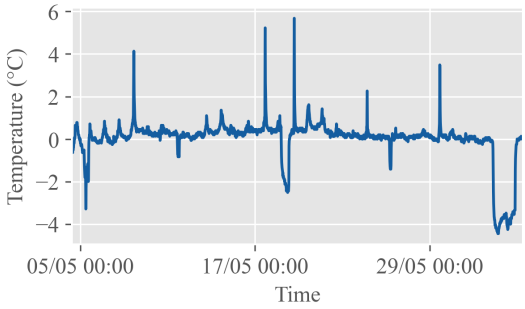
(m) Temperature measurements sensor 4



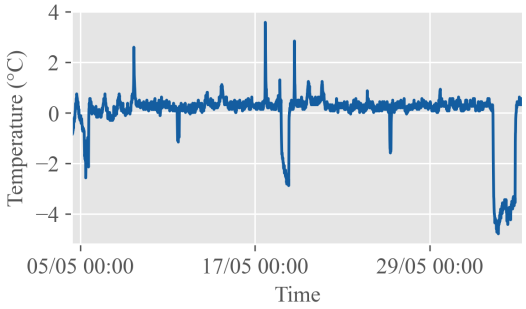
(n) Temperature measurements sensor 5



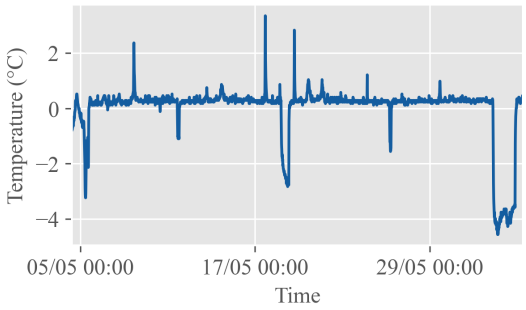
(o) Temperature measurements sensor 6



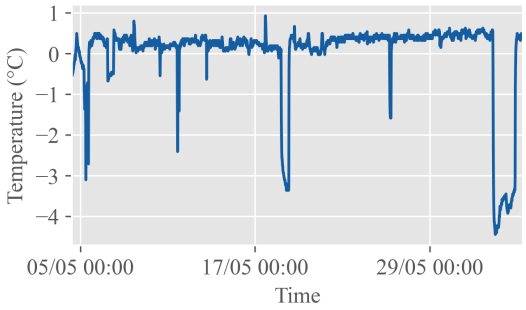
(p) Temperature measurements sensor 7



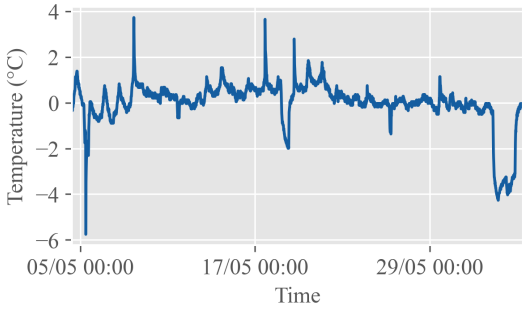
(q) Temperature measurements sensor 8



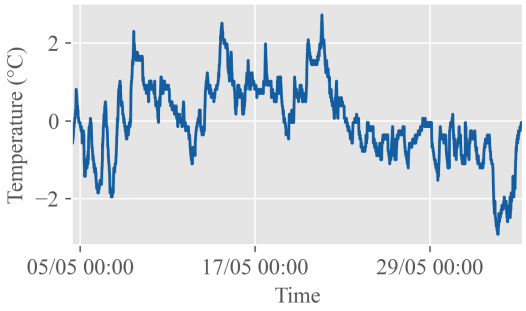
(r) Temperature measurements sensor 9



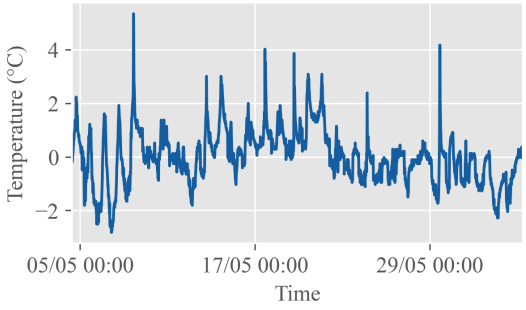
(s) Temperature measurements sensor 10



(t) Temperature measurements sensor 11



(u) Temperature measurements sensor 12



(v) Temperature measurements sensor 13

Figure 3.2.: Temperature measurements for the distributed temperature sensors. Note how each sensor has its distinct set of dynamics and how some of the sensors. The effects of temperature change are also delayed for some of the sensors due to spatial distances.

MAE are less sensitive to outliers as it is the absolute value of the error and not the square. The MAE are easy to interpret, as it gives the average error of the original unit. In contrast to RMSE, it is not sensitive to outlier and gives equal weight to all errors. Consequently, MAE are less sensitive to noise in the actual data or predictions.

MAPE are calculated as mean of the absolute difference between the actual value and the prediction, divided by the actual value, multiplied by 100:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{x_i - \tilde{x}_i}{x_i}\right|. \tag{3.3}$$

MAPE are independent of scale and can be used to compare models across different data sets an scales. It is easy to interpret, as it gives the average error as a percentage of the actual value. This makes the metric scale-free and the result can be easily compared across different data sets or units of measurements. However, MAPE are not defines when the actual value is zero something that affects the metric.

To test how to models generalize different experiments with different combinations of training and test data was conducted both for greenhouse and oil drilling.

3.4.3. Scenarios for Prediction

To evaluate how well the models generalizes on other measurements from similar processes that are exhibiting other patterns, different scenarios are presented for both greenhouse and oil drilling. For the laboratory-scale data, the same training set is used for all scenarios, while for drilling, both training and test set are changed in each scenario.

Laboratory Scale: Greenhouse

Depending on the control inputs, the data from the greenhouse exhibits totally different patterns. First, the heater duty cycle and fans had a cyclical pattern, something that resulted in cyclical patterns in temperature and humidity as well. Second, the fans were shift to a more random pattern, this resulted in high frequency components in the temperature and a more high frequent humidity. Last, both the heater duty cycle and fans were switched to a random pattern, resulting in a non-cyclical pattern.

The first data set with the simple cyclical pattern are split in to a training and test set. The other data sets with more complex patterns are all used as test sets. This creates three scenarios:

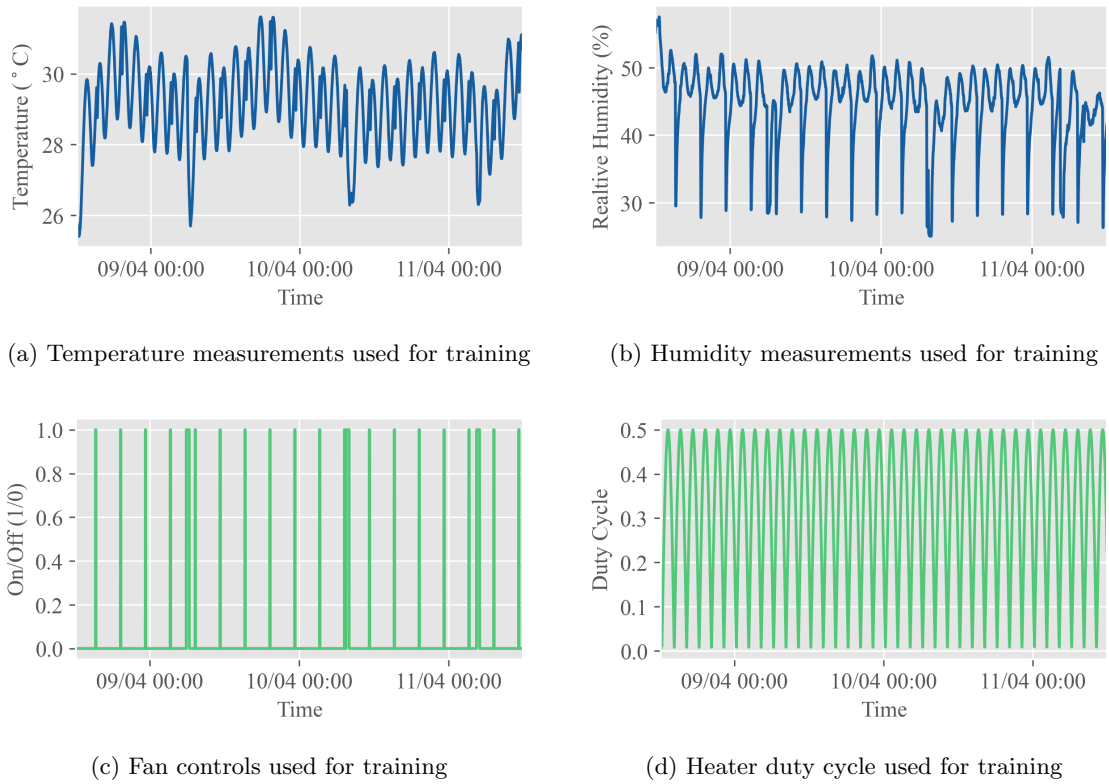
- Scenario 1: Train on simple, cyclical pattern, test on similar pattern.
- Scenario 2: Train on simple, cyclical pattern, test on more complex cyclical pattern with high frequency temperature components and high frequency humidity.
- Scenario 3: Train on simple, cyclical pattern, test on non-cyclical pattern with random heater duty cycle and fan controls.

These three scenarios will test how well the models perform on data that are similar to the one it has seen before, data that is similar, but with different fan controls resulting in high frequency components in temperature and high frequency humidity, and data that are non-cyclical with random controls. This will provide a good overview of model performance and generalization.

The training data for all scenarios are shown in figure 3.3. Statistical properties of the variables are shown in table 3.2.

Table 3.2.: Mean, median, standard deviation, range, max and min of training data for labatory-scale greenhouse data.

	Mean	Median	STD	Range	Max	Min
Temperature	29.16	29.26	1.16	6.20	31.61	25.41
Humidity	44.76	45.85	5.58	32.52	57.54	25.02
Fans	0.04	0.00	0.20	1.00	1.00	0.00
Heater Duty Cycle	0.32	0.35	0.15	0.49	0.50	0.01



(a) Temperature measurements used for training

(b) Humidity measurements used for training

(c) Fan controls used for training

(d) Heater duty cycle used for training

Figure 3.3.: Tranining data used to train the models on greenhouse data. Figure 3.3a and figure 3.3b shows the measurements and figure 3.3c and figure 3.3d shows the corresponding control inputs.

Table 3.3.: Mean, median, standard deviation, range, max, and min for the test set used in scenario 1 for testing models on the laboratory-scale greenhouse data.

	Mean	Median	STD	Range	Max	Min
Temperature	28.43	28.54	1.12	6.92	31.52	24.60
Humidity	42.95	43.63	4.30	23.79	49.9100	26.12
Fans	0.04	0.00	0.20	1.00	1.00	0.00
Heater Duty Cycle	0.32	0.35	0.15	0.49	0.50	0.01

Table 3.4.: Mean, median, standard deviation, range, max, and min for test set used in scenario 2 for testing models on the laboratory-scale greenhouse data.

	Mean	Median	STD	Range	Max	Min
Temperature	27.73	27.78	0.98	4.21	29.77	25.56
Humidity	28.79	28.75	3.40	16.33	38.43	22.10
Fans	0.24	0.00	0.42	1.00	1.00	0.00
Heater Duty Cycle	0.32	0.36	0.15	0.48	0.50	0.02

The first test set, test set 1, have a similar pattern to the training data, but with different trend, mean and standard deviation. The statistical properties of the test data is shown in table 3.3. Here you can see how the standard deviation, especially for the humidity, is different from the training set. The data is visualized in figure 3.4

The next test set, for scenario 2, still obtains the same cycles in temperature seen in both in the training set (figure 3.3) and test set 1 (figure 3.4), but with the addition of high frequency components. The humidity in this test set are more high frequent and has a signature that differs to the one in the two other data sets. This is a results of high fan activity seen both in figure 3.5c, by looking at the fan controls, and in table 3.4, as a significant rise in mean value for fans. The data is visualised in figure 3.5 and its statistical properties are summarized in table 3.4.

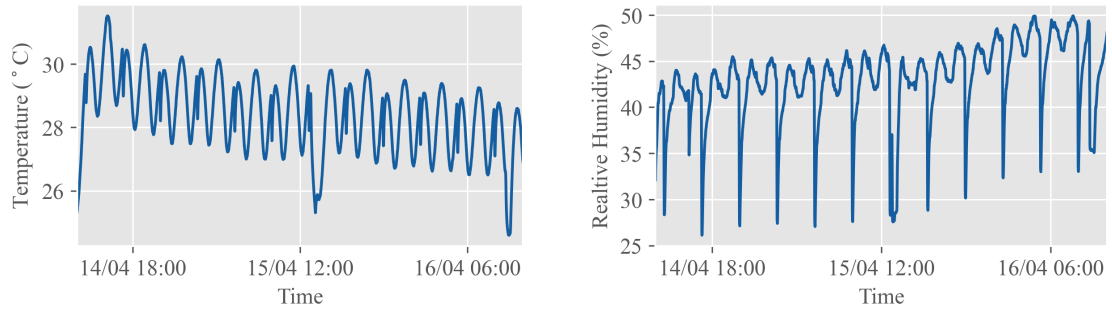
The test set for scenario 3 is the most challenging test set for the model. This data set exhibits a pattern that is non-cyclic both in temperature and humidity. This is caused by both the heater and fan working randomly, see figure 3.6d and figure 3.6c. The hole data set and its statistical properties are shown in figure 3.6 and table 3.5, respectively.

Field Scale: Oil Drilling Hole Cleaning Process

Two drilling cases are available for the oil drilling data. These are different drilling operations, but from nearby locations. In the first drilling case, case A, the pattern of the ECD changes significantly during drilling as the operators changes the drilling parameters. In the second case, case B, the data has somewhat the same pattern throughout

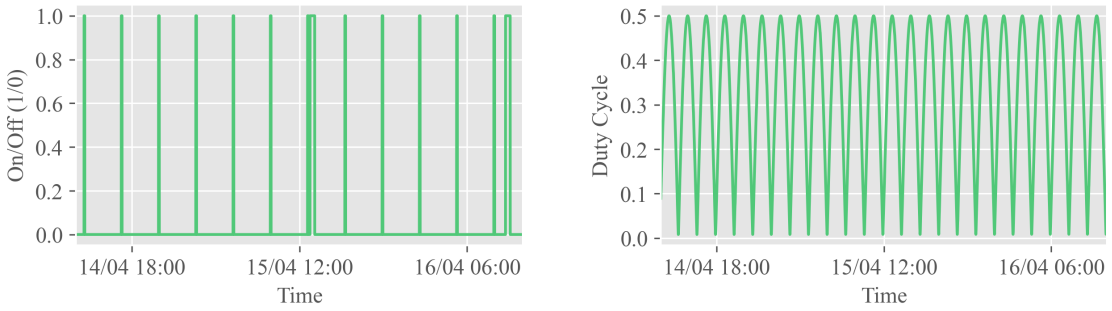
Table 3.5.: Mean, median, standard deviation, range, max, and min for test set used in scenario 3 for testing models on the laboratory-scale greenhouse data.

	Mean	Median	STD	Range	Max	Min
Temperature	30.13	30.23	1.76	6.99	33.72	26.73
Humidity	28.48	27.32	4.93	21.98	44.12	22.14
Fans	0.35	0.00	0.49	1.00	1.00	0.00
Heater Duty Cycle	0.48	0.41	0.31	0.93	0.95	0.02



(a) Temperature measurements used for test in scenario 1

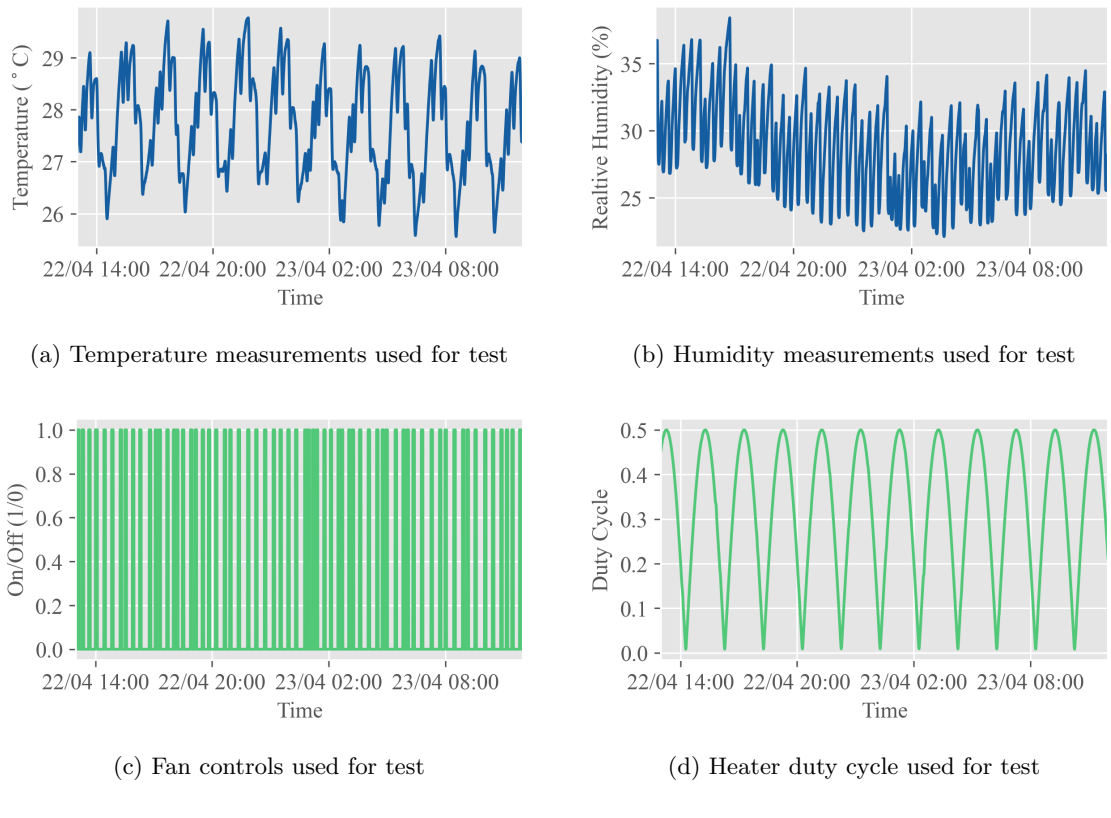
(b) Humidity measurements used for test in scenario 1



(c) Fan controls used for test in scenario 1

(d) Heater duty cycle used for test in scenario 1

Figure 3.4.: Test data for prediction in greenhouse scenario 1. Figure 3.4a and figure 3.4b shows the measurements and figure 3.4c and figure 3.4d shows the corresponding control inputs. This data has mostly the same type of pattern as the training data in figure 3.3.



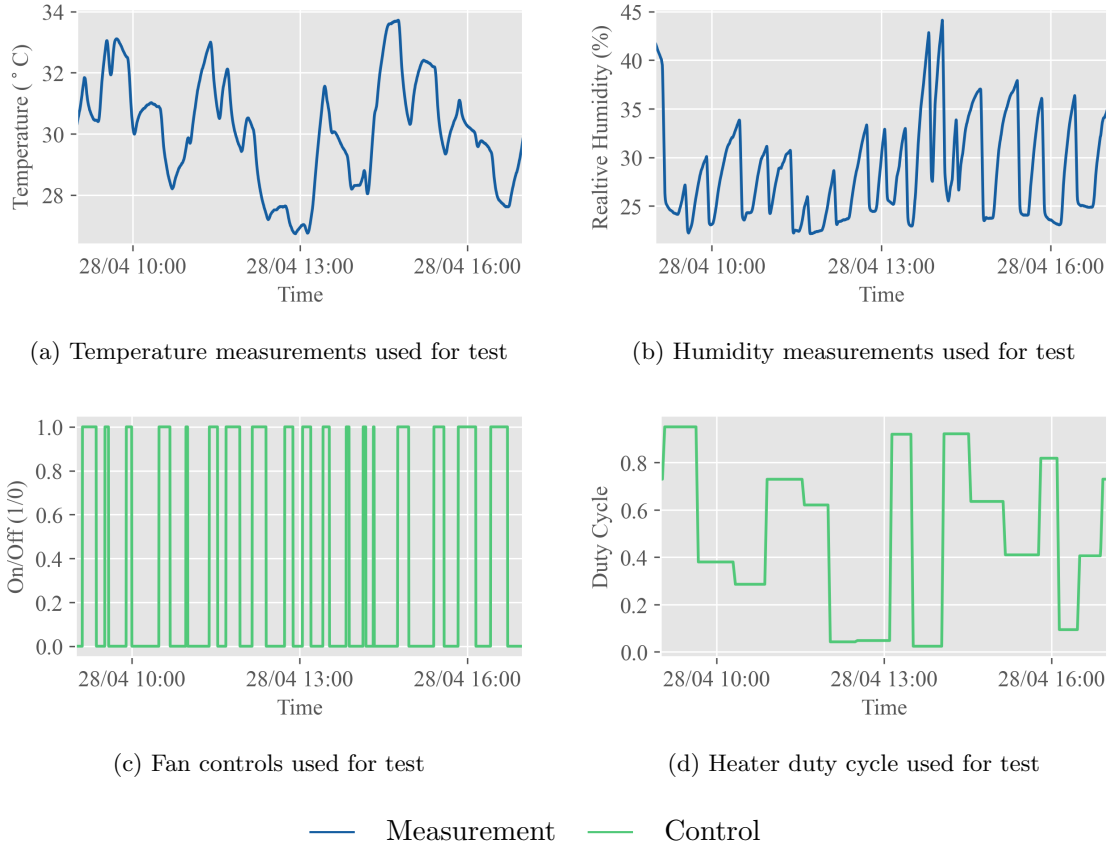
(a) Temperature measurements used for test

(b) Humidity measurements used for test

(c) Fan controls used for test

(d) Heater duty cycle used for test

Figure 3.5.: Test data for prediction in greenhouse scenario 2. Figure 3.5a and figure 3.5b shows the measurements and figure 3.5c and figure 3.5d shows the corresponding control inputs. Note how this data differs quite significantly from the training data in figure 3.3. This is due to an increase in the on/off frequency of the fans.



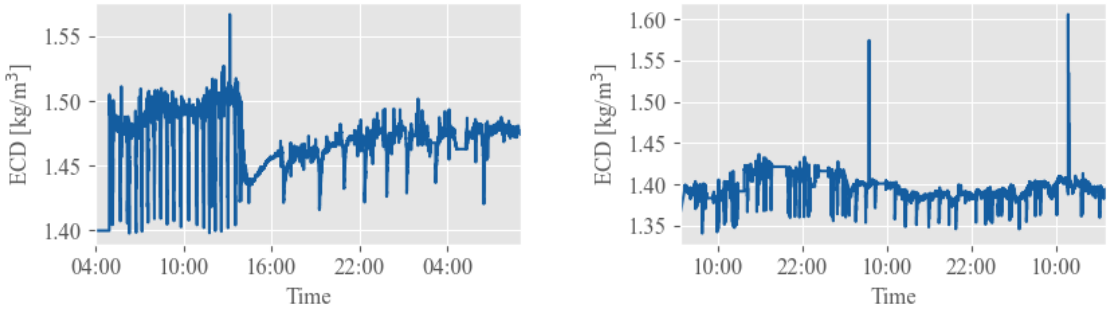
(a) Temperature measurements used for test

(b) Humidity measurements used for test

(c) Fan controls used for test

(d) Heater duty cycle used for test

Figure 3.6.: Test data for prediction in greenhouse scenario 3. Figure 3.6a and figure 3.6b shows the measurements and figure 3.6c and figure 3.6d shows the corresponding control inputs. Note how this data differs quite significantly both other data sets with random heater and fan controls.



(a) Case A

(b) Case B

Measurement

Figure 3.7.: ECD in the drilling case A and drilling case B.

the drilling.

The first data set are split into training and test, in two different ways. In the first split, the training data includes both the initial pattern of the ECD as well as some of the pattern that emerged after change of drilling parameters, while the test data includes only the pattern after change of parameters. In the second split, the training data includes only the initial pattern, while the drilling data includes only the pattern from after the change of parameters. Lastly, since the two data sets are from different drilling operations in nearby locations, the first case is used as training set and the last is used for testing.

This creates the following three scenarios:

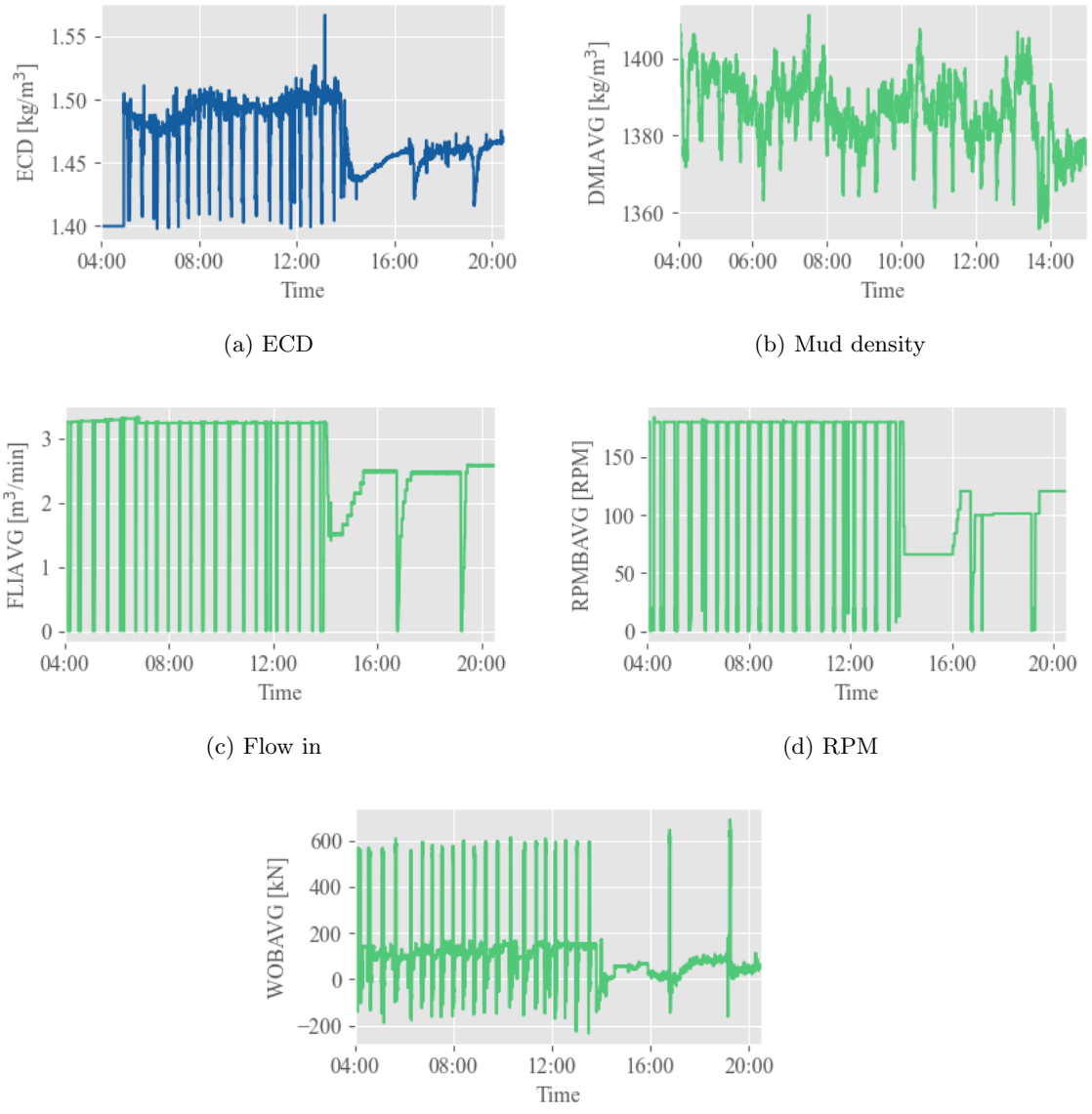
- Scenario 1: Case A are split into two data sets. The model are trained mainly on the first pattern, but also some on the second. The performance is evaluated at the second pattern.
- Scenario 2: Case A are split into two data sets. The model are trained only on the first pattern. The performance is evaluated at the second pattern.
- Scenario 3: Case A and case B forms training and test set, respectively. The model is trained on case A and its performance is evaluated on case B.

These three scenarios will show how well the model will perform both on seen and unseen patterns. The first scenario presents whether seeing only a small part of the pattern makes the model perform better. While the second scenario shows how the model performs on data with different patterns from the same drilling session, while scenario three shows how models can generalize across drilling operations.

The data sets used for the field scale models are based on two drilling cases, case A and case B. The ECD of these cases are shown in figure 3.7.

The training and test set pair for scenario 1 is shown in figure 3.8 and figure 3.9, respectively. Here the session is split so that the training set includes mainly the initial dynamics of the drilling and only a small part of the shifted dynamics. The statistical properties of the training set for this scenario is shown in table 3.6 and the test set are shown in table 3.7.

For scenario 2, the pair of training and test sets are shown in figure 3.10 and figure 3.11, respectively. Here the data is split into a train set with the initial ECD pattern, and a



(a) ECD

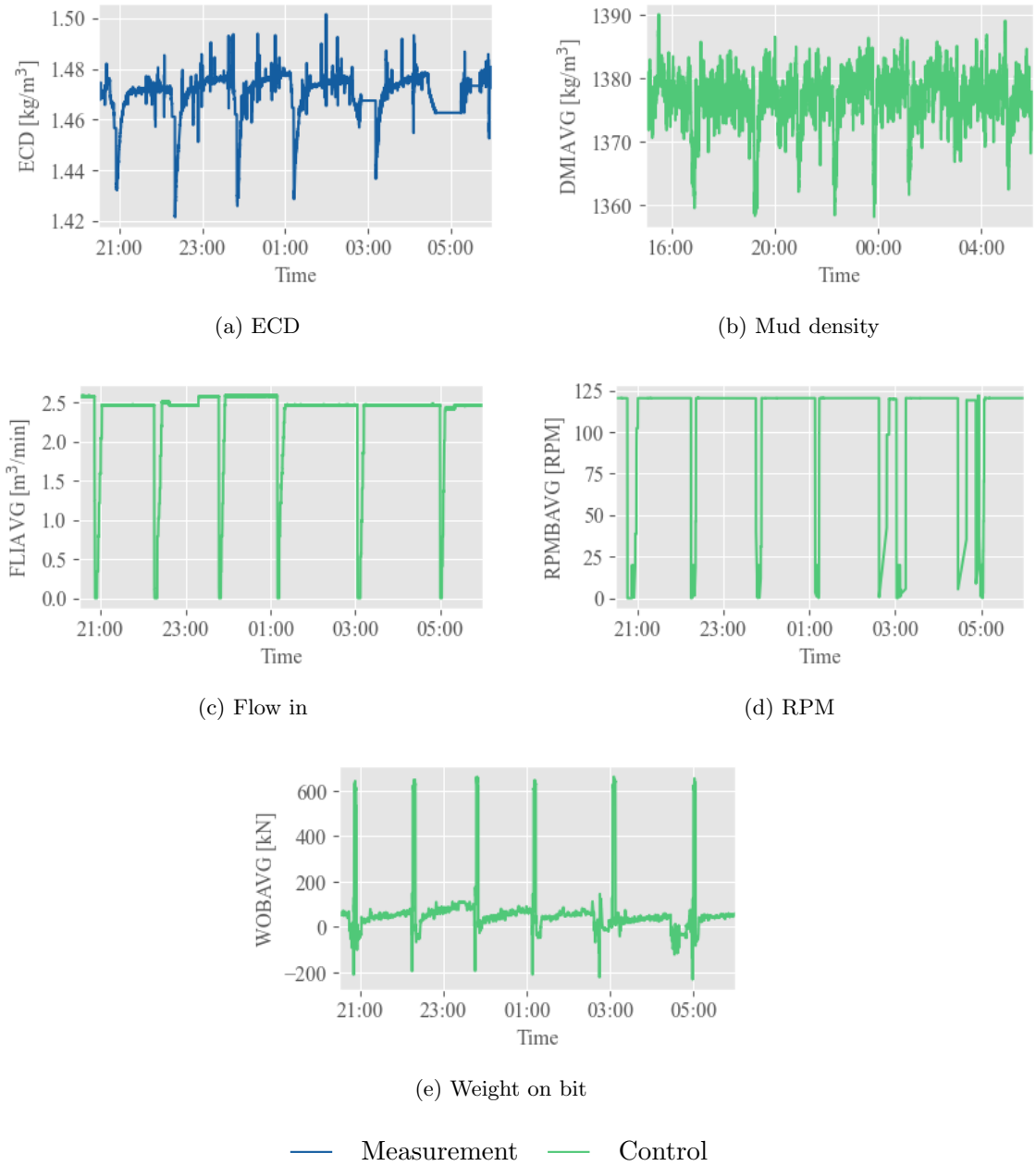
(b) Mud density

(c) Flow in

(d) RPM

(e) Weight on bit

Figure 3.8.: ECD measurements and corresponding controls of training set for scenario 1 on case A. Note how the controls changes significantly causing a change in the ECD.



(a) ECD

(b) Mud density

(c) Flow in

(d) RPM

(e) Weight on bit

Figure 3.9.: ECD measurements and corresponding controls for test set for first scenario on case A.

Table 3.6.: Mean, median, standard deviation, range, max, and min for the train set for scenario 1 for the oil drilling data.

	Mean	Median	STD	Range	Max	Min
ASMECD1_T	1.47	1.46	0.03	0.17	1.57	1.40
RPMBAVG	124.49	164.04	62.83	184.04	184.05	0.01
FLIAVG	2.61	3.24	0.93	3.33	3.33	0.00
WOBAVG	114.78	93.52	139.87	923.97	690.57	-233.40
DMIAVG	1382.98	1381.41	9.08	55.47	1411.20	1355.73

Table 3.7.: Mean, median, standard deviation, range, max, and min for the test set for scenario 1 for the oil drilling data.

	Mean	Median	STD	Range	Max	Min
ASMECD1_T	1.47	1.47	0.01	0.08	1.5016	1.42
RPMBAVG	106.13	120.41	36.31	121.99	122.00	0.01
FLIAVG	2.35	2.47	0.53	2.59	2.59	0.00
WOBAVG	61.05	48.61	110.46	893.08	662.41	-230.67
DMIAVG	1377.11	1378.05	4.03	31.01	1389.14	1358.13

test set with the new pattern emerging after change of parameters. The statistics of the training set are presented in table 3.8, while the statistics for the test set are presented in table 3.9.

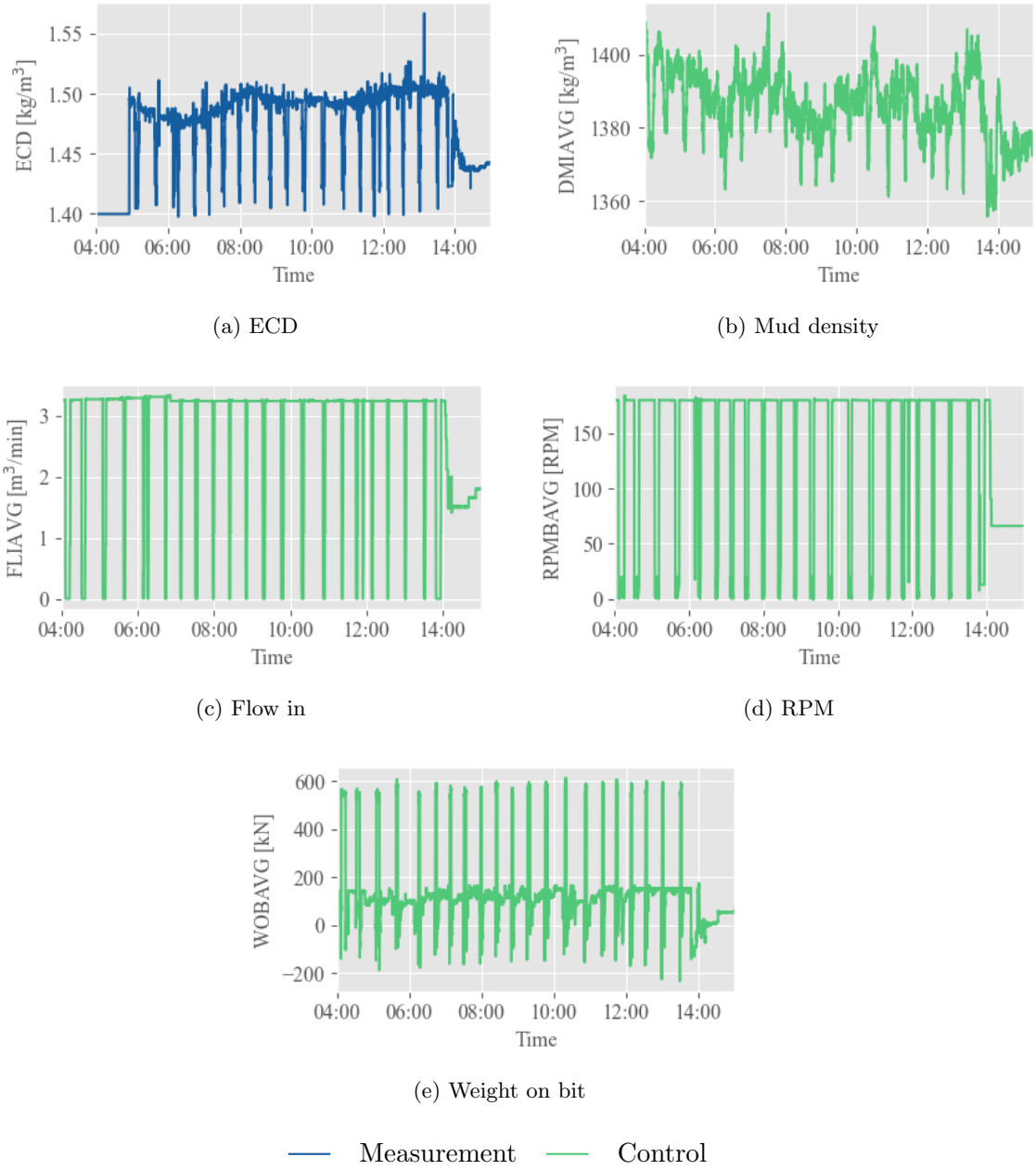
In scenario 3, all data from case A is used for training, while all data for case B is used for testing. The training set, case A, are shown in figure 3.12, while the set, case B, are shown in figure 3.13. The two drilling cases differs in pattern. Case A has a change in pattern about half way through the operation, while case B has the same pattern throughout the operation. The statistics of these data sets are presented in table 3.10 and table 3.11, respectively.

Table 3.8.: Mean, median, standard deviation, range, max, and min for training data for oil drilling scenario 2.

	Mean	Median	STD	Range	Max	Min
ASMECD1_T	1.47	1.49	0.03	0.17	1.57	1.40
RPMBAVG	139.60	179.90	68.45	184.04	184.05	0.01
FLIAVG	2.75	3.24	1.04	3.33	3.33	0.0000
WOBAVG	140.20	116.30	152.70	847.82	614.42	-233.40
DMIAVG	1385.83	1386.46	9.25	55.47	1411.20	1355.73

Table 3.9.: Mean, median, standard deviation, range, max, and min for test data for oil drilling scenario 2.

	Mean	Median	STD	Range	Max	Min
ASMECD1_T	1.46	1.47	0.01	0.09	1.50	1.42
RPMBAVG	101.17	120.39	34.18	121.99	122.00	0.01
FLIAVG	2.34	2.47	0.51	2.59	2.59	0.00
WOBAVG	60.32	49.43	101.92	921.24	690.57	-230.67
DMIAVG	1376.97	1377.84	4.12	32.04	1390.17	1358.13



(a) ECD

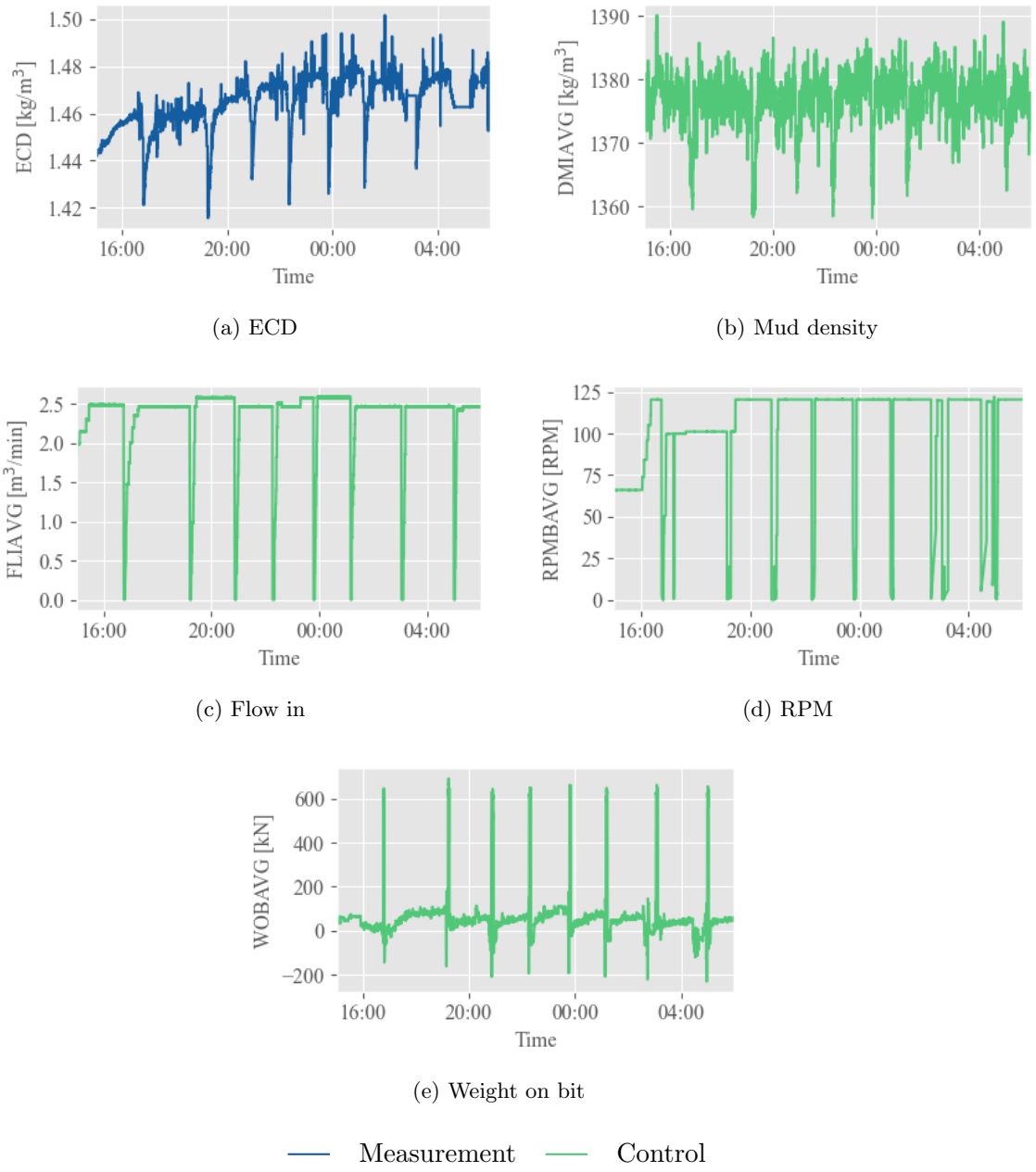
(b) Mud density

(c) Flow in

(d) RPM

(e) Weight on bit

Figure 3.10.: ECD measurements and corresponding controls of training set for scenario 2 on case A. Here the pattern that changes as the drilling parameters change are not included in the data set.



(a) ECD

(b) Mud density

(c) Flow in

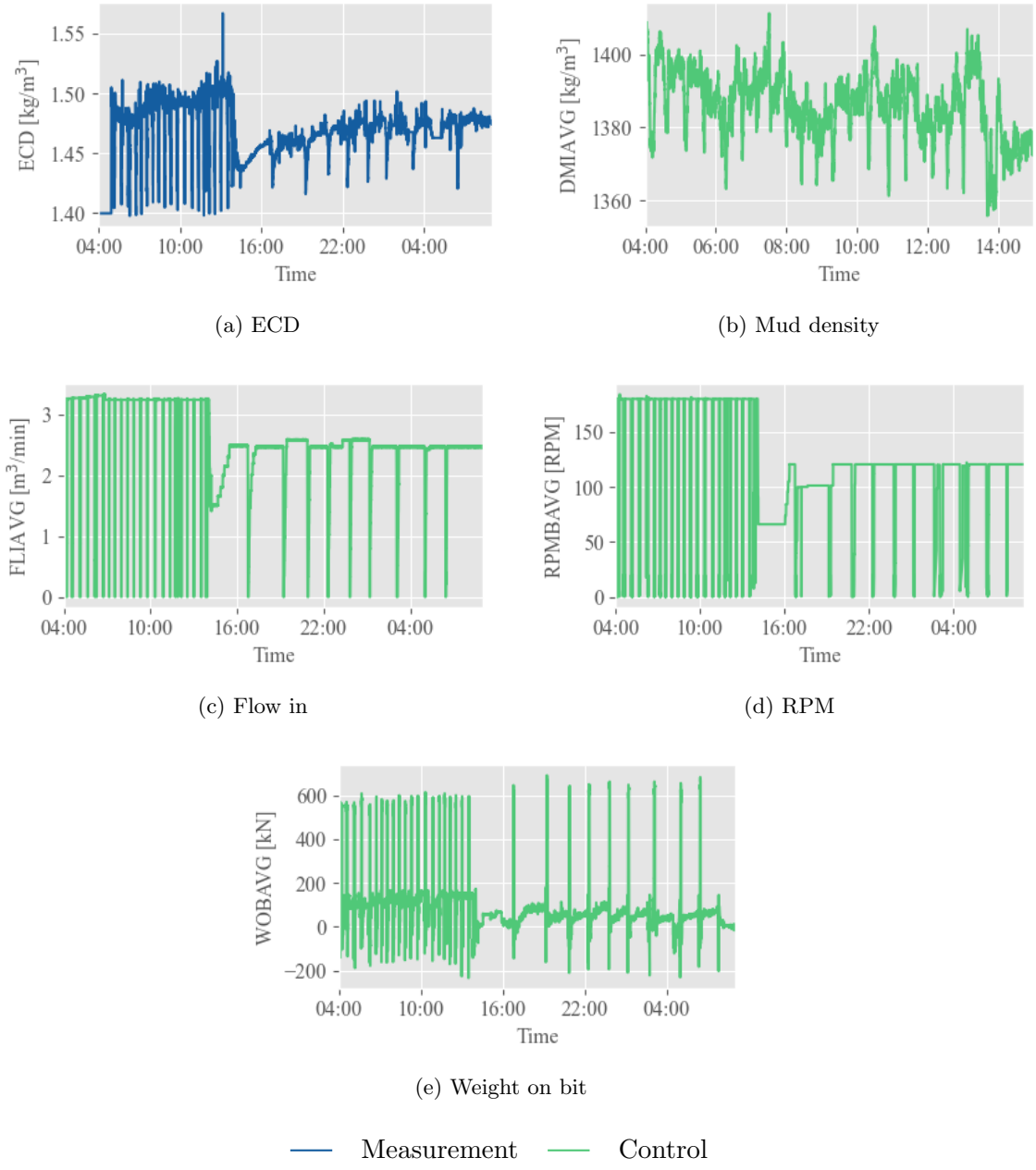
(d) RPM

(e) Weight on bit

Figure 3.11.: ECD measurements and corresponding controls for test set for scenario 2. The pattern in this data differs significantly from the one in figure 3.10.

Table 3.10.: Mean, median, standard deviation, range, max, and min for training data for oil drilling scenario 3.

	Mean	Median	STD	Range	Max	Min
ASMECD1_T	1.47	1.47	0.02	0.17	1.57	1.40
RPMBAVG	116.84	120.41	52.88	184.04	184.05	0.01
FLIAVG	2.50	2.47	0.77	3.33	3.33	0.0000
WOBAVG	88.96	61.95	128.33	923.97	690.57	-233.40
DMIAVG	1380.59	1378.80	7.77	55.47	1411.20	1355.73



(a) ECD

(b) Mud density

(c) Flow in

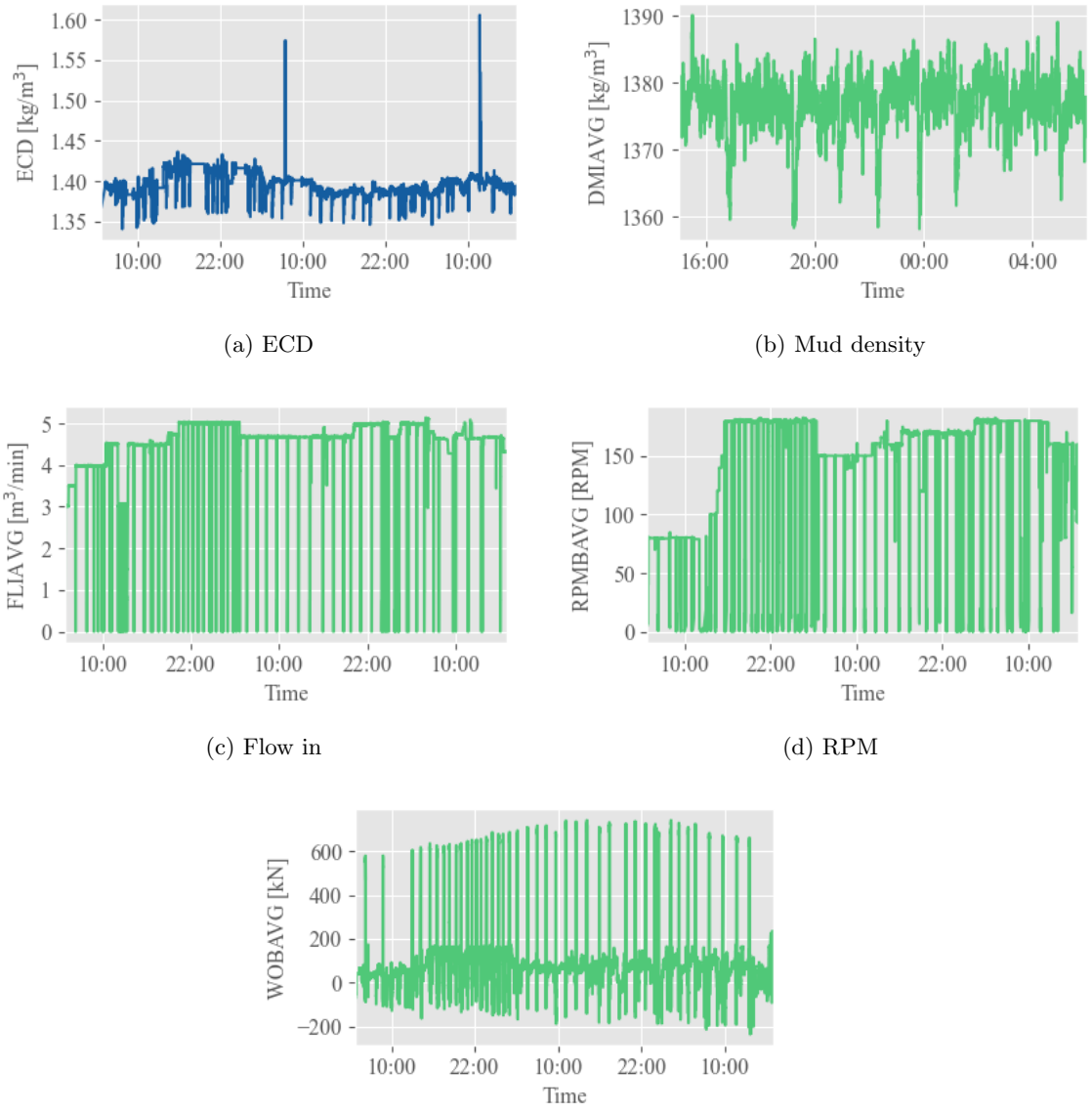
(d) RPM

(e) Weight on bit

Figure 3.12.: ECD measurements and corresponding controls of training set for scenario 1 on case A. Note how the controls changes significantly causing a change in the ECD.

Table 3.11.: Mean, median, standard deviation, range, max, and min for test data for oil drilling scenario 3.

	Mean	Median	STD	Range	Max	Min
ASMECD1_T	1.46	1.47	0.01	0.09	1.50	1.42
RPMBAVG	101.17	120.39	34.18	121.99	122.00	0.01
FLIAVG	2.34	2.47	0.51	2.59	2.59	0.00
WOBAVG	60.32	49.43	101.92	921.24	690.57	-230.67
DMIAVG	1376.97	1377.84	4.12	32.04	1390.17	1358.13



(a) ECD

(b) Mud density

(c) Flow in

(d) RPM

(e) Weight on bit

Figure 3.13.: ECD measurements and corresponding controls for test set in scenario 3 on oil drilling data. For this scenario the whole data set for case B are used.

4. Results and Discussions

The results section of this thesis presents the outcomes of data collection, data preprocessing, model architecture, and model evaluation steps from chapter 3. In the first part, the methods of analysis, PCA, EMD, EEMD, and FFT, are applied to the data and the results are discussed. In the second part, the methods for prediction, FFNN for prediction of derivative, LSTM for prediction of next value, and LSTM for prediction of derivative are applied to both greenhouse and drilling data. Their performance is evaluated using different metrics, MSE, MAE, and MAPE. The models are also validated on different data sets to investigate the generalization ability of the models. The chapter aims to answer the research questions and objectives of the thesis.

4.1. Exploratory Data Analysis

In this section the results of applying the methods of analysis: PCA, EMD, EEMD, and FFT to the data. The purpose of applying these methods is to explore the patterns and connections in the data and to extract useful connections between controls and measurements. The methods are first applied to greenhouse data before the best-performing ones are applied to the drilling data.

4.1.1. Laboratory Scale: Greenhouse

Dashboard for Real-Time Data Visualization and Exploration

For data analysis, it is often useful to have a dashboard showing real-time data. Such a dashboard makes it possible to quickly explore the data and do initial analysis before digging deeper into the data. As there are many different sensors in the greenhouse, it is, in this case, also useful with an interactive 3D plot showing the sensor's locations as well as the locations of the plant, heater and fans.

A screenshot of the dashboard is in figure 4.1. The dashboard was created using the Dash library from Plotly [25]. This is a library for creating deployable dashboards in Python,

PCA

PCA was performed on the single sensors and controls to investigate their interactions. Figure 4.2 shows the loadings for the first two PCs. These PCs explain $\approx 48\%$ of the total variance in the data.

Naturally, as the temperature is proportional to the heater duty cycles and the light intensity is proportional to the lights, these measurements and controls are grouped, pairwise. The temperature will increase if the heater duty cycle increase and the light intensity will increase if the lights are on.

In both PC1 and PC2, the humidity is negatively correlated with temperature and the heater duty cycle. As explained in the theory section (chapter 2) there is an inverse relationship between temperature and humidity. The results from the PCA confirm this.

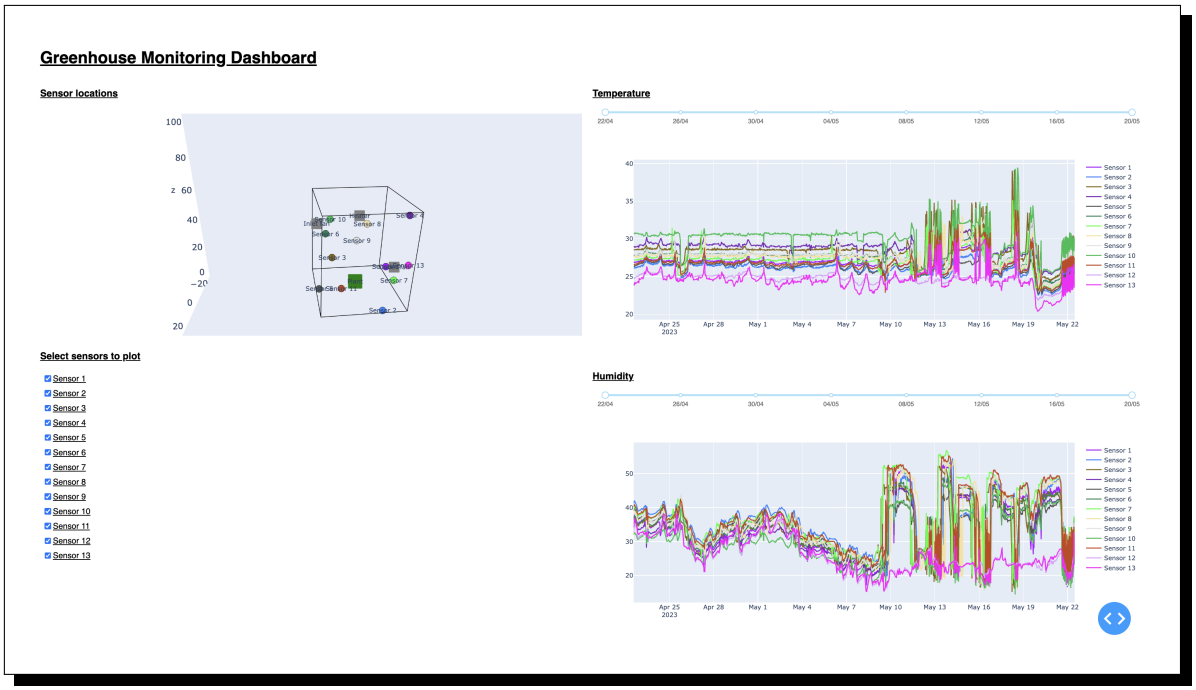


Figure 4.1.: Screenshot of greenhouse dashboard.

The PCA also suggest that there is some correlation between the fans and the heater duty cycle and temperature. Looking at figure 3.1, there is a clear similarity between the pattern in the heater duty cycle and the fans.

The CO_2 concentration seems to correlate negatively with the humidity and positively with the fans. As the CO_2 level is proportional with the fans, it is the air blown into the greenhouse that mainly contributes to the CO_2 level in the greenhouse, indicating that human interaction is an important factor for CO_2 level. Also, as the humidity and moisture are negatively correlated with the fans, the humidity level in the office can be said to generally be lower than the one inside the greenhouse.

The aim of applying PCA to the greenhouse measurements and controls was to investigate the interactions between the sensor measurements and controls. The results show how PCA manages to find both negative and positive correlations between the measurements and controls that were presented in the theory section. It finds a correlation between, temperature and heater duty cycle, light intensity and lighting, and CO_2 concentration and fans. Also, it finds negative correlations between humidity and temperature, and humidity and fans. This resulted in most of the theoretical dynamics presented in the system being extracted from the data using this PCA approach.

On the other hand, according to the theory, moisture and temperature should not be that negatively correlated. This may be a result of the simplified dynamics of this laboratory-scale greenhouse, compared to real-life greenhouses. It is also noteworthy that many of the dynamics and relationships presented in the theory section are simple and could be identified by just looking at the data.

Within and outside the greenhouse, there are also 13 distributed sensors that measure temperature and humidity. These sensors are placed at different distances from the heater and fans. Therefore, it is expected to see some dynamics in the data that describe the relationship between the different sensors and the heater and the fans. PCA was applied to investigate this relationship.

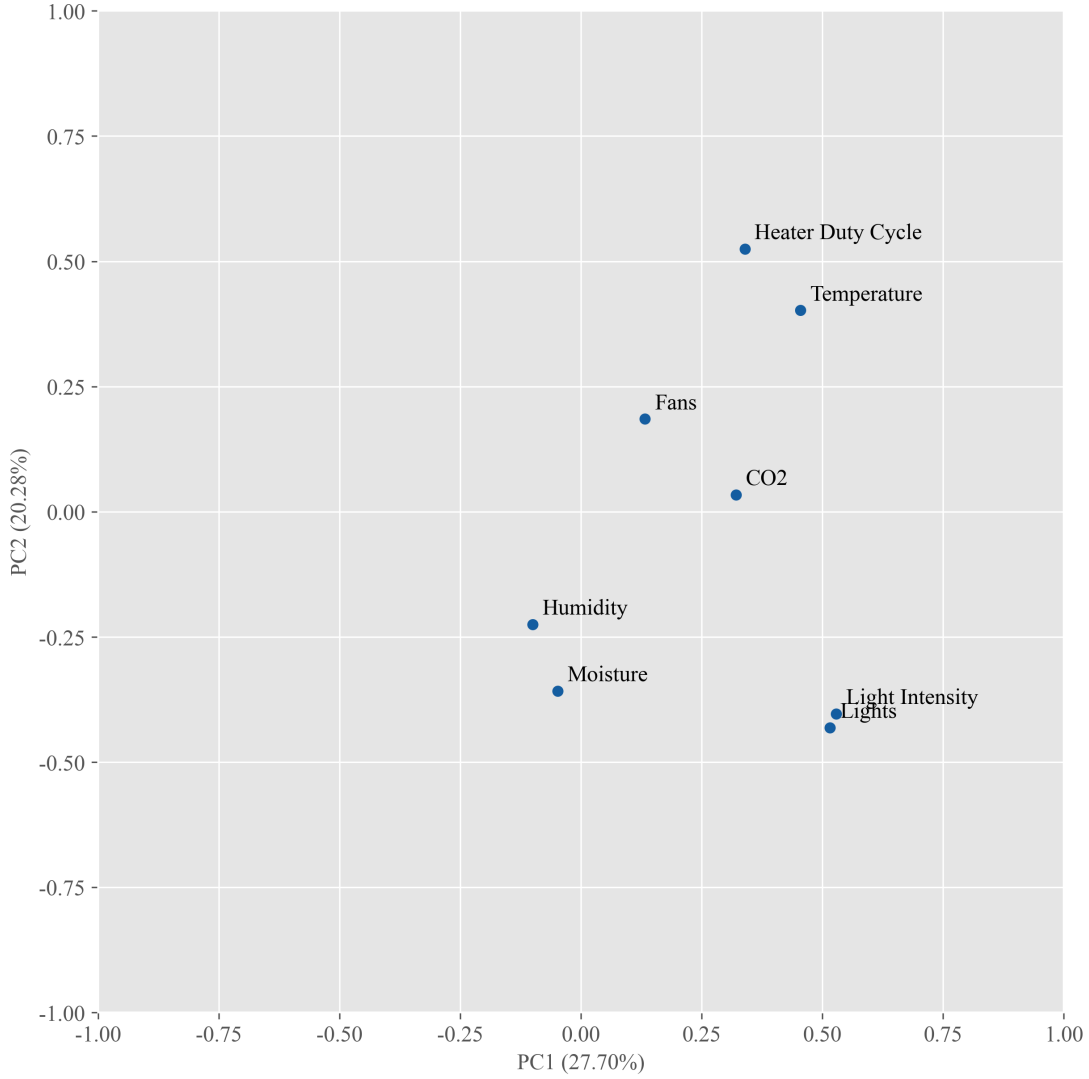


Figure 4.2.: Scatter plot of PCA loadings for PC1 and PC2 for sensors and controls in the greenhouse.

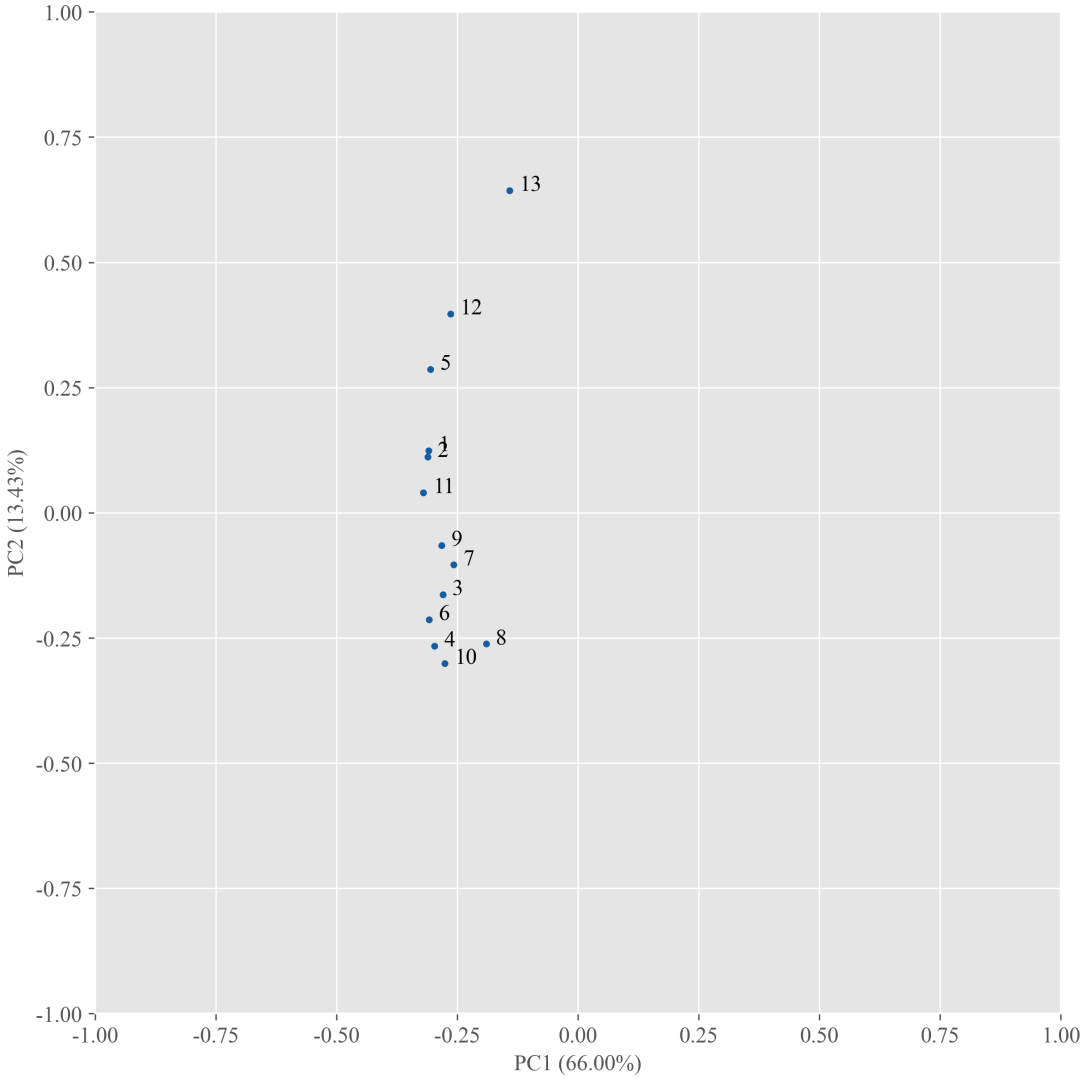


Figure 4.3.: Scatter plot of PCA loadings from the temperature data from the 13 distributed sensors in the greenhouse. PC2 seems to capture the distance from the sensor to the heater. The numbers indicates the sensor number.

Figure 4.3 shows the first and second principal components of the data from the different humidity and temperature sensors distributed in the greenhouse. PC1 and PC2 describe 66% and 13.43% of the total variance in the data, respectively.

In PC1 all sensors correlate. As all sensors measure the same variable this is not surprising. If the temperature increase in the location of one sensor, it will increase at the location of the other sensors as well.

But in PC2 it looks more interesting. The different sensors seem to be arranged in rising order, from the lowest PC2 value for sensor 10 to the highest PC2 value for sensor 13. Plotting this against the distance from each sensor to the heater yields the plot in figure 4.4. Here there is a clear, for most sensors, linear relationship between their distance to the heater and their PC2 value. Table 3.1 provides an overview of the sensor locations and their distance from the heater.

Compared to the earlier results, where PCA was used to investigate relationships

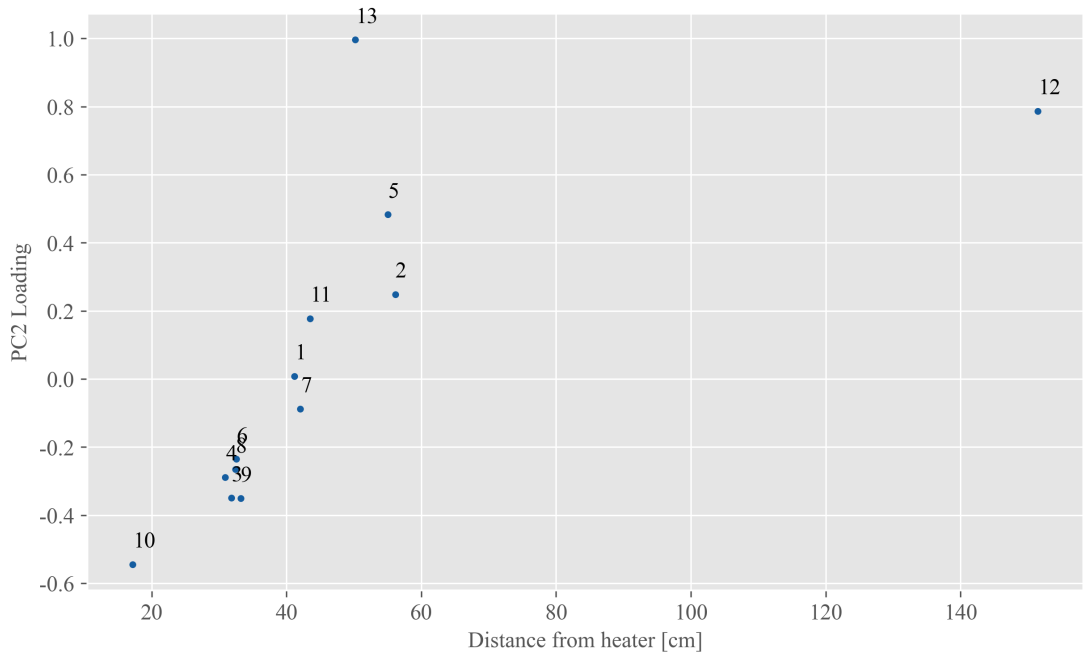


Figure 4.4.: PC2 loading for sensors against their respective Euclidean distance to the heater. There is a clear trend showing that PC2 captures mostly their distance from the heater, with lower PC2 loading meaning a smaller distance to the heater. The number indicates the sensor number.

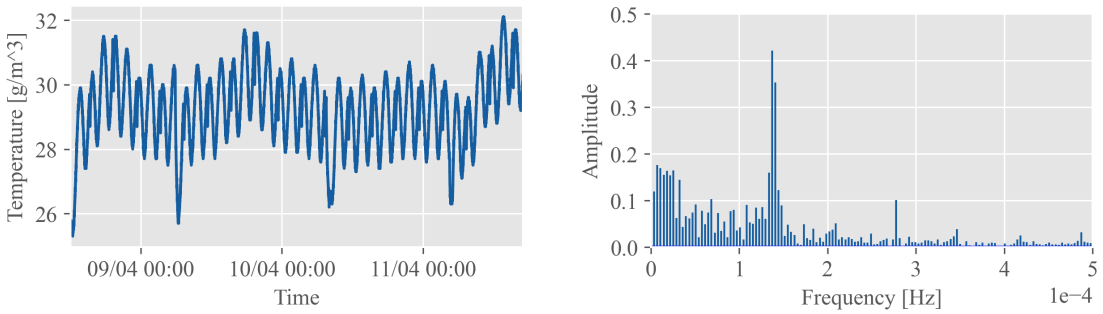
between controls and measurements, these results show how PCA can be used to explore heat transfer dynamics. PCA manages to unveil the distance from the sensors to the heater through PC2. These results would be challenging to find by just looking at the data. On the other hand, this way of interpreting PC2 is hard to find without knowledge about sensor placement.

FFT

To determine the dominant frequencies in the data, the Fourier transform was applied to the temperature and moisture measurements from the greenhouse. As discussed in section 2.1.2, Fourier transform requires the signal to be stationary over time for a correct transformation. However, the data from the greenhouse system, especially the moisture measurements, is likely to be non-stationary due to the non-linear dynamics and the random and cyclic control inputs. Therefore, the results of the FFT analysis should be interpreted with caution.

Figure 4.5 shows the temperature data with its corresponding FFT. The temperature data exhibits a clear periodic pattern with a frequency of 0.5 cycles per hour, which corresponds to the heater duty cycle of 2 hours. The temperature data also shows some fluctuations due to the effect of the fans, which are turned on and off at irregular intervals.

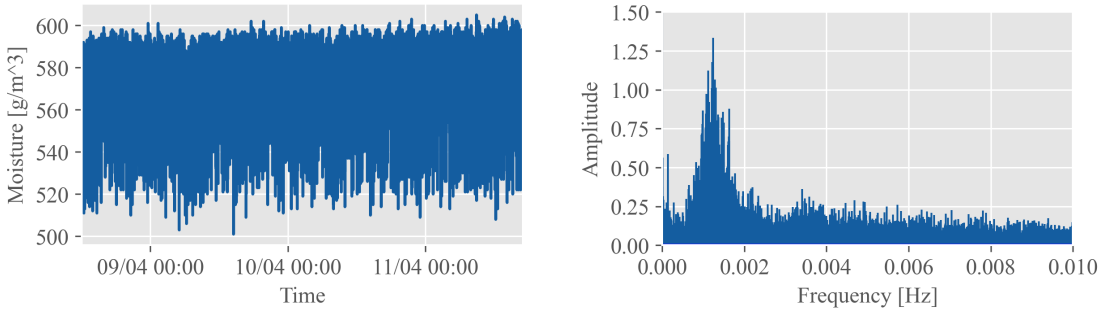
Figure 4.5b shows the magnitude spectrum of the temperature data obtained by taking the absolute value of the FFT output. The magnitude spectrum indicates the relative strength of each frequency component in the signal. The spectrum has a prominent peak at 0.00015 Hz, which corresponds to a period of 2 hours, confirming the visual



(a) Temperature measurments from greenhouse.

(b) FFT of temperature measurements from greenhouse.

Figure 4.5.: Temperature measurements and its corresponding FFT from greenhouse data. Note how the time domain measurements look noisy with no clear patterns and how the energy is spread in many different frequencies.



(a) Moisture measurments from greenhouse.

(b) FFT of moisture measurements from greenhouse.

Figure 4.6.: Moisture measurements and its corresponding FFT from greenhouse data. Note how the time domain measurements look noisy with no clear patterns and how the energy is spread in many different frequencies.

observation of the temperature data. The spectrum also has some smaller peaks at higher frequencies, which may represent some noise or other effects in the signal.

The same procedure was applied to the moisture measurements from the greenhouse, shown in figure 4.6. The moisture data is more noisy and less periodic than the temperature data, making it harder to identify any dominant frequencies by visual inspection.

Figure 4.6b shows the frequency spectrum of the moisture data obtained by taking the absolute value of the FFT output. The spectrum has a broad peak around 0.0012 Hz, which corresponds to a period of about 13 minutes. This frequency does not match any known control input or physical phenomenon in the greenhouse system and may be an artefact of noise or measurement error. The spectrum also has a smaller peak at 0.00013 Hz, which corresponds to a period of about 2 hours. This frequency matches the heater duty cycle, suggesting that there is some influence of temperature on moisture.

These results demonstrate that FFT analysis can reveal some information about the frequency content of the signals from the greenhouse system, but it also has some limitations due to noise and non-stationarity. A more advanced method or model may be needed to capture the complexity and variability of the greenhouse system.

EMD and FFT

To extract the periodic information from the moisture measurements, which appeared to be noisy and complex, EMD was applied to the data. Figure 4.6 shows the moisture measurement and the corresponding FFT. The moisture signal looks rather noisy and it is difficult to discern any patterns or dominant frequencies by visual inspection. The FFT of the signal, from the last section, shows that the energy is spread over a wide range of frequencies, with a slight peak at about 0.00125 Hz, corresponding to a period of about 13 minutes.

Figure 4.7 shows some selected IMFs and their FFTs obtained by applying EMD to the moisture signal. Most of the IMFs are mainly noise, but some contain useful information. For example, IMF 11 and its FFT, shown in figure 4.7a and figure 4.7b, respectively, reveal a frequency component with a period of 2 hours. This matches the heater duty cycle, which is the main driver for temperature change in the greenhouse. As the theory suggests, an increase in the temperature has some effect on the moisture.

Another example is IMF 12 and its FFT, shown in figure 4.7c and figure 4.7d, respectively. They show a frequency component with a period of about 4 hours. This matches the frequency of when the fans are turned on for a brief moment and air is blown through the greenhouse. As the temperature inside the greenhouse is different from the one in the outside office, air blowing through the greenhouse will affect the temperature. And, as mentioned earlier, an increase in temperature will affect the moisture level.

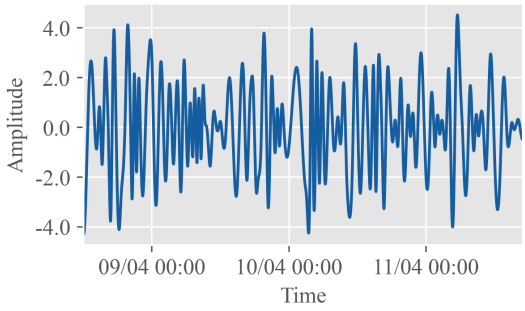
Finally, IMF 14 and its FFT, shown in figure 4.7g and figure 4.7h, respectively, show a frequency component with a period of about 24 hours. This matches the period of the light intensity in the greenhouse, as the greenhouse is placed in an office with windows. In addition to increased temperature, the increased light intensity will also affect the temperature and the moisture.

The results indicate that EMD can successfully extract different periodic components from the moisture signal and uncover their hidden information. The IMFs obtained by EMD reflect the influence of both the temperature and light intensity on the moisture level in the greenhouse. However, it is important to acknowledge that when looking for patterns and information, one may find what one expects to find. Therefore, it is possible that the temperature, fans, and light, do not have a significant impact on the moisture, but rather that their frequencies are detected by coincidence.

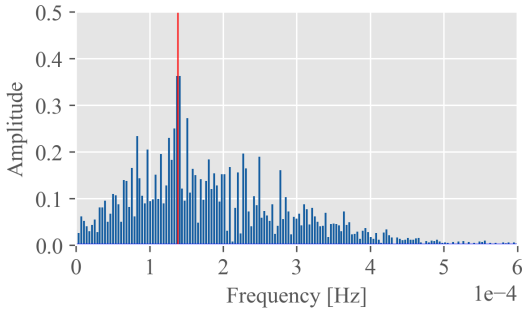
The results in figure 4.7 also demonstrate how EMD removes high-frequency noise components from the signal. All IMFs from EMD are shown in figure D.1. Analyzing the first 10 IMFs shows that they contain a range of frequencies that either do not seem to have any meaningful information or have their energy spread over a wide range of frequencies. Therefore, these first 10 IMFs can be considered as containing mostly noisy parts of the signal and thus, in some sense, act as a noise filter.

EEMD and FFT

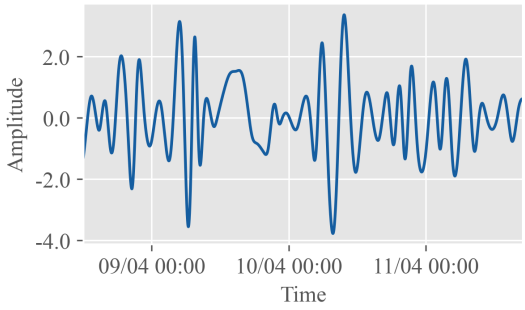
As discussed in section 4.1.1, EMD can decompose the data into IMFs that each contain information about the dynamics of the greenhouse. However, EMD is prone to mode mixing, which means that the IMFs may not represent distinct scales of oscillations in the signal. In figure 4.7, the IMFs look noisy and their FFTs have energy spread over a wide range of frequencies, suggesting that the modes of the IMFs are not well separated. To overcome this problem, EEMD, instead of EMD, is used to decompose the signal into IMFs with a lower degree of mode mixing. All IMFs from EEMD are shown in figure E.1.



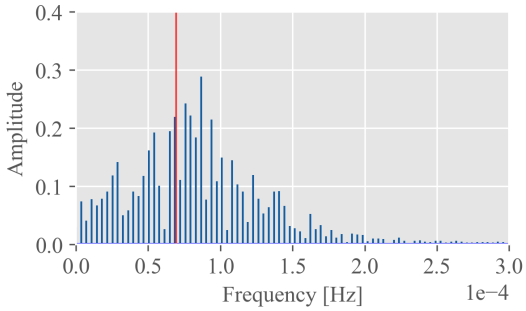
(a) IMF 11



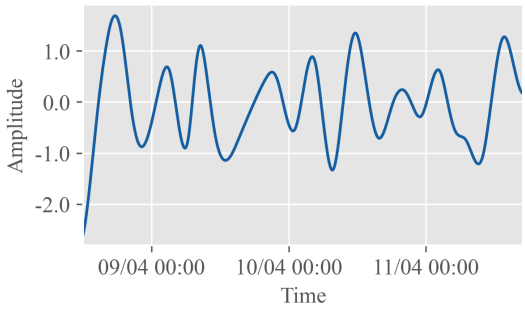
(b) FFT of IMF 11. The red stem marks the frequency of the sinusoidal heater duty cycle variations (2 hours).



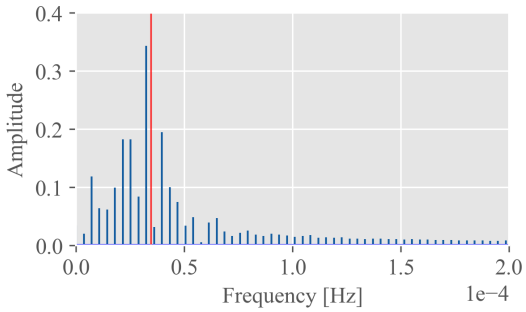
(c) IMF 12



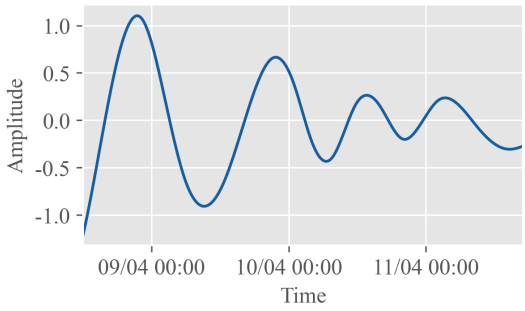
(d) FFT of IMF 12. The red stem marks the frequency of when the fans are on (4 hours).



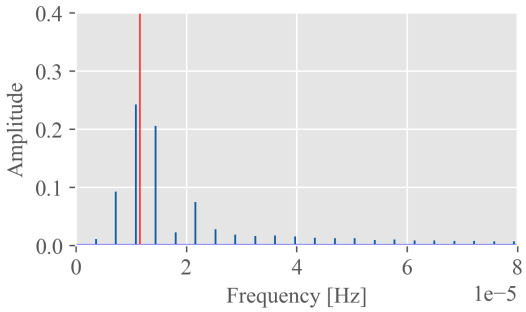
(e) IMF 13



(f) FFT of IMF 13. The red stem marks a frequency of twice the frequency of fans are on (8 hours)



(g) IMF 14



(h) FFT of IMF 14. The red stem marks a frequency of 24 hours.

Figure 4.7.: IMF and its FFT frequency plot for selected IMFs. Figure 4.7a, figure 4.7c, figure 4.7e and figure 4.7g shows the 11th, 12th, 13th and 14th IMF of the signal, respectively. And figure 4.7b, figure 4.7d, figure 4.7f and figure 4.7h shows their corresponding FFTs.

Figure 4.8 shows some of the IMFs of the EEMD and their corresponding FFTs. The IMFs in this case still look noisy, indicating that the modes are still somewhat mixed. However, the FFTs of the IMFs from EEMD show some clearer peaks at certain frequencies, which are further investigated.

Figure 4.8a and figure 4.8b show the 14th IMF and its corresponding FFT. The FFT has a prominent peak at a frequency corresponding to a period of 2 hours, which matches the heater duty cycle.

Figure 4.8c and figure 4.8d show the 15th IMF and its corresponding FFT. The FFT has two dominant peaks, one at a frequency corresponding to a period of 2 hours and another at a frequency corresponding to a period of 4 hours. As the heater duty cycle has a period of 2 hours and the fan has a period of 4 hours, this IMF seems to capture a combination of the effects of these controls.

Figure 4.8e and figure 4.8f show the 16th IMF and its corresponding FFT. The FFT has two peaks, one at a frequency corresponding to a period of 8 hours and another at a frequency corresponding to a period of 24 hours. The period of 8 hours is a multiple of both the heater duty cycle period and the fan control period and may be a result of their interaction. The period of 24 hours matches the light intensity in the room, indicating that the light may affect the moisture level.

Finally, figure 4.8g and figure 4.8h show the 17th IMF and its corresponding FFT. The FFT has a single peak at a frequency corresponding to a period of 24 hours, confirming the influence of light intensity on moisture.

Based on the results, EEMD did not seem to improve mode mixing and performance significantly compared to EMD on this data set. The IMFs obtained by EEMD still looked noisy and their FFTs still had energy spread over a wide range of frequencies. This may be due to several different reasons. The moisture signal may be inherently noisy and complex, making it hard to decompose it into distinct modes of oscillations or the added white noise may not be sufficient or appropriate to separate the modes of oscillations in the moisture signal.

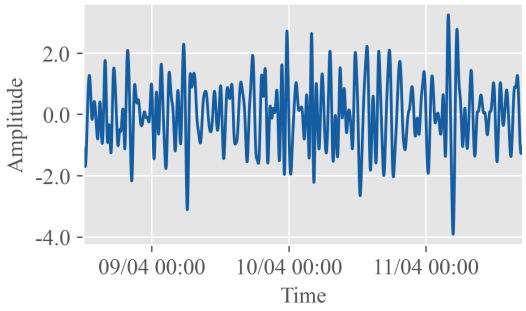
4.1.2. Field Scale: Oil Drilling Hole Cleaning Process

PCA

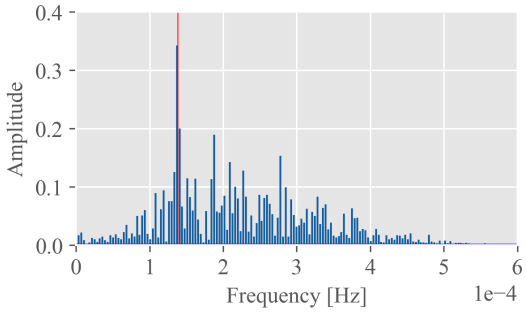
Figure 4.9 shows a scatter plot of the control variables along PC1 and PC2 axes. The PCA is performed on the control variables, mud density (DMIAVG), flow in (FLIAVG), weight on bit (WOBAVG) and revolutions per minute (RPMBAVG) to investigate their relationship with ECD.

PC1 explains 48.03% of the variance. Along PC1, DMIAVG, FLIAVG, and RPMBAVG are positively correlated with ECD, while WOBAVG is negatively correlated. This indicates that increasing DMIAVG, FLIAVG, and RPMBAVG increases ECD while increasing WOBAVG decreases ECD. For the positively correlated variables, this is consistent with the theoretical equation of ECD (equation (2.24)), which shows that ECD depends on the weight and pressure of the fluid exiting the hole, the annular pressure loss, and the cuttings concentration.

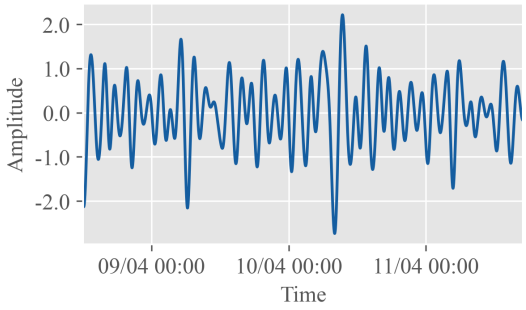
Increasing DMIAVG will increase the mud weight at the exit, consequently increasing ECD. Increasing the FLIAVG will increase the annular pressure loss. While the RPMAVG can be adjusted in a way to increase the rate of penetration and therefore also the amount of cuttings left in the hole. This increased amount of cuttings will lead to increased mud weight at the exit. Therefore, PC1 captures the proportional relationship between some of the control variables and ECD.



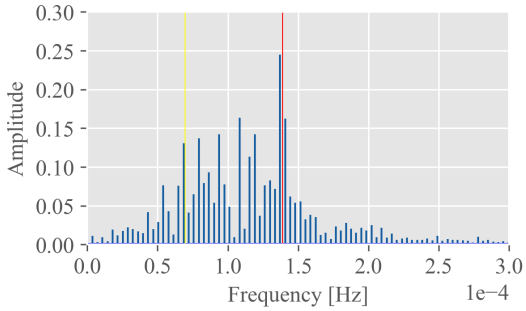
(a) IMF 14



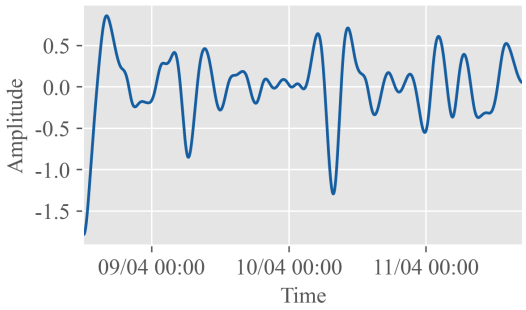
(b) FFT of IMF 14. The red stem marks the frequency of the sinusoidal heater duty cycle variations (2 hours)



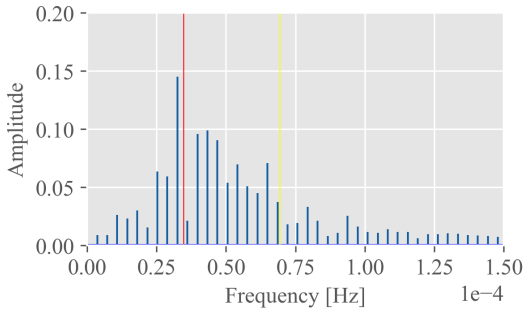
(c) IMF 15



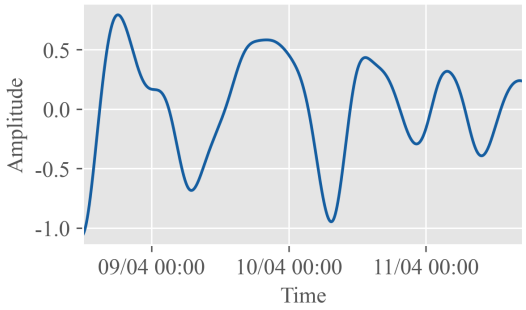
(d) FFT of IMF 15. The red stem marks a frequency of 2 hours and the yellow stem marks a frequency of 4 hours.



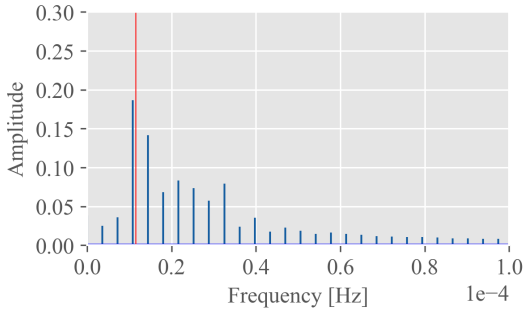
(e) IMF 16



(f) FFT of IMF 16. The red stem marks a frequency of 8 hours and the yellow stem marks a frequency of 4 hours.



(g) IMF 17



(h) FFT of IMF 17. The red stem marks a frequency with a period of 24 hours.

Figure 4.8.: IMF and its FFT frequency plot for selected IMFs. Figure 4.8a, figure 4.8c, figure 4.8e and figure 4.8g shows the 14th, 15th, 16th and 17th IMF of the signal, respectively. And figure 4.8b, figure 4.8d, figure 4.8f and figure 4.8h shows their corresponding FFTs.

WOBAVG has a negative correlation with ECD along PC1. This implies that WOBAVG has an inverse relationship with ECD during drilling. However, the exact mechanism of how WOBAVG affects ECD is not clear. It may affect other parameters that in turn affect ECD. For example, decreasing WOBAVG may reduce the rate of penetration, which may reduce the cuttings and ECD. Therefore, it is not unreasonable that WOBAVG has a negative correlation with ECD.

PC2 explains 21.99% of the variance. The variables have different directions along PC2, indicating that they have different effects on ECD. FLIAVG and RPMBAVG are positively correlated with ECD along PC2, while DMIAVG and WOBAVG are negatively correlated. This is consistent with the results in PC1. DMIAVG and WOBAVG are negatively correlated with ECD in this PC. For WOBAVG, this is consistent with the results in PC1, but not for DMIAVG.

DMIAVG has a different correlation with ECD along PC1 and PC2. This shows that DMIAVG can affect ECD both positively and negatively. Figure 3.12 shows the controls and ECD. The DMIAVG has the same pattern at the beginning of the drilling operation, but its mean decreases towards the end. However, the ECD keeps increasing. This may explain the negative relationship, but the theoretical explanation is not clear.

The PCA analysis reveals that some of the control variables and ECD have different correlations along PC1 and PC2. This reflects the different effects of the control parameters on ECD depending on how they are adjusted during drilling operations. This implies that the operators manipulate the parameters to change ECD in various ways, sometimes proportionally and sometimes inversely. This is according to theory.

The relationships between the different variables found using PCA are also mostly consistent with the theory. Theory agrees with the positive correlations between DMIAVG, RPMAVG, FLIAVG and ECD in PC1. The inverse relationship between ECD and WOBAVG is harder to explain using theory, but the relationship is not direct and may be complex. For PC2, the positive correlations between FLIAVG, RPMAVG and ECD remain, but the DMIAVG now has a negative correlation. In the same way as WOBAVG, this is difficult to explain using theory, but looking at the plots, it is not unreasonable. WOBAVG still correlates negatively, also in PC2.

EMD and EEMD

each method to illustrate the range of the IMFs, without implying any correspondence between the same numbered IMFs from different methods. EMD produced 24 IMFs and EEMD produced 20 IMFs. For the curious reader, all IMFs from EMD and EEMD are shown in figure F.1 and figure G.1, respectively.

As can be seen from figure 4.10, EMD failed to preserve the important information in the ECD signal, such as the points where the signal characteristics changed significantly. These points were distributed over different IMFs, resulting in a loss of interpretability. Therefore, the IMFs from EMD were not suitable for further analysis.

On the other hand, EEMD maintained the signal features and patterns in the IMFs, as shown in figure 4.11. The points where the ECD characteristics changed were visible in all IMFs, indicating a better decomposition quality. Therefore, the IMFs from EEMD were more informative and useful for further analysis.

Based on these results, EEMD outperforms EMD in terms of preserving the signal features and patterns in the IMFs for the ECD. EMD distributes important information over different IMFs, resulting in a loss of interpretability. EEMD maintains the signal features and patterns in the IMFs, indicating a better decomposition quality. Therefore, for this case, EEMD had been a more suitable decomposition method to apply to the

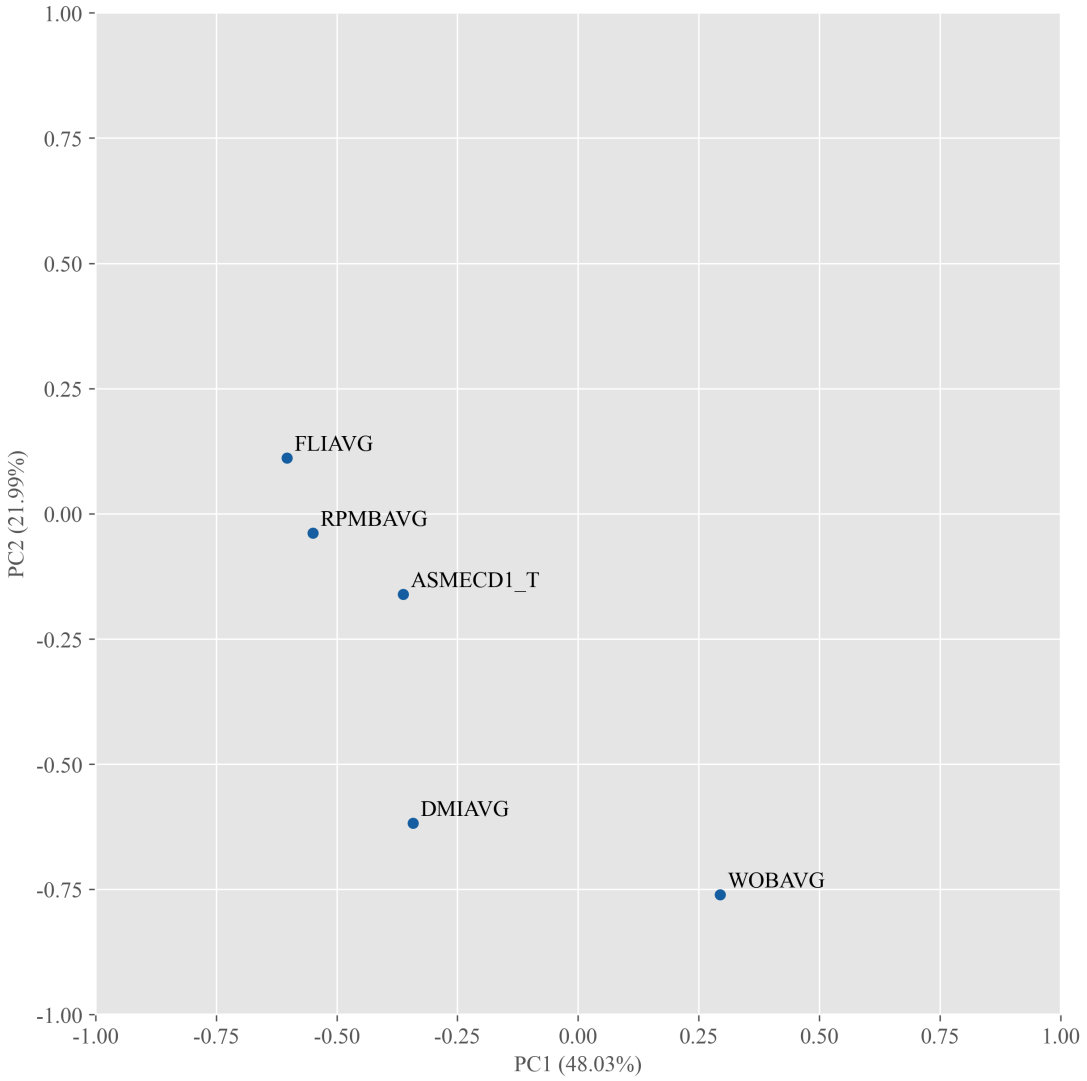
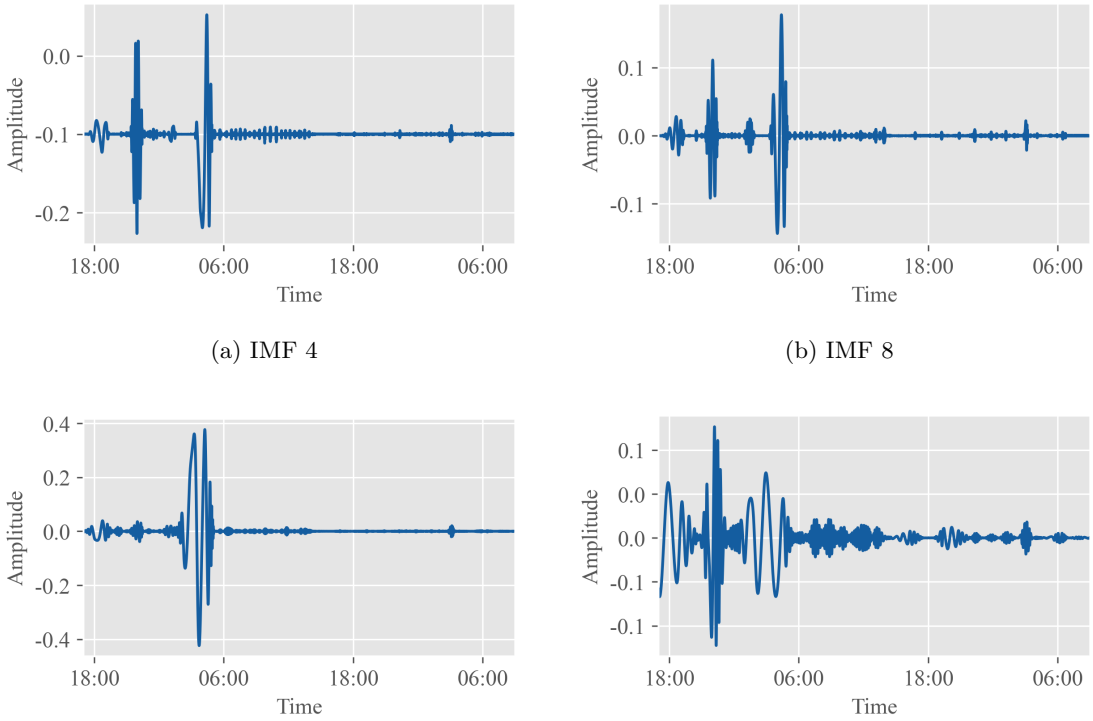


Figure 4.9.: Loadings for PC1 and PC2 for drilling data. PCA is performed on the control variables, mud density, weight on bit, flow in and RPM to investigate their relationship with ECD. Weight on bit, flow in and RPM seems to be negatively correlated with ECD, while mud density seems to be positively correlated.



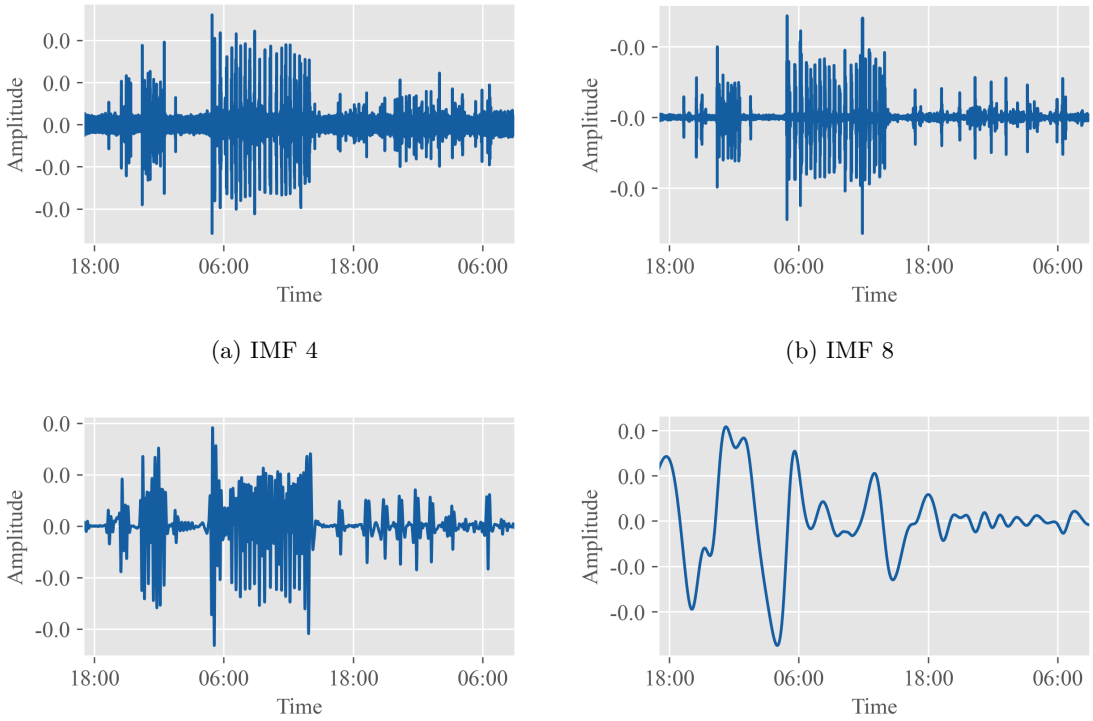
(a) IMF 4

(b) IMF 8

(c) IMF 12

(d) IMF 16

Figure 4.10.: IMFs of EMD of ECD from drilling case A. Note how the important information about changes in pattern seems to vanish after decomposition.



(a) IMF 4

(b) IMF 8

(c) IMF 12

(d) IMF 16

Figure 4.11.: IMFs from EEMD of ECD from drilling case A. Note how the important changes in the original data are kept after the decomposition.

Table 4.1.: Metrics for FFNN model performance on scenario 1.

	Temperature	Humidity
RMSE	0.90	2.96
MAE	0.81	2.52
MAPE	0.03	0.06

data before further analysis. These findings also show how EEMD can be useful when it comes to anomaly detection in drilling operations, as it manages to detect significant changes in the ECD measurements.

4.2. Predictive Analysis and Forecasting

This section presents the results of applying the FFNN model for predicting derivative, the LSTM model for predicting the next value, and the LSTM model for predicting derivative to greenhouse data. The predictions are presented, one by one, with plots and tables with metrics. Their performance is evaluated based on metrics and visual inspection of the scenarios presented in chapter 3.

4.2.1. Laboratory Scale: Greenhouse

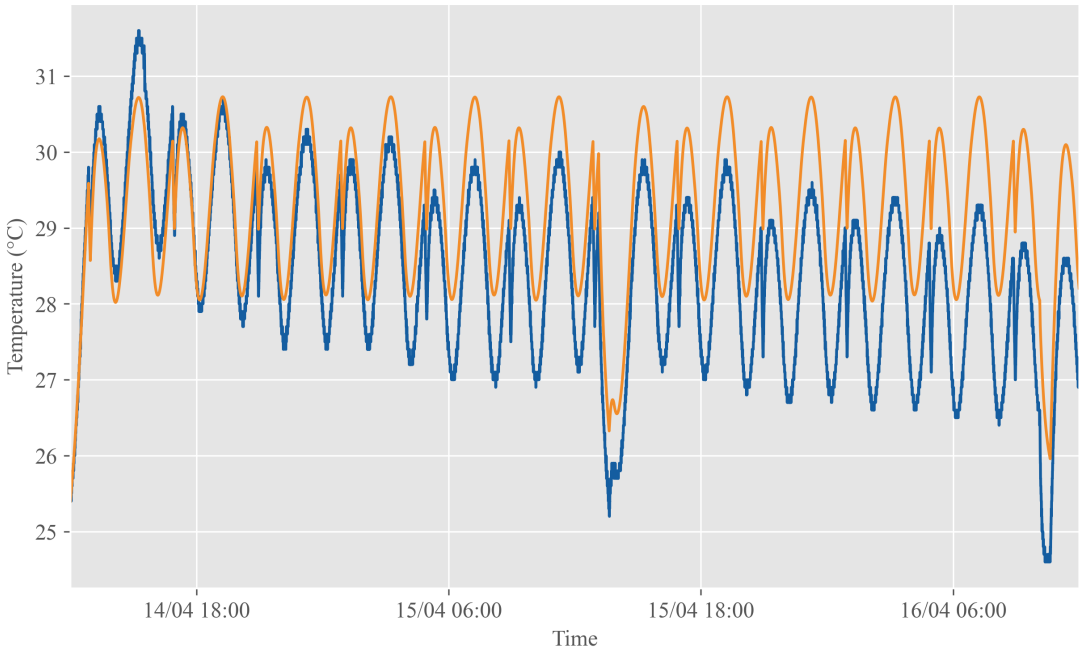
FFNN to Predict Derivative

The first model was to use a FFNN to predict the derivative. This model only uses information about the states in the last time step to predict the derivative that "moves" the system from the current time step to the next.

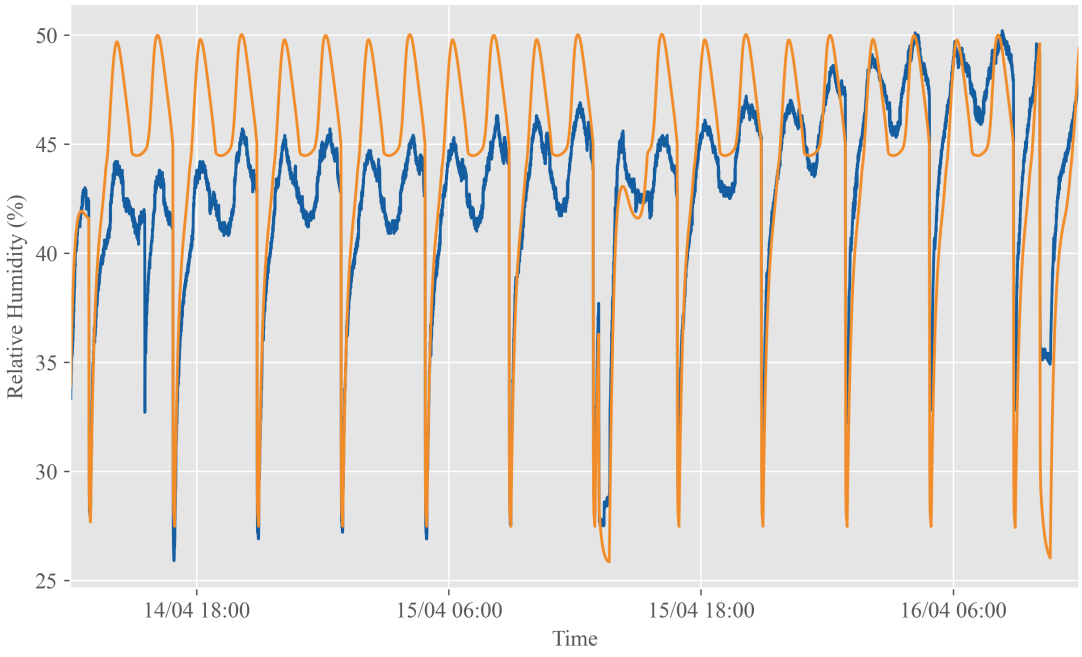
The performance of the model on the data from scenario 1 is evaluated by comparing the predicted value of temperature and humidity, as shown in figure 4.12. The corresponding metrics are shown in table 4.1. In this scenario, the test set exhibits much of the same dynamics as the training set. The model predicts the temperature fairly well but drifts away from the actual values towards the end. This may be because the test data has a drift in the temperature that is not present in the training data. The model also predicts the humidity with reasonable accuracy, but it overestimates the peaks and dips. This may be due to the standardization and de-standardization process for the model's inputs and outputs. The model is trained on standardized data using the mean and standard deviation from the training data. The same mean and standard deviation are used to standardize the input before feeding the data into the model and to de-standardize the output of the model to get predictions. However, the test data has smaller peaks and dips in the humidity, as seen in figure 3.3 and figure 3.4. Therefore, using the mean and standard deviation of the training data may introduce some distortions to the data.

The metrics for the model performance on scenario 1 are shown in table 4.1. The prediction has a RMSE of 0.90, which is low compared to the range of the temperature in the test set, which is 6.92 (see table 3.3). For the humidity, the RMSE are 2.96, which also is low compared to the range of the humidity, 23.79 (see table 3.3). The MAE, suggest the same as the RMSE, 0.81 for temperature and 2.52 for humidity, which is low compared to their range. The MAPE further emphasizes the good fit, with values of, 0.03 for temperature and 0.06 for humidity.

In scenario 2, the model is tested on more complex data. The results are shown in



(a) Measurement and prediction of temperature for scenario 1 using FFNN to predict derivative



(b) Measurement and prediction of humidity for scenario 1 using FFNN to predict derivative

Figure 4.12.: Measurement of temperature and humidity in the greenhouse along with predictions on the test set from scenario 1 using the FFNN model. Figure 4.12a shows the temperature and the corresponding prediction and figure 4.12b shows the humidity and the corresponding prediction.

Table 4.2.: Metrics for FFNN model performance on scenario 2.

	Temperature	Humidity
RMSE	1.49	12.19
MAE	1.44	11.85
MAPE	0.05	0.43

Table 4.3.: Metrics for FFNN model performance on scenario 3.

	Temperature	Humidity
RMSE	1.35	14.02
MAE	1.08	13.36
MAPE	0.03	0.51

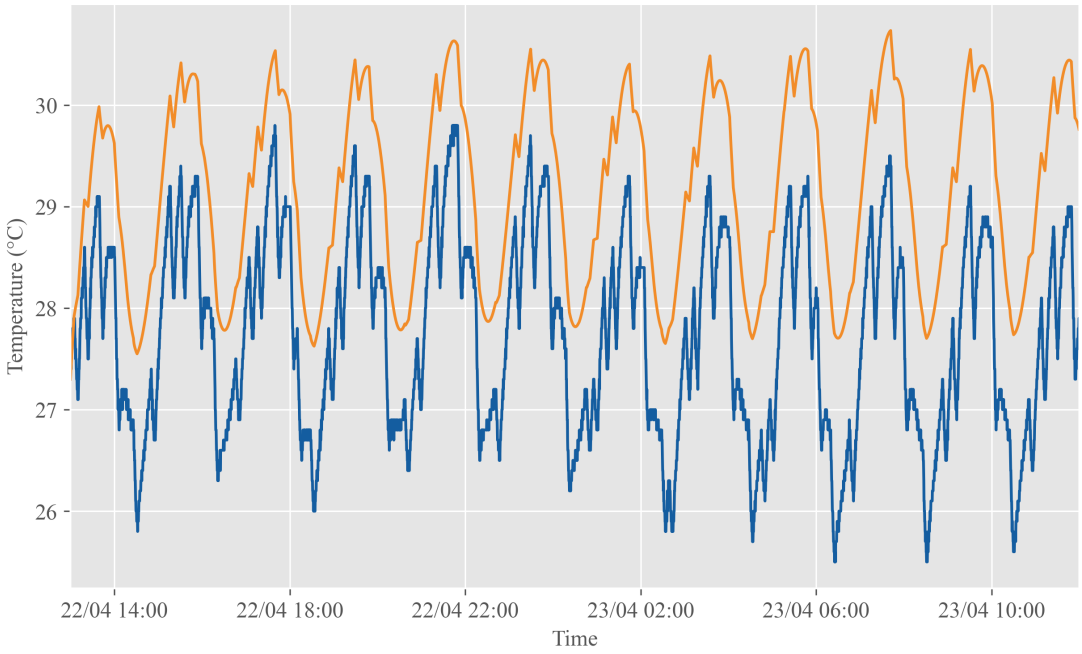
Figure 4.13 and table 4.2. For the temperature, the model follows the pattern of the actual data, but it has a systematic bias, probably due to the standardization and de-standardization reasons discussed earlier. For the humidity, the model has a bias and a different scale than the actual data. The model does not capture the depth of the dips and the height of the peaks of the actual humidity data, again probably due to standardization issues.

For scenario 2, the test set has a range of 4.21 for temperature and 16.33 for humidity, as shown in table 3.4. The model has a RMSE of 1.49 for temperature and 12.19 for humidity, which are high compared to the ranges of temperature and humidity values. This means that the model has large errors in predicting the temperature and humidity. The MAE also indicates a poor fit, with values of 1.44 for temperature and 11.85 for humidity, which are high compared to their ranges. The MAPE shows that the model has small relative errors in percentage terms for the temperature, with values of 0.05, and large relative percentage errors for humidity, with 0.43. The difference in the pattern from the training set to the test set makes it difficult for the model to predict, and combining this with standardization issues will result in bad predictions.

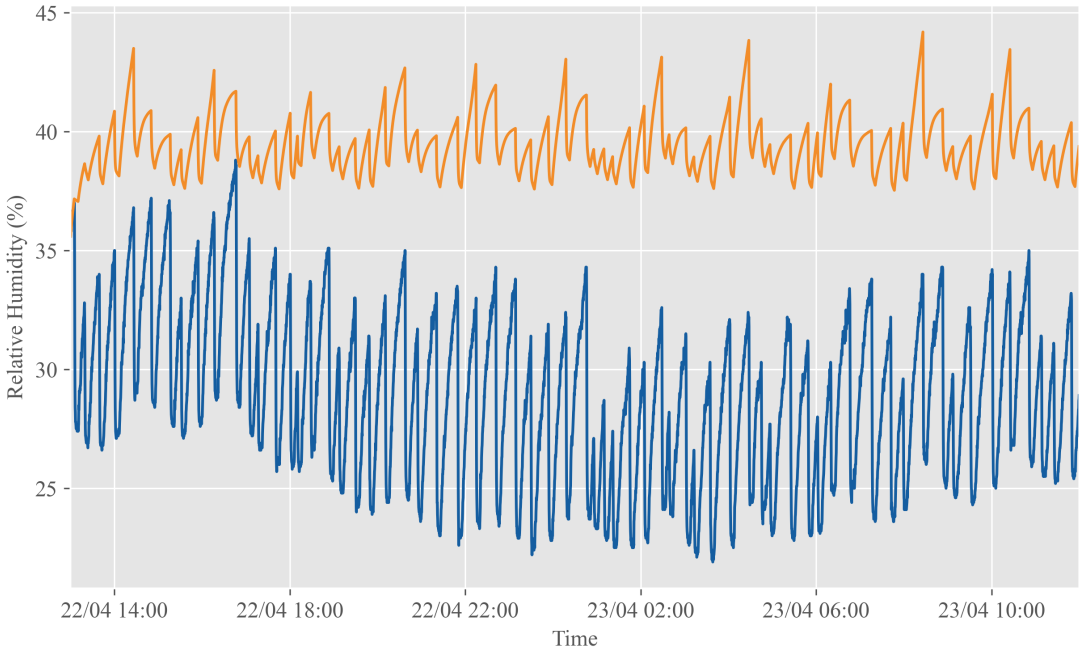
Lastly, the model was used for prediction on the most challenging scenario. Here, the test data has no cyclical patterns and different controls compared to the two other test sets and the training set. The prediction of the model is shown in figure 4.14. For the temperature, the model seems to have learnt how the different control inputs affect the patterns in the data. This is seen as the prediction of temperature following the same pattern as the actual data. The humidity is worse. The model is heavily biased, with too high humidity prediction almost all the way.

Based on the metrics in table 4.3, the model has moderate accuracy when predicting the temperature, but struggles with the humidity. The temperature and humidity range for the test set in this scenario is 6.99 and 21.98 (see table 3.5). The RMSE are 1.35 for the temperature and 14.02 for the humidity, given the ranges, this indicates a moderate accuracy on the temperature and bad accuracy on the humidity. The MAE indicates the same, with 1.08 for the temperature and 13.36 for the humidity. The prediction has a fairly low MAPE for the temperature, at 0.03. A MAPE of 0.51 for the humidity indicates a bad fit.

To summarize, prediction of the derivative with a FFNN model performs well when the data is cyclical and stationary. When the data is not the temperature predictions tend to be better. This is probably due to more direct interactions between the temperature and heater duty cycle making the temperature easier to predict.



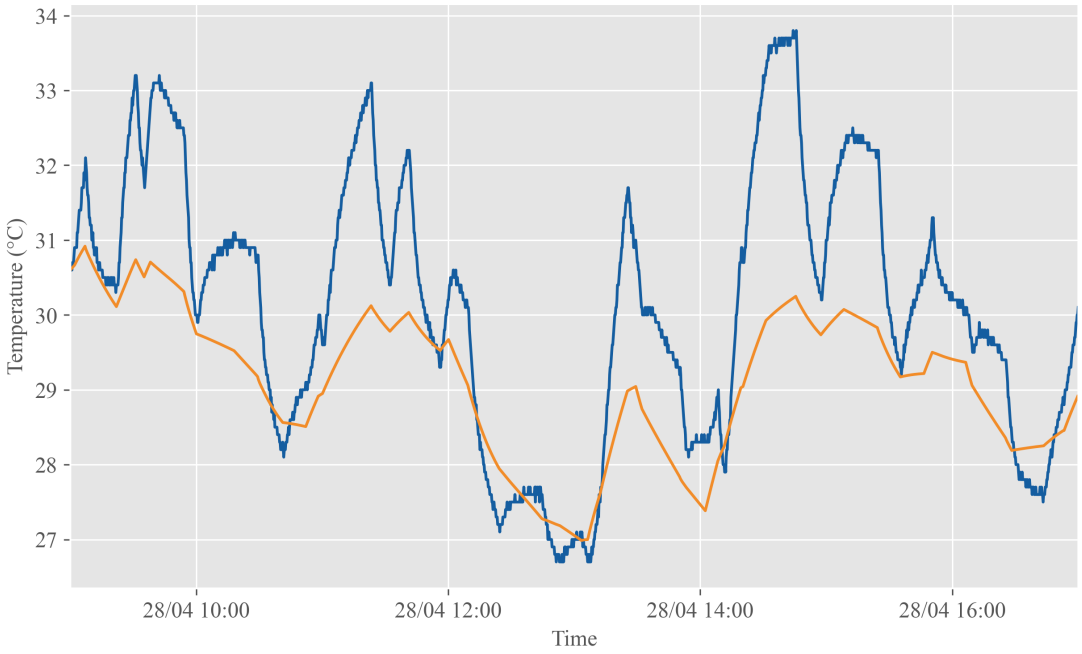
(a) Measurement and prediction of temperature for scenario 2 using FFNN to predict derivative



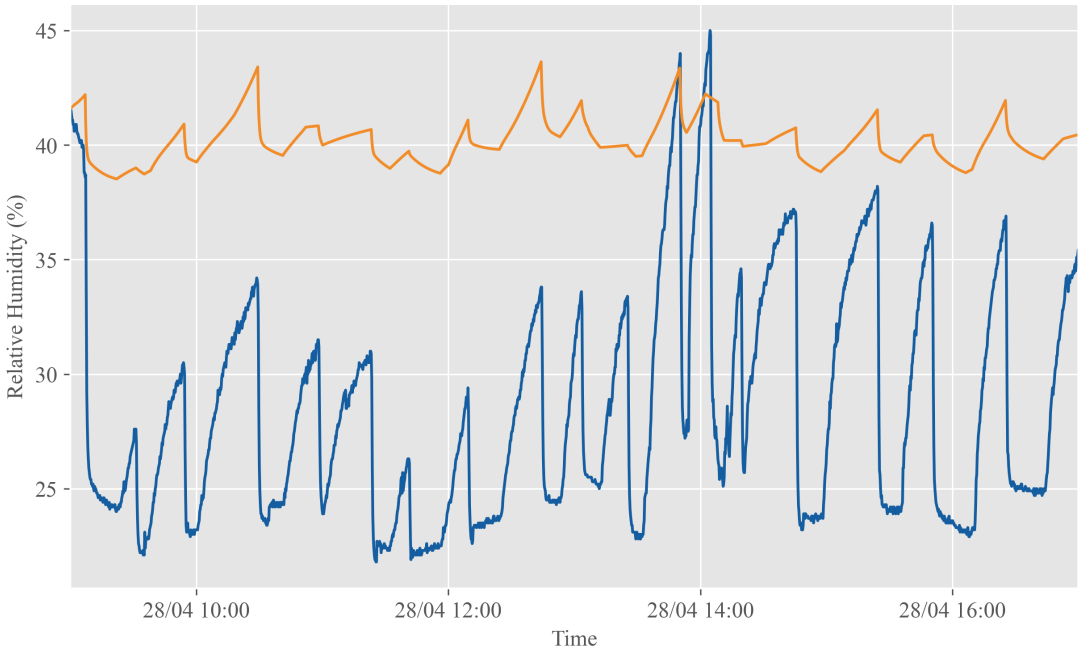
(b) Measurement and prediction of humidity from for scenario 2 using FFNN to predict derivative

Measurement Prediction

Figure 4.13.: Measurement of temperature and humidity in the greenhouse along with predictions on scenario 2 from the FFNN model. Figure 4.13a shows the temperature and the corresponding prediction and figure 4.13b shows the humidity and the corresponding prediction.



(a) Measurement and prediction of temperature for scenario 3 using FFNN to predict derivative



(b) Measurement and prediction of humidity for scenario 3 using FFNN to predict derivative

Figure 4.14.: Measurement of temperature and humidity in the greenhouse along with predictions on scenario 3 from the FFNN model. Figure 4.14a shows the temperature and the corresponding prediction and figure 4.14b shows the humidity and the corresponding prediction.

Table 4.4.: Metrics for LSTM model prediction on greenhouse scenario 1

	Temperature	Humidity
RMSE	1.26	3.70
MAE	1.13	3.14
MAPE	0.04	0.07

Table 4.5.: Metrics for LSTM model prediction on greenhouse scenario 2

	Temperature	Humidity
RMSE	2.17	11.22
MAE	1.96	10.54
MAPE	0.07	0.38

LSTM to Predict Next Timestep

In this section, a LSTM was used to estimate the next temperature and humidity values given the last 50 (last 12 minutes and 24 seconds) measurements and/or predictions.

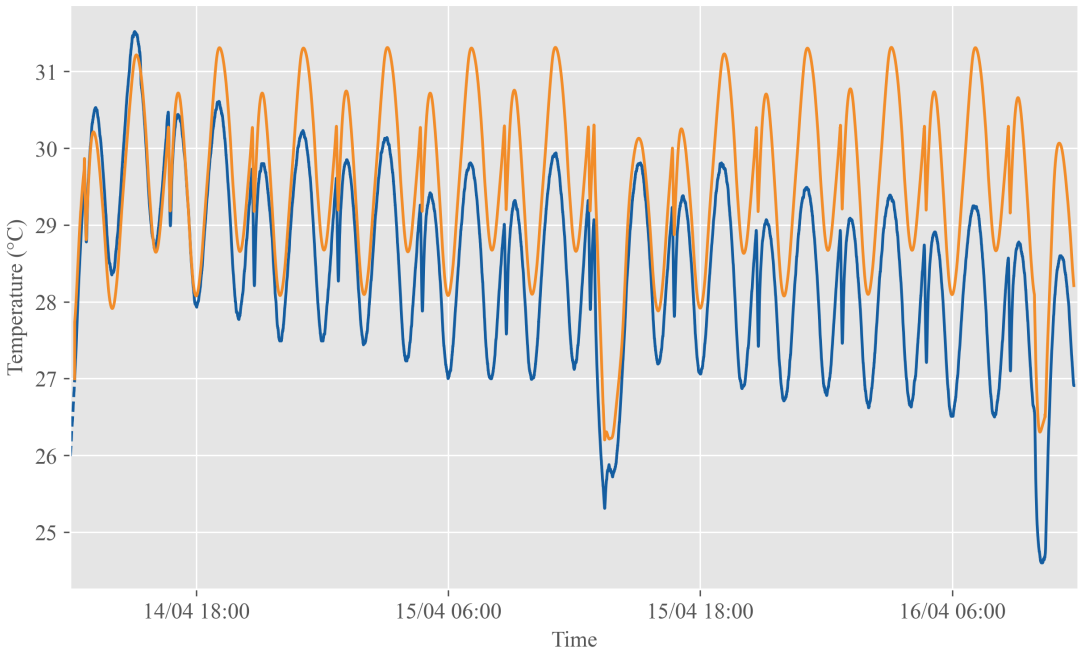
A plot of the prediction on scenario 1 is shown in figure 4.15 with corresponding metrics shown in table 4.4. Looking at the plot of the temperature, the prediction is good at the start but deviates towards the end. For the humidity it is the opposite, the predictions are biased in the start but get better towards the end when the actual humidity is rising. Also for this model, the data is standardised before being fed into the model as input and de-standardized before becoming a prediction. Therefore, the bias may be a result of this difference in mean and standard deviation between the training set and test set.

The metrics for the LSTM model prediction on scenario 1 are shown in table 4.4. The results show that the model has low accuracy in predicting the temperature, with a RMSE of 1.26, a MAE of 1.13, and a MAPE of 0.04, with a temperature range of 6.92. On the other hand, the model had moderate accuracy in predicting the humidity, with a RMSE of 3.70, a MAE of 3.14, and a MAPE of 0.07, with a humidity range of 23.79. This suggests that the model can capture some of the variability and trend of the data, but the plot suggests that it is biased.

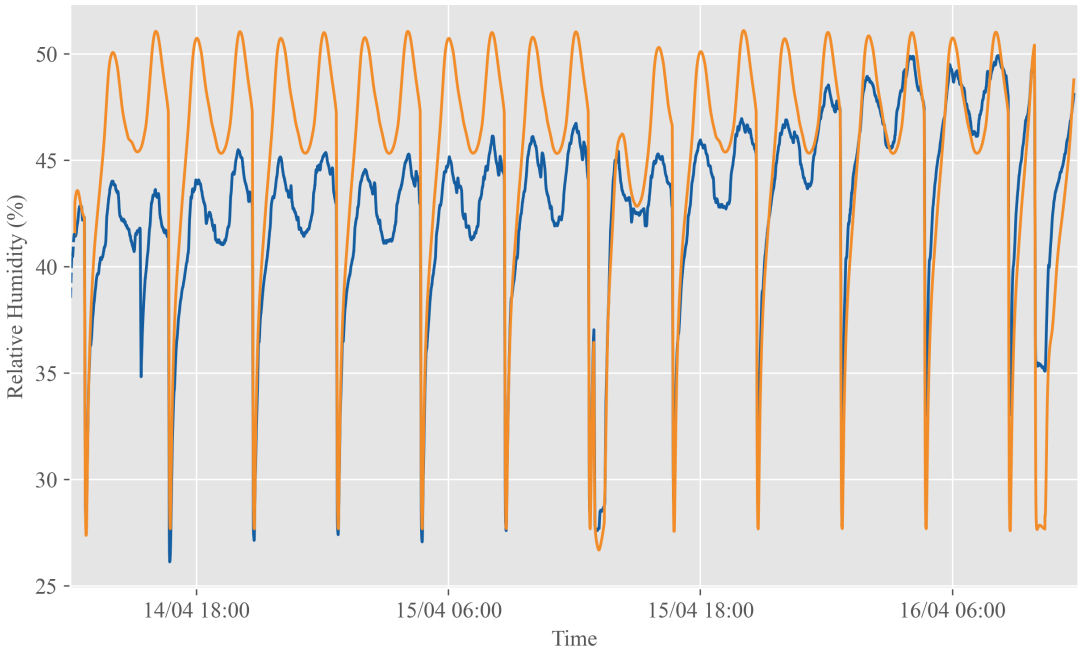
The LSTM model prediction on scenario 2 is shown in figure 4.16 and its corresponding metrics are shown in table 4.5. For this scenario, the LSTM model suffers from some of the same problems as in scenario 1. For both humidity and temperature, it starts well with good predictions for both measurements, but the model gets more and more biased towards the end resulting in some pretty bad predictions towards the end. Again, this is probably due to the difference in mean and standard deviation between the training set and the test set.

The metrics for scenario 2 are shown in table 4.5. The results show that the model has low accuracy in predicting both the temperature and the humidity, with a RMSE of 2.17 and 11.22, a MAE of 1.96 and 10.54, and a MAPE of 0.07 and 0.38, respectively. The range of the temperature data was 4.21, and the range of the humidity data was 16.33.

Lastly, this model was also on the more challenging data from scenario 3, shown in figure 4.17. For the temperature, the model manages to capture both the pattern and scales of the test data in a good way. It struggles a bit with the highest peaks but captures the lows. For the humidity data, the model struggles a bit more and predicts



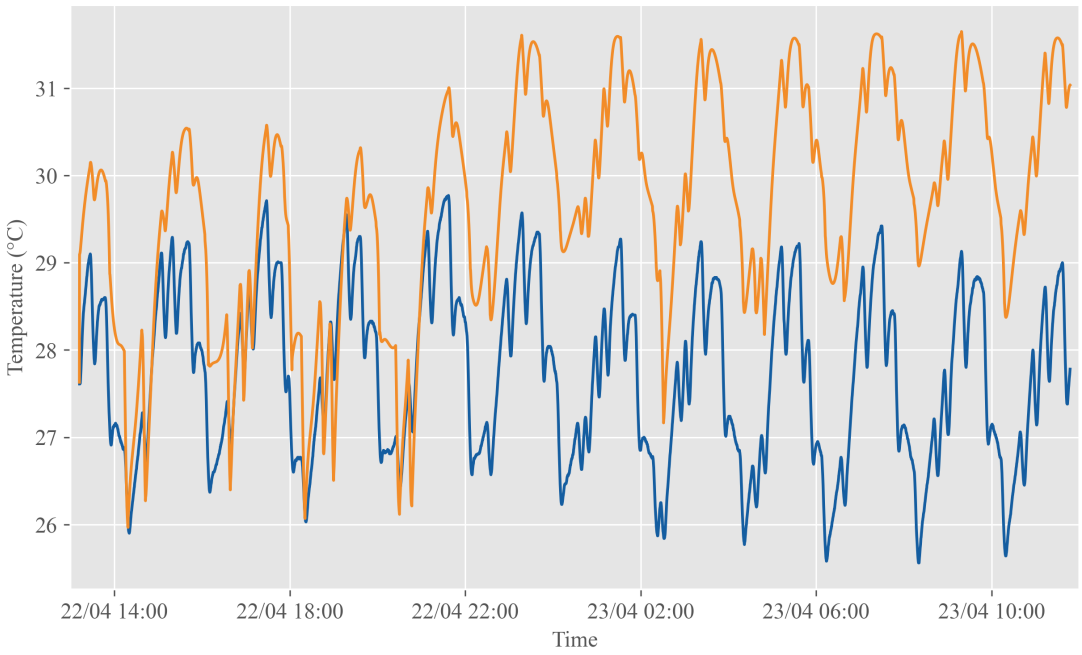
(a) Measurement and prediction of temperature for scenario 1 using LSTM to predict next value



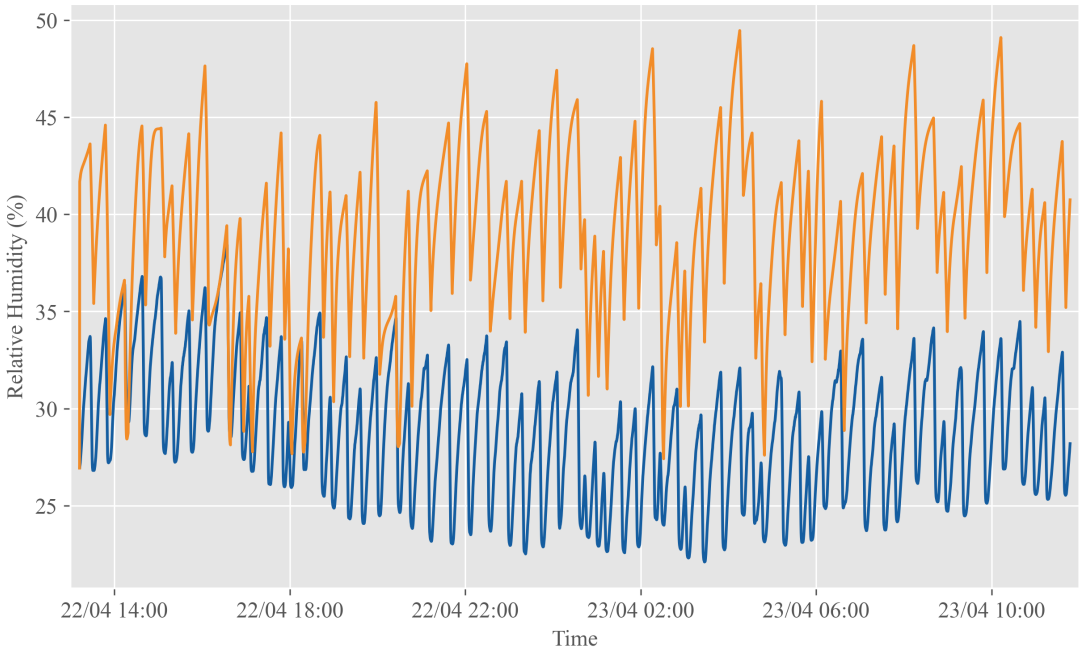
(b) Measurement and prediction of humidity for scenario 1 using LSTM to predict next value

Measurement Prediction

Figure 4.15.: Measurements and prediction of temperature (figure 4.15a) and humidity (figure 4.15b) on scenario 1. The predictions are made using a LSTM to predict the temperature and humidity at each time step.



(a) Measurement and prediction of temperature for scenario 2 using LSTM to predict next value



(b) Measurement and prediction of humidity for scenario 2 using LSTM to predict next value

Measurement Prediction

Figure 4.16.: Measurements and prediction of temperature (figure 4.16a) and humidity (figure 4.16b) on scenario 2. The predictions are made using a LSTM to predict the temperature and humidity at each time step.

Table 4.6.: Metrics for LSTM model prediction in greenhouse scenario 3.

	Temperature	Humidity
RMSE	0.95	12.19
MAE	0.75	11.42
MAPE	0.02	0.42

Table 4.7.: Metrics for LSTM model predicting the derivative in greenhouse scenario 1.

	Temperature	Humidity
RMSE	0.84	3.05
MAE	0.75	2.58
MAPE	0.03	0.06

too high humidity throughout the time horizon, but manages to capture the pattern.

The metrics for the predictions in scenario 3 are shown in table 4.6. The results indicate that the model has a high accuracy in predicting the temperature, with a low RMSE of 0.95, a low MAE of 0.75, and a low MAPE of 0.02, with a temperature range of 6.99. However, the model has low accuracy in predicting the humidity, with a high RMSE of 12.19, a high MAE of 11.42, and a high MAPE of 0.42, with a humidity range of 21.98.

To summarize, the results show that the model performs well in predicting the temperature in all scenarios, but poorly in predicting the humidity in scenarios 1 and 2. The model also suffered from some bias issues, especially in scenarios 1 and 2, where the predictions deviate significantly from the actual measurements. As for the FFNN model, one possible reason for this bias is that the data was standardized before feeding into the model, and de-standardized before becoming a prediction. This might have caused some discrepancy between the mean and standard deviation of the training and test sets.

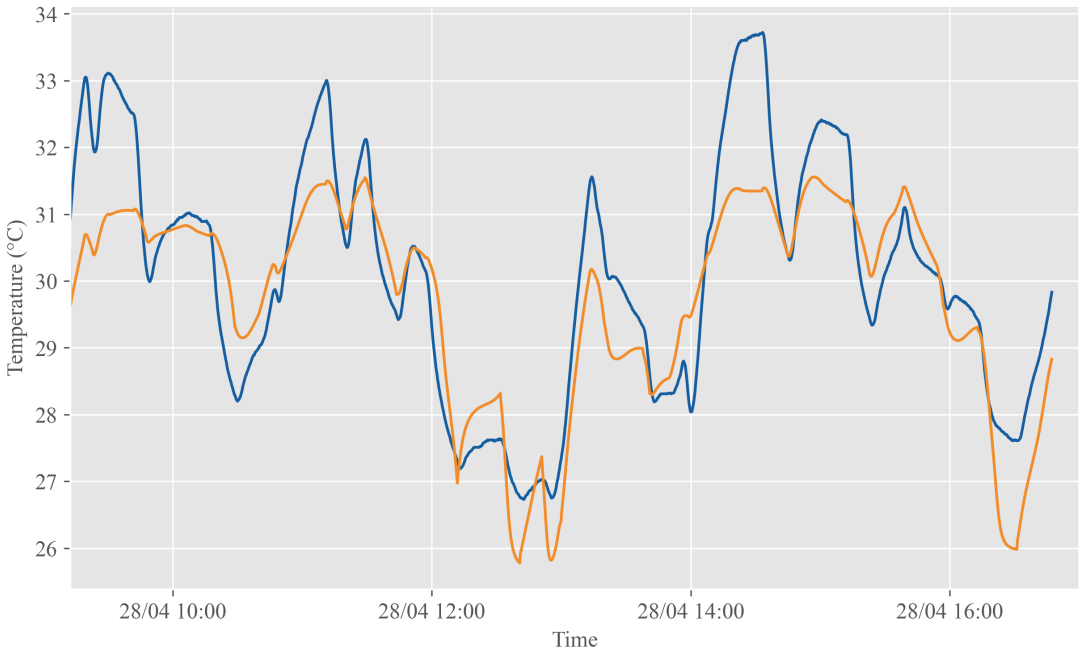
LSTM to Predict Derivative

The last model is the LSTM model used to predict the derivative. This model uses the last data from the last 50 timesteps (the last 12 minutes and 24 seconds) to predict the derivatives that take the temperature and humidity to their next values.

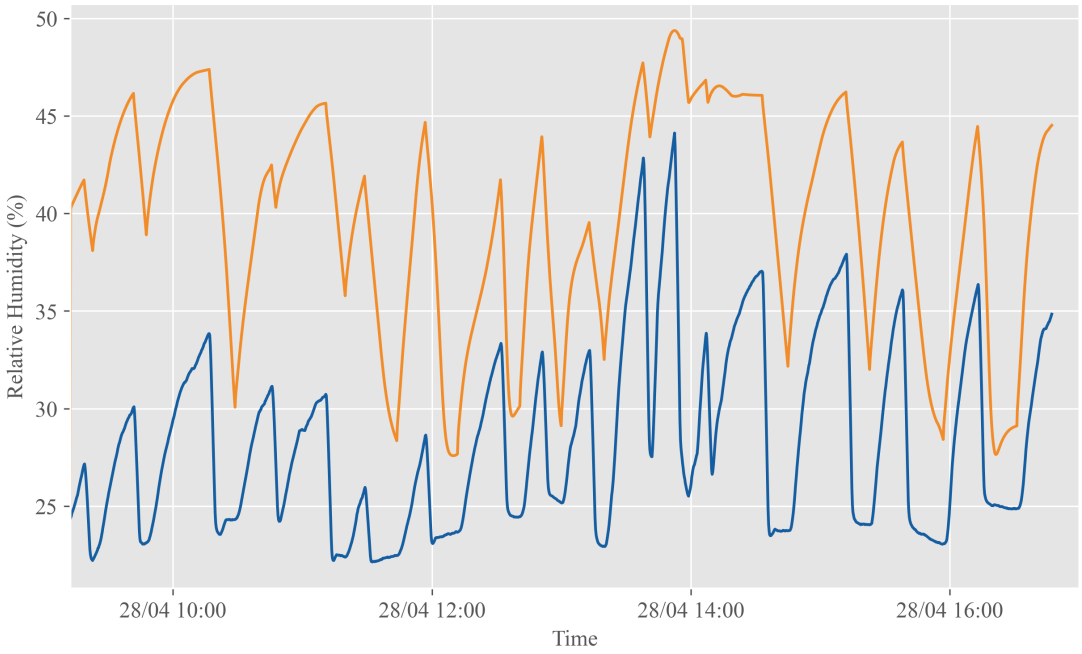
The results on the first, simple data set are seen in figure 4.18. Here you can see that, similar to the other models, this model performs well in predicting both temperature and humidity. But, similar to the other models, this model also struggles to capture the downward trend in temperature and the upward trend in humidity. Also here, the difference in the standard deviation from the training data makes the predictions deviate from the true data in the highs and lows.

Looking at the metrics for the prediction in table 4.7, the model performs well for both temperature and humidity. In this scenario, the temperature range is 6.92 and the humidity range is 23.79. For temperature prediction, the RMSE are 0.84, the MAE are 0.75, and the MAPE are 0.03. Given the range, this indicates a good fit. For humidity prediction, the RMSE are 3.05, the MAE are 2.58, and the MAPE are 0.06. Again, given the range, this indicates a good fit. This is the best fit among all the other models.

The model was then tested using the test data from scenario 2, shown in figure 3.5. This dataset contains more high-frequency components in the temperature data and a different humidity profile. Both the humidity and temperature predictions are seen in



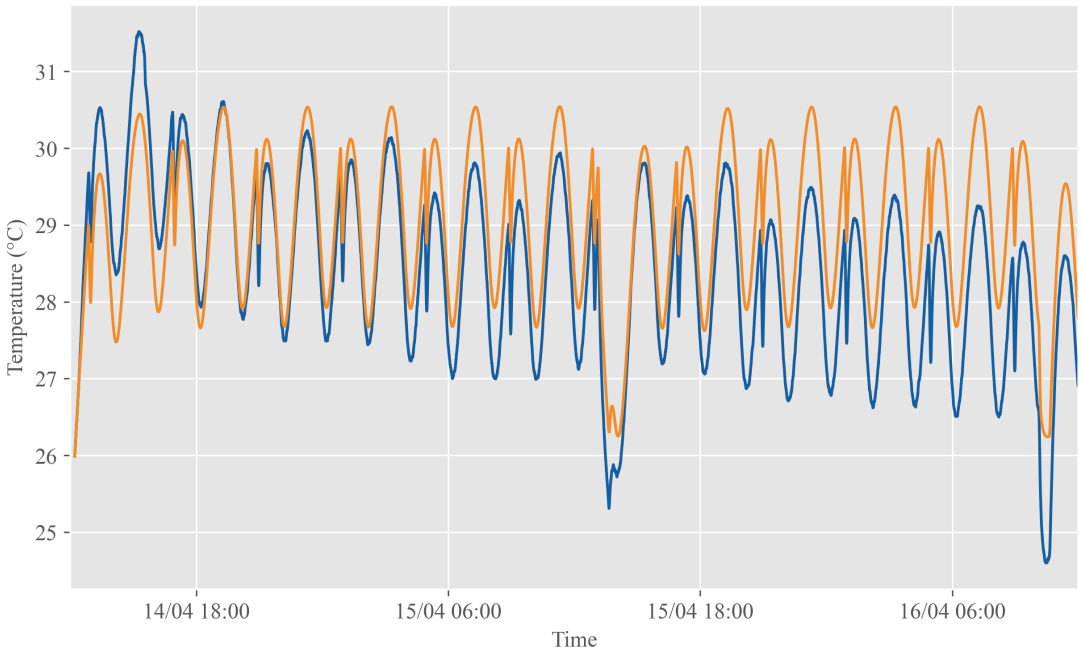
(a) Measurement and prediction of temperature for scenario 3 using LSTM to predict next value



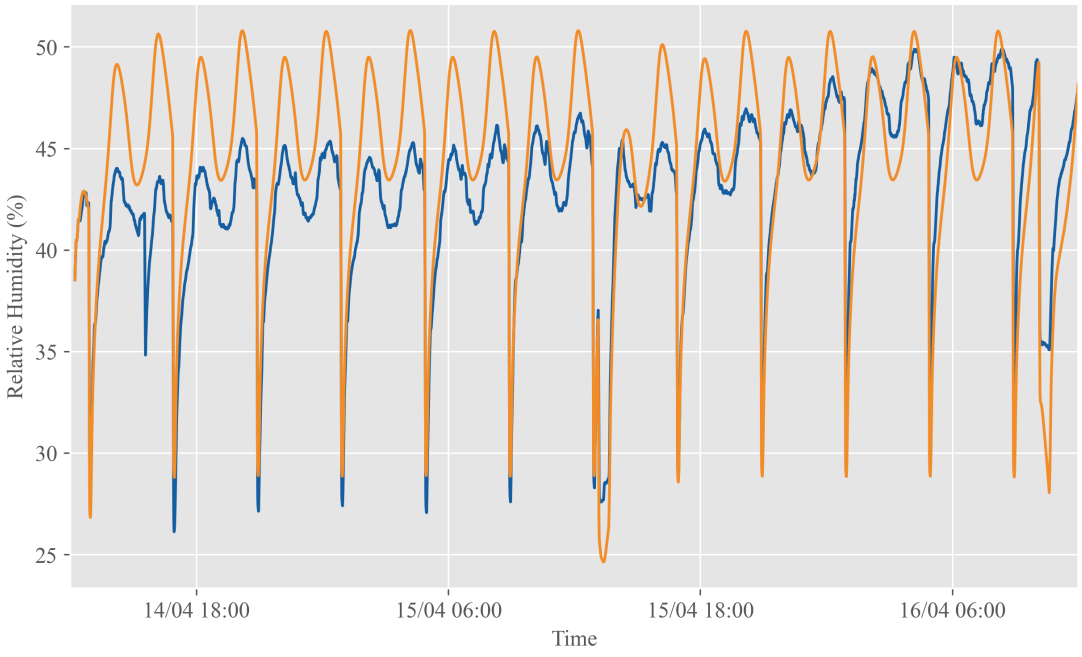
(b) Measurement and prediction of humidity for scenario 3 using LSTM to predict next value

Measurement Prediction

Figure 4.17.: Measurements and prediction of temperature (figure 4.17a) and humidity (figure 4.17b) on scenario 3. The predictions are made using a LSTM to predict the temperature and humidity at each time step.



(a) Measurement and prediction of temperature for scenario 1 using LSTM to predict derivative



(b) Measurement and prediction of humidity for scenario 1 using LSTM to predict derivative

Figure 4.18.: Measurements and predictions of temperature (figure 4.18a) and humidity (figure 4.18b) on scenario 1. The predictions are made using a LSTM to predict the derivative at each time step.

Table 4.8.: Metrics for LSTM model predicting the derivative in greenhouse scenario 2.

	Temperature	Humidity
RMSE	2.00	13.11
MAE	1.72	12.92
MAPE	0.06	0.46

Table 4.9.: Metrics for LSTM model predicting the derivative in greenhouse scenario 3.

	Temperature	Humidity
RMSE	2.19	5.00
MAE	1.88	4.09
MAPE	0.06	0.16

figure 4.19. As seen in figure 4.19a, the model manages to capture the pattern temperature pattern in a good way, but there is a significant offset between the prediction and true data. For the temperature, the peaks are too high, while for the humidity, the prediction is biased throughout the data set.

The test set for scenario 2 has a temperature and humidity range of 4.21 and 16.33, respectively. The metrics for model performance on the data are shown in table 4.8. The RMSE for the prediction of temperature is 2.00 and for the prediction of humidity, it is 13.11. This indicates bad performance for both measurements. The MAE shows the same with a MAE of 1.72 for temperature and 12.92 for humidity. The MAPE are 0.06 for temperature and 0.46 for humidity, this indicates that the temperature prediction is better than the humidity prediction.

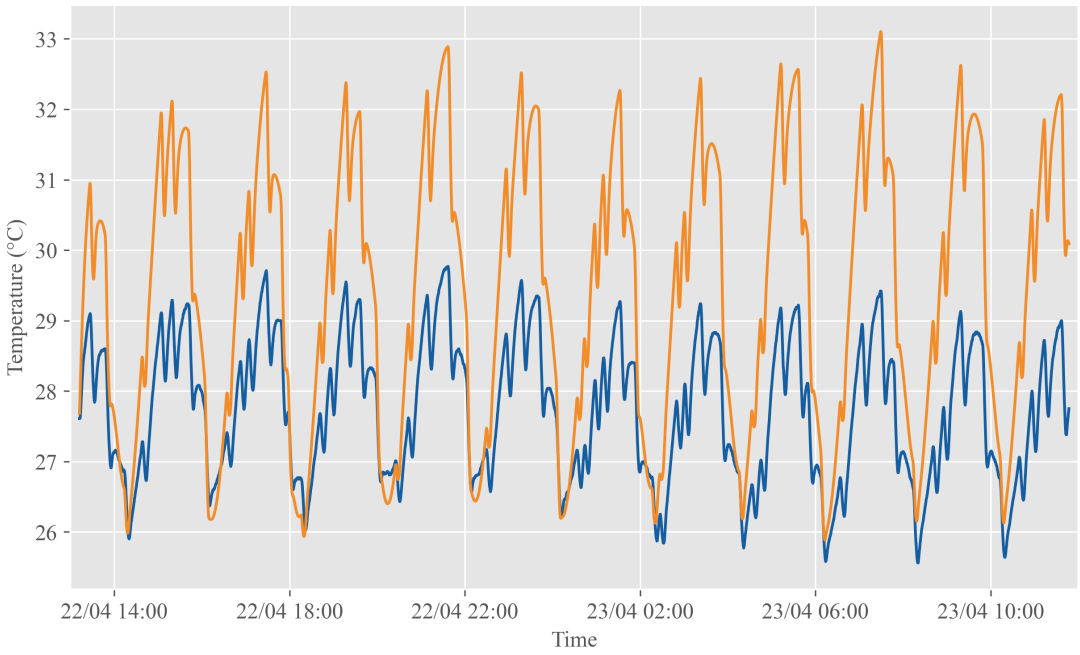
Lastly, for this model, we have its performance on the most challenging scenario 3. Looking only at the plot in figure 4.20 the model seems to perform badly. For the temperature, the model is off from the very start and struggles to capture the dynamics. For the humidity, the model does not seem to learn how the interactions in the greenhouse are and it predicts almost a straight line with an upward trend.

The metrics for scenario 3 are shown in table 4.9. The temperature and humidity range for this data set are 6.99 and 21.98, respectively. An RMSE of 2.19 and a MAE of 1.88 for the temperature predictions indicate that the predictions are off. For the humidity, a RMSE of 5.00 and MAE of 4.09 is good compared to the other models. The RMSE are 0.06 for the temperature and 0.16 for the humidity, which is also low compared to the other models.

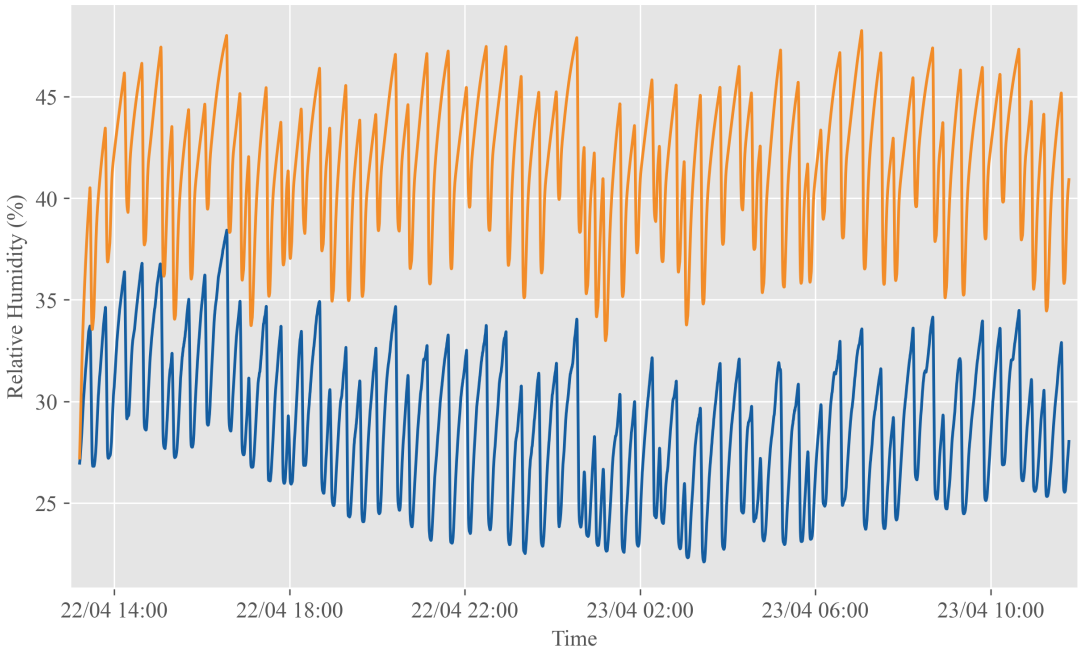
To summarize, the LSTM model for predicting the derivative also struggles with the same problems as the other models. It performs best in scenario 1 but struggles more with the more complex data in the other scenarios. For the most complex scenario, scenario 3, it achieves good RMSE, MAE and MAPE values, especially for temperature.

Model Comparison

For the prediction of oil drilling data, only the best model will be used. The FFNN model has the highest performance metrics across all scenarios and variables. However, the main goal is to find a model that can generalize well and handle new dynamics, which are best represented by greenhouse scenario 3. Looking only at the metrics, the FFNN model still has the highest performance metrics for both humidity and temperature in this scenario, with the LSTM model right behind.



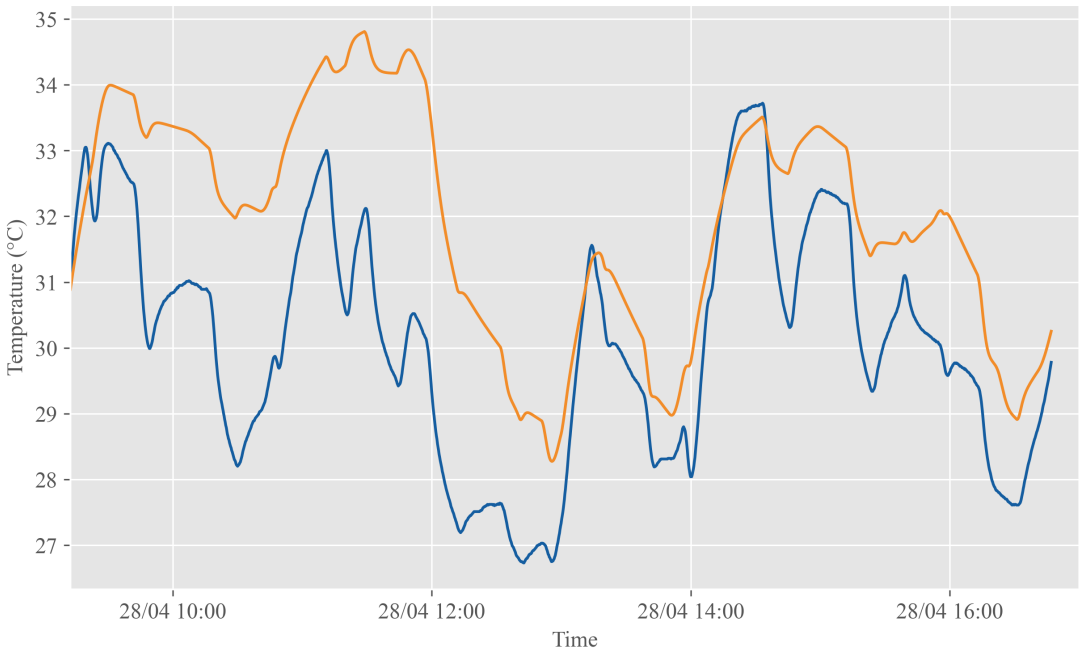
(a) Measurement and prediction of temperature for scenario 2 using LSTM to predict derivative



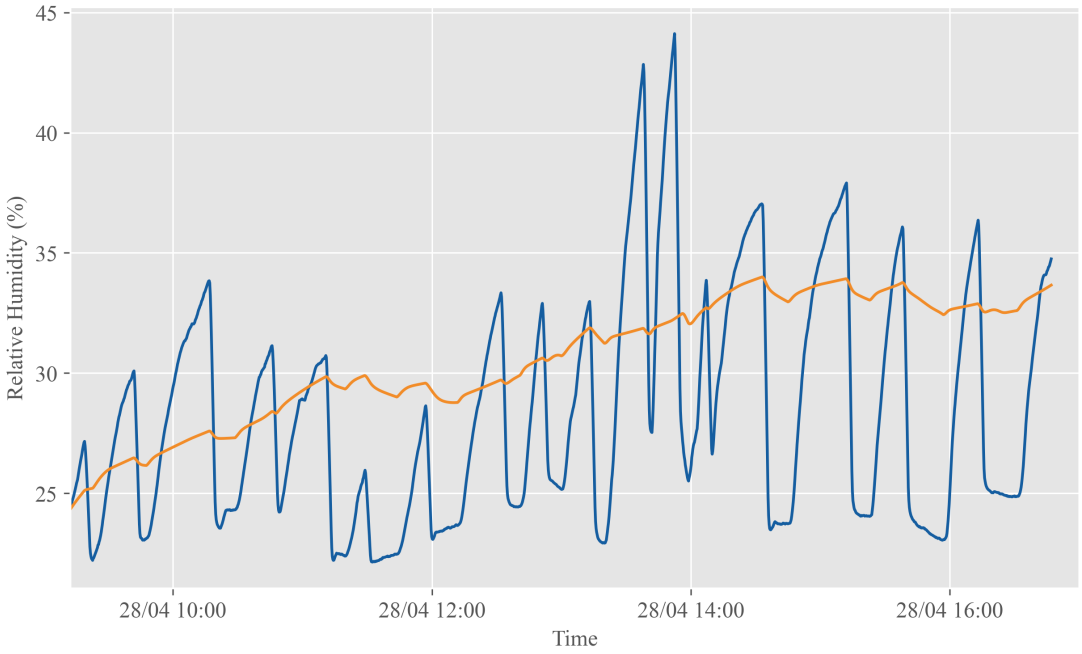
(b) Measurement and prediction of humidity for scenario 2 using LSTM to predict derivative

Measurement Prediction

Figure 4.19.: Measurements and predictions of temperature (figure 4.19a) and humidity (figure 4.19b) on scenario 2. The predictions are made using a LSTM to predict the derivative at each time step.



(a) Measurement and prediction of temperature in greenhouse using LSTM to predict derivative



(b) Measurement and prediction of humidity in greenhouse using LSTM to predict derivative

Figure 4.20.: Measurements and predictions of temperature (figure 4.20a) and humidity (figure 4.20b) on scenario 3. The predictions are made using a LSTM to predict the derivative at each time step.

Table 4.10.: Metrics for prediction of ECD in scenario 1.

	Value
RMSE	0.0156
MAE	0.0086
MAPE	0.0059

All models have difficulty in predicting humidity, especially in scenarios 2 and 3, while they perform better on temperature. Therefore, only temperature prediction will be considered for selecting the best model for oil drilling data. For temperature prediction in scenario 3, the FFNN and LSTM models have similar performance metrics, with the LSTM model having slightly better performance.

To further distinguish between the models, the plots of the predicted and actual values are examined. Figures figure 4.14 and figure 4.17 show the plots of the FFNN and LSTM models for temperature prediction in scenario 3, respectively. The LSTM model seems to capture the pattern of the temperature more accurately than the FFNN model. Therefore, based on both quantitative and qualitative analysis, the LSTM model is selected as the best model for predicting oil drilling data.

4.2.2. Field-scale: Oil Drilling Hole Cleaning Process

Evaluation of the model's performance on the greenhouse data suggests that the LSTM model may be the best-performing model, both when it comes to both performances on the simple model and when generalizing to more complex data. To further examine the possibilities of this model, it is tested on the different scenarios of the oil drilling data.

LSTM to Predict Next Value

The predictions for scenario 1 are shown in figure 4.21 and the corresponding metrics are shown in table 4.10. In this scenario, the model was trained on a data set that contained a different pattern from the test set, with some samples of the same pattern also included. The plot of the actual and predicted ECD shows that the model was able to learn the pattern from the data with a small exposure to it.

The range of the ECD in the test set for scenario 1 was 0.08 (see table 3.7). The RMSE for the ECD prediction is 0.016, which is relatively high compared to the ECD range. The MAE are 0.0086, which is much lower than the RMSE. This suggests that there are some outliers in the errors that increased the RMSE. By looking at figure 3.9, it can be seen that the prediction follows the pattern well, but some points, especially towards the end, have errors. The MAPE was 0.0059, which indicates that the prediction has a low error relative to the actual values and therefore a good performance considering the values of the actual ECD at each time step.

In scenario 2, the model is trained only on the initial pattern in the training data and tested on the other pattern. The prediction and actual value of the ECD are shown in figure 4.22. From this, it is clear that when the new pattern is not shown to the model during training, it struggled with predictions. The model started too low and then gave too high predictions throughout the prediction horizon. However, the predictions follow the dips in the data well. As for the greenhouse data, one possible reason for this bias could be that the mean and standard deviation used for standardizing and de-standardizing the data is different for the training and test sets. The standard deviation differs significantly, with a value of 0.03 in the training set and 0.01 in the test set.

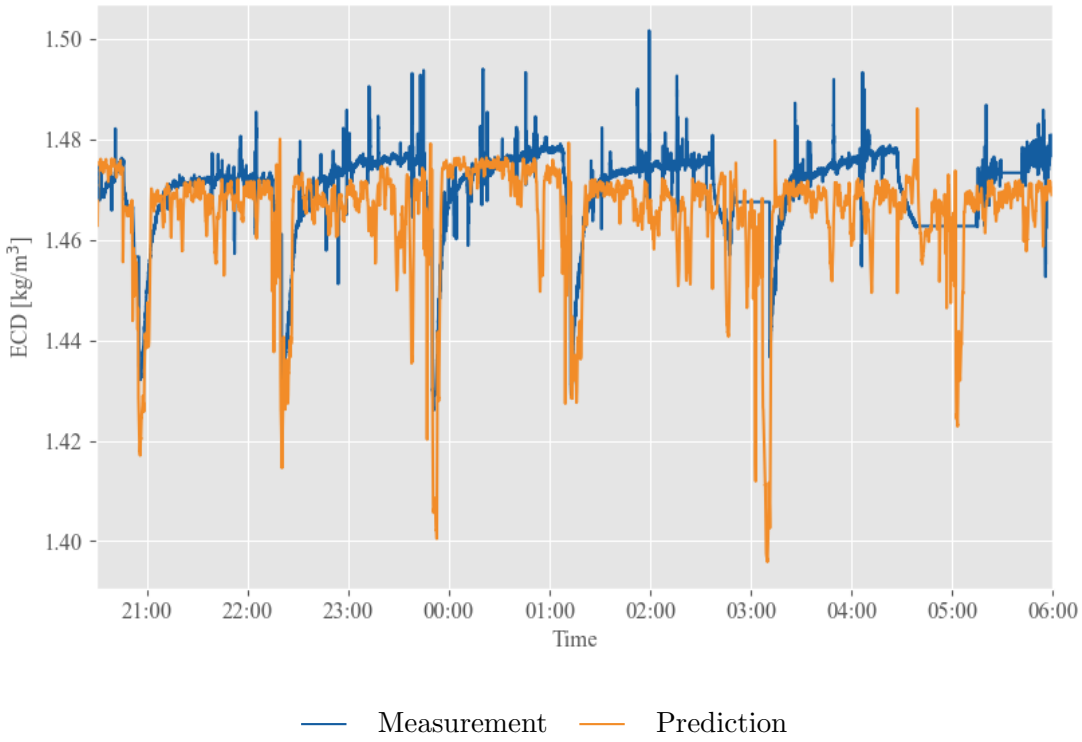


Figure 4.21.: Measurement of ECD and prediction of ECD in scenario 1. The model gets a brief taste of the new pattern during training and manages to predict well.

Table 4.11.: Metrics for prediction of ECD in scenario 2.

	Value
RMSE	0.0372
MAE	0.0310
MAPE	0.0212

The metrics for scenario 2 are presented in table 4.11. For this scenario, the range of the ECD is 0.09 (see table 3.9). This is a similar range as in scenario 1, but comparing it with the RMSE of 0.372, a MAE of 0.0310 and a MAPE of 0.0212 with those in scenario 1, this indicates a worse fit. As already mentioned, the model is biased throughout the prediction, therefore a MAE larger than the range of the data is not surprising.

Scenario 3 also involved training the model on data with one pattern and testing it on data with another pattern, so a similar performance as in scenario 2 was expected. The prediction and actual ECD for this scenario are shown in figure 4.23. As can be seen, the prediction tends to overestimate the ECD compared to the actual ECD. However, the model also captures the dips in the data well, although they seemed to be of a greater magnitude than the ones in the actual ECD measurements. Looking at the statistical properties of case A and case B in table 3.10 and table 3.11, respectively, case A has a standard deviation of 0.02, while case B has a standard deviation of 0.01. Considering that the data were standardized and de-standardized during prediction using the standard deviation of case A, this mismatch is expected.

The metrics for scenario 3 are presented in table 4.12. As mentioned in the previous paragraph, the model performed similarly to the one in scenario 2. This is also reflected

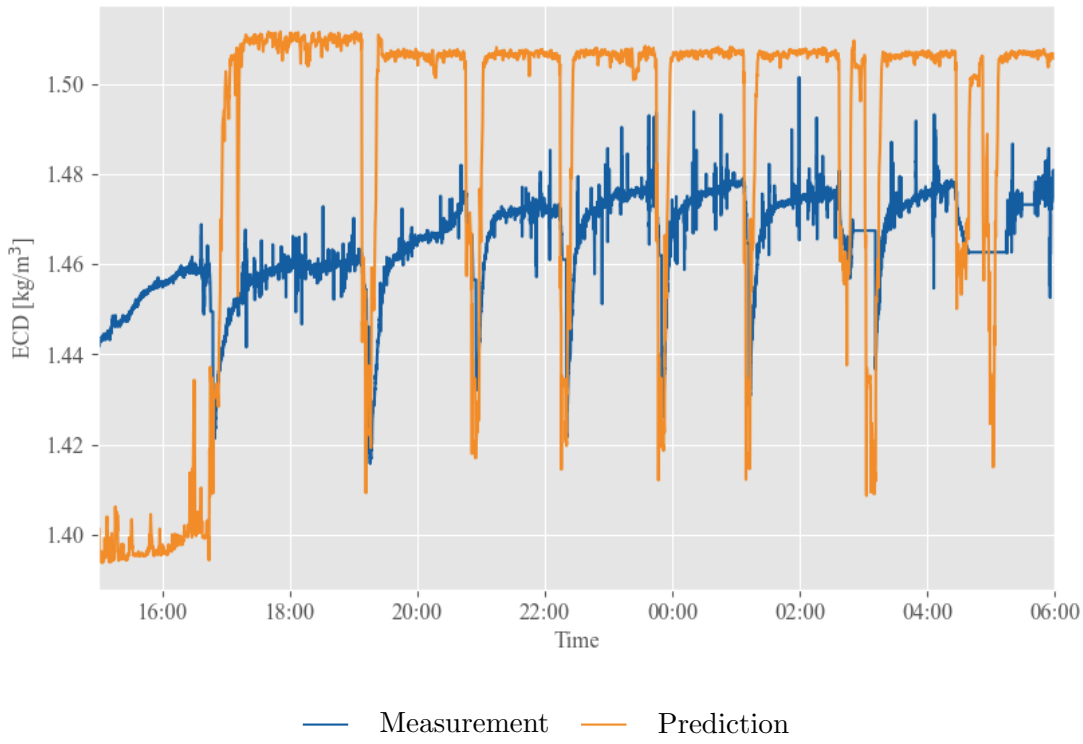


Figure 4.22.: Actual ECD and prediction of ECD using LSTM model for scenario 2. The model is trained solely on a different pattern than in the data set, resulting in biased predictions. But the prediction seems to follow the pattern of the actual ECD in a good way.

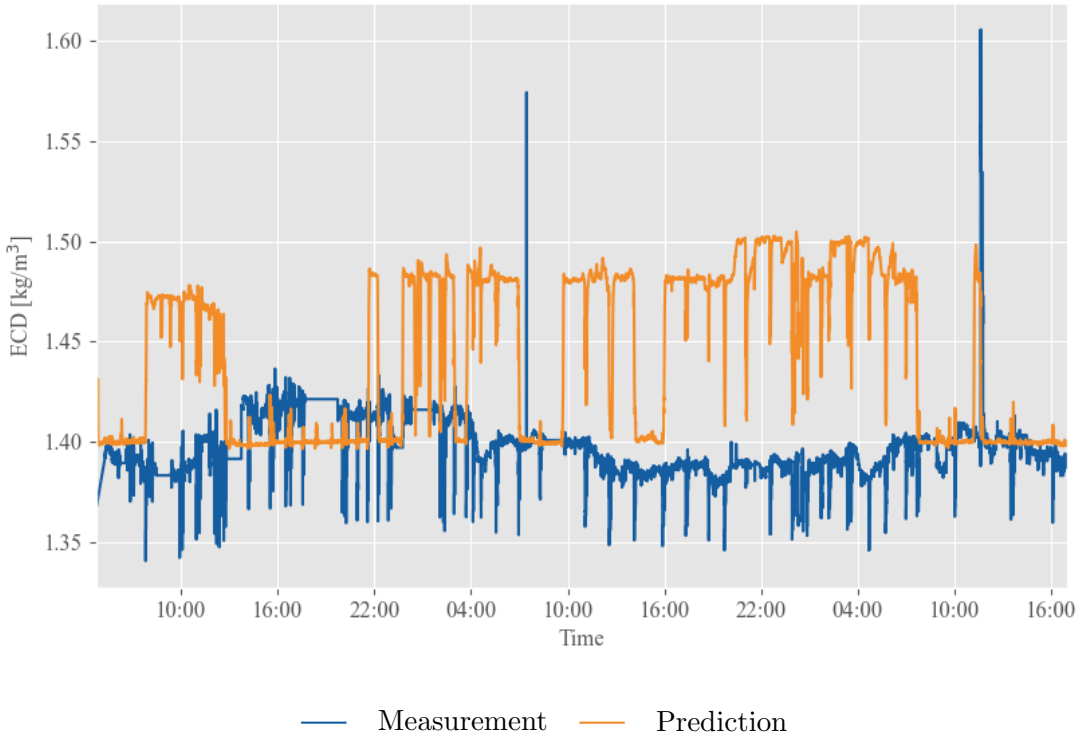


Figure 4.23.: Actual ECD measurements and predictions of ECD using LSRM model for scenario 3. The model is trained on case A and the predictions are done on case B.

Table 4.12.: Metrics for prediction of ECD in scenario 3.

	Value
RMSE	0.0664
MAE	0.0526
MAPE	0.0378

by the performance metrics. The range of the ECD in case B is 0.085, the RMSE for the prediction is 0.0664, the MAE is 0.0529, and the MAPE is 0.0378. This is in the same range as for scenario 2.

The above results show that the LSTM model also performed well on the oil drilling data when it is trained on a data set that contained some samples of the same pattern as the test set. However, when the model is trained on a data set that has a different pattern from the test set, it fails to predict the ECD accurately. This suggests that the LSTM model is sensitive to the pattern of the data and requires some exposure to the new pattern during training to adapt to it. The model also seems to be affected by the difference in the mean and standard deviation of the training and test sets, which caused a bias in the predictions. Therefore, getting some estimate of the mean and standard deviation of the teest set and using this for standardization and de-standardization of the data would probably improve the results. The model can capture the dips in the data well but sometimes exaggerates them compared to the actual ECD measurements. This could be due to the high variability of the ECD data and the difficulty of predicting its dynamics. Overall, the LSTM model shows promising results for predicting ECD in both greenhouse and field-scale scenarios, but it needs further improvement and tuning to handle different patterns and reduce errors.

5. Conclusion and Further Work

This thesis aimed to show how readily available (and on-demand generated) lab scale data from a greenhouse can be used to evaluate and compare different time series analysis and forecasting methods applied for hole cleaning in oil drilling. The main goal was to demonstrate how the greenhouse data can serve as a proxy for the oil drilling data, and how the methods and techniques can help to understand and predict hole cleaning performance in oil drilling.

The first objective was to collect and pre-process the greenhouse data and the oil drilling data for time series analysis and forecasting. The data were re-sampled and standardized to make them suitable for the methods and techniques.

The second objective was to apply various time series analysis methods, such as PCA, EMD, EEMD and FFT to the greenhouse data and the oil drilling data to get a better understanding of the underlying dynamics. These methods were able to reveal the interactions between the measurements and controls in the greenhouse data, such as the temperature and heater duty cycle, light intensity and lighting, and humidity and temperature. Therefore, PCA was applied to investigate the dynamics of the oil drilling controls and measurements, such as the flow rate, pressure and ECD. The results showed that PCA could capture the main variations in the data and identify the most influential variables for each scenario.

The third objective was to identify the best methods for prediction of ECD in oil drilling, using the greenhouse data as a proxy. The greenhouse data was used to generate different data with different patterns and three scenarios was constructed, each with different patterns and complexities. A FNN model for predicting the derivative, a LSTM model for predicting the next value and a LSTM mode for predicting the derivative were applied to greenhouse data and evaluated. All models performed well on simple data, but for more complex data, especially the humidity predictions were poor. The LSTM and FFNN models had good generalizability, performing well on totally different patterns. The LSTM model was the best model and it was used for prediction of ECD on oil drilling data.

For ECD predictions, the LSTM model performed well when predicting on already seen patterns, even if it was just a small peak. For the models with different patterns, it struggled more. It got the patterns, but was biased due to a difference in the mean and standard deviation of the different data sets. These were used for standardization and de-standardization of the data before feeding it to the model and after getting the model prediction.

The main contribution of this thesis is to show how inexpensive lab scale data from a greenhouse can be used to develop, test and validate different time series methods and techniques for analysis and forecasting on hole cleaning data from oil drilling. This can help to reduce the cost and risk of collecting and processing real oil drilling data, which is often scarce, noisy and expensive. The thesis also demonstrated how PCA, EMD, EEMD, FFT, FFNN and LSTM can be applied to both greenhouse data and oil drilling data to understand and predict their dynamics and performance. The results showed that these methods and techniques can provide useful insights and accurate predictions for both scenarios, but they also have some limitations and challenges.

A limitation of this research is that the greenhouse data and the oil drilling data are not exactly equivalent, and there are major differences in the physical processes and their noise levels. Therefore, the generalizability and the validity of the results may be limited to the specific scenarios and the data sets used in this research.

Some directions for future work are:

- Create lab scale setups that has more similarities with hole cleaning in oil drilling. This will increase the similarities of the dynamics between the inexpensive lab scale data and the expensive field scale data. Consequently, the evaluation of the models on the lab scale data would better resemble their performance on the field scale data.
- Clearly, using a simple LSTM for prediction on data with such different patterns as the ones presented in this thesis is not enough. Further improvements on the models architectures, optimizers, loss functions and regularization techniques needs to be investigated. Continuous learning with LSTM models seems to be a good approach.
- Explore other aspects of hole cleaning performance that were not covered in this research, such as detecting anomalies. This can help to provide more comprehensive and useful information for improving hole cleaning performance in oil drilling. Also, this can help removing anomalies from training sets and consequently improve model performance.

Bibliography

[1] Hole cleaning. https://petrowiki.spe.org/Hole_cleaning, 2021. Accessed: 2022-05-27.

[2] Ahmed Alsaihati, Salaheldin Elkatatny, and Abdulazeez Abdulraheem. Real-time prediction of equivalent circulation density for horizontal wells using intelligent machines. *ACS Omega*, 6(1):934–942, 2021. doi: 10.1021/acsomega.0c05570. URL https://doi.org/10.1021/acsomega.0c05570. PMID: 33458545.

[3] Parimal Arjun Patil and Catalin Teodoriu. Model Development of Torsional Drillstring and Investigating Parametrically the Stick-Slips Influencing Factors. *Journal of Energy Resources Technology*, 135(1), 12 2012. ISSN 0195-0738. doi: 10.1115/1.4007915. URL https://doi.org/10.1115/1.4007915. 013103.

[4] Kenichi Azuma, Naoki Kagi, U. Yanagi, and Haruki Osawa. Effects of low-level inhalation exposure to carbon dioxide in indoor environments: A short review on human health and psychomotor performance. *Environment International*, 121:51–56, 2018. ISSN 0160-4120. doi: https://doi.org/10.1016/j.envint.2018.08.059. URL https://www.sciencedirect.com/science/article/pii/S0160412018312807.

[5] Foued Badrouchi, Vamegh Rasouli, and Nidhal Badrouchi. Impact of hole cleaning and drilling performance on the equivalent circulating density. *Journal of Petroleum Science and Engineering*, 211:110150, 2022. ISSN 0920-4105. doi: https://doi.org/10.1016/j.petrol.2022.110150. URL https://www.sciencedirect.com/science/article/pii/S0920410522000444.

[6] Cristina Baglivo, Domenico Mazzeo, Simone Panico, Sara Bonuso, Nicoletta Matera, Paolo Maria Congedo, and Giuseppe Oliveti. Complete greenhouse dynamic simulation tool to assess the crop thermal well-being and energy needs. *Applied Thermal Engineering*, 179:115698, 2020. ISSN 1359-4311. doi: https://doi.org/10.1016/j.applthermaleng.2020.115698. URL https://www.sciencedirect.com/science/article/pii/S135943112033180X.

[7] Estela Bee Dagum. Time series modeling and decomposition. *Statistica*, 70 (4):433–457, Jan. 2010. doi: 10.6092/issn.1973-2201/3597. URL https://rivista-statistica.unibo.it/article/view/3597.

[8] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. doi: 10.1109/72.279181.

[9] Mohsen Besanjideh. Nonlinear and non-stationary vibration analysis for mechanical fault detection using emd-fft method. *International Journal of Engineering*, 25, 12 2012. doi: 10.5829/idosi.ije.2012.25.04c.11.

[10] Peter Bloomfield. *Fourier analysis of time series: an introduction*. John Wiley & Sons, 2004.

[11] Abdel-O Boudraa, Jean-Christophe Cexus, Salem Benramdane, and Azeddine Beghdadi. Noise filtering using empirical mode decomposition. pages 1 – 4, 03 2007. doi: 10.1109/ISSPA.2007.4555624.

[12] Endre Bruaset. Experimental set-up for research on digital twins. 2022.

[13] Bashir Busahmin. Review on hole cleaning for horizontal wells. *Journal of Engineering and Applied Sciences*, Volume 12:NO. 16, 08 2017.

[14] Gaofeng Cheng, Vijayaditya Peddinti, Daniel Povey, Vimal Manohar, Sanjeev Khudanpur, and Yonghong Yan. An exploration of dropout with lstms. In *Interspeech*, pages 1586–1590, 2017.

[15] Salah Elkatatny. New approach to optimize the rate of penetration using artificial neural network. *Arab J Sci Eng*, 43:6297–6304, 2018. doi: 10.1007/s13369-017-3022-0.

[16] K.A. Fattah and A. Lashin. Investigation of mud density and weighting materials effect on drilling fluid filter cake properties and formation damage. *Journal of African Earth Sciences*, 117:345–357, 2016. ISSN 1464-343X. doi: https://doi.org/10.1016/j.jafrearsci.2016.02.003. URL https://www.sciencedirect.com/science/article/pii/S1464343X16300498.

[17] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901. doi: 10.1080/14786440109462720. URL https://doi.org/10.1080/14786440109462720.

[18] John Cristian Borges Gamboa. Deep learning for time-series analysis. *CoRR*, abs/1701.01887, 2017. URL http://arxiv.org/abs/1701.01887.

[19] Geetikaverma and Vikramjit Singh. Empirical wavelet transform & its comparison with empirical mode decomposition: A review. *International journal of engineering research and technology*, 4, 2018.

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[21] S.L. Ho and M. Xie. The use of arima models for reliability forecasting and analysis. *Computers & Industrial Engineering*, 35(1):213–216, 1998. ISSN 0360-8352. doi: https://doi.org/10.1016/S0360-8352(98)00066-7. URL https://www.sciencedirect.com/science/article/pii/S0360835298000667.

[22] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(89)90020-8. URL https://www.sciencedirect.com/science/article/pii/0893608089900208.

[23] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:498–520, 1933.

[24] Norden E. Huang, Zheng Shen, Steven R. Long, Manli C. Wu, Hsing H. Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H. Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary

time series analysis. *Proceedings: Mathematical, Physical and Engineering Sciences*, 454(1971):903–995, 1998. ISSN 13645021. URL http://www.jstor.org/stable/53161.

[25] Plotly Technologies Inc. Collaborative data science, 2015. URL https://plot.ly.

[26] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical,Physical and Engineering Sciences*, 374(2065):20150202, 2016. doi: 10.1098/rsta.2015.0202. URL http://doi.org/10.1098/rsta.2015.0202.

[27] Nishant Joshi et al. Analysis of drilling fluid flow in the annulus of drill string through computational fluid dynamics. *Journal of Physics: Conference Series*, 1913:012128, 2021. doi: DOI10.1088/1742-6596/1913/1/012128.

[28] Mingu Kang, Jaehyo Jung, Siho Shin, Kyeung Ho Kang, and Youn Tae Kim. Multi bio-signal based algorithm using emd and fft for stress analysis. In *2020 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–4, 2020. doi: 10.1109/ICCE46568.2020.9043087.

[29] Moji Karimi. Drill-cuttings analysis for real-time problem diagnosis and drilling performance optimization. In *SPE Asia Pacific oil and gas conference and exhibition*. OnePetro, 2013.

[30] L. Karthikeyan and D. Nagesh Kumar. Predictability of nonstationary time series using wavelet and emd based arma models. *Journal of Hydrology*, 502:103–119, 2013. ISSN 0022-1694. doi: https://doi.org/10.1016/j.jhydrol.2013.08.030. URL https://www.sciencedirect.com/science/article/pii/S0022169413006173.

[31] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[32] G. Ganesan Kumar, Sushanta K. Sahoo, and Pramod K. Meher. 50 years of fft algorithms and applications. *Circuits, Systems, and Signal Processing*, 38:5665–5698, 2019. doi: 10.1007/s00034-019-01136-8.

[33] L.W. Lake and Society of Petroleum Engineers (U.S.). *Petroleum Engineering Handbook*. Society of Petroleum Engineers, 2007. ISBN 9781555631260. URL https://books.google.no/books?id=0ggiSgAACAAJ.

[34] Dawid Laszuk. Python implementation of empirical mode decomposition algorithm. https://github.com/laszukdawid/PyEMD, 2017.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015.

[36] Christian Møldrup Legaard, Thomas Schranz, Gerald Schweiger, Ján Drgoňa, Basak Falay, Cláudio Gomes, Alexandros Iosifidis, Mahdi Abkar, and Peter Gorm Larsen. Constructing neural network-based models for simulating dynamical systems, 2021. URL https://arxiv.org/abs/2111.01495.

[37] Sonam Maheshwari and Ankur Kumar. Empirical mode decomposition : Theory & applications. *International Journal of Electronic and Electrical Engineering*, 7(8), 2014. ISSN 09742174.

[38] Chris V. Nicholson. A beginner's guide to lstms and recurrent neural networks. URL https://wiki.pathmind.com/lstm.

[39] Michael A. Nielsen. *Neural Networks and Deep learning*. Determination Press, 2015. URL http://neuralnetworksanddeeplearning.com/.

[40] Yanrui Ning, Hossein Kazemi, and Pejman Tahmasebi. A comparative machine learning study for time series oil production forecasting: Arima, lstm, and prophet. *Computers & Geosciences*, 164:105126, 2022. ISSN 0098-3004. doi: https://doi.org/10.1016/j.cageo.2022.105126. URL https://www.sciencedirect.com/science/article/pii/S009830042200084X.

[41] Alison Nugent, David DeCou, Shintaro Russell, and Christina Karamperidou. *Atmospheric Processes and Phenomena*. University of Hawai'i Pressbooks, 2021. URL http://pressbooks-dev.oer.hawaii.edu/atmo/.

[42] Sebin Park, Myeong-Seon Gil, Hyeonseung Im, and Yang-Sae Moon. Measurement noise recommendation for efficient kalman filtering over a large amount of sensor data. *Sensors*, 19(5), 2019. ISSN 1424-8220. URL https://www.mdpi.com/1424-8220/19/5/1168.

[43] Warren M Persons. Indices of business conditions. *Review of Economic Statistics*, 1(1):5–107, 1919.

[44] Lutz Prechelt. *Early Stopping - But When?*, pages 55–69. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[45] Mohammed H. Rasheed, Omar M. Salih, Mohammed M. Siddeq, and Marcos A. Rodrigues. Image compression based on 2d discrete fourier transform and matrix minimization algorithm. *Array*, 6:100024, 2020. ISSN 2590-0056. doi: https://doi.org/10.1016/j.array.2020.100024. URL https://www.sciencedirect.com/science/article/pii/S2590005620300096.

[46] Moin Rezvani, Redmond Shamshiri, Ibrahim Hameed, Hamid Abyane, Mohsen Godarzi, Davood Momeni, and Siva Balasundram. *Greenhouse Crop Simulation Models and Microclimate Control Systems, A Review*. 05 2021. ISBN 978-1-83968-075-5. doi: 10.5772/intechopen.97361.

[47] F. Rodríguez, L.J. Yebra, M. Berenguel, and S. Dormido. Modelling and simulation of greenhouse climate using dymola. *IFAC Proceedings Volumes*, 35(1):79–84, 2002. ISSN 1474-6670. doi: https://doi.org/10.3182/20020721-6-ES-1901.01322. URL https://www.sciencedirect.com/science/article/pii/S1474667015397433. 15th IFAC World Congress.

[48] Shakir Saat, Sing Kiong Nguang, and Alireza Nasiri. Chapter 1 - introduction. In Shakir Saat, Sing Kiong Nguang, and Alireza Nasiri, editors, *Analysis and Synthesis of Polynomial Discrete-Time Systems*, pages 1–27. Butterworth-Heinemann, 2017. ISBN 978-0-08-101901-6. doi: https://doi.org/10.1016/B978-0-08-101901-6.00001-3. URL https://www.sciencedirect.com/science/article/pii/B9780081019016000013.

[49] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[50] Robin M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview. *CoRR*, abs/1912.05911, 2019. URL http://arxiv.org/abs/1912.05911.

[51] Ervin Sejdić, Igor Djurović, and Jin Jiang. Time–frequency feature representation using energy concentration: An overview of recent advances. *Digital Signal Processing*, 19(1):153–183, 2009. ISSN 1051-2004. doi: https://doi.org/10.1016/j.dsp.2007.12.004. URL https://www.sciencedirect.com/science/article/pii/S105120040800002X.

[52] Artemios-Anargyros Semenoglou, Evangelos Spiliotis, Spyros Makridakis, and Vassilios Assimakopoulos. Investigating the accuracy of cross-learning time series forecasting methods. *International Journal of Forecasting*, 37(3):1072–1084, 2021. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2020.11.009. URL https://www.sciencedirect.com/science/article/pii/S0169207020301850.

[53] Sonia I. Seneviratne, Thierry Corti, Edouard L. Davin, Martin Hirschi, Eric B. Jaeger, Irene Lehner, Boris Orlowsky, and Adriaan J. Teuling. Investigating soil moisture–climate interactions in a changing climate: A review. *Earth-Science Reviews*, 99(3):125–161, 2010. ISSN 0012-8252. doi: https://doi.org/10.1016/j.earscirev.2010.02.004. URL https://www.sciencedirect.com/science/article/pii/S0012825210000139.

[54] Olli Seppänen and Jarek Kurnitski. Moisture control and ventilation. 2009. URL https://www.ncbi.nlm.nih.gov/books/NBK143947/.

[55] Farah Shahid, Aneela Zameer, and Muhammad Muneeb. Predictions for covid-19 with deep learning models of lstm, gru and bi-lstm. *Chaos, Solitons & Fractals*, 140:110212, 2020.

[56] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *towards data science*, 6(12):310–316, 2017.

[57] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. The performance of lstm and bilstm in forecasting time series. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 3285–3292, 2019. doi: 10.1109/BigData47090.2019.9005997.

[58] Patrik Sleziak, Kamila Hlavčová, and Ján Szolgay. Advantages of a time series analysis using wavelet transform as compared with a fourier analysis. *Slovak Journal of Civil Engineering*, 23(2):30–36, 2015. doi: doi:10.1515/sjce-2015-0010. URL https://doi.org/10.1515/sjce-2015-0010.

[59] Kalamkas Smagulova and Alex P. James. A survey on lstm memristive neural network architectures and applications. *The European Physical Journal Special Topics*, 228(11):2313–2324, 2019. doi: 10.1140/epjst/e2019-900046-x.

[60] Steven W. Smith. Chapter 12 - the fast fourier transform. In Steven W. Smith, editor, *Digital Signal Processing*, pages 225–242. Newnes, Boston, 2003. ISBN 978-0-7506-7444-7. doi: https://doi.org/10.1016/B978-0-7506-7444-7/50049-2. URL https://www.sciencedirect.com/science/article/pii/B9780750674447500492.

[61] J. Sola and J. Sevilla. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science*, 44(3):1464–1468, 1997. doi: 10.1109/23.589532.

[62] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[63] Gian Antonio Susto, Angelo Cenedese, and Matteo Terzi. Chapter 9 - time-series classification methods: Review and applications to power systems data. In Reza Arghandeh and Yuxun Zhou, editors, *Big Data Application in Power Systems*, pages 179–220. Elsevier, 2018. ISBN 978-0-12-811968-6. doi: https://doi.org/10.1016/B978-0-12-811968-6.00009-7. URL https://www.sciencedirect.com/science/article/pii/B9780128119686000097.

[64] Yajiao Tang, Zhenyu Song, Yulin Zhu, Huaiyu Yuan, Maozhang Hou, Junkai Ji, Cheng Tang, and Jianqiang Li. A survey on machine learning models for financial time series forecasting. *Neurocomputing*, 512:363–380, 2022. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2022.09.003. URL https://www.sciencedirect.com/science/article/pii/S092523122201089X.

[65] Diana Taylor. Time-series analysis: Use of autocorrelation as an analytic strategy for describing pattern and change. *Western Journal of Nursing Research*, 12(2): 254–261, 1990.

[66] Yingjie Tian and Yuqi Zhang. A comprehensive survey on regularization strategies in machine learning. *Information Fusion*, 80:146–166, 2022. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2021.11.005. URL https://www.sciencedirect.com/science/article/pii/S156625352100230X.

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[68] M. Farooq Wahab, Fabrice Gritti, and Thomas C. O'Haver. Discrete fourier transform techniques for noise reduction and digital enhancement of analytical signals. *TrAC Trends in Analytical Chemistry*, 143:116354, 2021. ISSN 0165-9936. doi: https://doi.org/10.1016/j.trac.2021.116354. URL https://www.sciencedirect.com/science/article/pii/S0165993621001771.

[69] Xingyu Wang, Hui Liu, Junzhao Du, Xiyao Dong, and Zhihan Yang. A long-term multivariate time series forecasting network combining series decomposition and convolutional neural networks. *Applied Soft Computing*, 139:110214, 2023. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2023.110214. URL https://www.sciencedirect.com/science/article/pii/S1568494623002326.

[70] Yao-Ping Wang, Yan-Rong Zou, Jian-Ting Shi, and Jun Shi. Review of the chemometrics application in oil-oil and oil-source rock correlations. *Journal of Natural Gas Geoscience*, 3(4):217–232, 2018. ISSN 2468-256X. doi: https://doi.org/10.1016/j.jnggs.2018.08.003. URL https://www.sciencedirect.com/science/article/pii/S2468256X1830052X.

[71] Eric W. Weisstein. Discrete Fourier Transform. https://mathworld.wolfram.com/DiscreteFourierTransform.html, accessed on 2023-05-06.

[72] Ronald J. Williams and Jing Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501, 1990. doi: 10.1162/neco.1990.2.4.490.

[73] Monika Woloszyn, Targo Kalamees, Marc Olivier Abadie, Marijke Steeman, and Angela Sasic Kalagasidis. The effect of combining a relative-humidity-sensitive ventilation system with the moisture-buffering capacity of materials on indoor climate and energy efficiency of buildings. *Building and Environment*, 44(3):515–524, 2009. ISSN 0360-1323. doi: https://doi.org/10.1016/j.buildenv.2008.04.017. URL https://www.sciencedirect.com/science/article/pii/S0360132308000772.

[74] Zhaohua Wu and Norden Huang. Ensemble empirical mode decomposition: a noise-assisted data analysis method. *Advances in Adaptive Data Analysis*, 1:1–41, 01 2009. doi: 10.1142/S1793536909000047.

[75] Mehmet Çetin and Hakan Sevik. Measuring the impact of selected plants on indoor co2 concentrations. *Polish Journal of Environmental Studies*, 25, 01 2016. doi: 10.15244/pjoes/61744.

A. Greenhouse Sensor Information

Table A.1.: The sensors used for measuring in the greenhouse.

Sensor:	**Link:**
DHT22 Temperature-Humidity Sensor	www.adafruit.com/product/385
Grove Light Sensor	wiki.seeedstudio.com/Grove-Light_Sensor/
Grove CO2 Sensor	wiki.seeedstudio.com/Grove-CO2_Sensor/
Grove Capacitive Moisture Sensor	wiki.seeedstudio.com/Grove-Capacitive_Moisture...

B. Code

There is quite a significant code base created along with this thesis. It will be available at: https://github.com/emilhaugstvedt/master

C. Model Hyperparameters

Table C.1.: Hyperparameters of FFNN model used for prediction of the derivative of temperature and humidity.

Hyperparameters:	**Hyperparameter value:**
Number of layers	4
Input layer size	4
Hidden layers size	32, 32
Output layer size	2
Activation function	ReLU
Network optimizer	Adam
Learning rate	0.0001
Number of epochs	12
L1	0.001

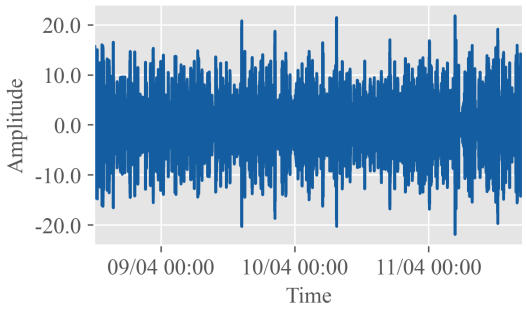
Table C.2.: Hyperparameters of LSTM model used for prediction of next value for temperature and humidity. The LSTM network consists of a LSTM layer and a fully connected output, with each part having its own set of parameters. The first inputs to the table, marked with "(LSTM)" is for the LSTM layer, the next ones, marked with "(ON)" is for the output network, and the last set of parameters is for training. In some parameters there are difference between the model for x and $\dot{x}$, in these cases "(a)" means model for x and (b) means model for $\dot{x}$.

Hyperparameters:	**Hyperparameter value:**
Input size (LSTM)	4
Number of hidden layers (LSTM)	1
Hidden size (LSTM)	32 (a), 64 (b)
Dropout (LSTM)	0.4
Input size (ON)	32 (a), 64 (b)
Number of layers (ON)	3
Output size (ON)	2
Activation function (ON)	ReLU
Network optimizer	Adam
Learning rate	0.0001
Number of epochs	30
L1	0.0001
Lookback	30 (a), 50 (b)

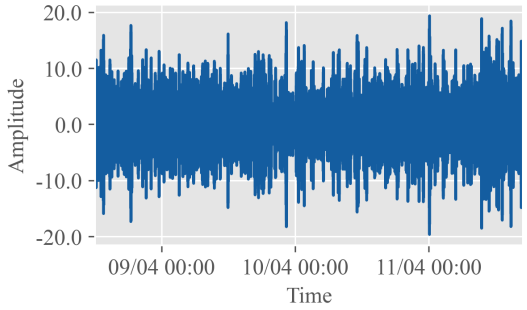
Table C.3.: Hyperparameters for LSTM model used for prediction on drilling data. The first inputs to the table, marked with "(LSTM)" is for the LSTM layer, the next ones, marked with "(ON)" is for the output network, and the last set of parameters is for training.

Hyperparameters:	**Hyperparameter value:**
Input size (LSTM)	5
Number of hidden layers (LSTM)	1
Hidden size (LSTM)	32
Dropout (LSTM)	0.4
Input size (ON)	32
Number of layers (ON)	3
Output size (ON)	2
Activation function (ON)	ReLU
Network optimizer	Adam
Learning rate	0.0001
Number of epochs	20
L1	0.0001
Lookback	30

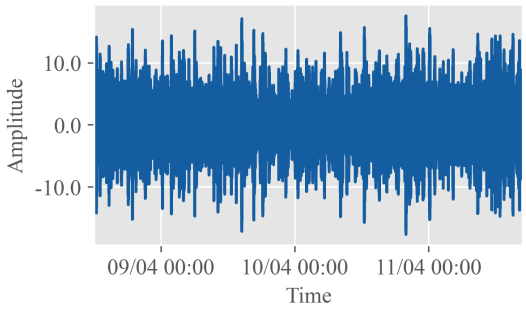
D. IMFs from EMD of Moisture



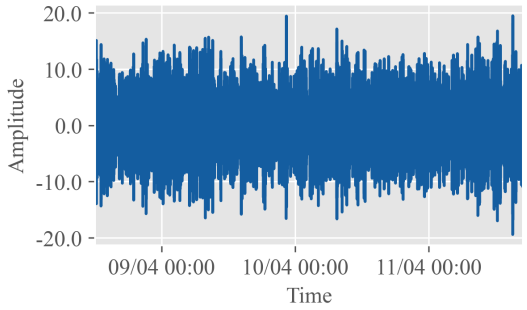
(a) IMF 1



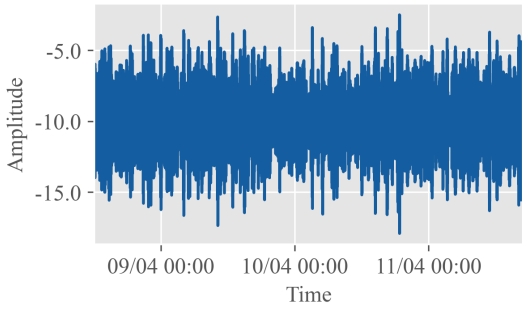
(b) IMF 2



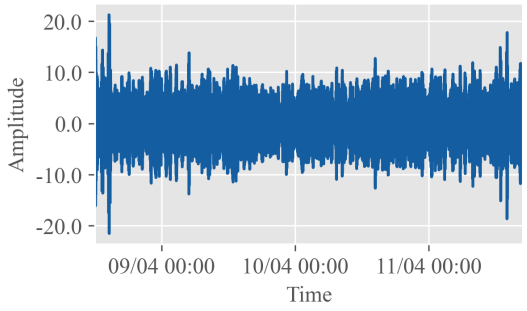
(c) IMF 3



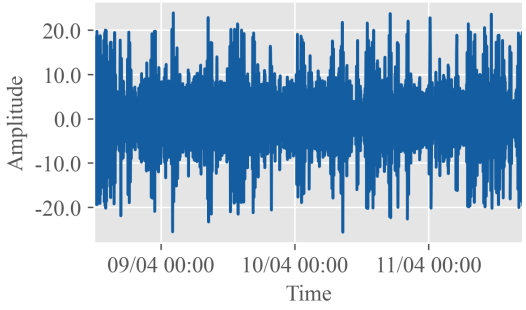
(d) IMF 4



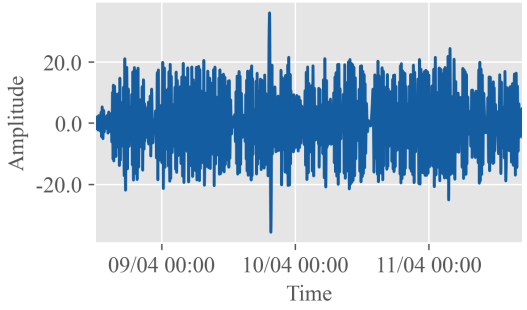
(e) IMF 5



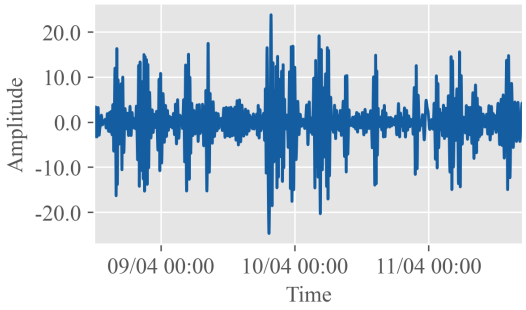
(f) IMF 6



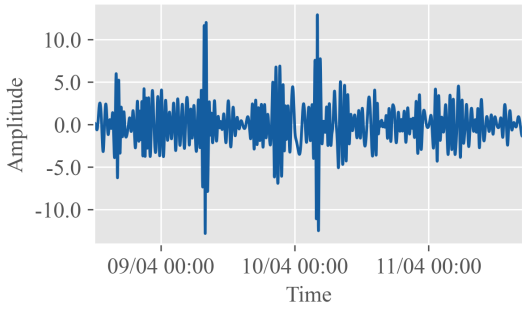
(g) IMF 7



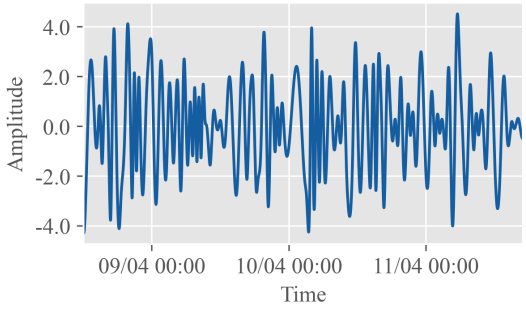
(h) IMF 8



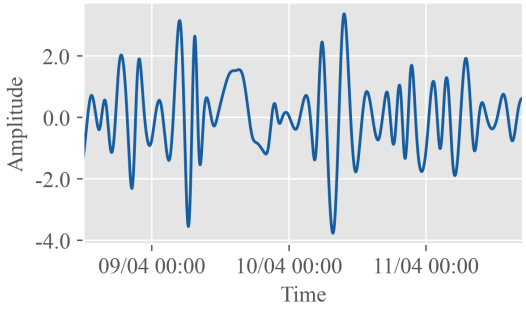
(i) IMF 9



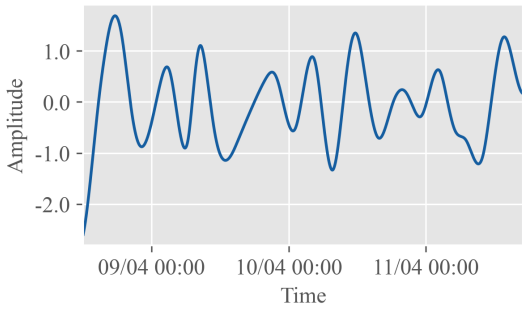
(j) IMF 10



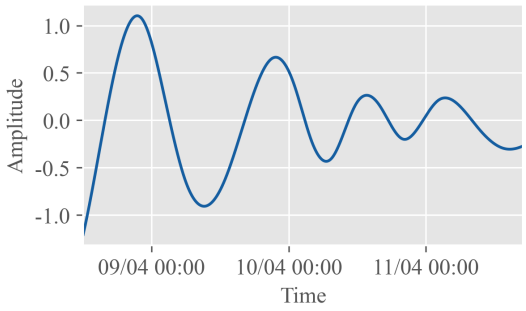
(k) IMF 11



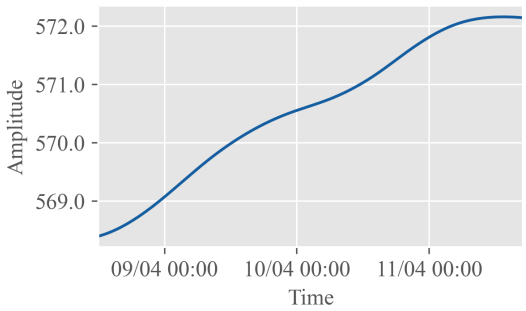
(l) IMF 12



(m) IMF 13



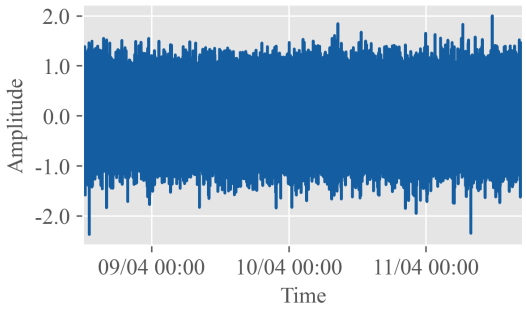
(n) IMF 14



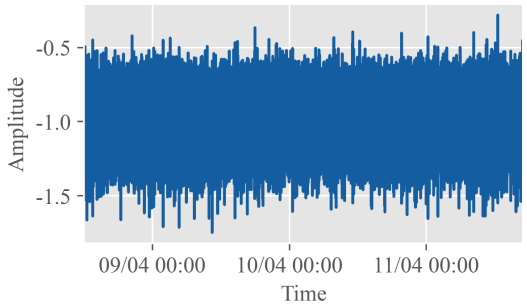
(o) IMF 15

Figure D.1.: All IMFs from EMD of the moisture measurements.

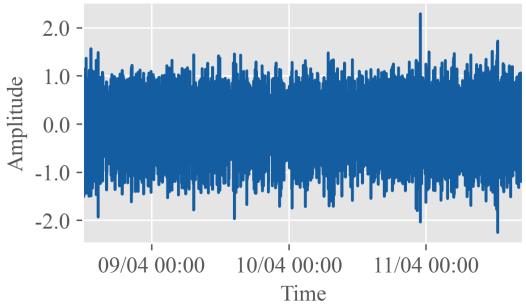
E. IMFs from EEMD of Moisture



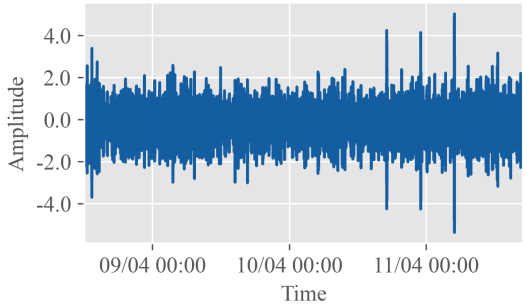
(a) IMF 1



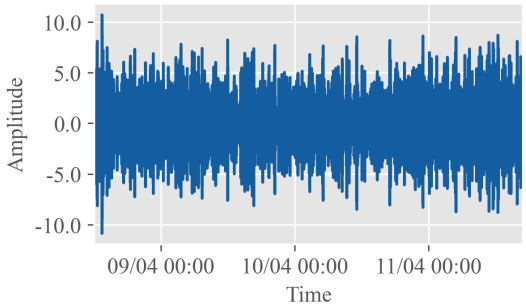
(b) IMF 2



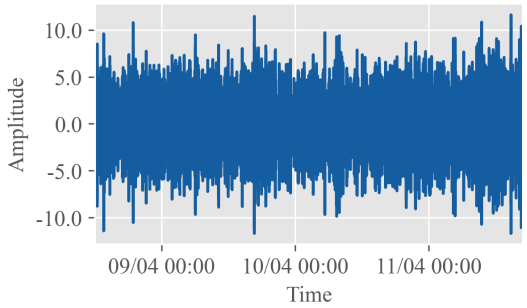
(c) IMF 3



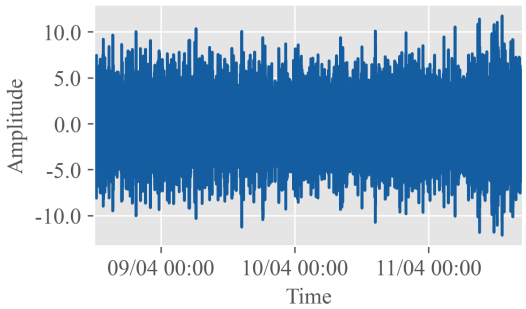
(d) IMF 4



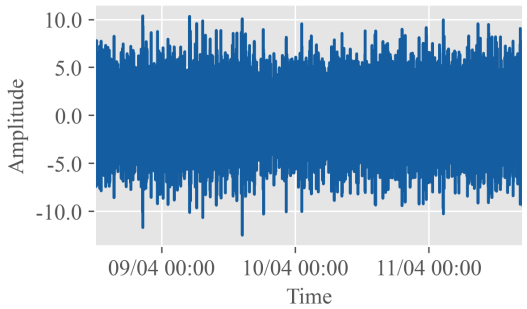
(e) IMF 5



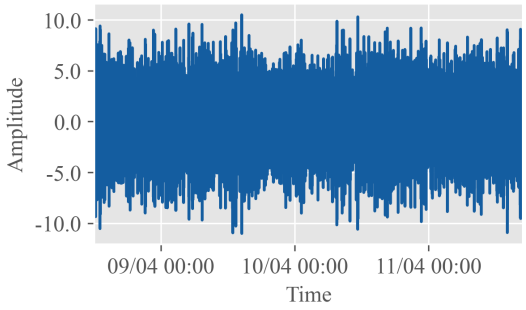
(f) IMF 6



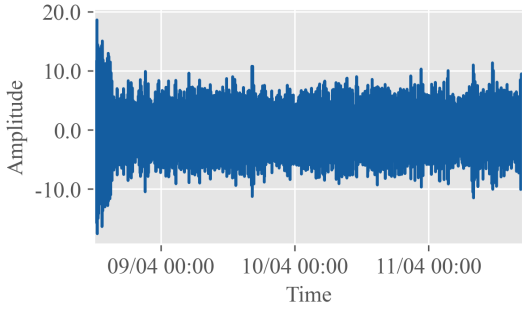
(g) IMF 7



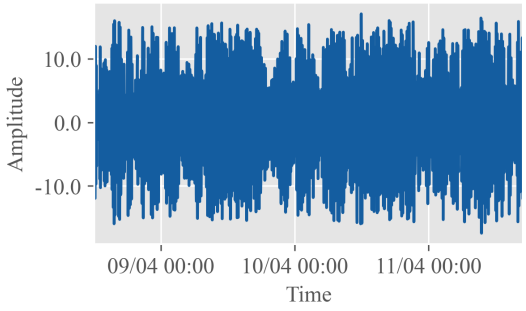
(h) IMF 8



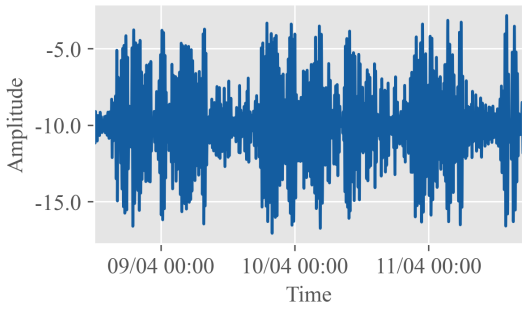
(i) IMF 9



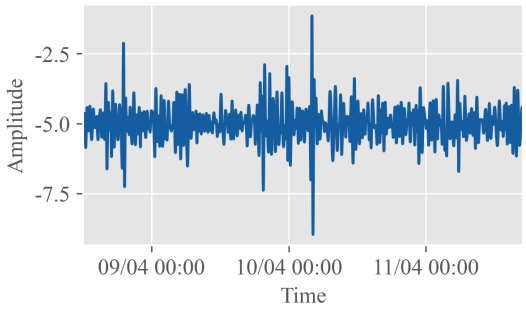
(j) IMF 10



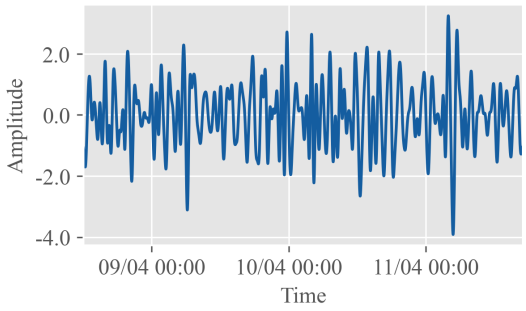
(k) IMF 11



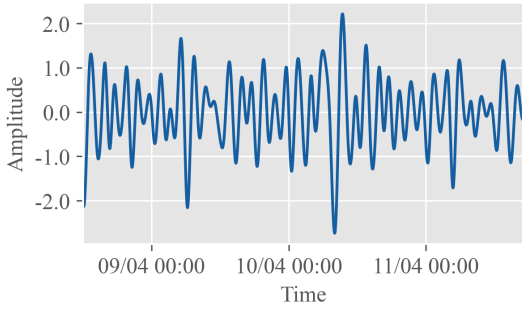
(l) IMF 12



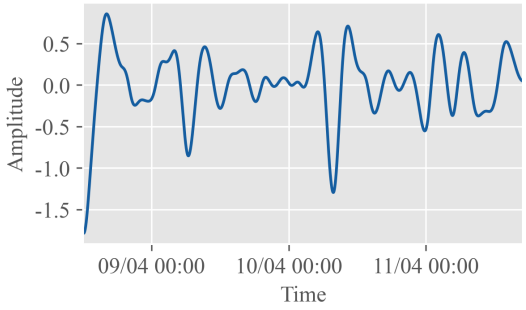
(m) IMF 13



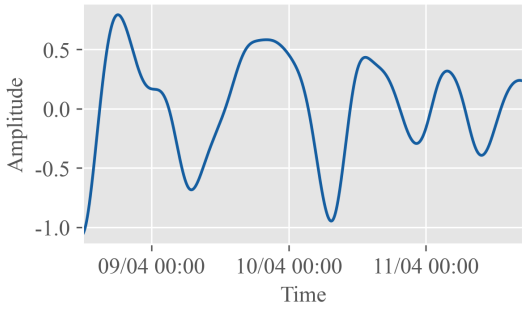
(n) IMF 14



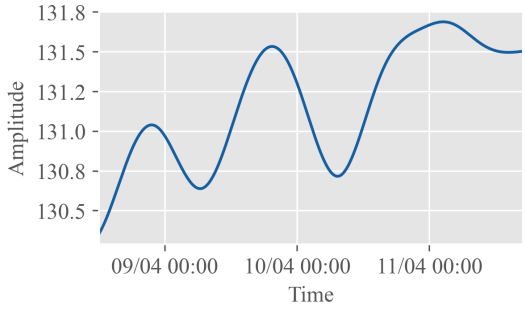
(o) IMF 15



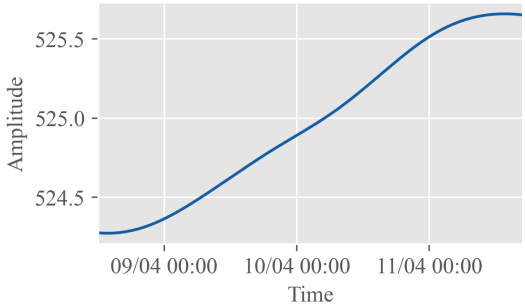
(p) IMF 16



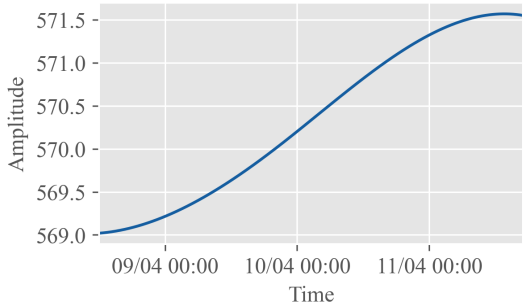
(q) IMF 17



(r) IMF 18



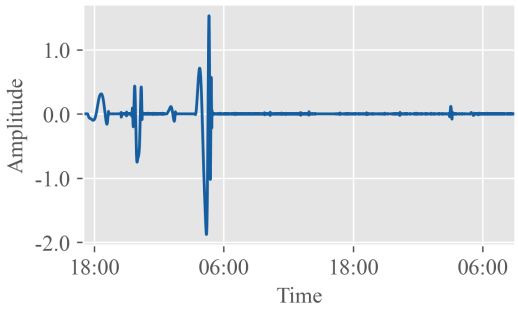
(s) IMF 19



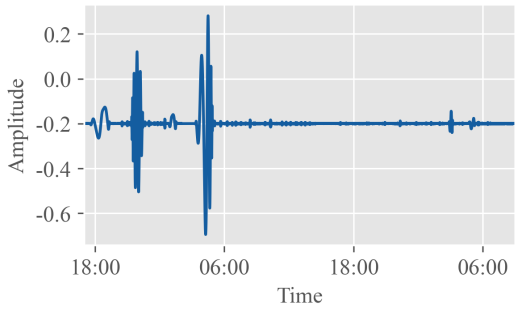
(t) IMF 20

Figure E.1.: All IMFs from EEMD of the moisture measurements.

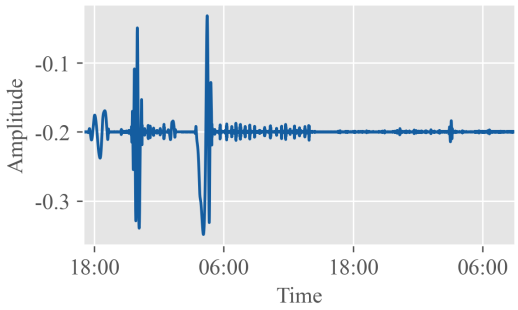
F. IMFs from EMD of ECD



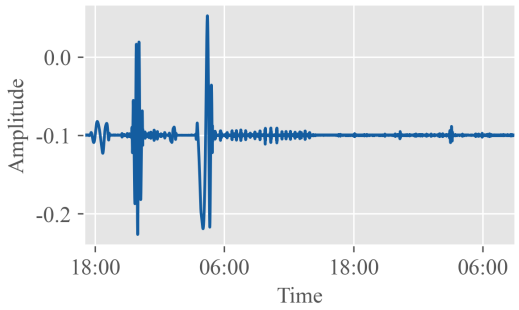
(a) IMF 1



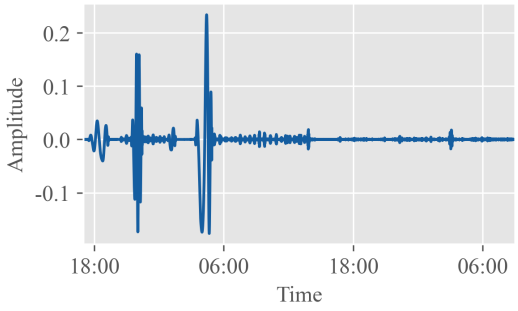
(b) IMF 2



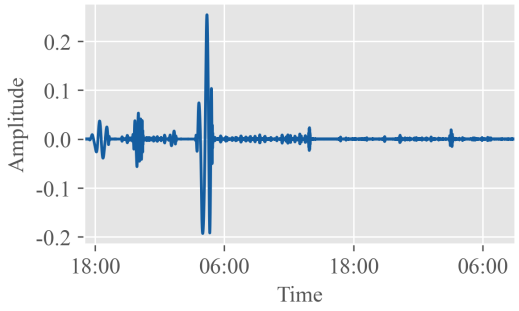
(c) IMF 3



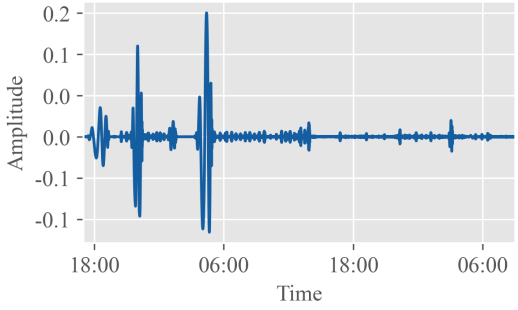
(d) IMF 4



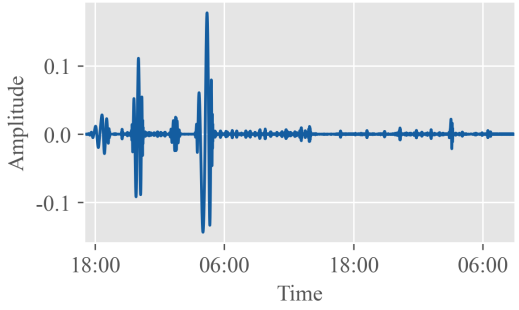
(e) IMF 5



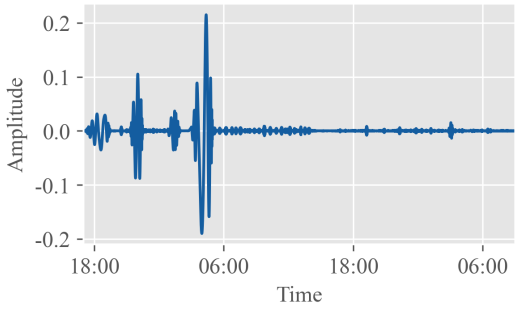
(f) IMF 6



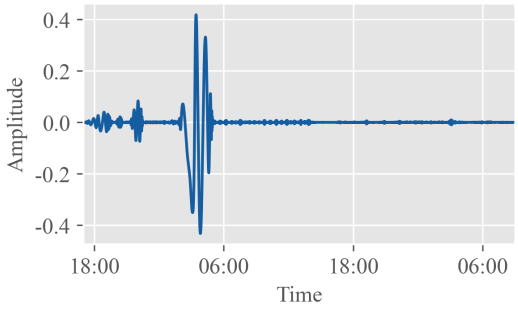
(g) IMF 7



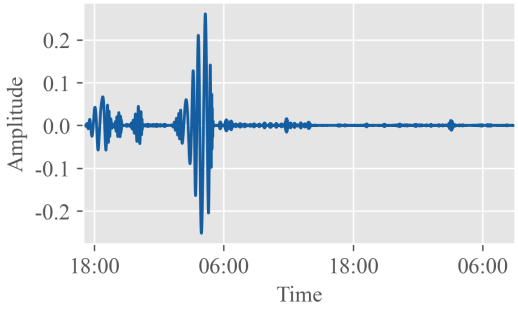
(h) IMF 8



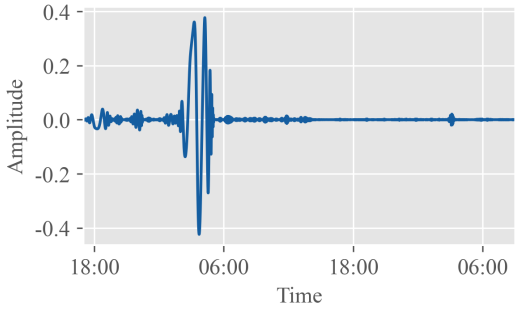
(i) IMF 9



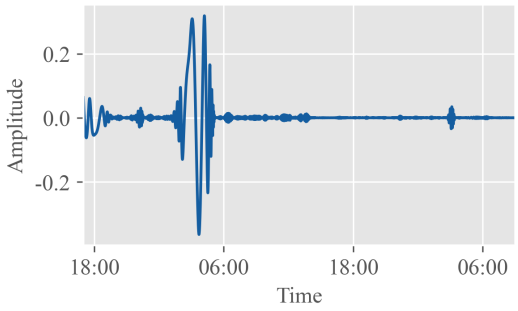
(j) IMF 10



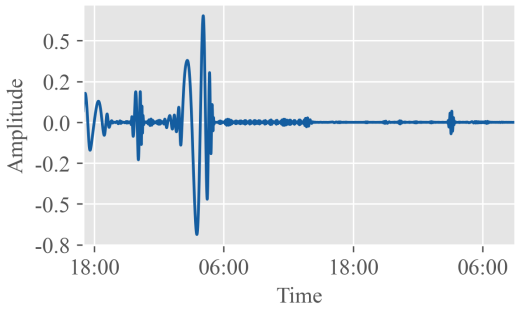
(k) IMF 11



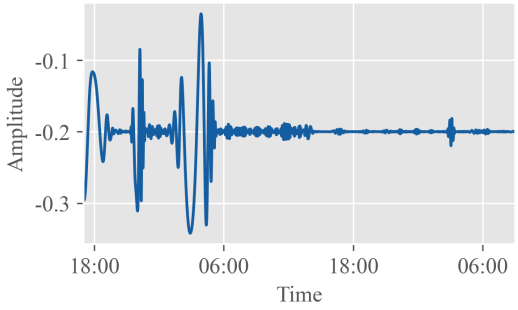
(l) IMF 12



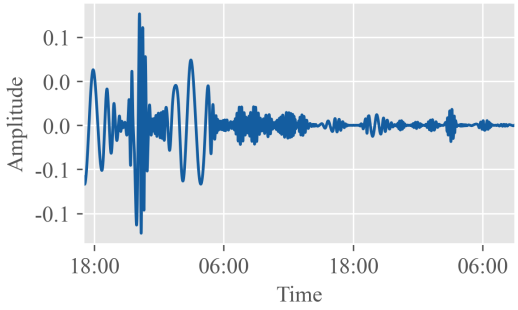
(m) IMF 13



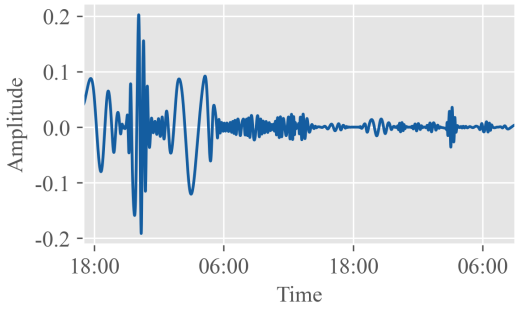
(n) IMF 14



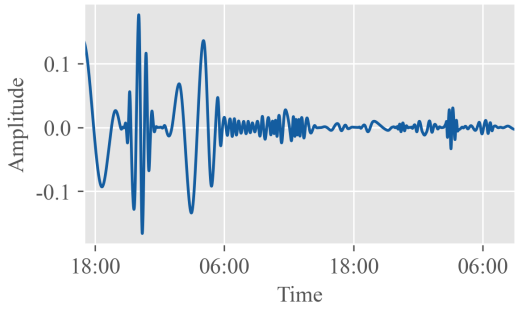
(o) IMF 15



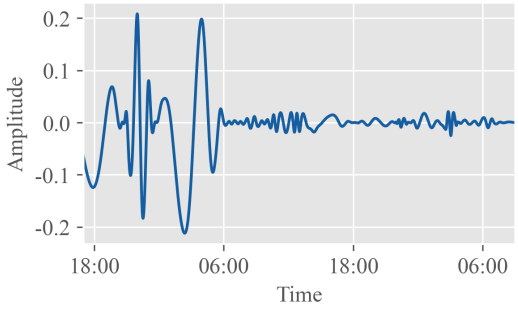
(p) IMF 16



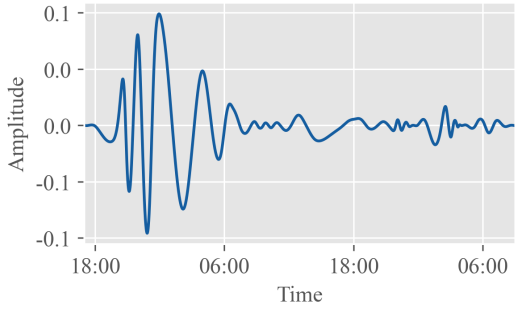
(q) IMF 17



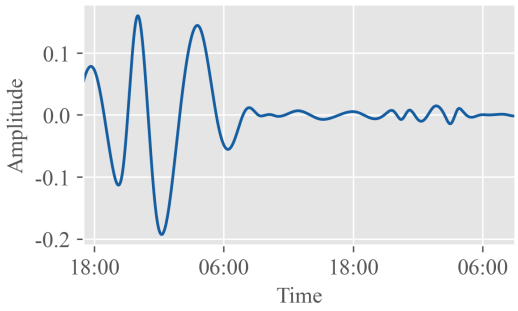
(r) IMF 18



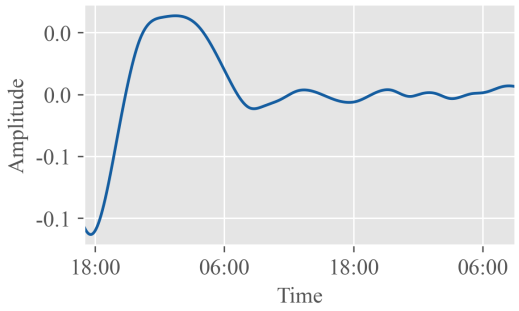
(s) IMF 19



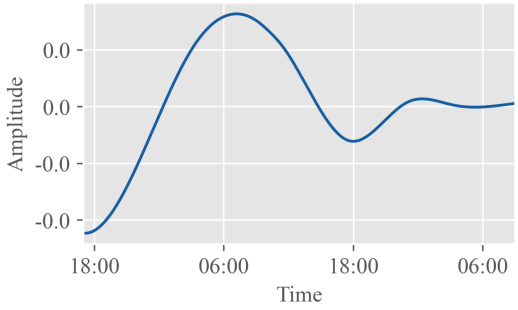
(t) IMF 20



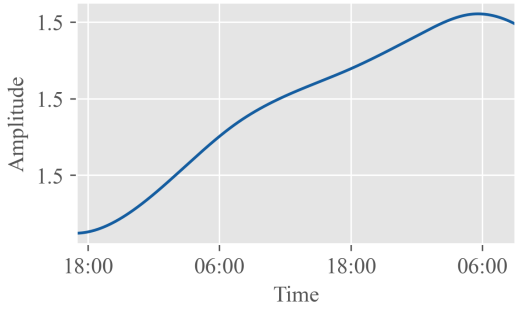
(u) IMF 21



(v) IMF 22



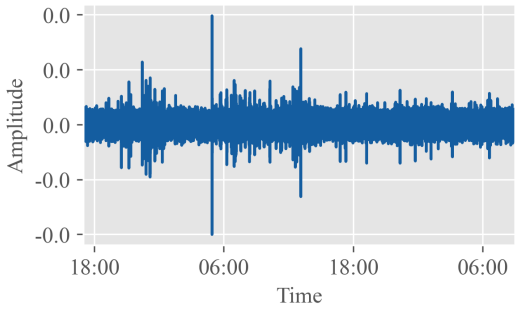
(w) IMF 23



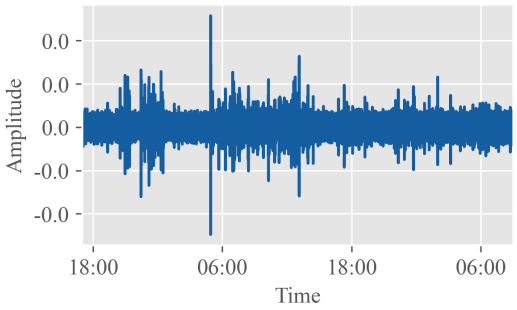
(x) IMF 24

Figure F.1.: All IMFs from EMD of the ECD measurements.

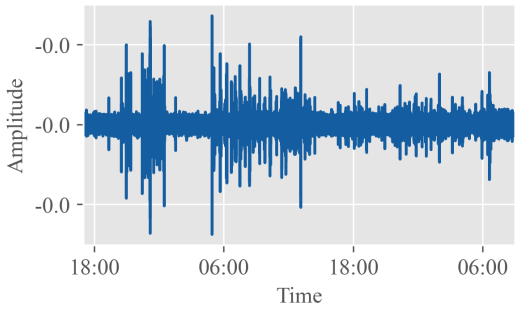
G. IMFs from EEMD of ECD



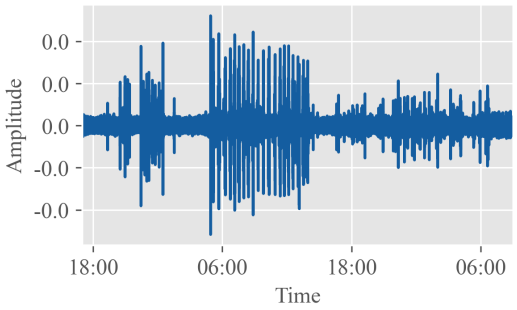
(a) IMF 1



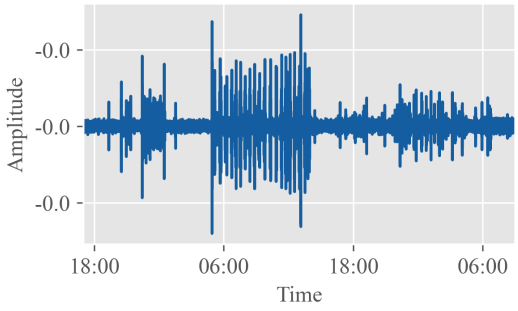
(b) IMF 2



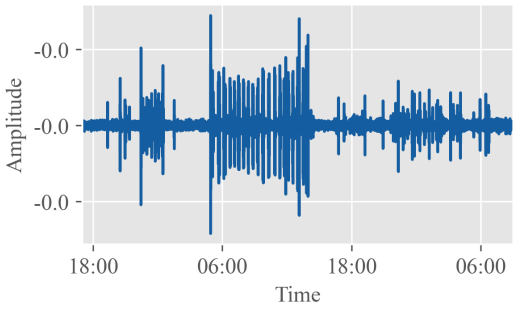
(c) IMF 3



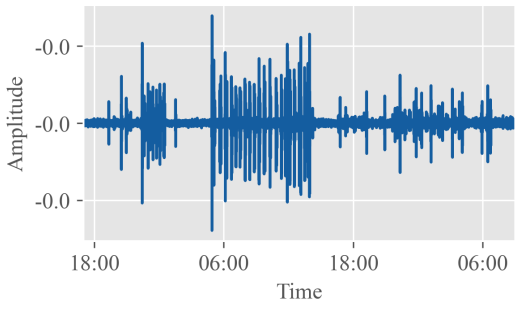
(d) IMF 4



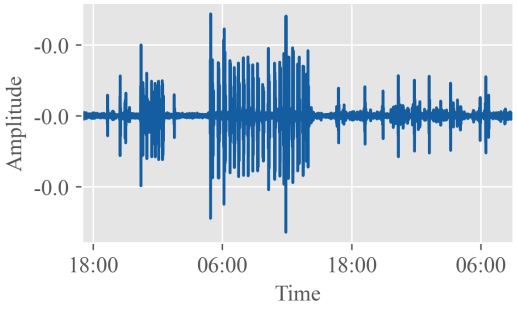
(e) IMF 5



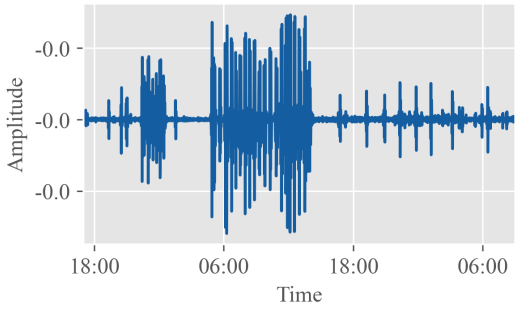
(f) IMF 6



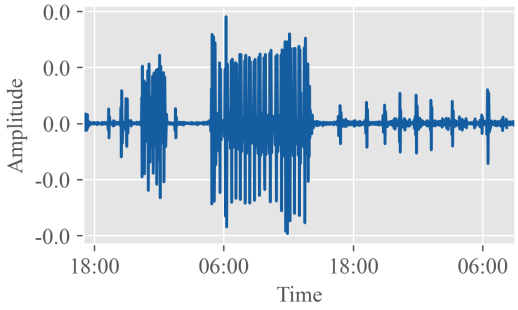
(g) IMF 7



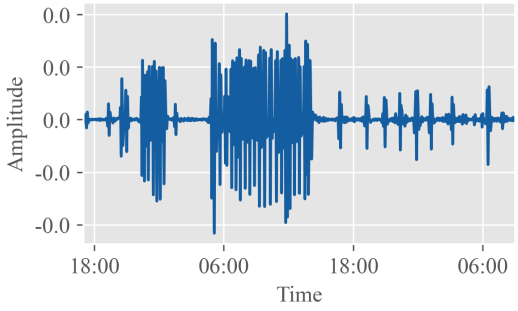
(h) IMF 8



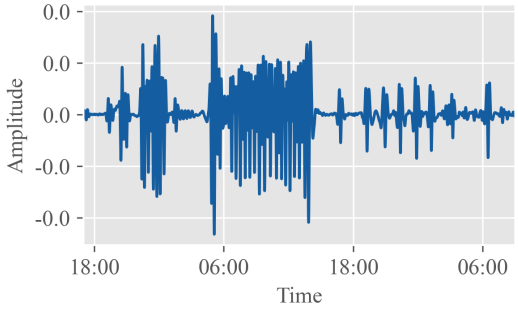
(i) IMF 9



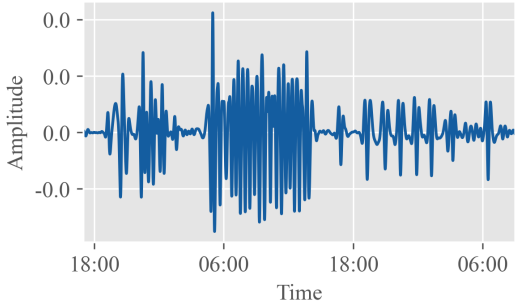
(j) IMF 10



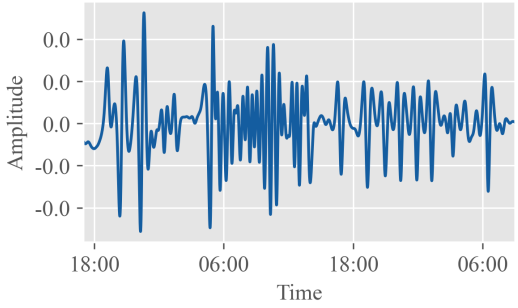
(k) IMF 11



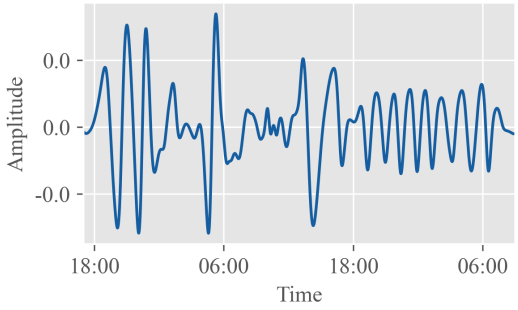
(l) IMF 12



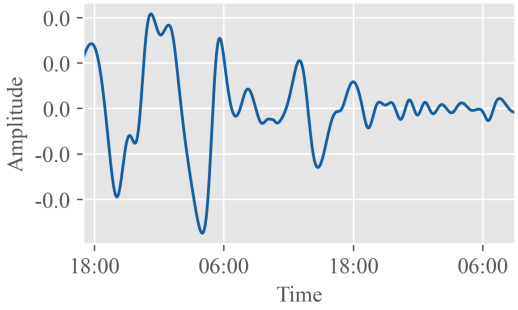
(m) IMF 13



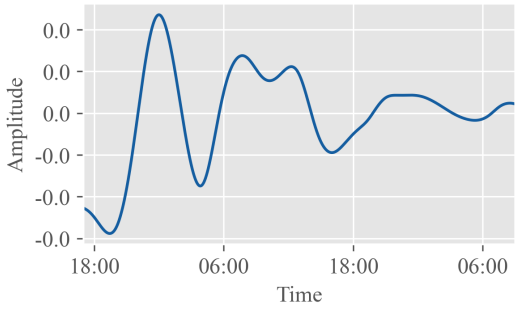
(n) IMF 14



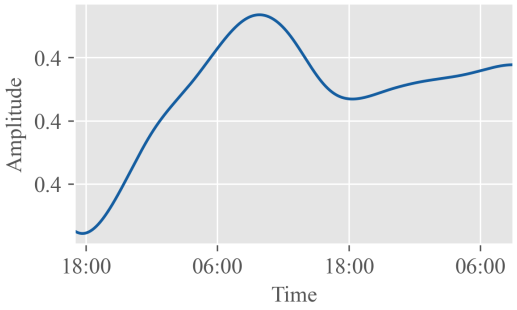
(o) IMF 15



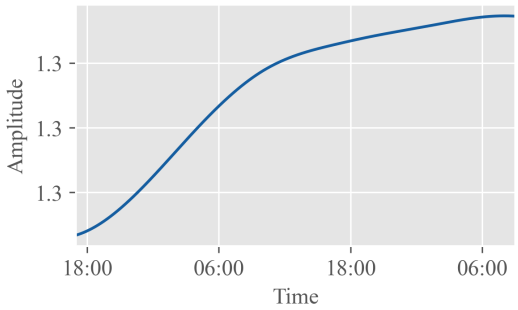
(p) IMF 16



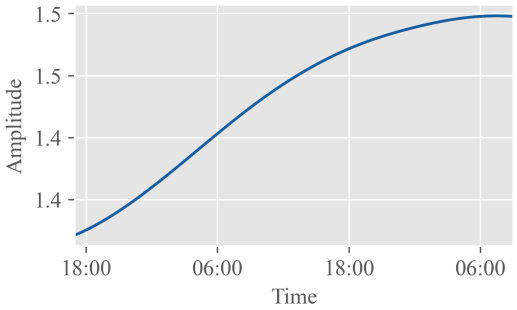
(q) IMF 17



(r) IMF 18

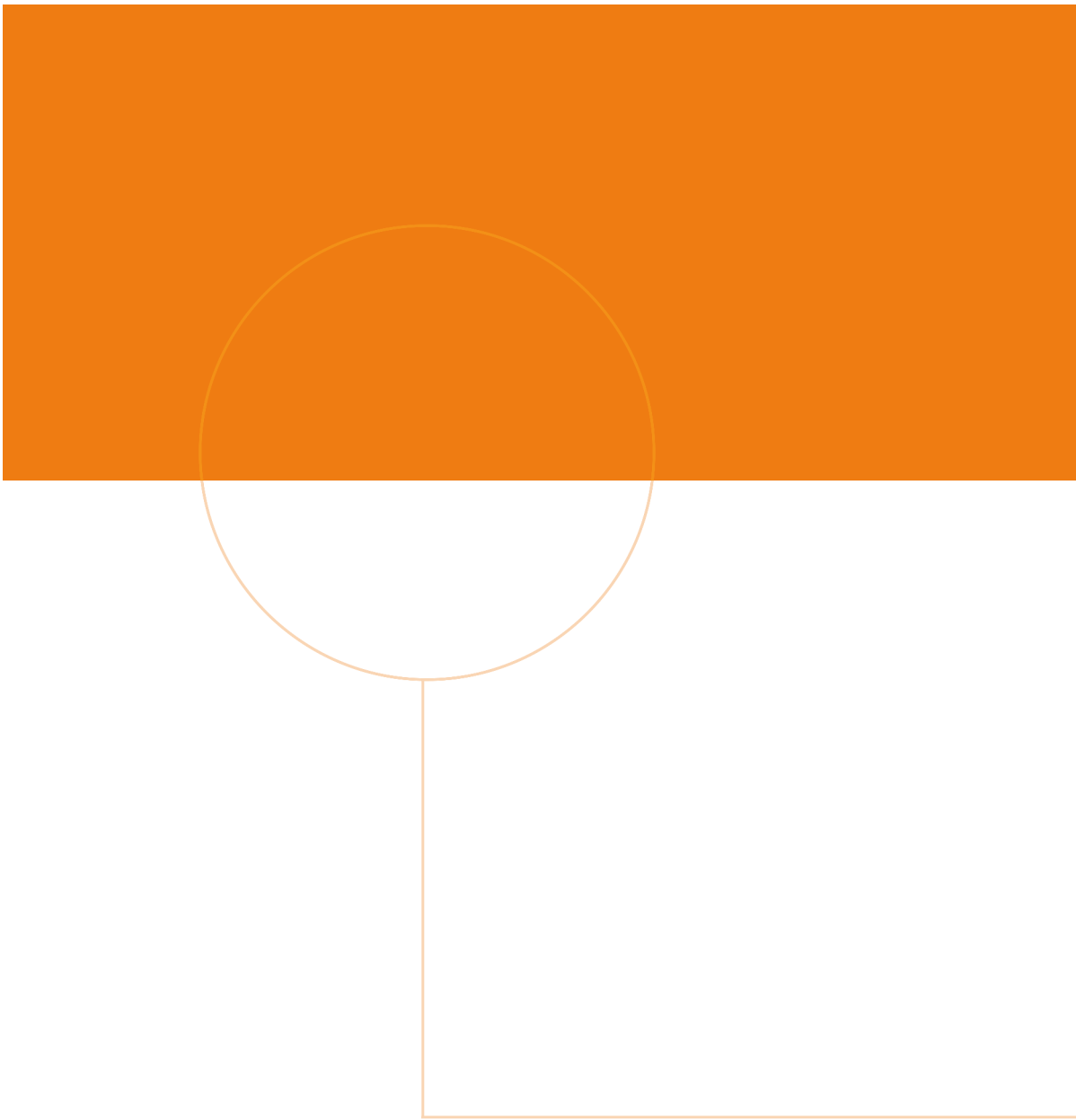


(s) IMF 19



(t) IMF 20

Figure G.1.: All IMFs from EEMD of the ECD measurements.



NTNU

Norwegian University of
Science and Technology