UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Faculty of Engineering, Built Environment and Information Technology

The Department of Mechanical and Aeronautical Engineering

Towards the discovery of boiler physics in a Kraft recovery boiler using sparse identification of non-linear dynamics

by

Ivan Ackermann

Towards the discovery of boiler physics in a Kraft recovery boiler using sparse identification of non-linear dynamics

by

Ivan Ackermann

A thesis submitted in partial fulfillment
of the requirements for the degree

Master of Engineering

in the

Department of Mechanical Engineering
Faculty of Engineering, the Built Environment and Information Technology

University of Pretoria
Pretoria

2021

Synopsis

Towards the discovery of boiler physics in a Kraft recovery boiler using sparse identification of non-linear dynamics

by

Ivan Ackermann

Supervisors:	Prof. P.S. Heyns and Prof. D.N. Wilke
Department:	Department of Mechanical and Aeronautical Engineering
University:	University of Pretoria
Degree:	Master of Engineering

A common problem found in boilers in many industries, is fouling accumulation on the heat transfer surfaces. Recovery boilers in certain paper and pulp industries are especially vulnerable to ash fouling due to the composition of the fuel used for them. Typically, soot blowers (long tubes through which pressurised steam or air is blown) are employed in boilers to deal with fouling accumulation. The high-pressure steam or air knocks off deposits that have formed on the heat transfer surfaces in the boiler. Recently, it has become necessary to optimise these soot blowing strategies, to increase boiler efficiency and longevity. Many methodologies have been implemented that attempt to optimise the soot blowing schedule and duration of soot blowing in different boilers. Typically, traditional machine learning models, such as artificial neural networks, support vector machines and long short-term memory networks, have been used to predict the fouling levels in boilers as well as the fouling change during soot blowing processes. However, these models all possess the same flaw, which is a lack of interpretability. These models are typically described as 'black-box' models, and, while their predictions are generally accurate, they are very difficult to interpret. It is often impossible to determine where errors come from for these machine learning models.

In this study, a novel approach to machine learning models is implemented, that attempts to not only fit a dataset with a model, but rather to extract the underlying physics equations from sensor measurement data, obtained from the recovery boiler at SAPPI's Ngodwana mill. A thermodynamic model is first proposed that takes sensor measurements and boiler parameters into account to calculate an estimated level of fouling in the boiler. Previously, no such metric for fouling existed for the boiler, and the thermodynamic model alone is already a positive result, since the fouling in the boiler can now be monitored more accurately and can be used in different machine learning applications. The thermodynamic model also allows one to understand which physics equations drive the processes within the boiler and which input parameters are important. Once the fouling levels are determined for the boiler over a fixed period, the model is validated using the soot blowing schedule and observing the changes in the estimated fouling levels.

The sparse identification of non-linear system dynamics (SINDy) algorithm is then introduced, which is an algorithm that can extract the underlying physics equations from measurement data and was developed by Brunton et al. (2016). The SINDy algorithm is prospectively applied to the Ngodwana dataset, with a default algorithm setup (algorithm parameters were not specifically selected), to establish basic model forms typically found in the dataset. The idea was not to test the algorithm on the real dataset yet, but rather just to extract a few default models that can be used for setting up verification problems. These verification models are used to test the identifiability of the underlying physics as well as the recovery ability of the SINDy algorithm.

At first, the basic models' coefficients are sequentially changed, to determine how they affect the curve shape of the simulated models. Hence, one can determine in which coefficient ranges the physics are more identifiable. The recovery ability of the SINDy algorithm is subsequently tested, by varying the noise and initial conditions of the artificial dataset curves, which were generated by the verification models. It was seen that the SINDy algorithm was sensitive to noise and that noise could cause the algorithm to extract incorrect models. Further, it was seen that the SINDy algorithm would not extract consistent models when the curve shapes of the data being fitted changed, or when the initial values of these curves shifted. In a final verification problem, first and second-order models were fitted to data that was generated by a second-order model, using the SINDy algorithm. It was seen that the lower order models were naturally not the optimal fit for the data, however, the results were still interpretable and conveyed basic information regarding the soot blowing curves.

The SINDy algorithm was then applied to the Ngodwana measurement dataset, once the verification problem results were interpreted. The optimal polynomial order models were initially determined, by using the results from the actual dataset as well as knowledge gained through the verification problems. A baseline algorithm setup was established and the soot blowing sequence models were extracted for a specific soot blower pair in the boiler. It was noted that the model coefficients fluctuated severely from one sequence to the next, even though the sequences came from the same soot blower pair. This was expected, since there was a lot of inconsistency regarding the soot blowing curve shapes and initial values as well as the fact that there was inevitably noise present in the data that could further impact the algorithm's performance. A positive result from the baseline implementation, was the fact that interpretable models could already be seen even if they were not consistent. The model forms obtained were logical and showed that one could potentially extract working physics models from the Ngodwana boiler.

In an attempt to circumvent the data inconsistency, additional measurement inputs were given to the algorithm to increase possible model complexity. In some cases, the additional inputs improved the model consistency, however, general models could still not be extracted. Some of the additional inputs did improve the prediction accuracy of the extracted models dramatically and showed which sensor measurements were important. This was promising, since accurate predictions in a complex physics environment such as the recovery boiler were difficult to come by. Several additional methods were tested to try and overcome the data inconsistency. This included optimising the threshold parameter of the optimiser algorithm sequentially, scaling the fouling factor and normalising the dataset. None of these proved to be very successful and only slightly improved model consistency in some cases. Finally, soot blowing sequences, that had the same basic curve shape and initial values, were manually selected, to introduce some consistency in the data.The models that were extracted, were slightly more consistent and it was shown that a general model could then be found for the simple fouling factor input only models. The more complex models, with additional inputs, however, were too sensitive to even slight data variation. The experiment also showed that sensor measurement errors were likely one of the causes of data inconsistency that was causing coefficient fluctuation and further highlighted how complex the dataset was that was being worked with.

Overall, the results of this study were positive, since an estimated measurement of fouling was introduced in the boiler that had not been available before. Further, the SINDy algorithm had no trouble extracting interpretable models and these models were able to predict the soot blowing sequences with a high degree of accuracy, despite the complex nature of the data. Lastly, the study has allowed one to determine how one would have to approach measurement datasets from the boiler in the future, especially regarding the data processing and sequence selection. If one would want to implement the extracted models in a more predictive capacity, one would have to be able to extract more general models for which this information is invaluable. Overall, the study has laid a strong foundation for future research regarding the use of SINDy for the extraction of boiler physics, which could potentially be used in soot blowing optimisation.

Acknowledgements

A memorable thank you to the following people for their contribution to this work:

- Prof. Stephan Heyns for his steady guidance, willingness to help, extensive knowledge and respectful manner in which he helps his students.
- Prof. Nico Wilke for his immeasurable help in the project with his deep insights into data science and many other fields as well as his friendly and respectful manner towards his students.
- Dr Stephan Schmidt for allowing us to knock down his door each time we had a question regarding statistics and for spending the time to ensure we understood it all.
- Wayne Davis, for his substantial contribution in the formulation of this study, as well as his guidance throughout the process, where he made many resources available without which, this study would not have been possible.
- Andre van Zyl, for his substantial help in setting up the thermodynamic model and his support throughout the study.

Furthermore, I would like to thank my friends and family, especially my parents for their continuous support throughout my study.

Lastly, I would like to thank everyone at SAPPI in the data science team that helped us with the project and made the project possible as the work in this study was conducted on a recovery boiler at one of their plants.

Contents

List of Figures

List of Tables

Nomenclature

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
CFD	Computational Fluid Dynamics
DCS	Distributed control system
FF	Fouling Factor
GDP	Gross Domestic Product
HTC	Heat transfer coefficient
ISP	Intermediate Sized Particles
ISP	Intermediate sized particles
LMTD	Log Mean Temperature difference
LSTM	Long Short term memory
ODE	Ordinary differential equation
PSO	Particle Swarm Optimisation
RMSE	Root Mean squared error
RNN	Recurrent Neural Network
SAPPI	South African Pulp and Paper industries

SH	Superheater
SINDy	Sparse Identification of Non-linear System Dynamics
STD	Standard deviation
STLSQ	Sequentially Thresholded Least squares
SVM	Support vector machine

LIST OF SYMBOLS

α	Absorbtivity
$\bar{x}$	Mean of measurement samples
ΔT	Temperature difference [^oC]
ΔT_{LMTD}	Log mean temperature difference
$\dot{m}$	Mass flow rate [kg/s]
$\dot{Q}$	Heat transfer [W]
$\dot{\mathbf{X}}$	Derivative of the input dataset
$\boldsymbol{\Xi}$	Coefficient vector
$\mathbf{u}$	Control input to the system
$\mathbf{x(t)}$	State variables of the system
$\mathbf{X}$	Input dataset
ν	Kinematic viscosity [m^2/s]
σ	Boltzmann's constant [W/m^2K^4]
$\Sigma(\mathbf{X})$	Feature library
σ_{STD}	Standard deviation of measurement samples
ε	Emissivity
A	Surface area [m^2]
C_x	Coefficient x
D	Diameter [m]

F	Correction factor
f	Friction factor
H	Enthalpy $[kJ/kg]$
h	Heat transfer coefficient $[W/m^2K]$
k	Thermal Conductivity constant $[W/mK]$
L	Tube length [m]
N_L	Number of longitudinal tubes
N_{tubes}	Number of tubes
Nu	Nusselt number
P	Pressure [kPa]
Pr	Prandlt Number
q	Rate of heat transfer $[J/kg]$
Re	Reynolds number
S_T	Transverse tube spacing [m]
T	Temperature $[^oC]$
V	Velocity [m/s]
x	Measurement sample
Z	Z score number

CHAPTER 1

Overview

1.1 Background

All over the world, the manufacturing and power generation sectors are moving towards becoming more autonomous and requiring less human intervention. Companies are moving toward the use of Artificial Intelligence (AI) to increase the efficiency of manufacturing and power generation processes. Most of these industrial processes generate large amounts of data, that can be utilized by AI systems to extract useful insights for the optimisation of plant processes. Many of these AI systems can then monitor the plants online, and make adjustments on the fly, to optimize manufacturing processes and hence increase efficiency (Keith Mills 2019). The integration of AI in manufacturing processes as companies move toward sustainability is of vital importance and many companies have already managed to integrate such systems with great success (Cioffi et al. 2020).

A study published by PwC shows that AI systems could boost the worldwide GDP by as much as 14 % over the next 10 years. China could see an increase in their GDP of up to 26 % while North America could see an increase of up to 14 % (Verweij & Rao 2017). It is therefore clear that the use of artificial intelligence in industrial processes could result in more efficient manufacturing and an increase in a company's profit. Furthermore, there has been a worldwide movement toward sustainability and companies are under pressure from government regulations and other organizations to evaluate their environmental impact critically (Gunasekaran & Spalanzani 2012). Increasing manufacturing efficiency, through the use of AI, is directly contributing to a company's sustainability as they will consume fewer resources to produce the same number of products.

In the manufacturing and power generation sectors, boilers are widely used to generate steam for plant processes and energy generation. Because of the high temperatures present in boilers, and the nature of the fuels used for combustion, boilers typically come with a variety of problems and require extensive maintenance throughout their service lifetimes. One such problem is the accumulation of deposits on the heat transfer surfaces of the boiler, known as fouling. These deposits reduce the heat transfer efficiency of the boiler, as the deposits act as insulation, which prevents heat from passing through the surfaces to the working fluid. Should the deposit build-up be left unchecked, the boiler will eventually plug in some sections. Plugging refers to the case where the deposit build-up in a section of the boiler is so severe that the entire section is blocked, preventing all airflow (Tran 2007). This requires the boiler to be shut down and cleaned out before operations can continue, and results in large financial losses for a company.

Currently, soot blowing is used to prevent as much deposit formation as possible. Soot blowing refers to the use of high-pressure steam or air, blasted onto the heat transfer surfaces, to remove deposits. In most boilers found in the industry, very little has been done to optimise the frequency and duration of soot blowing procedures. Usually, soot blowing is performed, based on the experience of an operator or by establishing a schedule through trial and error (Peña et al. 2011). The problem with this approach is that both over-blowing and under-blowing leads to decreased efficiency. Over-blowing can lead to corrosion of the heat transfer surfaces, which could incur expensive maintenance costs. Under-blowing could lead to excessive deposit accumulation and boiler plugging which will lead to plant shutdowns and maintenance costs (Tran 2007).

Recently, the manufacturing and power generation industries have shown an interest in the optimisation of soot blowing procedures using of artificial intelligence. Peña et al. (2011) tried to optimise soot blowing through using an artificial neural network (ANN). Anitha Kumari & Srinivasan (2019) utilised support vector regression and gaussian process regression to monitor ash fouling and optimised the soot blowing in a reheater of a thermal power plant. Shi et al. used basic thermodynamic modeling and a Particle Swarm Optimisation (PSO) algorithm, to optimize the soot blowing phase and duration (Shi et al. 2019), and Zhang et al. (2018) used a dynamic heat transfer model in conjunction with other machine learning models to predict the flue gas exit temperature of a recovery boiler to name but a few examples. It is therefore clear that AI can play an important role in the optimisation of boilers in the industry.

There are many difficulties when one wants to optimize soot blowing in a boiler. It is very difficult to estimate the level of fouling in boilers, and measurement sensors that could approximate the level of fouling, such as heat flux sensors, are very expensive. They also require extensive maintenance because of the high temperatures they have to operate in and are prone to deterioration. Typically, modifications have to be made to the boiler to install them. The non-linear nature of the dynamics in a boiler makes it difficult to build accurate thermodynamic models. Boilers rarely operate at steady-state conditions as the load requirements change continuously. Additionally, the conditions within the boiler do not remain constant due to deposit buildup and corrosion. It is difficult to determine the radiation heat transfer within the boiler accurately, due to the complex nature of radiation heat transfer. There are substantial challenges to overcome when trying to optimize the soot blowing procedures, which is why the use of AI models to help with the optimisation process has become very popular.

SAPPI in South Africa operates the Ngodwana Paper and Pulp mill. The mill produces paper-grade pulp for various industries around the world. The mill also has its own recovery boiler to generate electricity and steam for other plant processes. Some of the excess energy, generated by the turbines that use steam from the recovery boiler, is sent back into the South African electricity grid. The recovery boiler utilises black liquor (a by-product of the paper-making process) as fuel. Black liquor contains up to 50 % inorganic materials, which, when burnt, form ash. The ash from the combustion process deposits on the heat transfer surfaces within the boiler and leads to fouling and boiler plugging. As is standard in the industry, SAPPI employs soot blowers to alleviate this problem. However, the soot blowing schedule has not been optimised and excessive boiler fouling and plugging is a common problem. This results in 2 plant shutdowns per year, where the boiler has to be cooled off and water washed, in addition to smaller water washes within the same year. Each day that the boiler is shut down translates to a monetary loss in excess of R20 million. The current method for determining the soot blowing schedule is to observe at each water wash where the most deposit build-up has occurred and then to increase the frequency of the soot blowing in that area. This is not ideal in terms of boiler efficiency or for obtaining reliable results.

There is thus a need to develop models that can predict the fouling within the boiler and can potentially be used for the optimisation of the soot blowing schedule to reduce the number of water washes per year. Since the internal conditions in the boiler are not well known, the models that are developed for the prediction of the fouling should also be as interpretable as possible to aid in the understanding of the internal boiler physics.

1.2 Literature Review

A literature review is performed on the optimization of soot blowing procedures in the industry, with the focus being on the use of artificial intelligence and machine learning models to achieve this. The first few sections look at the characteristics of fouling deposits and soot blowing operations, followed by sections dealing with the estimation or measurement of fouling in a boiler. The final sections deal with the types of machine learning models that can be used for the prediction of fouling.

1.2.1 Black liquor and deposit characteristics

Black liquor is a fuel that is commonly used in the pulp and paper industry. It is a by-product that is extracted from the pulp washing process. Black liquor typically contains up to 50 % inorganic solids, making it a difficult fuel to work with, especially regarding fouling. When black liquor is combusted, the inorganic solids produce ash that is prone to deposit on heat transfer surfaces. There are three deposit types: namely, i) Carry-over, ii) Intermediate Sized Particle (ISP) deposits and iii) fume deposits. (Tran 2007)

Carry-over particles are partially combusted black liquor compounds that are usually larger than the other deposit sources. These deposits are mainly found in the lower and upper superheater regions of boilers close to the combustion region. These particles fuse with heat transfer surfaces and become very hard. ISP deposits are slightly smaller particles than carry-over deposit particles and are usually found in the same regions as the carry-over deposits, but may also be found in subsequent sections of the boiler. ISP particles are usually carried through the boiler by the flue gas. Fume deposits occur because of the condensation of flue gas. Flue gas comes into contact with cooler surfaces in the boiler and condenses to form a soft, white deposit. These deposits are mostly found in the later sections of the boiler near the boiler bank and economizers (Tran 2007).

Spectrum analysis of the deposits revealed that most of these deposits comprise of alkali compounds (Bernath et al. 1998). Most of these alkalies were found to be compounds of sodium or potassium. A problem with these alkali compounds is that they build up rapidly on heat transfer surfaces, making heat transfer inefficient. The presence of chlorine in some compounds, also becomes problematic when temperatures in the boiler exceed 490 °C. At this temperature, the compounds containing chlorine become corrosive and may damage the boiler pipes and surfaces (Leppänen et al. 2014).

The primary type of deposit found on heat transfer surfaces depends on the location in the boiler as mentioned previously. Carry-over and ISP deposits are mostly found in the superheater regions near the combustion chamber as these particles are larger and cannot be carried further into the boiler by the flue gas. These deposits also form on the windward side of the heat transfer surfaces (the side that is closest to the combustion chamber). Fume deposits, on the other hand, form in the cooler sections of the boiler and deposits on the leeward side of the heat transfer surfaces which are typically cooler, with the exception being the economizer sections of the boiler where temperatures are typically cold enough for deposits to form anywhere on the pipes. Figure 1.1 shows the prevalence of the deposits in the different sections of the boiler. The thickness of the arrows corresponds to the amount of deposit formation on each of the heat transfer surfaces.

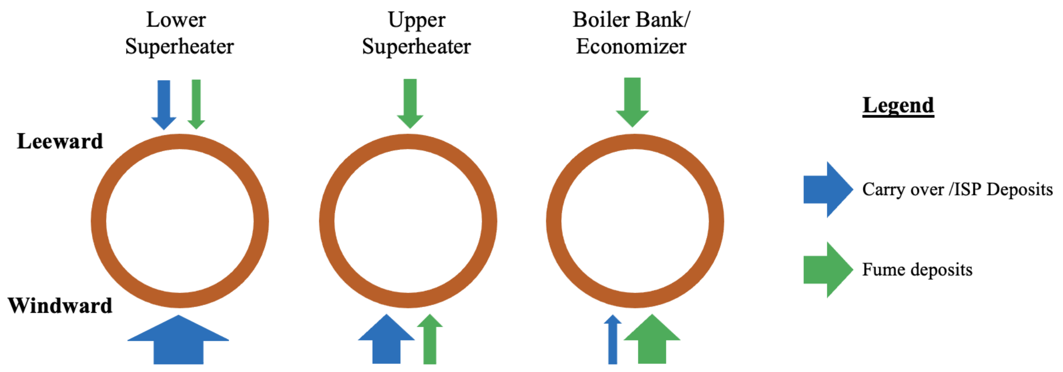


Figure 1.1: Deposit prevalence in different sections of the boiler (Adapted from (Tran 2007)).

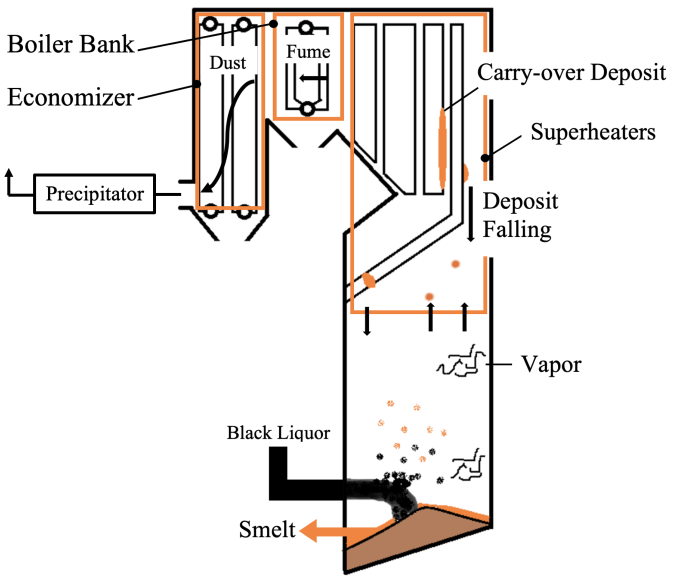


Figure 1.2: Typical layout of recovery boiler (Adapted from (Tran 2007)).

Figure 1.2 shows the typical structure and layout of a recovery boiler as well as the deposit formation process. The sections indicated in figure 1.1 are also indicated here.

Even though deposits occur throughout the boiler, some section are more problematic than others. Identifying these regions is important, as one can then determine where the focus of the soot blowing optimisation should be. To understand why regions experience more deposit build-up than others, it is necessary to define two important temperatures, namely the sticky temperature and the radical deformation temperature T_{RD}. The sticky temperature is the point at which deposits contain 15-20 % liquid. At this temperature, deposit accumulation occurs unbounded as the deposits cling to each other. Therefore, the risk of boiler plugging is high at this point. The radical deformation temperature is the temperature at which deposits contain 70 % liquid. At this temperature, the weight of the deposits themselves are enough to cause them to drop from the heat transfer surfaces and deposit accumulation is bounded. Thus, the risk of plugging is low (Tran 2007).

In the lower superheater, the temperature is typically above 820 °C. The carry-over and ISP particles are molten, and they fuse to the heat transfer surfaces in this region when they come into contact with them to form a hard deposit. As more deposits accumulate, the surface temperature of the pipes increases, due to less heat transfer taking place. At some point, the surface temperature surpasses T_{RD}, and the deposits melt. Deposits then simply run off of the tubes and do not continue to grow. A steady-state point is reached, where the deposit accumulation equals the deposit run-off. It can be concluded that this might not be such a problematic area in the boiler, as the deposition rate eventually goes to zero and deposit growth is not unbounded (Tran 2007).

In the upper superheater, the temperature is lower, and deposits fall within their sticky temperature range. Carry-over and ISP particles continue to fuse to these tubes as they are always sticky. Deposit accumulation, therefore, occurs rapidly. In this section of the boiler, the surface temperature never reaches T_{RD} and no steady state is ever reached. Plugging is therefore a significant risk in this area, as deposits grow uncontrolled. Rigorous soot blowing is usually employed in this area to knock off deposits (Tran 2007).

The boiler bank region is particularly susceptible to plugging, especially when the upper superheater has experienced significant fouling. This is due to the increase in the flue gas temperature since less heat is being transferred in the upper superheater. The spacing between the heat transfer tubes is also very narrow. The deposit particles are in their sticky temperature range or enter their sticky temperature range once the upper superheater has fouled. Any significant deposit growth blocks these narrow passages and therefore the boiler bank inlet is the most common location for plugging to occur

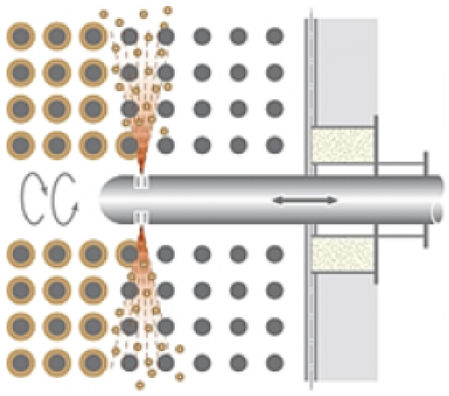


Figure 1.3: Illustration of sootblower operation.

(Tran 2007). The most important thing to note is that plugging tends to occur here if the upper superheater (Secondary superheater) has already experienced much fouling. Therefore, if soot blowing is ineffective in the upper superheater, plugging eventually becomes inevitable in this region. This makes effective soot blowing in the upper superheater even more critically important.

1.2.2 Removing deposit accumulation through soot blowing

A standard procedure for removing deposits from tubes within boilers in the industry is to employ soot blowers. This is also the case in the recovery boiler at the Ngodwana mill. Soot blowers usually employ high-pressure steam or air to knock off fouling deposits from the heat transfer surfaces. Soot blowers are simply long pipes with nozzles at the end, which are used to disperse the pressurised steam over the heat transfer surfaces in the boiler internals. While the steam or high-pressure air is being discharged, the soot blower tubes are rotated to clean the area surrounding it. The recovery boiler at the mill makes use of high-pressure steam, taken directly from the boiler itself, to operate the soot blowers. It is therefore not possible to run the soot blowers continuously, as the efficiency of the boiler will be severely affected. These soot blowers are currently running on a schedule that has been determined through trial and error. There is currently no way of estimating the fouling within the boiler and therefore, the soot blowers are operated manually, and the schedules are determined by inspecting the fouling accumulation when the boiler is water-washed. Figure 1.3 illustrates the working of a soot blower as it is pushed through a section of the boiler. The soot blower tube is inserted between the boiler tubes or platens, and the steam knocks off some of the deposits as the soot blower is rotated and moved through a section.

1.2.3 Estimating the level of fouling in boilers

Estimating the level of fouling within the boiler is no simple task and is a field of ongoing research in many sectors. It can be an arduous task to determine the level of fouling within boilers, because of the many non-linearities in the internal physics and random occurrences taking place within the boiler (Tran 2007). Nonetheless, some techniques have been established to give one an indication of the level of fouling in a boiler.

Direct measurement techniques

Some of the more accurate methods of measuring fouling in a boiler are through the use of direct measurement instrumentation. An example of this would be heat flux sensors. Heat flux sensors can measure the amount of heat that is transferred from one body to another. A heat flux sensor generates an electrical current that is proportional to the amount of heat transferred through it. Heat flux sensors can give one an indication of the amount of heat transferred through the heat transfer surfaces to the steam. When fouling levels increase, the amount of heat being transferred will decrease. Therefore, these sensors can be used as an indirect measurement of fouling. This method of estimating the fouling is used in the paper by Teruel et al. (2005) where the statement is also made that heat flux sensors are some of the most reliable methods of measuring fouling in a boiler. Heat flux sensors tend to deteriorate the longer they are in the boiler and therefore require substantial maintenance. Because of this deterioration, they are also not very well suited to online monitoring of fouling and should rather be used to obtain a dataset containing fouling estimations that can be used in machine learning models.

In the paper by Peña et al. (2011), heat flux sensors were used to estimate fouling, however, it was seen that the heat flux sensors experienced measurement drift because of this deterioration, and the data had to be inspected closely to adjust for this change. Other direct measurement techniques include cameras designed to withstand the harsh conditions within a boiler. The cameras can record the fouling development on the pipes and the amount of fouling can be gauged from these images, as well as the rate of deposit growth. While this method might not be the most accurate, it is a very direct method for determining the fouling level in the boiler. The paper by Prem et al. (2014) mentioned a third method of estimating the fouling accumulation directly by using strain gauges. Strain gauges are attached to the top of the superheater platens and measure the weight of the platen. When deposit accumulation increases, the weight of the platens also increases and is picked up by the strain gauges. Thus, one can estimate the weight of the fouling deposits.

While direct measurement techniques can work very well for estimating the fouling in the boiler, there are usually a couple of problems that accompany them. The instrumentation is expensive, not only to buy, but to maintain as well (Teruel et al. 2005). Installing the instrumentation usually takes time and requires significant modifications to the boiler, which can be costly and unfeasible since the boilers must be shut down. Lastly, the instrumentation and sensor accuracy deteriorate over time due to the harsh conditions in the boiler, making them unsuited for online monitoring of fouling.

Mathematical and thermodynamic modeling

When direct measurement techniques are not feasible, one might want to investigate mathematical and thermodynamic modelling methods to determine the fouling in a boiler. These methods are inevitably not as accurate as direct measurements, as many thermodynamic models make use of simplifying assumptions and approximations. Building a mathematical model to capture all the non-linearities and random occurrences in the boiler would be a very complicated endeavour that is not currently feasible. Therefore, one would have to make approximations. Most of the mathematical methods used to estimate the fouling in the boiler revolve around the calculation of the heat transfer coefficients in the boiler. The fouling accumulation will cause a decrease in the actual heat transfer coefficient in the boiler, thus giving one an indication of the level of fouling in the boiler. Most thermodynamic models of boilers depend on temperature differences in the boilers. In the paper by Shuiguang Tong (2019), the log mean temperature difference is used as a temperature difference measure to determine the heat transfer coefficients. Furthermore, the paper makes use of an energy balance equation between the heat released by the flue gas and the heat absorbed by the steam to solve for the heat transfer coefficients. Shi et al. (2015) make use of a similar approach. However, the paper states that the LMTD method assumes steady-state conditions in the boiler. They, therefore, adapted their approach to incorporate some of the transient behaviour in the boiler. Finally, in the paper by Peña et al. (2013) it is stated that one cannot neglect the radiation heat transfer occurring in the boiler, which the previously first mentioned paper did not explicitly consider. Therefore, their approach, while being similar to the previous two papers, also incorporates a radiation heat transfer term in their heat transfer coefficient calculations.

Despite the differences in the approaches of the papers mentioned above, all of them used basic thermodynamic relations and equations that share the same base in literature. It can be noted in these papers, however, that estimating the fouling factor is no simple task and some methods require extensive measurements and boiler parameters, which are not always readily available.

Alternative fouling estimation techniques

If no direct measurement techniques can be used in the boiler and mathematical models are too complex or one does not have enough measurements available, one could resort to methods that try to combine these two methods to estimate the level of fouling. One such a method is proposed in the paper by Anitha Kumari & Srinivasan (2019), where a dual extended Kalman filter is used to estimate the real heat transfer coefficient, while basic heat transfer equations are used to estimate the clean heat transfer coefficient. A dual extended Kalman filter comprises two extended Kalman filters that simultaneously estimate the states of a system as well as the parameters of the equations of the system (Wassiliadis et al. 2018). The technique used in the paper by Anitha Kumari & Srinivasan (2019), is therefore, an example of combining basic thermodynamic equations and an indirect measurement technique that estimates the system states to obtain a fouling factor estimation of a boiler.

Ramirez (2000) proposed that one use a fouling status, rather than using a specific fouling factor, when it is not possible to determine the fouling factor accurately. In this paper, the fouling status is rather described as a cleanliness status and works similarly to that of 'fuzzy logic' in computer algorithms. Rather than having a specific value for the fouling, such as 80% fouling in the boiler, the cleanliness status will say that the boiler fouling level is 'Severe', since the fouling lies somewhere between 75 and 95%. The paper states that this allows one to use simple measurements to estimate the level of fouling and, in their case, the flue gas exit temperature is used for this purpose. No complex mathematical models have to be built with this method. However, one does not get a very accurate value for the level of fouling in the boiler, but merely a range of potential values. This could be problematic when more accurate fouling estimations are required for optimisation.

Discussion of fouling estimation techniques

It has been discussed how one might determine the amount of fouling in a boiler using one of three methods, namely direct measurement, mathematical and thermodynamic modelling, and a combination of the other two techniques. While direct measurement techniques can give one a very accurate estimate of the amount of fouling in a boiler, the cost of installing and maintaining the instrumentation is very high, especially when the boiler does not have such measurement devices by default. These direct measurement devices have short life cycles in the boiler because of the corrosive environment and high temperatures, which makes them unsuited for online fouling monitoring. In this project, online monitoring is crucial as the plant engineers would like to know how conditions in the boiler change over a long period, as the physics would undoubtedly change. Therefore, direct measurement techniques would not be well suited to the current long-term project objective for a recovery boiler.

The fuzzy logic and Kalman filter methods have potential, however, these techniques do not always give one a very accurate or usable unit of measure for the fouling in the boiler. If one would want to optimise the soot blowing in the boiler, one would need a fouling estimation that can be used in thermodynamic calculations to calculate energy losses.

For the optimisation end goal of the overall boiler project, the thermodynamic and mathematical models are well suited. While these models still only approximate the level of fouling in the boiler, the resulting fouling factor estimation should be accurate enough to be used in energy calculations, as the fouling level is based on the true physics that governs the boiler. These methods also allow one to have a good understanding of the thermodynamics within a boiler that can potentially be used to understand how the deposit buildup works in a boiler.

1.2.4 Modeling the fouling in boilers

While being able to obtain the fouling factor with thermodynamic models or direct measurements is useful, it is not always practical to use these methods in an online and always-on capacity. As mentioned, direct measurement sensors do not last long due to the sensors corroding in the boiler's high-temperature environment. Thermodynamic models, on the other hand, often rely on multiple sensor readings, such as pressure and temperature measurements as well as steam flow and fuel flow measurements. For this reason, applying these models in an online capacity is not always reliable, as sensors are prone to faulty measurements or can break altogether. The more sensors one requires (as is the case with complex thermodynamic models), the larger the chances become of one sensor failing at some point. This could cause the models to not work for an extended period or the models could become extremely inaccurate. Therefore, in most industry cases, models are built, that attempt to predict the fouling factor using fewer input sensors to a model to reduce the risk of sensor drift or failure in the long run. Typically, one of two modeling options is used, namely physical modeling and data-driven modeling.

Physical modeling

Physical modeling, in this case, mostly refers to the use of Computational Fluid Dynamics (CFD) to build a computer-simulated replica of a system that can simulate different conditions and scenarios to obtain, for example, a fouling factor. Physical models can be very accurate when one wants to know what exactly is happening in the boiler over time, and can give one an accurate view of the level of fouling in a boiler. In the paper by Leppänen et al. (2014), a full-scale 3D model of a boiler was built to determine where fume deposits will form in the boiler. The model was very successful in determining the location where severe deposit formation occurs, as the predicted location of deposit formation was confirmed later on by looking at the boiler itself. However, a pitfall of these types of physical models is that degradation in the boiler often leads to these models becoming inaccurate, as the physics in the boiler change over time. Often, assumptions regarding conditions and physics in the boiler are made when these types of models are built and if these assumptions do not hold as the boiler conditions degrade, the models become obsolete (Smrekar et al. 2009). Furthermore, these models are usually quite complex to build and are not well suited to online predictions of the fouling factor, as simulations take a long time to run if the computers running them are not top-of-the-range products. If less computationally expensive models are required, one would rather consider data-driven models as they are usually faster to apply in online scenarios where frequent predictions are needed.

Data-driven modeling

Data-driven models, especially machine learning models, are nowadays becoming more and more popular for use in many industry sectors in the world. The reason for their popularity is due to their ability to take relatively few input parameters and extract enough information from them to make accurate predictions. Many machine learning models can also fit very non-linear data sets that were previously not an easy problem to solve. Recently, due to the increased availability of data in the engineering sectors, data-driven models have become very popular (Parente et al. 2019). Data-driven machine learning models can be separated into different categories. The most well-known and used category being supervised learning methods. Another form of data-driven model that have become very popular in recent years is the unsupervised or semi-supervised type machine learning model because of its ability to extract valuable information from data without having a fixed goal or labeled dataset.

Supervised machine learning models

Examples of supervised machine learning models include Artificial Neural Networks (ANNs), Support Vector Machines (SVMs), Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks and dimensionality reducing models, such as Sparse Partial Least Squares (SPLS) methods. These models have the requirement of a labeled dataset, where the target prediction that needs to be made is defined in a dataset. For example, the fouling factor must already be known for a set of input sensor measurements. These models have been used extensively in the industry to solve similar problems to this project.

Teruel et al. (2005) and Peña et al. (2011) both used ANNs to predict, for example, efficiency of soot blowing, and the cleanliness of the heat transfer surfaces after soot blowing in a recovery boiler in China. The models were implemented to a large degree of success and improved the soot blowing efficiency in the boiler. In the paper by Anitha Kumari & Srinivasan (2019), an SVM was used to predict the cleanliness factor of the boiler heat transfer surfaces, which could be used to optimise the soot blowing strategy in the boiler itself. Radhakrishnan et al. (2007) used an LMST model to predict the outlet temperatures of fluids from a shell-and-tube heat exchanger, which allowed them to use the log mean temperature difference method to predict the heat transfer coefficient of the heat transfer surfaces.

While these methods are often very successful in predicting system parameters in non-linear systems, the models are seldom interpretable. For example, a neural network, that has multiple layers, is very difficult to decipher, and it is nearly impossible to understand all the weights, as there might be millions of interconnected neurons, each with their own weight. Models such as ANNs are often considered 'Black box' models that one cannot interpret beyond what the inputs and outputs are (Robbins 2017) because of the complexity of the models themselves. When these machine learning algorithms inevitably make mistakes, it is challenging to identify where exactly in the models these mistakes came from, not to mention how to fix them (Wiley 2020).

These models are often also not able to generalise when, for example, conditions in the boiler change drastically. This is because the models were trained on a specific input data range and if this range changes significantly, the models usually cannot compensate for the change. These models simply fit a 'line' through data instead of extracting the true underlying models that govern the system that is being investigated. This is the reason for the rise in popularity of unsupervised machine learning models in recent years, where some models try to address the issues of 'black box' models and try to extract better general models that can be interpreted and used in the long run. In the paper by Rudin (2019), several compelling arguments are made why one would rather use interpretable models, the key reason being that machine learning models are often used for high stakes decision making and having an interpretable model avoids unnecessary or inexplicable problems, that could have dire consequences and often accompany black box models.

Unsupervised machine learning models

Unsupervised machine learning models are commonly used to cluster data and to identify underlying patterns in data sets without being led by a target value that the models have to reach. For unsupervised machine learning models, it is up to the models themselves to find structure and patterns in the data that is presented to them (Mishra 2017). This can be very useful where it is not clear what one wants to extract from a particular dataset or when labeling the dataset is simply not possible. An interesting type of machine learning model that has gained some traction recently are models that extract the system dynamics or physics from the data itself instead of simply fitting a line through the data to make predictions. These types of models are classified as unsupervised, as one does not know what the underlying system dynamics are beforehand. Rather, it is up to the algorithms to extract the correct underlying models. An advantage of this is the fact that the extracted dynamics or models are more interpretable and can be used to understand the data-generating systems better. If one understands the ground truth of a system, it is easier to generalise and make predictions beyond the training range of the model, as the system dynamics have been extracted and not just a model that fits data that one already has.

A very recent example of such an algorithm is the Sparse Identification of Non-Linear System Dynamics (SINDy) algorithm. The SINDy algorithm tries to extract physics from a dataset that can be used to interpret a system. In the paper by Brunton et al. (2016), the SINDy algorithm was successfully implemented and discovered the governing equations of fluid flow past a cylinder where vortex shedding occurs. The algorithm could extract the equations efficiently from measurement data, where these equations took scholars years to derive and describe. This shows the potential that unsupervised learning algorithms could have and may allow one to obtain general models that can be interpreted and used for long periods without having to retrain complex ‘black box’ models. The SINDy algorithm in particular may be of significant value for this project, specifically where where the fouling factor development is concerned.

1.2.5 Sparse Identification of Non-linear System Dynamics (SINDy)

The sparse identification of non-linear system dynamics or SINDy algorithm was developed to try to solve the problems of generalisation and interpretability often found in black-box models. As mentioned earlier, advances in machine learning and artificial intelligence have made it possible to extract patterns from complex data streams that would not have been found otherwise. However, extracting models from data that can predict instances outside of the training data range has been a very slow process (Brunton et al. 2016). The SINDy algorithm was developed to discover the underlying governing equations of a system. These governing equations should then be able to generalize outside the domain of the training data and result in a model that is both robust and interpretable.

The assumption is made with the SINDy algorithm, that most dynamic models can be described using only a few significant equations. Therefore, relative to the number of possible functions and relations that can describe a system, the actual number of functions describing a particular system is sparse (Brunton et al. 2016). Sparse regression is a central concept used in the SINDy algorithm, which allows one to determine the fewest terms needed to describe the training data. This results in an accurate model, whic is also robust to noise and outliers, as over-fitting is avoided. (Brunton et al. 2016) Since these models are developed to promote sparsity in combination with the extraction of physical equations, interpretability is usually an added benefit of using the SINDy algorithm.

The SINDy algorithm tries to solve the following equation, which is the assumed form of the dynamical system:

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)) \tag{1.1}$$

The idea of sparsity enters with the function $\mathbf{f}$ (the feature library of possible functions that could describe the physics), which is assumed to have only a few terms that are active and combined to describe the physics. If the feature library is well chosen to support the physics, and the terms in the feature library are sparsely active, the resulting model is a simple differential equation that can generally be interpreted easily. It is also possible that this model can be used outside the domain of the training data, which is not possible for many traditional machine learning models. In the paper by Brunton et al. (2016), the SINDy algorithm is successfully applied to the chaotic Lorenz system and the problem of fluid flow past a cylinder. For both cases, the SINDy algorithm could discover the underlying governing equations of the two systems from noisy measurement data. In another study by Kaiser et al. (2018), the SINDY algorithm is combined with control inputs. The idea is to predict the system dynamics that also include the control inputs so that the effects of different control inputs on the system can also be accounted for in the models and their predictions. The assumed dynamical form takes on a new form as follows:

$$\frac{d}{dt}\mathbf{x} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{1.2}$$

where $\mathbf{x}$ is still a function of time and $\mathbf{u}$ refers to the control inputs to the system. This paper proves that adding control to the SINDy algorithm can be done, by adding control inputs to the chaotic Lorenz system. The algorithm could recover the system dynamics, including the control influences, efficiently. Using a predator-prey model, this algorithm with added control is further illustrated to work, as the SINDy model is once again able to recover the underlying equations of the true model.

The SINDy algorithm is a robust algorithm that can extract the underlying physics from data despite noise being present. The resulting models can likely generalize outside the domain of the training data, which is ideal for case studies where the distribution of the data does not always remain constant, however generalization is not always guaranteed. Furthermore, the resulting models can be interpreted relatively easily as the algorithm returns simple differential equations with clearly defined inputs.

1.3 Scope of research

Boiler fouling estimation and prevention is an ongoing field of research that has gained much traction because of the worldwide movement towards sustainability and, hence, higher efficiency requirements. Different methodologies have been developed in papers by Shi et al. (2019), Peña et al. (2011), Anitha Kumari & Srinivasan (2019) and many others, where they attempt to predict the fouling level and optimise the soot blowing strategy in different boiler types. Many of these methodologies incorporated machine learning models and algorithms to predict the fouling level based on a few sensor measurements from the boiler. Even though these methods work very well, the models used to predict the fouling and hence optimise the soot blowing have not been particularly interpretable. Many of the approaches include black-box type models such as ANN's, which makes it difficult to understand why the model is obtaining certain results and, should predictions be wrong, it is difficult to establish where the fault came from. Understanding why these errors occur or how the internals of the boilers work is invaluable information that can be used for asset integrity management in the long run and can help with the decision-making regarding soot blowing procedures. Having a clear picture of the internal mechanisms that drive fouling accumulation and soot blowing efficiency could provide support for decision-making tasks, such as soot-blowing schedules and planned downtime of the boilers.

The aim of this study is therefore to develop a fouling prediction methodology that,

- Can estimate the level of fouling in different sections of the boiler based on local sensor measurement inputs, such as temperature and pressure.
- Investigate the identifiability of the physics in the recovery boiler using verification problems.
- Obtain basic predictive models that are interpretable and can be used to better understand the underlying physics within the recovery boiler.
- Investigate methods of obtaining consistent model results for a fixed period in the boiler that could potentially be used for soot blowing optimisation.
- Allows the predictive models and results to be interpreted by people who are not necessarily experts in the field of soot blowing.

Several thermodynamic models have been developed in the papers by Shi et al. (2015) and Peña et al. (2011) that can estimate the level of fouling within a boiler. These methods are based on simple heat and mass transfer relations that incorporate both convection and radiation heat transfer. These models often only approximate the physics within the boiler, but they are sufficient for the estimation of fouling levels and can be used in machine learning models further on, as was demonstrated in the papers mentioned above (Shi et al. 2019), (Peña et al. 2011), (Anitha Kumari & Srinivasan 2019).

Interpretable models have become more of a necessity as the need to understand why errors occur in machine learning models has grown and as models that can generalize outside the domain of the training data have become necessary. Recent research into this topic has resulted in an algorithm known as the sparse identification of non-linear system dynamics (SINDy) algorithm. This algorithm, developed by Brunton et al. (2016), can extract the physics of a system from measurement data from the same system. Being able to extract these physics equations is invaluable for obtaining models that can be interpreted. Since the physics equations are extracted, the models can also be used to generalise beyond the domain of the training data and can be used in the decision-making processes regarding the specific system.

A method is proposed in this study that is not only computationally inexpensive, relative to other machine learning models, but can also extract the underlying physics during a soot blowing operation, which results in interpretable models. The method is based on studies that were conducted using the SINDy algorithm in the papers by Brunton et al. (2016) and Kaiser et al. (2018), which is combined with fouling estimation techniques proposed by Shi et al. (2019) and Peña et al. (2011). Initially the algorithm is tested in a controlled environment with artificial datasets and models, to determine the strengths and weaknesses of the algorithm. The algorithm is tested under varying noise and initial value conditions, as well as varying model conditions, to determine the algorithm's recovery ability. This has the added benefit of showing one what results can be expected when the algorithm is implemented on the real dataset and to determine what possible complications may be encountered. In the preliminary results, it is seen that the algorithm is more than capable of extracting logical models from typical soot blowing sequences, however it is sensitive to noise, initial value changes and changes in the soot blowing sequence curve shape.

The SINDy algorithm is subsequently implemented on the recovery boiler measurement dataset, and the results are seen to be the same as was expected, based on the initial verification problems. Several attempts to achieve model consistency are made to try and circumvent the severe variation in the data. Soot blowing sequences are then manually selected to illustrate what would be needed should one want to extract an average model from the soot blowing sequence data and to further highlight the data complexity.

1.4 Document Layout

The development of the thermodynamic model, used to determine the fouling level within the Kraft recovery boiler, is presented and explained in detail in the second chapter of this work. Furthermore, the sensor measurement data from the boiler is analysed and preprocessed before being sent through the thermodynamic model. Finally, the thermodynamic model is briefly validated and some of the complexities of the data are highlighted to conclude chapter 2.

In chapter 3, the SINDy algorithm is introduced as well as the methodology and working of the algorithm. The algorithm is then tested using a verification model to establish the algorithm's model extraction capabilities. The soot blowing sequences with their corresponding fouling factors are then extracted from the Ngodwana dataset. The SINDy algorithm is broadly applied (a default algorithm setup is used) to these sequences to establish a few basic model forms for use in additional validation experiments. The experiments that follow, look into the identifiability of the underlying physics models and the influence of the model coefficients on artificial, but representative soot blowing curves. Lastly, the SINDy algorithm's recovery ability is tested under different noise and initial conditions. The algorithm's recovery ability is also tested when training samples are limited and lower-order models are fitted to higher-order data.

Chapter 4 sees the implementation of the SINDy algorithm on the Ngodwana dataset. To start, the optimal polynomial order is determined based on the work done in chapter 3 and results obtained from the measurement dataset. A baseline algorithm setup is established, and the extracted models are investigated. New inputs are added to the SINDy algorithm to attempt to improve model consistency in terms of the coefficients that are recovered as well as the overall prediction accuracy of the models. Once the inputs are tested and the results interpreted, the threshold parameter of the optimisation algorithm is sequentially optimised, and the dataset is normalised and scaled in an attempt to achieve model consistency. Finally, soot blowing sequences are manually selected to remove some data variation and to try and illustrate the complexity of the problem and dataset.

The work is concluded in chapter 5 and several recommendations are given for future research possibilities regarding this specific project and problem.

CHAPTER 2

Fouling Accumulation Data

To optimise the soot blowing sequences within SAPPI's recovery boiler, one would require an indication of the level of fouling within the boiler. Without such a measurement, optimisation is not possible, as one would have no tangible way of estimating the effectiveness of the soot blowers and hence no way of determining whether it is cost-effective to soot blow. Furthermore, if any predictive models are to be built, one would need a 'label' to predict, or a metric of fouling to extract models from. The lack of a fouling metric is the reason for the current soot blowing methodology at the plant. The soot blowing sequences are simply scheduled beforehand and altered manually after a boiler inspection, instead of having an adaptive program that activates soot blowers when necessary.

Due to the highly corrosive environment and high temperatures within the boiler, it is difficult to maintain instrumentation, such as heat flux sensors and cameras, to measure and monitor the degree of fouling on the heat transfer surfaces. These sensors are also expensive, not only to purchase but also to install in a boiler, as modifications need to be made to the boiler itself and the proper infrastructure needs to be installed so that these sensors can integrate with the existing monitoring system. For this reason, it is not a practical solution for monitoring the fouling levels in this boiler, as explained before.

Currently, many sensors within the boiler are used to monitor the internal conditions and to adjust the fuel and air rates as necessary to obtain a specific boiler output. Typical measurements include pressure and temperature, which are ideal for developing basic thermodynamic models for the estimation of fouling. While it is very difficult to develop a model that can capture all the non-linearities and random occurrences in the boiler, it is also unnecessary to have such a model. A detailed model would be helpful if the exact level of fouling had to be calculated. However, in our case, we simply need a representation of fouling within the boiler that is accurate enough to perform basic calculations with, to extract physics models and hence potentially optimise the soot blowing sequences.

To represent the degree of fouling within the recovery boiler, a Fouling Factor (FF) is used which uses the heat transfer as proxy, since the fouling is difficult to measure. The fouling factor combines real-time data from the data storage system of the plant (DCS system) as well as some basic thermodynamic calculations and data to obtain an estimated level of fouling within the boiler. The fouling factor is defined as follows, adapted from Shi et al. (2019) :

$$FF = \frac{h_{theoretical} - h_{actual}}{h_{theoretical}} \tag{2.1}$$

where $h_{theoretical}$ refers to the theoretical heat transfer coefficient of the boiler's heat transfer surfaces and h_{actual} refers to the real heat transfer coefficient of the boiler's heat transfer surfaces. The fouling factor's definition is chosen so that an increase in the amount of soot on the heat transfer surfaces would correlate with an increase in the fouling factor value. The definition also bounds the fouling factor between 0 and 1, where 0 refers to an ideal clean state in the boiler and 1 refers to a completely fouled boiler. Any value for the FF above 0 would show that some measure of soot deposition is present on the boiler heat transfer surfaces. Therefore, to define a fouling factor, it is necessary to obtain the approximate actual and theoretical heat transfer coefficients in the boiler.

In this chapter, a basic thermodynamic model is built, to determine the level of fouling within the boiler. This metric can be used for data-driven modeling and also allows one to determine the types of equations that form part of the underlying physics equations in the boiler, which can help with the SINDy modelling. It should be noted that the thermodynamic model was built in a joint effort with André van Zyl. The reason for this collaboration is because the same boiler problem is being approached in both studies, however, different machine learning approaches are followed. To ensure that the different approaches can be compared (should SAPPI need to choose an approach to implement on the plant), the same method of obtaining a fouling estimation, and thus a labeled dataset, was used.

2.1 The theoretical heat transfer coefficient

To obtain the theoretical heat transfer coefficient of the recovery boiler, basic thermodynamic equations and empirical correlations are used. In the boiler, heat is transferred from the flue gas, through the superheater pipes, into the steam. Therefore, a boiler can be seen as a type of crossover heat exchanger and thermodynamic calculations pertaining to heat exchangers are used. In a typical heat exchanger, the internal and external heat transfer coefficients are not the same. The difference can be due to the internal and external surface areas not being the same size or the fluid types used in the heat exchanger. It is therefore necessary to calculate both the external and internal heat transfer coefficients and then combine them into an overall heat transfer coefficient to ensure that the model is as accurate as possible. The overall heat transfer coefficient is usually dominated by the smaller internal or external heat transfer coefficients. This is because of the method of calculating the overall heat transfer coefficient, where the smaller of the internal or external coefficients creates a bottleneck (Cengel & Ghajar 2015).

For the calculation of the overall theoretical HTC, the following **assumptions** are made:

- Pseudo Steady-state conditions in terms of the mass going in equals the mass going out. Since the dataset used will be time-series data, it is expected that the system will evolve over time, however, for short sections of the data, conditions should remain fairly steady making it a fair assumption.
- Constant surface temperature of heat transfer surfaces over time, as the boiler conditions on the fireside are kept as consistent as possible and there is no way of measuring the surface temperature in the boiler currently.
- Smooth tube surfaces.

2.1.1 The internal theoretical heat transfer coefficient

To calculate the internal heat transfer coefficient, the Gnielinski (1976) correlations are used. This method is also used in the paper by Peña et al. (2013). The Gnielinski correlation is a relation suited for turbulent flow with a Reynolds number between 3×10^3 and 5×10^6. To ensure that turbulent flow is present, the Reynolds number must be confirmed. The Reynolds number can be calculated as follows:

$$Re = \frac{V_{avg} \times D}{\nu} \tag{2.2}$$

where V_{avg} refers to the average velocity of the steam, D refers to the diameter of the pipe, and ν refers to the kinematic viscosity of the steam. Typically, the flow is turbulent in the superheaters of boilers and is confirmed with equation 2.2

The Gnielinski relation is then defined as follows:

$$Nu = \frac{(f/8)(Re - 1000)Pr}{1 + 12.7(f/8)^{0.5}(Pr^{2/3} - 1)} \qquad \begin{pmatrix} 0.5 \leq Pr \leq 2000 \\ 3 \times 10^3 < Re < 5 \times 10^6 \end{pmatrix} \tag{2.3}$$

where Nu refers to the Nusselt number, Pr to the Prandtl number of the steam, and f refers to the friction factor that can be calculated for smooth tubes as follows:

$$f = (0.790 lnRe - 1.64)^{-2} \tag{2.4}$$

Once the Nusselt number has been calculated, the internal heat transfer coefficient can be calculated by simply rearranging the equation for the Nusselt number:

$$h_{int} = \frac{kNu}{D} \tag{2.5}$$

where k refers to the thermal conductivity constant of the fluid.

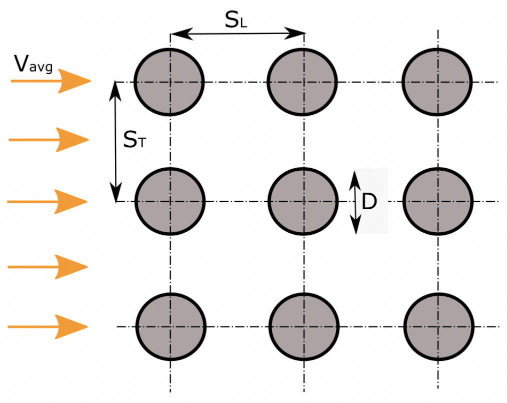
2.1.2 The external theoretical heat transfer coefficient

For the calculation of the external heat transfer coefficient, the Zukauskas correlations (1987) are used. These correlations are suited for heat transfer calculations of fluid flow through tube banks. Since the superheaters of the recovery boiler comprise of platens of tubes forming tube banks, these correlations are well suited for the heat transfer calculations in the boiler. The same methodology is also employed in the papers by Peña et al. (2013), Shi et al. (2019) and Shi et al. (2015).

For the Zukauskas correlations, it is once again necessary to calculate the Reynolds number, however, the average fluid velocity (V_{avg}), as defined in equation 2.2, is now replaced with the maximum fluid velocity through the tube bank. The maximum velocity through the tube bank can be calculated as follows:

$$V_{max} = \frac{S_T}{S_T - D} V_{avg} \tag{2.6}$$

where S_T refers to the transverse spacing between the tubes. It should be noted that this calculation for the maximum velocity in the tube bank only holds when the arrangement of the tube bank is an in-line arrangement. In the case of the recovery boiler at Ngodwana, all the superheater tube banks are in-line arrangements. An example of



In-line Arrangement

Figure 2.1: Example of a typical in-line arrangement adapted from Cengel & Ghajar (2015).

an in-line arrangement can be seen in figure 2.1 where the transverse spacing is also shown.

Once the Reynolds number is known, the Zukauskas correlations can be used for calculating the Nusselt number. The Zukauskas correlations are shown in Table 2.1 for an inline arrangement. It should be noted that these correlations only hold when there are over 16 tubes in the longitudinal direction ($N_L > 16$). If there are fewer tubes, however, Table 2.2 should be used to obtain a correction factor for the Nusselt number. It should be noted that all the fluid properties should be evaluated at the arithmetic mean temperature of the fluid, except for Pr_s, which should be evaluated at the surface temperature of the tube bank.

Table 2.1: Zukauskas Correlations for crossflow over tube banks when $N_L > 16$ adapted from Cengel & Ghajar (2015).

Range of Re	**Correlation**
0-100	$Nu = 0.9Re^{0.4}Pr^{0.36}(Pr/Pr_s)^{0.25}$
100-1000	$Nu = 0.52Re^{0.5}Pr^{0.36}(Pr/Pr_s)^{0.25}$
1000 - 2×10^5	$Nu = 0.27Re^{0.63}Pr^{0.36}(Pr/Pr_s)^{0.25}$
2×10^5 - 2×10^6	$Nu = 0.033Re^{0.8}Pr^{0.4}(Pr/Pr_s)^{0.25}$

Table 2.2: Correction factor for Zukauskas Correlations when $N_L < 16$ adapted from Cengel & Ghajar (2015). $Nu = FNu_{N>16}$

N_L	1	2	3	4	5	7	10	13
F	0.70	0.80	0.86	0.90	0.93	0.96	0.98	0.99

Once the Nusselt number has been determined, Equation 2.5 can once again be used to calculate the external heat transfer coefficient for a section of the boiler.

2.1.3 The theoretical radiation heat transfer coefficient

In a boiler, there are two fundamental mechanisms of heat transfer, namely convective heat transfer and radiation heat transfer. So far, the external and internal coefficients have not considered radiation heat transfer, which plays a significant role in a boiler's total heat transfer (Peña et al. 2013). This is especially true for the earlier sections in the boiler that are closer to the furnace. This was confirmed when the plant engineer at Ngodwana stated that the primary mechanism of heat transfer in the Primary 1 and 2 superheaters of the boiler was radiation. Furthermore, the paper by Peña et al. (2013) states that in some boiler cases, neglecting the radiation could result in a deviation of up to 30 % between the heat released by the flue gas and the heat absorbed by the steam.

To calculate the radiation heat transfer coefficient, the Hottel correlations (1954) can be used. The same radiation heat transfer equations are used in the papers by Peña et al. (2013), Shi et al. (2015) and Shi et al. (2019) to calculate the radiation heat transfer coefficient. Hottel's correlation is defined as follows:

$$q_{net,gray} = \frac{\varepsilon_s + 1}{2} A_s \sigma (\varepsilon_g T_g^4 - \alpha_g T_s^4) \tag{2.7}$$

where ϵ_s refers to the surface emissivity of the walls and pipes, ϵ_g refers to the emissivity of the gas and α_g refers to the absorptivity of the gas (Cengel & Ghajar 2015). T_g and T_s refer to the temperature of the gas and of the surface, respectively. It should be noted that Hottel's correlation is developed for nearly black surfaces, or surfaces with $\epsilon_s >$ 0.7. It is stated in the book by Cengel & Ghajar (2015) that the surfaces of the walls in furnaces and boilers typically have an emissivity larger than 0.7, therefore this model is well suited to the boiler problem. Most of the heat transfer surface area in the boiler is manufactured from ASME 210 A1 steel. Comparing this steel's estimated emissivity and other similar steels' emissivities, it was seen that the typical emissivity was around 0.75 to 0.88. This once again proves that the Hottel correlation should work well for this case study.

To obtain the emissivity of the flue gas, one would have to know the approximate constituents of the gas mixture. This is because the emissivity is a function of the temperature and the partial pressure of the constituent gases. Typically, flue gas is made up of a mixture of N_2, CO_2, H_2O, O_2 and CO. This is also the case for the flue gas in the recovery boiler. In the book by Cengel & Ghajar (2015), it is stated that CO_2 and H_2O are typically gasses that participate in radiation heat exchange, while gases such as N_2 and O_2, do not participate in this exchange. The concentration levels of CO is typically so low in flue gas mixtures, that it can be safely neglected for the emissivity calculations. Plants would typically actively prevent CO from forming by preventing incomplete combustion.

Once the flue gas constituents are assumed, one can use the Hottel emissivity tables to obtain the approximate gas emissivity. First, the emissivity of a gas containing H_2O and other non-participating gases is found in Figure 2.2a. The L in the graph refers to the mean distance traveled by the radiation beam. Similarly, Figure 2.2b is used to find the emissivity of a gas containing CO_2. Finally, the correction factor for a gas containing both H_2O and CO_2 is found using Figure 2.3. To find the overall emissivity, the following equation can be used:

$$\varepsilon_g = \varepsilon_{CO_2} + \varepsilon_{H_2O} - \Delta\varepsilon \tag{2.8}$$

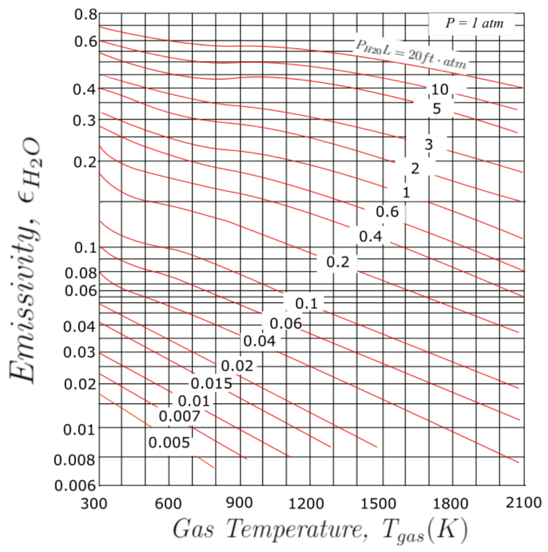
The absorptivity of the flue gas can then be determined. Once again, the absorptivity is determined for the participating gases in the gas mixture separately and then combined (Cengel & Ghajar 2015):

$$\begin{aligned} CO_2: \quad & \alpha_{CO_2} = (T_g/T_s)^{0.65} \times \varepsilon_{CO_2} \\ H_2O: \quad & \alpha_{H_2O} = (T_g/T_s)^{0.45} \times \varepsilon_{H_2O} \end{aligned} \tag{2.9}$$

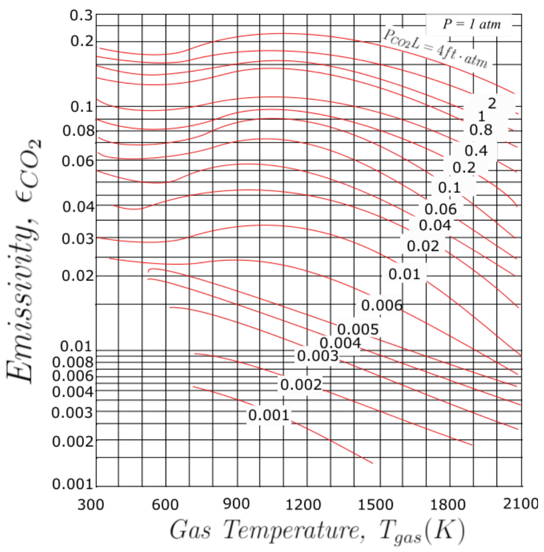
To find the final gas absorptivity, we use the following equation:

$$\alpha_g = \alpha_{CO_2} + \alpha_{H_2O} - \Delta\alpha \tag{2.10}$$

where $\Delta\alpha = \Delta\varepsilon$



(a) H_2O emissivity



(b) CO_2 emissivity

Figure 2.2: Emissivity of gases with CO_2 or H_2O and other non-participating gases at atmospheric pressure (Adapted from: Cengel & Ghajar (2015)).

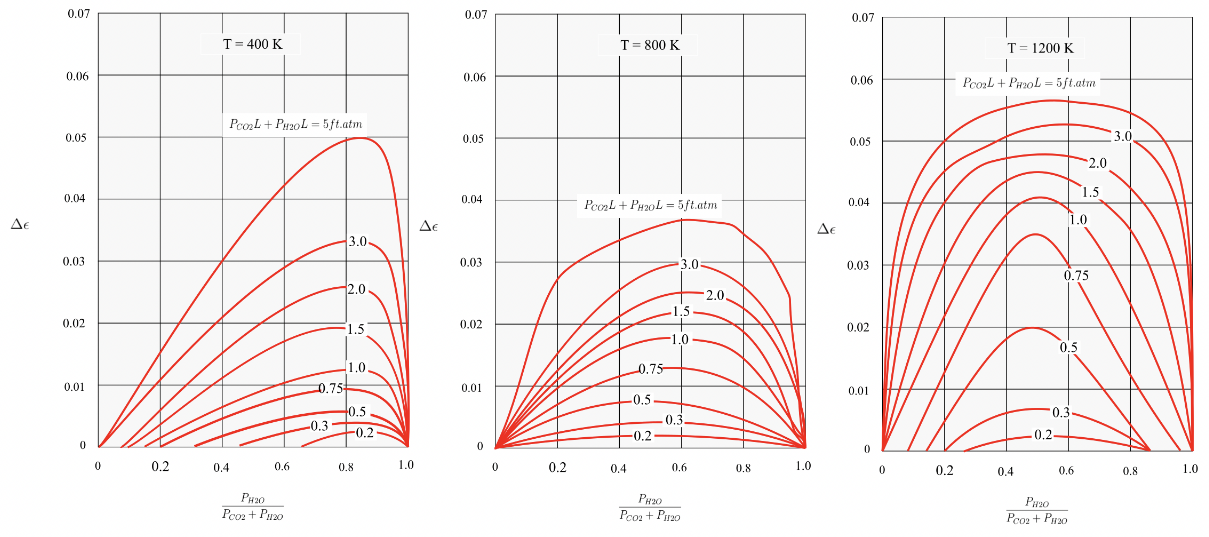


Figure 2.3: Correction emissivity for gases containing both CO_2 and H_2O (Adapted from: Cengel & Ghajar (2015)).

Once all the factors have been determined, $q_{net,gray}$ is determined and is used to find the radiation heat transfer coefficient as follows:

$$h_{rad} = \frac{q_{net,gray}}{A_s(T_g - T_s)} \tag{2.11}$$

where A_s is once again the heat transfer surface area.

2.1.4 The overall theoretical heat transfer coefficient

Once the external, internal and radiation heat transfer coefficients are determined, the overall theoretical heat transfer coefficient can be calculated using equation 2.12

$$\frac{1}{h_{theoretical}A} = R = \frac{1}{h_{int}A_i} + \frac{ln(D_o/D_i)}{2\pi kLN_{tubes}} + \frac{1}{h_{ext}A_o + h_{rad}A_o} \tag{2.12}$$

where A_i and A_o refer to the internal and external surface areas respectively, N_{tubes} refer to the number of tubes in the boiler and L refers to the length of the tubes. D_o and D_i refer to the outer and inner diameters of the tubes, respectively, and k refers to the conduction heat transfer coefficient of the tubes. Sometimes, the tubes will be relatively thin and the material will have a high conductivity. This will cause the middle term of equation 2.12 to become negligibly small relative to the other two terms (Cengel & Ghajar 2015). However, for the sake of accuracy, all the terms were included in this study. The overall heat transfer coefficient should now be a relatively accurate approximation of the true theoretical heat transfer coefficient and can be used in the calculation of the fouling factor.

2.2 The actual heat transfer coefficient

The actual heat transfer coefficient is calculated using an energy balance equation. The same methodology is used in the papers by Shi et al. (2015) and Shi et al. (2019). Another soot blowing optimisation project, conducted at SAPPI's Saiccor mill, used a similar method of finding the heat transfer coefficients. In this project, the log-mean temperature difference (LMTD) was also used in their analysis and the engineers confirmed that it was sufficiently accurate for fouling factor calculations for their project. The following assumptions are made when calculating the actual heat transfer coefficient:

- Steady-state conditions, similar to the theoretical heat transfer coefficient.
- Boiler loads stay relatively constant and do not fluctuate continuously.Typically, conditions in the boiler remain constant for at least 6 hours at a time at a minimum, while the conditions also change very slowly should they be changed as was observed from the dataset.
- No large leakages in the boiler relative to the amount of air pushed through it, as was seen when on a plant visit to the boiler itself.
- Boiler conditions remain stable, i.e. the temperature of the metal surfaces and fluid parameters does not change drastically or fluctuate often similar to that of the boiler load time frames.

To calculate the actual heat transfer coefficient, the heat absorbed by the steam is calculated using equation 2.13

$$\dot{Q}_{absorbed} = \dot{m}_s(H_{out} - H_{in}) \tag{2.13}$$

where m_s refers to the mass flow rate of the steam and H refers to the enthalpy. The enthalpy of the steam is a function of temperature and pressure and can be determined from the thermodynamic tables for super-heated steam, once the temperature and pressures are known.

To determine the actual heat transfer, the log-mean temperature difference (LMTD) method is used. The LMTD method is a steady-state method that can be used when the surface temperature of a heat exchanger remains relatively constant (Cengel & Ghajar 2015). In this case, it is assumed that the surface temperature does not fluctuate often in the boiler and hence the LMTD method should be valid. However, it should be noted that a dynamic analysis, which incorporates changes to the temperature of the surface would likely result in a more accurate estimation of the actual heat transfer coefficient. In this case, however, it is not possible to incorporate such terms, since no surface temperature measurements are available in an online capacity in the boiler and can thus not be incorporated into the model. Since it has been confirmed by the other SAPPI project that using the LMTD is sufficiently accurate for fouling estimation, this method will be used. The actual heat transfer coefficient is hence defined as follows:

$$h_{actual} = \frac{\dot{Q}_{absorbed}}{A_s \Delta T_{LMTD}} \tag{2.14}$$

and the log-mean temperature can be calculated using equation 2.15

$$\Delta T_{LMTD} = \frac{\Delta T_1 - \Delta T_2}{ln(\Delta T_1 / \Delta T_2)} \tag{2.15}$$

where $\Delta T_1 = T_{fi} - T_{so}$ and $\Delta T_2 = T_{fo} - T_{si}$. Here, T_{fi} and T_{fo} refer to the flue gas inlet and outlet temperature respectively, while T_{si} and T_{so} refer to the inlet and outlet steam temperatures, respectively.

2.3 Data acquisition and pre-processing

From the thermodynamic model, it is possible to determine which parameters and sensor measurements are required, to determine the fouling factor in the respective sections of the boiler. The thermodynamic model was be applied to the different sections of the boiler, respectively, and not to the boiler as a whole. This is done to obtain a more detailed picture of where in the boiler problems occur and to ensure higher accuracy in terms of the fouling factor. Since each section also has its respective soot blowers and heat exchanger configuration, it makes sense to apply the model to each section individually. The sensor measurements in the boiler are also separated by the sections of the boiler and therefore, this method is also more practical. The parameters and sensor measurements required can be seen in table 2.3. In most cases, the sensors mentioned in the table are roughly located in the middle, referring to the height of the platen, before and after each superheater or heat exchanger platen.

The boiler consists of several sections, namely, the primary 1, 2, and 3 superheaters, the secondary superheater, the boiler bank, and the economiser. Sections that are experiencing the highest amount of fouling are the superheaters and the boiler bank according to the operators of the boiler at the plant, since they monitor fouling accumulation hot spots when the boilers are water washed. The only sections that have sufficient sensor measurements and information available to build the thermodynamic model, are the secondary superheater and the primary superheater 2. As mentioned in the literature review, most boilers experience severe fouling and sometimes plugging in the boiler bank

Table 2.3: Boiler Parameters and sensor measurements required for the thermodynamic model.

SYMBOL	DESCRIPTION	UNITS	OBTAIN FROM
m_s	Steam mass flow rate	kg/s	Sensor measurement
m_f	Flue gas mass flow rate	kg/s	Approximate from measurements
V_w	Velocity of the steam through the pipes	m/s	Derive from measurements
T_{iw}	Inlet temperature of the steam/water	$^{\circ}C$	Sensor measurement
T_{ow}	Outlet temperature of the steam/water	$^{\circ}C$	Sensor measurement
T_{if}	Flue gas Inlet temperature	$^{\circ}C$	Sensor measurement
T_{ef}	Flue Gas Outlet temperature	$^{\circ}C$	Sensor measurement
T_s	Surface temperature of the tubes	$^{\circ}C$	Approximate from measurements
V_{in_f}	Flue gas velocity into tube banks	m/s	Approximate from measurements
D_o	Outside diameter of the tubes	m	Design parameter
D_i	Inner diameter of the tubes	m	Design parameter
ST	Transverse spacing of tube bank	m	Design Parameter
SL	Longitudinal spacing of tube banks	m	Design Parameter
N	Number of tubes in tube bank	-	Design Parameter
L_t	Lenth of the tubes in tube bank	m	Design Parameter
e_s	Surface emmissivity of tubes	-	Design Parameter
k	Tube thermal conductivity	$W/m.K$	Design Parameter
L	Mean distance traveled by radiation beam	m	Approximate from schematics
P	Flue gas pressure	Pa	Sensor measurement
P_{iw}	Inlet water pressure	kPa	Sensor measurement
P_{ow}	Outlet water pressure	kPa	Sensor measurement

and the superheaters leading up to the boiler bank. Unfortunately, there are currently no measurements showing the steam to water ratio in the boiler bank and temperature measurements are scarce. It is therefore not possible to build our thermodynamic model for the boiler bank section at this stage.

The two superheaters leading up to the boiler bank are the primary 2 superheater and the secondary superheater for which there are enough sensor measurements and information. Thus, the study focuses on determining the level of fouling in these two sections. The other superheaters and economiser have not experienced as much fouling according to Johan Kok, a plant engineer at Ngodwana, and optimisation of their soot blowing sequences are not as critical for the time being.

As seen in table 2.3 some inputs required for the thermodynamic model have to be derived or approximated from other sensor measurements. To derive the velocity of the steam, the mass flow rate sensor measurement value is taken and is divided by the density of steam at the arithmetic mean temperature of the steam as can be seen in equation 2.16. The volumetric flow rate is then obtained, which is divided by the inlet flow area of the steam in the boiler section to get the mean velocity seen in equation 2.17.

$$\dot{V} = \frac{\dot{m}}{\rho_s} \tag{2.16}$$

$$v_{steam} = \frac{\dot{V}}{A_{inlet}} \tag{2.17}$$

The surface temperature of the tubes is assumed to be constant at 540 $^{\circ}C$, which is a value that was obtained as an estimate when visiting the plant. Currently, there are no sensors measuring the surface temperature online and this assumption must therefore be used. The surface temperature is only used for the calculation of the theoretical heat transfer coefficient and it was seen, when performing a basic sensitivity study, that the surface temperature does not significantly affect the theoretical HTC value. In the sensitivity study, the surface temperature is varied from -10 % to 10 % of its original value, which only makes a -1.07 % to 1.14 % difference in the theoretical HTC.

To get an approximation of the flue gas mass flow, the mass balance for the combustion process is used. The approximate mass balance was obtained from Kendrick Mashego,the process engineer at the plant, and is used in conjunction with the sensor measurement, measuring black liquor flow, to obtain the amount of black liquor converted to gas. It was seen that, on average, 42 % of the black liquor is converted to gas when combusted. This gas flow is added to the total amount of air actively blown into the boiler, to get the flue gas mass flow. The flue gas calculation can be seen in equation 2.18. The flue gas velocity is calculated using the same principle as the steam velocity calculation mentioned above.

$$\dot{m}_{Flue} = \dot{m}_{air} + 0.42 \times \dot{m}_{BlackLiquor} \tag{2.18}$$

2.3.1 Data Acquisition

All the sensor measurement data from the entire Ngodwana plant is stored on a server called the Distributed Control System (DCS). The dataset required for the thermodynamic model was drawn from this server. The data was drawn for approximately 4 months from 10 February 2020 to 10 May 2020. This time interval was chosen as there WAS no boiler shuts in between those dates due to broken or malfunctioning equipment. The boiler was water washed the day before the commencement of the dataset and was shut down for a water wash shortly after the last entry in the dataset. Therefore, the dataset was chosen in such a way that one could see the fouling factor development over time, and so that any problems occurring in the boiler, are only because of fouling and not malfunctioning equipment. The time interval for samples was chosen to be 10 seconds to take the necessary sensor measurements for the thermodynamic model. This allows the steady-state assumption to hold as boiler conditions do not change as quickly and hold steady for time frames typically more than 6 hours as was seen when investigating the data. Typically conditions also change over several hours, thus small intervals will allow the pseudo-steady-state assumption to hold. This interval was also chosen because of the length of the soot blowing sequences. A typical soot blowing sequence lasts between two and a half to three and a half minutes and, having ten-second interval measurements, should give one a good idea of the fouling factor change during a soot blowing sequence. Drawing the data in such small time steps also ensures that one has enough data to build accurate models. The final dataset drawn had over a million samples with approximately 800 sequences per soot blowing pair where this pair of soot blowers were active over the period.

2.3.2 Data pre-processing

Once the data was drawn from the DCS, pre-processing was required, as the data was contaminated with incorrect measurements and false values. Since the data came from live sensor measurements, it is also bound to be corrupted by noise, further increasing the need for pre-processing. Some of the measurement samples will be outliers and will have to be filtered out. Many of the measurement units from the sensors and the DCS system are not in standard engineering units or in the unit of measure required for the thermodynamic model. Therefore, an additional step was required, where the data was converted into the correct units of measure.

To start with the data pre-processing, all the units of measure were converted to standard engineering units. The mass flow rates were converted from 'T/Hr' to 'kg/s', the pressures were converted from 'mmH_2O' gauge pressure to 'kPa' absolute pressure and the volumetric flow rates were converted from 'l/min' to 'm^3/s'.Once the units were all converted, equation 2.18 was used to create a new data feature, namely the flue gas mass flow rate.

When all the features were created and the units are correct, it was necessary to deal with the empty or 'Nan' values in the dataset. These values occur when sensors malfunction and fail to generate an entry for the dataset. In literature, there are different ways of dealing with these types of values. According to an article by Kumar (2020), one can either remove samples with 'Nan' values, impute them with specific values such as 0 or with statistical metrics such as the mean or median. Similar statements are made in the paper by Sharpe & Solly (1995). Imputing these samples with values, such as the mean or median, may skew your data to some extent or replace the sample with a completely incorrect value. This is especially true for this dataset, as some outliers may affect the mean so much that the value makes no sense in the dataset. Similarly, imputing the value with a specific entry such as zero might work well in some cases, however one must have a very good idea of what the actual value should be. Finally, one can handle these 'Nan' values by simply removing them. Removing 'Nan' values is not a good idea when one has a very small dataset, as one loses some information. Since the data set for this study contains over one million samples, this should not be a problem.

While removing Nan values could negatively affect the sampling rate, it was noticed in the dataset that the Nan values were not often found within soot blowing sequences, and if they were the entire soot blowing sequence would consist of Nan values and would therefore be dropped for this project. It should however be kept in mind that in some cases the sampling rate might have been affected slightly, but since there was no viable alternative, the samples containing 'Nan' values were dropped.

Lastly, the dataset had to be filtered for outliers as these samples might slow down the machine learning models by causing them to converge slower and could potentially prevent them from finding any solutions. To filter the outliers, each measured feature was taken, and the mean and standard deviation of that feature was determined for all the samples. Every sample for the specific feature was then inspected. Should the sample lie outside of the 99% confidence interval, the entire sample, for all features in the dataset, was dropped. This filter is a very simple one, however, advanced outlier detection algorithms are extensive fields of research on their own and are not necessary nor possible for this specific study. Once this rudimentary filter was applied, the dataset was ready for use in the thermodynamic model. The fouling factor could be calculated for each sample using the equations described earlier in the chapter.

2.3.3 Validating the Fouling Factor data and thermodynamic model

Once the thermodynamic model is applied to every sample in the dataset, it is necessary to ensure that plausible answers for the fouling factor are obtained and that the fouling factor development makes sense. To validate the model to some extent, the fouling factor is plotted, together with the soot blower sequences. This is done to see whether the fouling factor drops during a soot blowing sequence and increases when no soot blowers are active. The plot can be seen in Figure 2.4. The fouling factor does indeed drop when a soot blower is active and rises when no soot blowers are active. The response of the fouling factor to the soot blowers makes sense and is expected. It is thus assumed that the thermodynamic model works relatively well and that the model can pick up the changes in the system when soot blowers are active. Some soot blowers have more of an influence than other soot blowers, which also helps to validate that the model works correctly, as their influence depends on their location in the boiler. It should be noted that the fouling factor data shown in the figure is from the last part of the dataset, when the fouling in the boiler was already significant and the boiler was close to being shut down for a water wash. The fouling factor value is relatively high, at approximately 0.7 which is an expected value, as this part of the dataset that was tested, was near the end of the boiler cycle, just before it was water washed.

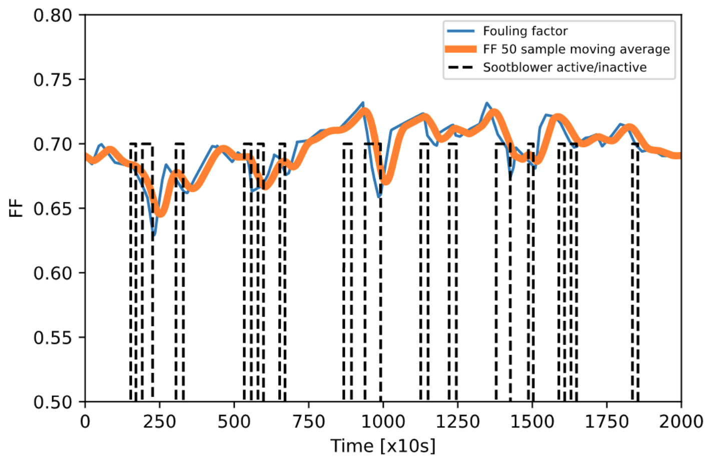


Figure 2.4: The influence that the soot blowers have on the fouling factor development in the secondary superheater.

Therefore, the boiler should be fouled heavily and the thermodynamic model seems to be obtaining plausible answers. It is therefore assumed that the thermodynamic model is behaving as is expected and the fouling factor data can be combined with the rest of the measurement dataset for use in machine learning models. The fact that one now has a measure of fouling in this boiler, is already a very positive result, as it was mentioned before, that previously, no such measurement or indication existed for the plant. Before any machine learning models have been applied, one already has a better view of what is happening in the boiler and the process engineers at the plant can most likely already use this model to some extent with their soot blowing decision making.

It should be noted that the data from the DCS system as well as the calculated fouling factor as is, is still very raw and most likely extensively corrupted with noise. How the data is to be used must therefore be considered carefully as there are several possible options. One possibility is to take the measurement data and fouling factor for the entire boiler and pass the entire set to a machine learning model. This however is most likely not a good idea, as there are too many changes that still occur over time in the entire boiler for a machine learning model to extract a consistent model. Especially regarding the SINDy algorithm, this would be a problem, as the soot blowers entering and leaving the boiler most likely change the physics and would therefore hinder the algorithm's capabilities. Another option is to focus on a specific section within the boiler. This option has many of the same potential problems as focusing on the entire boiler, as each section still has multiple soot blowers changing conditions in the section. However, the physics should remain more consistent compared to the entire boiler. This method is a possibility as each section in the boiler has its own temperature and pressure measurement sensors as was seen when building the thermodynamic model.

Lastly one could focus on a pair of sootblowers in a specific section, which would most likely result in the physics changing the least. In other words, the data set can be split into sequences where a specific soot blower pair is active. This should minimise the influence of other soot blowers on the physics, during a specific soot blowing sequence. However, as is clear, the dataset is complex and it is very difficult to decouple a section or sootblower pair from the others all together, as there are often more than one soot blower pair active in the boiler at a time. Most likely, this will cause difficulties in model extraction with the SINDy algorithm, as the models are likely very intertwined. To therefore, first test the capabilities of the SINDy algorithm, as well as discover potential problems one could encounter when using the real dataset, verification problems are built and tested to ensure the algorithm is capable of extracting these types of models. Once a full investigation has been completed on a virtual level, the real dataset can be used to confirm or disprove theories from the verification problems.

CHAPTER 3

Identifiability of the soot blowing physics

In the literature review, it is mentioned that traditional machine learning methods have developed much over the last decade and can fit very complex data sets. However, a caveat that usually comes with these models is the loss of interpretability, which makes it difficult to determine where errors come from, why they occur, and how to fix them. These machine learning models are also rarely able to generalize beyond the domain of the training data set and, therefore, are not well suited to adapt to new conditions. It is also mentioned in the literature review that the SINDy algorithm was developed to circumvent these problems. This is because the algorithm extracts the governing equations of a system from raw measurement data, which results in simple and interpretable models that can generalize well. In this chapter, the SINDy algorithm is tested using different artificial models and verification problems to establish the strengths and shortcomings of the algorithm. Basic model forms that can be expected from the Ngodwana dataset are found and the SINDy algorithm's recovery ability, under different initial, noise and underlying model conditions, is tested. These experiments are done, so that one can determine what can be expected once the SINDy algorithm is applied to the real Ngodwana dataset.

3.1 The SINDy algorithm

The SINDy algorithm is a relatively new method of looking at machine learning and model fitting. The algorithm does not fit a model to the input dataset directly, but rather tries to extract an ordinary differential equation or set of equations, through sparsity-promoting methods. The resulting models, assuming the feature library was well defined, should describe the underlying equations that govern a system. Typically these models can be interpreted fairly easily and can be used to generalise outside the domain of the training data (Brunton et al. 2016). As mentioned in the literature review, the SINDy algorithm tries to extract an ordinary differential equation (ODE) with the following form (all equations are adapted from the paper by Brunton et al. (2016).):

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)) \tag{3.1}$$

To determine the function f one has to gather time-series data of the states $\mathbf{x}(t)$. Therefore, the inputs to the function are measured at different time steps to form the input dataset $\mathbf{X}$. To obtain $\dot{\mathbf{x}}(t)$ one can either measure the derivative of the states of the system or calculate it using numerical methods, such as the finite difference method. Once again, the derivative is determined at different time steps to form $\dot{\mathbf{X}}$. $\mathbf{X}$ and $\dot{\mathbf{X}}$, therefore, form the following two matrices, where n is the number of variables and m refers to the number of samples:

$$\begin{gathered}
\mathbf{X} = \begin{bmatrix} x_1(t_1) & ... & x_{n-1}(t_1) & x_n(t_1) \\ ... & ... & ... & ... \\ x_1(t_{m-1}) & ... & x_{n-1}(t_{m-1}) & x_n(t_{m-1}) \\ x_1(t_m) & ... & x_{n-1}(t_m) & x_n(t_m) \end{bmatrix} \\
\dot{\mathbf{X}} = \begin{bmatrix} \dot{x_1}(t_1) & ... & \dot{x_{n-1}}(t_1) & \dot{x_n}(t_1) \\ ... & ... & ... & ... \\ \dot{x_1}(t_{m-1}) & ... & \dot{x_{n-1}}(t_{m-1}) & \dot{x_n}(t_{m-1}) \\ \dot{x_1}(t_m) & ... & \dot{x_{n-1}}(t_m) & \dot{x_n}(t_m) \end{bmatrix}
\end{gathered} \tag{3.2}$$

Once the matrices are constructed, a feature library is constructed, where the feature library comprises the possible functions that can form part of the ODEs describing the system. These functions can vary from being constant terms to polynomials and even trigonometric functions. A typical feature library is shown in equation 3.3.

$$\Theta(\mathbf{X}) = \begin{bmatrix} \mathbf{1} & \mathbf{X} & \mathbf{X^2} & \mathbf{X^3} & ... & \mathbf{tan}(\mathbf{X}) & \mathbf{cosec}(\mathbf{X}) \end{bmatrix} \tag{3.3}$$

where $\mathbf{X^2}$ refers to second-order functions of the state $\mathbf{x}$ such as $x_n^2(t)$ and $x_n(t)x_{n-1}(t)$. $\mathbf{X^3}$ refers to third-order functions of state $\mathbf{x}$. As mentioned in the paper by Brunton et al. (2016), the functions used in the feature library matrix can be chosen freely, as the sparse regression is assumed to only activate a few of these functions. The feature library functions should be chosen to represent some of the underlying physics equations, if some basic knowledge of these equations is found beforehand. Choosing representative functions could result in SINDy obtaining correlations closer to the true underlying physics. This is once again, one of the other reasons for building a basic thermodynamic model in chapter 2. The thermodynamic model gives one a basic idea of which functions one would expect to work well with the SINDy model and could help with the construction of the feature library.

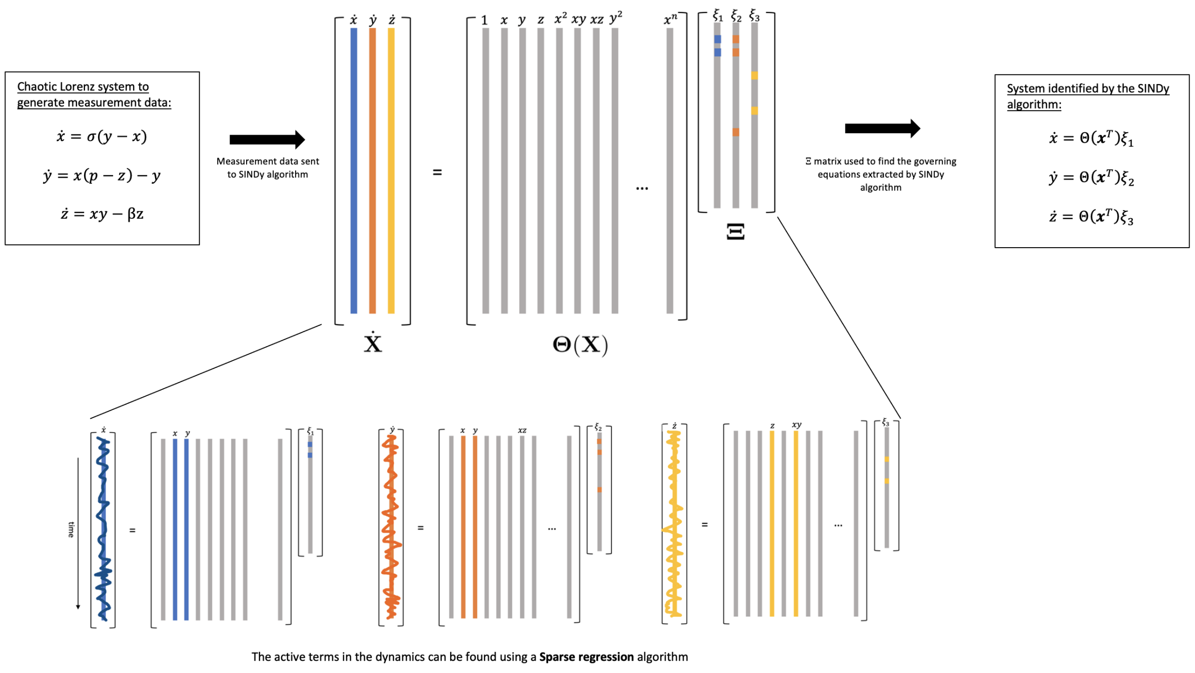
A sparse regression problem can be set up, once the feature library has been constructed and the derivative of the states has been measured or calculated. The sparse regression problem is set up as follows:

$$\dot{\mathbf{X}} = \Theta(\mathbf{X})\Xi \tag{3.4}$$

where $\Xi = \begin{bmatrix} \xi_1 & \xi_2 & \dots & \xi_n \end{bmatrix}$ refers to the vector of coefficients of the feature library functions. This vector is assumed to be sparse, due to it being calculated using a sparse regression optimizer, such as the sequentially thresholded least squares (STLSQ) optimizer (Brunton et al. 2016). Once Ξ has been determined, each of the governing equations can be found using the following equation:

$$\dot{\mathbf{x_k}} = \Theta(\mathbf{x}^T)\xi_k \tag{3.5}$$

A schematic of the SINDy algorithm can be seen in figure 3.1 which was adapted from the paper by Brunton et al. (2016). The schematic shows how the SINDy algorithm is applied to the chaotic Lorenz system example from the paper, by using sparse regression to find the active terms in the feature library. Once again it is also shown how the Ξ matrix is used to obtain each of the equations for the variables in the state $\mathbf{x}$.



Chaotic Lorenz system to generate measurement data:
$\dot{x} = \sigma(y - x)$
$\dot{y} = x(p - z) - y$
$\dot{z} = xy - \beta z$
Measurement data sent to SINDy algorithm
$\dot{x}$ $\dot{y}$ $\dot{z}$
=
1 x y z x^2 xy xz y^2 x^n
...
ξ_1 ξ_2 ξ_3
$\dot{\mathbf{X}}$
$\Theta(\mathbf{X})$
Ξ
Ξ matrix used to find the governing equations extracted by SINDy algorithm
System identified by the SINDy algorithm:
$\dot{x} = \Theta(\mathbf{x}^T)\xi_1$
$\dot{y} = \Theta(\mathbf{x}^T)\xi_2$
$\dot{z} = \Theta(\mathbf{x}^T)\xi_3$
time
x y
xz
z xy
The active terms in the dynamics can be found using a **Sparse regression** algorithm

Figure 3.1: Schematic of the SINDy algorithm, which is applied to the Chaotic Lorenz system from the paper by Brunton et al. (2016).

3.2 The verification problem

To ensure that the SINDy algorithm has the capabilities to recover the underlying equations for a typical soot blowing sequence, a verification problem is set up using equations from the paper by Shi et al. (2019). A verification problem is used, to ensure that the data can be controlled when confirming that the algorithm is working correctly before it is applied to noise corrupted measurement data. In the paper, they attempt to find the time-dependent equations that describe the fouling factor development during and after a soot blowing sequence. The following two equations are given in the paper to represent the two possible states:

$$F_d = 10.54 - 10.31e^{-0.001716t} \tag{3.6}$$

$$F_b = 0.328e^{-0.4515t} \tag{3.7}$$

where F_d and F_b refer to the fouling rate change curve during deposition and the fouling rate change during a soot blow, respectively. To obtain the differential equations of these equations, the derivative with respect to time is taken. After some mathematical manipulation, the differential equations can be written in the following form:

$$F_d' = 0.018086 - 0.001716F_d \tag{3.8}$$

$$F_b' = -0.4515F_b \tag{3.9}$$

These differential equations are the equations that the SINDy algorithm should be able to extract from the time series data. An artificial time series data set is generated using equations 3.6 and 3.7 and python's ODE integrator. The initial value is chosen to be 0.016, based on the paper by (Shi et al. 2019). Initially, no noise is added to the data set.

The SINDy model is constructed using the Python library known as PySINDy (de Silva et al. 2020). The SINDy model is constructed by first specifying the differentiation method (used to find the derivative of the input dataset), then specifying the library of base functions to be used (list of all possible functions that could make up the differential equations) and lastly specifying the optimizer to be used for the sparse regression. For the verification problem, most of the default parameters are used. The differentiation method is set as the central finite difference method, the function library is set to be a second-degree polynomial library and the optimizer is chosen to be the recommended sequentially thresholded least squares algorithm, which was also used in the papers by Brunton et al. (2016) and (Kaiser et al. 2018). The first dataset, generated by equation 3.6, is given as input to the SINDy model and the following differential equation is obtained:

$$F_d' = 0.018 - 0.002F_d \tag{3.10}$$

This equation corresponds well with equation 3.8. Similarly, the dataset generated by equation 3.7 is given as input to the model and an equation, almost identical to equation 3.9 is found and can be seen below:

$$F_b' = -0.452F_b \tag{3.11}$$

Satisfied that the SINDy algorithm does indeed extract the correct models, noise is added to the artificial data set from equation 3.6 to test the ability of the SINDy algorithm to handle different types of noise. Three types of noise are investigated, namely Gaussian, Laplacian and t-distributed noise. These noise distribution types are chosen, as they are commonly used to describe white noise in a system and are all continuous distribution types. For each noise type, the standard deviation of the noise is varied. The standard deviation is set to 0.05, 0.1, 0.2 and 0.5 for each noise type with a mean of zero, and the noise is added to the artificial data set. These noise percentages are chosen as it is highly likely that the real dataset will contain noise that lies within this range. The dataset is then given to the SINDy model to test whether the correct underlying equations are retrieved. Table 3.1 shows the results for this experiment and also indicates the root-mean-square error (RMSE) of the predicted function and the actual time series data. Even in very noisy cases, the SINDy algorithm can still retrieve the underlying model.

Table 3.1: SINDy algorithm model extraction capabilities with different types of noise.

Noise type	**Noise STD**	**Model found by SINDy**	**RMSE**
Gaussian Noise	0.05	$F_d' = 0.018 - 0.002F_d$	0.064
	0.1	$F_d' = 0.018 - 0.002F_d$	0.122
	0.2	$F_d' = 0.018 - 0.002F_d$	0.308
	0.5	$F_d' = 0.019 - 0.002F_d$	0.849
Laplacian Noise	0.05	$F_d' = 0.018 - 0.002F_d$	0.074
	0.1	$F_d' = 0.017 - 0.002F_d$	0.142
	0.2	$F_d' = 0.018 - 0.002F_d$	0.277
	0.5	$F_d' = 0.020 - 0.002F_d$	0.761
T-distributed Noise	0.05	$F_d' = 0.018 - 0.002F_d$	0.064
	0.1	$F_d' = 0.019 - 0.002F_d$	0.123
	0.2	$F_d' = 0.018 - 0.002F_d$	0.278
	0.5	$F_d' = 0.017 - 0.002F_d$	0.700

Furthermore, the RMSE does not spike to very large values in any of the experiments. All the RMSEs increase gradually as the noise standard deviation is increased, which is expected, as more noise makes it more difficult to identify the underlying physics model and the more noise is present the less exact a model will fit through the data which will result in a slight increase in the RMSE. The experiment shows that the SINDy algorithm

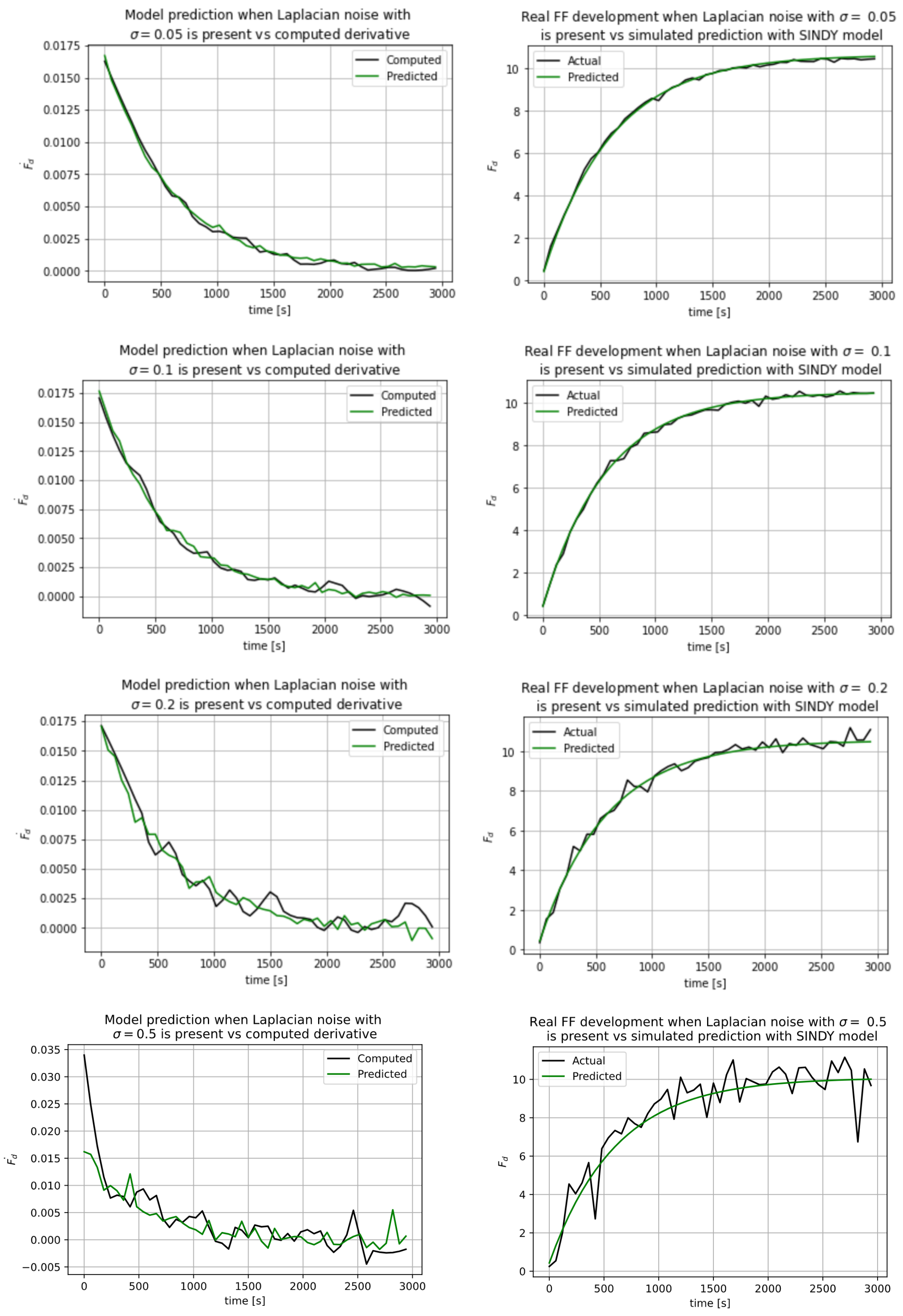


Figure 3.2: Predicted vs computed derivative (left) and the actual data vs simulated data (right) for a dataset with Laplacian distributed noise of various degrees.

seems to be robust against noise and should be able to determine the underlying governing equations in the boiler, even when noise is present. An example of the SINDy model's fit to the Laplace distributed noise with different standard deviations can be seen in figure 3.2. The figure shows the predicted and computed derivatives of the dataset, as well as the integrated function and the actual dataset.

It should be noted that the equations, used to build the artificial dataset, could be slightly different for the case study in this project, however the soot blowing curves obtained with these equations and the soot blowing curves one can extract from the real dataset looks very similar with the exception being the duration of the curves. This means that this verification problem is a fairly good representation of the type of physics the SINDy algorithm would have to extract, however more representative models will have to be tested that have comparable sampling rates to determine whether it is important for the algorithms extraction capabilities.

3.3 Establishing basic model forms from the Ngodwana dataset

To ensure that the verification problems going forward are more representative of the true models that are present in the boiler, the SINDy algorithm will be applied to the Ngodwana dataset in a broad sense to develop an idea of the model forms one can expect. Understanding what the models look like, will allow one to develop verification models that can be interpreted. Hence one can understand what can be expected when the SINDy algorithm is applied to the real dataset. The SINDy algorithm is therefore applied to the measurement dataset from the plant, not to extract fully developed models, but to simply obtain basic model forms that can be used to build representative artificial datasets.

3.3.1 Extracting the time series sequences for modeling

As mentioned before, the SINDy models try to determine the physics equations from the measurement data. When a soot blower activates or deactivates, the underlying physics change, because of the change in the inputs to the system. The different soot blowers might also affect the physics in different ways, as they are located in different sections of the boiler and have been shown to have different influences on the fouling factor. Therefore, it is decided that the models will be built from small sections of the dataset, where the physics should remain relatively constant.

Specific soot blowing sequences are thus extracted from the data set for specific soot blower pairs. The sequences are extracted for pairs, as the soot blower pair always enters the boiler at approximately the same time and works in unison. Thus, the measurement samples are extracted when a specific soot blower pair is active or, just after a specific soot blower pair has deactivated. Doing this should allow the SINDy algorithm to find the underlying governing equations more readily, because the physics does not change during the section of data given to the model. The extraction of these sequences also allows one to determine how the specific underlying physics change over time for each soot blower in a section, because of the deposit buildup in the boiler. To attempt to isolate the soot blower pairs as much as possible, a soot blower pair sequence is only selected if no other soot blowers in the specific boiler section has been active for at least 3 minutes and no other pairs activate after this pair for at least 2 minutes. This should ensure that as many external influences as possible are removed and the physics remains relatively constant.

Once the active soot blower sequences were extracted, it was seen that some of the sequences were extremely short and only contained 2 or 3 samples, while other sequences were too long containing over 100 samples, which relates to a period of 16 minutes. These outliers could be due to sensors malfunctioning and not picking up that a soot blower is no longer active or erroneously picking up that a soot blower is active. It was therefore necessary to filter the sequences again, to only extract sequences that were of the correct length. The mean lengths of the soot blowing sequences for each soot blower were determined, as well as the 25th, 50th and 75th percentiles. These values would ensure that the sequences are filtered in such a way, to best represent the nominal sequence length for each soot blower when a standard outlier filter is run. The statistical lengths of the sequences, for each of the soot blowers in the different boiler sections, are summarised in table 3.2. Once the sequences are filtered, the basic SINDy modeling can begin.

3.3.2 Building basic predictive models

In the verification problem, the only input to the differential equation was the fouling factor itself. Therefore, it is decided to start the SINDy modeling, using only the fouling factor as input. Additional inputs to the system can then be investigated once the basic verification models are built and tested.

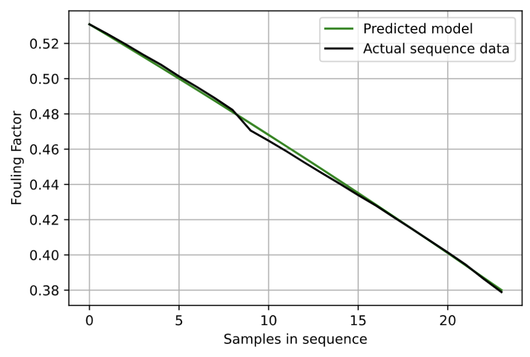
To build the initial model, the sequentially thresholded least squares regression (STLSQ) optimizer is used and the threshold for the weights is set to 0.0001. This threshold is small, as it was seen that using a higher threshold resulted in the optimizer eliminating almost all the function coefficients and trivial models are found as a result. Currently the objective is to obtain only basic model forms and the parameter values are not adjusted precisely, therefore a default value for the threshold should be sufficient.

Table 3.2: Statistical lengths of soot blowing sequences for all soot blowers in the different sections of the boiler.

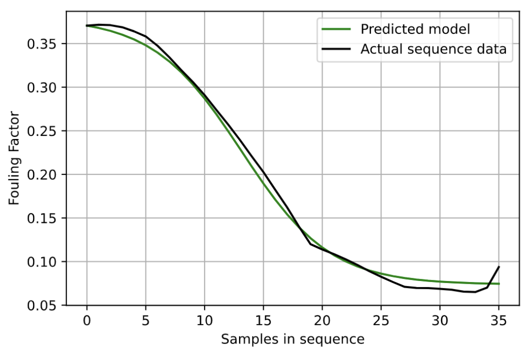
Boiler section	**Soot blower pair**	**Mean length**	**STD of length**	**25th percentile**	**50th percentile**	**75th percentile**
Primary SH 1	SB 1 and 2	26.571	3.770	24	26	30
	SB 3 and 4	29.743	6.108	24	31	35
	SB 11 and 12	25.087	3.884	22	24	28
	SB 13 and 14	21.478	3.214	19	21	24
	SB 21 and 22	26.591	3.279	24	25	30
	SB 23 and 24	21.180	5.115	18	20	24
	SB 31 and 32	27.568	25.760	22	25	28
	SB 33 and 34	28.114	5.801	25	27	31
	SB 39 and 40	26.567	4.682	24	26	30
Primary SH 2	SB 7 and 8	20.786	3.473	18	20	24
	SB 9 and 10	27.627	5.0137	25	26	31
	SB 17 and 18	20.526	3.643	18	20	24
	SB 19 and 20	27.034	3.850	24	26	30
	SB 27 and 28	21.125	3.320	18	20	24
	SB 29 and 30	27.181	5.781	24	26	30
	SB 37 and 38	20.847	3.566	18	20	24
	SB 43 and 44	27.096	3.639	24	26	30
Secondary SH	SB 3 and 4	29.743	6.108	24	31	35
	SB 5 and 6	21.134	3.370	18	20	24
	SB 13 and 14	21.478	3.214	19	21	24
	SB 15 and 16	21.171	3.627	18	20	24
	SB 23 and 24	21.180	5.115	18	20	24
	SB 25 and 26	20.909	3.867	18	20	24
	SB 33 and 34	28.114	5.801	25	27	31
	SB 35 and 36	27.716	4.11	25	26	31
	SB 39 and 40	26.567	4.682	24	26	30
	SB 41 and 42	20.974	4.520	18	20	24

The threshold value is investigated in-depth once the verification problems are complete. The differentiation method was chosen as the standard finite difference method. Both the STLSQ optimizer and the finite difference method is recommended in the paper by Brunton et al. (2016) The PySINDy implementation in Python has a range of pre-defined function libraries to choose from, to form the feature library as described in section 3.1 (de Silva et al. 2020). Since most of the non-linearities in the thermodynamic model were logarithmic and polynomial, the differential equations should also be polynomial. The verification problem, which was adapted from the paper by Shi et al. (2019), also showed that the derivative of the soot blowing sequence function seems to be polynomial. Thus, the polynomial feature library was chosen from the PySINDy libraries and the default second-order library was used for initial modelling. From the thermodynamic model validation, it was seen that the most effective soot blower pair was soot blowers 3 and 4. It was decided that initial modeling will be started on this pair of soot blowers, which were located in the secondary superheater. The same principles can be applied to the other soot blowers in the respective boiler sections, using the same methodology, if required.

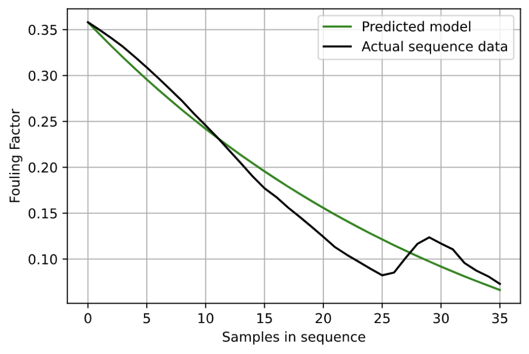
The fouling factor sequences of soot blowers 3 and 4 are given to the SINDy algorithm and the underlying equations are extracted per sequence. A few examples of the models found by this SINDy model can be seen in figure 3.3. The soot blowing curves do not all have the same shape or curve which may make it difficult for the SINDy algorithm to extract consistent models.



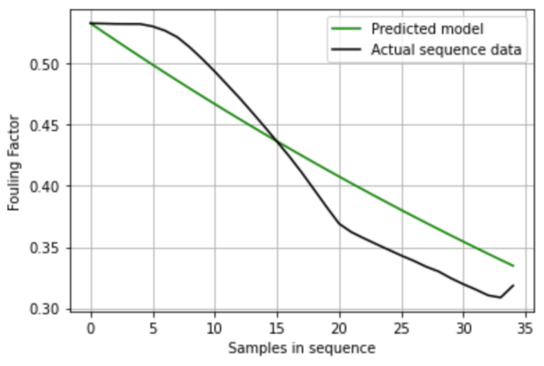
(a) Example 1



(b) Example 2



(c) Example 3



(d) Example 4

Figure 3.3: Examples of simulations of the models found by the SINDy algorithm and the data used in the model extraction.

While the model that is built is very simple (in the sense that it only has one input), the equations found by the SINDy algorithm, generally fit the data very well and capture the underlying trends in the sequences. As is expected, the model found is a polynomial of order 2 (from the second-order feature library) and generally has three coefficients. The basic form of the differential equations found by the algorithm can be seen in equation 3.12.

$$FF' = C1 + C2 \times FF + C3 \times FF^2 \tag{3.12}$$

where $C1, C2$ and $C3$ refer to the coefficients found through sparse regression. It was noted that in some sequence cases certain coefficients would be set to zero, while for other sequences all coefficients would be present. It was rarely seen that C2 was set to be zero, however the second-order coefficient was often driven to zero by the algorithm.

It can be seen that the soot blowing sequences do not seem to have excessive amounts of measurement noise in them, especially when comparing the examples in figure 3.3 to those of the initial verification problem in figure 3.2 where different noise levels were shown. The sequences in figure 3.3 have a noise level that seems to lie below a standard deviation of 0.2 if the figures are directly compared. Therefore, the models that the SINDy algorithm extracts, should be valid models to use for verification problems.

It was noted that, while the underlying model equations remained very constant over the different soot blowing sequences, the coefficient values changed with every sequence and were not consistent. This might be because of several factors including, the noise in the system resulting in the soot blowing sequences not having a consistent shapes or due external factors in the boiler influencing the data. The model might not have enough defined inputs or the model order may not be well suited to the problem.

Before the cause of the coefficient fluctuations can be discovered it is necessary to build a few verification problems to better understand what is being seen as well as determine when the underlying models are difficult to identify. The first verification problem will be set up to investigate the influence of the coefficients on the sequence shape and to determine which sequence lengths are required to easily identify model coefficients.

3.4 Identifiability of underlying physics models and the SINDy algorithm's recovery ability

3.4.1 Model coefficient influences for different polynomial order models and their identifiability

To determine the cause of the coefficient fluctuations for the different sequences, it is important to understand what each coefficient's effect is on the shape of the fouling factor curve for different polynomial order models. Hence, it can be determined whether the changes in the coefficients are due to shape changes in the soot blowing curves, caused by varying conditions in the boiler, or whether it is due to noise in the measurement data and/or other random occurrences. Three verification models are set up that have three different polynomial orders. The polynomial order will range from 2 to 0, meaning the first model is a second-order polynomial, similar to what was seen in the initial experiment on the Ngodwana data, the second model is a first-order polynomial and the last model will have only a constant term.

The second-order model

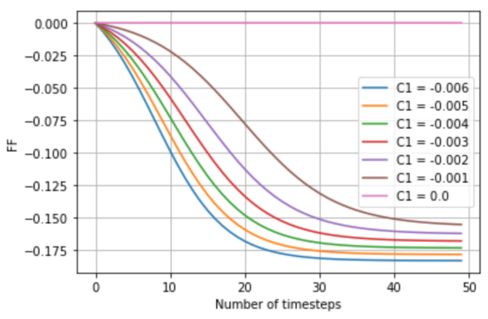
The second-order base model is chosen based on the average of the first 100 sequences' coefficients in the initial Ngodwana experiment of the previous section, as it was seen a representative soot blowing sequence was found using this method. For simplicity, the initial value is set to 0. To determine the coefficient influences, each coefficient is varied, while the other two are kept constant. This way, it can be determined what influence each coefficient has on the overall fouling factor curve. The equation of the second-order polynomial ODE chosen can be seen in equation 3.13.

$$FF' = -0.003 + 0.15FF + FF^2 \tag{3.13}$$

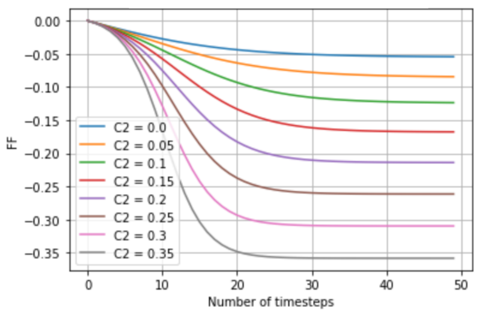
The first coefficient is varied between -0.006 and 0 and its influence can be seen in figure 3.4a. The second coefficient is then varied between 0 and 0.35, while the others remain constant. Its influence can be seen in figure 3.4b. Finally, the third coefficient is varied between 0.0 and 1.6 and its influence is shown in figure 3.4c. These values are chosen as they give one a good idea of how the coefficients influence the curve and how the identifiability of the models changes for the different coefficient values.

Upon closer inspection of these figures, it can be seen that the first coefficient has a large influence on the rate of change in the graphs. The more negative the value becomes, the faster the graph drops to reach an imaginary 'asymptote'. The second coefficient has a very large influence on the location of the asymptote in the graph. The location of the asymptote and the negative value of coefficient 2 are almost directly correlated. Finally, the third coefficient also has a strong influence on the location of the asymptote. The value of coefficient 3 and the value at which the asymptote is located, is indirectly correlated. This means that if coefficient 3 doubles, the value at which the asymptote is situated halves and vice versa.
It is therefore clear that the coefficient values are highly dependent on the shape of the fouling factor curves, as well as the end location of the curve, as it was seen that some of the coefficient values are highly correlated to the 'asymptotic' line that the curves approach. Furthermore, the respective coefficients seem to be correlated in such a way that different combinations of the same coefficient could result in the same answer, which could lead to possible ill-posedness (more than one combination gives one the same curve) in the problem.

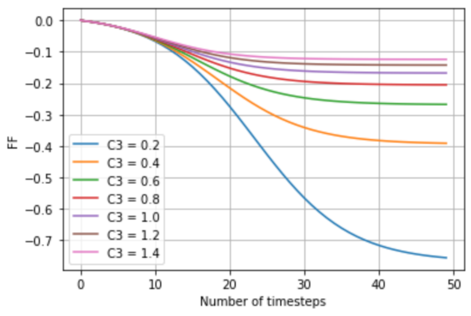
When investigating the figures from an identifiability point of view, it can be seen that the soot blowing curves become more difficult to discern from one another when coefficient C1 becomes more negative and when coefficient C3 becomes a larger positive



(a) Varying C1



(b) Varying C2



(c) Varying C3

Figure 3.4: Influence of the model coefficients on the shape of the fouling factor curve for a second-order polynomial model.

value. Coefficient C2 seems to have the least influence on the curve identifiability as the spacing between the curves remains fairly consistent when its values are varied. If coefficients C1 and C3 lie within a range that makes the curves less separable, it could potentially cause difficulties for the SINDy algorithm, as multiple coefficient values would result in the same basic curve shape and position. This could potentially result in an ill-posed problem, which could cause fluctuations in the model coefficients and prevent a consistent model from being extracted by the algorithm. When applying the SINDy algorithm to the real dataset, one should therefore be aware that this could be a potential problem that causes coefficient fluctuation and would be difficult to identify.

Finally one can also determine what the minimum sequence lengths need to be for the SINDy algorithm to efficiently extract distinctive models from the data. If the number of samples in the sequence is less than 10 time steps, the SINDY algorithm will likely struggle to extract distinctive models as the curves are very close to each other in the beginning. This was confirmed with the SINDy algorithm, as it was seen that the different sequences with only 10 samples obtained almost identical equations and were not distinguishable. Once more than 25 samples are available, the SINDy algorithm should be able to extract more distinct models as the curves can more easily be differentiated from one another. Since most of the soot blowing sequences in the Ngodwana dataset contain more than 25 samples, this should not be a problem in this particular study.

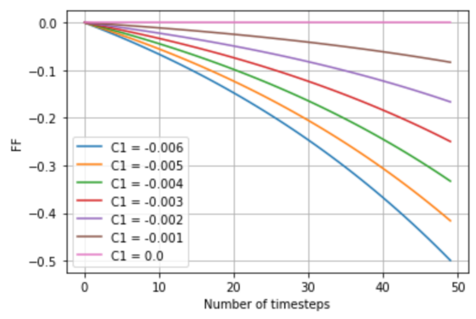
The first-order model

To better understand the influences of the coefficient and to determine a suitable polynomial order for the SINDy algorithm, the first-order polynomial model is also tested and the coefficient influences are noted. The first-order model looks similar to the second-order polynomial model, with the exception being the second-order term is dropped and the first-order term coefficient is altered to ensure the curve is still representative of a fouling factor development curve. Two different base equations will be used as it was seen that changing the sign of the second coefficient altered the shape of the curve dramatically. The first base equation used can be seen in equation 3.14 while the other can be seen in equation 3.15

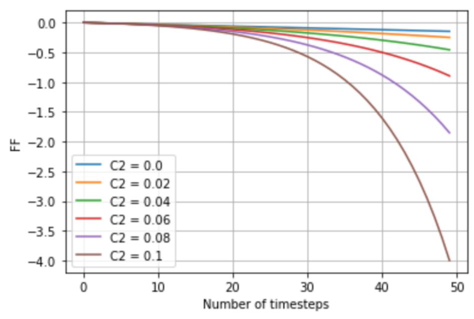
$$FF' = -0.003 + 0.02FF \tag{3.14}$$

$$FF' = -0.003 - 0.02FF \tag{3.15}$$

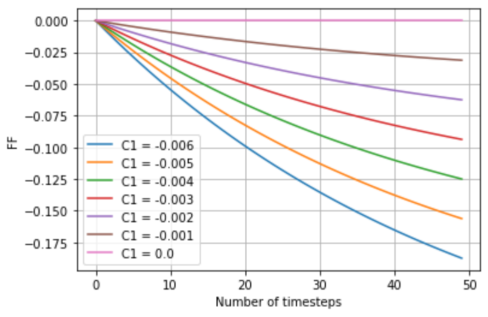
Similar to the second-order model, the constant coefficient is varied between -0.006 and 0 while the second coefficient in equations 3.14 and 3.15 is varied between 0 and 0.12.
In figure 3.5, the fouling factor curves are better spaced when the coefficients are being varied. Similar to the second-order polynomial model, very short sample sequences over the same range will be difficult to distinguish as the curves are very close to each other. However once the sequence lengths are more than 20 samples, the curves become more distinct for each coefficient value. The curves do not seem to converge as much when the C1 coefficients go closer to 0 as was the case with the second-order polynomial models. It was noted during the experiments that coefficient C2 was very influential regarding the curve shape. Very small changes in the coefficient value altered the curve drastically. This indicates that the SINDy algorithm should not have a problem with extracting the correct coefficient for a specific curve as the respective coefficient values have very distinct curves. As coefficient C2 goes closer to 0, the curves once again become difficult to distinguish from each other. However, as mentioned before it was noted that this coefficient is very sensitive and therefore, the SINDy algorithm should still be able to



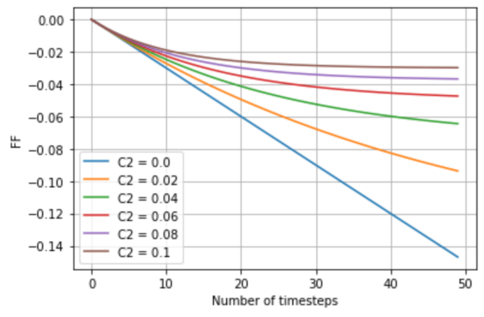
(a) Varying C1 in equation 3.14



(b) Varying C2 in equation 3.14



(c) Varying C1 in equation 3.15



(d) Varying C2 in equation 3.15

Figure 3.5: Influence of the model coefficients on the shape of the fouling factor curve for two respective first-order polynomial models.

extract consistent models, should the fouling factor curve shapes be consistent. So far this polynomial order model seems to result in the most distinguishable models and has enough complexity to fit the typical soot blowing curves seen in the Ngodwana dataset.

The constant ODE model

The constant model is a model that simply contains only a constant in the differential equation and does not take any fouling factor inputs into account. The model that is used in this verification problem, can be seen in equation 3.16 and the base value of the constant is from the previous two polynomial model's constant values.

$$FF' = -0.003 \tag{3.16}$$

Once again, the constant term is varied between -0.006 and 0 similar to the other experiments. Figure 3.6 shows the change in the fouling factor curve when the constant value is altered. The different fouling curves are very distinguishable in this case. This makes sense as the underlying model is elementary and the angle of the curve is only dependent on a single coefficient. The SINDy algorithm should be able to extract these types of models easily due to them being very distinguishable, however the likelihood that this is the true underlying physics model in the boiler is low. This model is very elementary and will not fit the more complex fouling factor curves of the Ngodwana dataset well, however it may be a fair approximation of a fouling factor curve, if no other options are available.

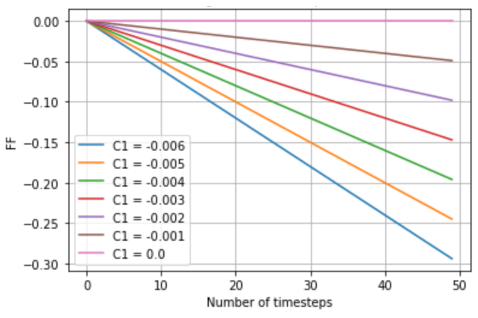


Figure 3.6: Influence of the model coefficients on the shape of the fouling factor curve for two respective first-order polynomial models.

The following section contains verification problems that test the SINDY algorithm's ability to recover representative fouling factor curves when different polynomial order models are used. SINDy's recovery ability will also be tested under different noise levels and initial conditions. This will allow one to further determine what the optimal polynomial order would be for the Ngodwana dataset and to understand which problems may be encountered when an actual dataset is used.

3.4.2 Recovery ability of the SINDy algorithm

This section's verification problems test the SINDy algorithm's ability to recover consistent models under different conditions and test whether over-fitting is a problem that should be addressed when using the algorithm on a real dataset. To test the model recovery consistency, an artificial soot blowing model will be chosen for each polynomial model type investigated in the previous section. Each of these models will be representative of a typical soot blowing sequence, as was seen when the basic models were extracted from the Ngodwana dataset. Therefore, the coefficients are chosen through trial and error to represent soot blowing sequences often seen in the dataset. It was noted in the basic model forms that the algorithm often made the second-order term zero, however, there were very few times that the algorithm set the first-order term to zero and kept the second-order one. Therefore, this model form is not considered. The artificial model equations can be seen in equation 3.17 in descending polynomial order. The typical shapes of the soot blowing sequences can also be seen in figure 3.7.

$$\begin{aligned} FF'_{2nd-order} &= 0.350 - 1.613FF + 1.806FF^2 \\ FF'_{1st-order} &= 0.013 - 0.036FF \\ FF'_{0-order} &= -0.0051 \end{aligned} \tag{3.17}$$

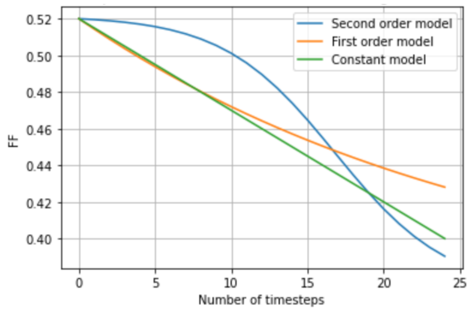


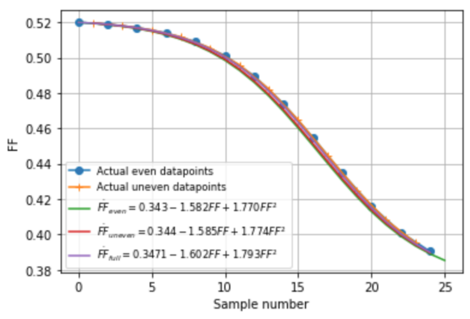
Figure 3.7: Representative soot blowing models of different polynomial orders.

General Recovery ability

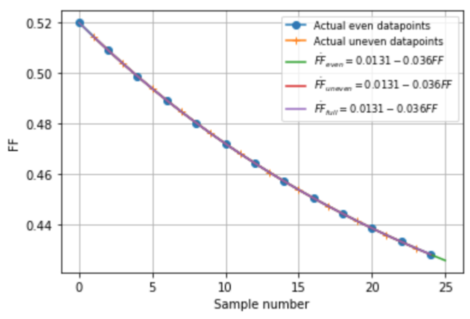
The first experiment will test the SINDy algorithm's recovery ability when the same soot blowing sequence is used but only the even or odd samples in the sequences are given to the algorithm. Should the SINDy algorithm be able to extract the same or similar model for the different samples from the same sequence, one can safely assume that the algorithm is not over-fitting the data given to it. Furthermore it will prove that the problem is not inherently ill-posed, as different samples with the same underlying model result in the same model being extracted. If the coefficients are vastly different depending on the samples used, it could indicate that over-fitting is a potential problem or that the problem is inherently ill-posed and multiple models can give one the same answer.

The SINDy algorithm is set up the same way as it was set up when the basic models were extracted from the Ngodwana dataset. This means the STLSQ optimizer is used with a threshold of 0.0001, the finite difference method is used for differentiation and the second-order polynomial feature library is chosen. The results of this test can be seen in figure 3.8 for the different polynomial models. Figure 3.8 also shows the equations of the models that were extracted by the SINDy algorithm for the respective sequence samples for easy comparison.

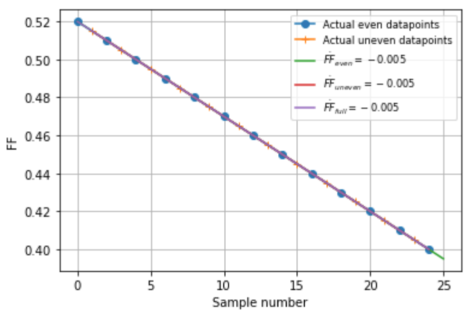
The SINDy model can extract consistent models despite the sample types changing. The second-order polynomial sequence has very slight variations in the extracted model coefficients when looking at the legend in figure 3.8a. However, these variations are negligibly small and are most likely only due to slight variations in the samples. This



(a) Second-order soot blowing model



(b) First-order soot blowing model

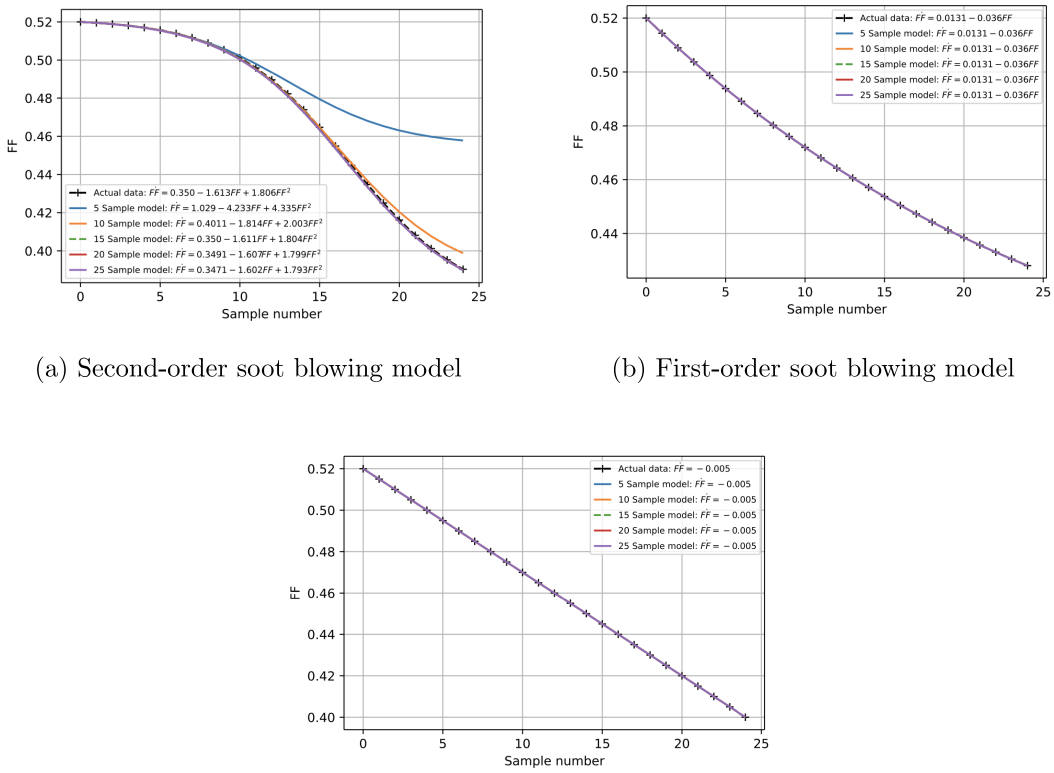


(c) Zero-order soot blowing model

Figure 3.8: SINDy algorithm recovered models when even and uneven numbered samples from the same sequence are used.

proves that the problem is not inherently ill-posed, as there exists only one solution for each type of soot blowing curve. Furthermore, it is clear that the SINDy algorithm does not seem to overfit the data, since the same model is extracted for the the even and uneven data samples. The first and zero-order models are the same regardless of the samples used. This indicates that these models are once again a good option to use when applying the SINDy algorithm to the real dataset, since many of the soot blowing sequences look very similar to the first-order polynomial model and the sample selection do not influence the algorithm's recovery ability. The results obtained from this model are representative of the results one can expect from models extracted from the real measurement dataset, should the same noise conditions or data quality be available.

To further test the algorithm extraction consistency, the number of data samples sent to the SINDy algorithm is varied. This is done, to see whether the model coefficients vary significantly if fewer data samples are used, as it was seen in the coefficient influence experiment that shorter sequences may be difficult to identify. An experiment is set up where the first 5, 10, 15, 20,and full 25 samples are given to the SINDy algorithm. The models that the algorithm extracts are then compared to determine how many samples are needed before the true underlying model is extracted and to determine when the extracted models become consistent. This allows one to determine how many training input samples one typically requires, for use in the Ngodwana dataset. The SINDy algorithm is set up the same way as the previous experiment and the artificial models, that are used to generate data, are the same models described in equation 3.17. Figure 3.9 shows the experimental results for different numbers of training data samples for the different polynomial models. Once again, the equations of the extracted models are indicated in each figure's legend for comparison purposes.



(a) Second-order soot blowing model (b) First-order soot blowing model

(c) Zero-order soot blowing model

Figure 3.9: SINDy algorithm recovered models when the number of training samples is varied.

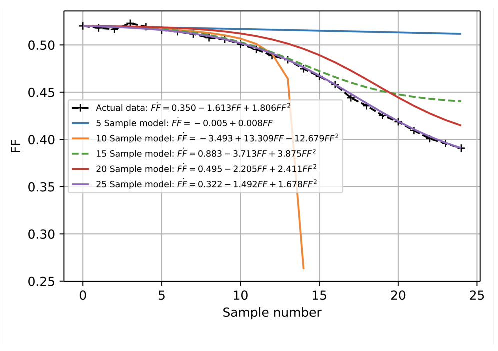
In figure 3.9 it can be seen that the SINDy algorithm extracts consistent models when the number of training samples is more than 10 for the second-order model and seems to extract consistent models for as little as five samples when the polynomial order is less than 2. This proves that the typical sample length of 25 samples, in the Ngodwana dataset, is more than sufficient for the extraction of the underlying physics models in conditions where little to no noise is present. Furthermore, the SINDy algorithm is capable of extracting not only consistent, but also the correct models fairly easily, even when data is sparse. This is especially true when the underlying physics models are first-order polynomials or lower. This makes sense, as these order models are fairly simple and would therefore not require as many training samples to discover.

So far no noise has been added to these polynomial curves. Therefore, the next experiment will test the algorithms recovery ability, when noise is once again added to more representative artificial sequences than what was used in the very first verification problem. This will allow one to truly see whether the SINDy algorithm can recover consistent models when a realistic measurement dataset is given to it.

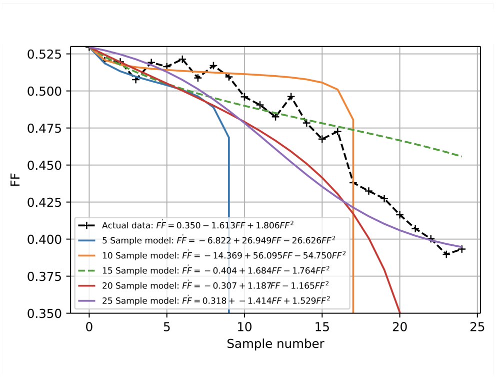
Recovery ability under different noise conditions

To test the SINDy algorithm's ability to recover the underlying physics from noisy measurement data, the same artificial physics models will be used as were tested in the general recovery section. The artificial dataset will only be altered slightly by adding Laplace distributed noise to each 'measurement' sample. Laplacian noise is chosen, as this is often assumed in other experiments involving measurement data. Pure Gaussian distributed noise is relatively rare in measurement datasets and Laplacian or t-distributed noise is often more common. It was also seen in the previous noise experiment that the SINDy algorithm seems capable of dealing with multiple types of noise equally. If it can deal with one type it should be able to deal with other types of noise as well. The noise standard deviations used in this experiment are 0.001, 0.005,0.01 and 0.05 as it was seen that the true dataset sequences have noise that seems to fall within this range and higher noise deviations would not be representative of the real dataset.

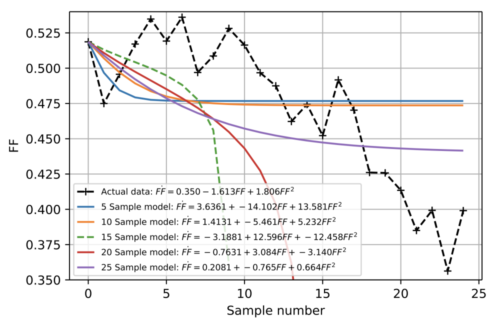
The artificial dataset with added measurement noise is once again sent to the SINDy algorithm in different training sample lengths, to determine whether noise impacts the algorithm's ability to extract the physics from shorter training sequences. The SINDy algorithm setup is not changed for this experiment. The results of the experiment can be seen in figures 3.10, 3.11 and 3.12 and show the recovered models for different polynomial order sequences, levels of noise and training sequence lengths. Once again, the recovered models' equations are shown in the legend of the plots.
In figures 3.10, 3.11 and 3.12, the addition of noise to the artificial dataset has impacted the algorithm's ability to recover the true underlying models significantly. As the polynomial order of the physics models increases the ability of the algorithm to recover the underlying model, when noise is added, deteriorates. This may be due to the added complexity of the models. The SINDy algorithm was able to recover the zero-order models more often at higher noise levels, than it could when the underlying models were



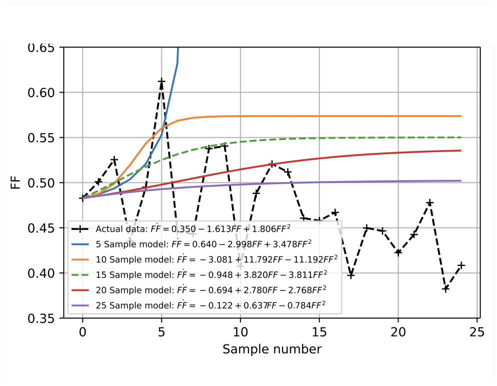
(a) Laplacian noise with a standard deviation of 0.001



(b) Laplacian noise with a standard deviation of 0.005



(c) Laplacian noise with a standard deviation of 0.01

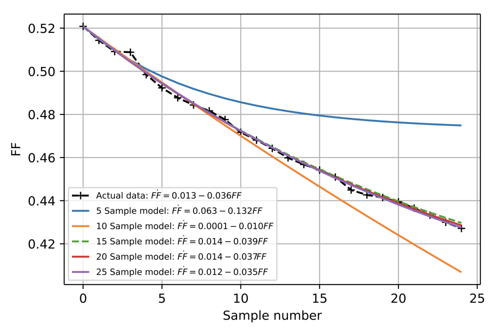


(d) Laplacian noise with a standard deviation of 0.05

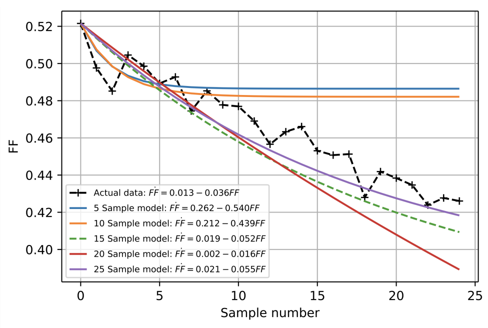
Figure 3.10: SINDy algorithm's recovery ability under different noise levels and varying training sequence lengths for a second-order physics model.

higher-order polynomials. This makes sense, since the more complex the shape of the curves become, the more possible equations there are, when noise obscures the true form of the curves.

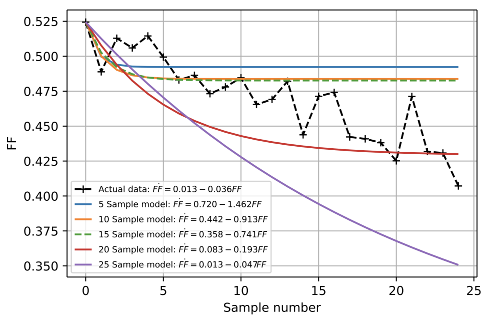
In most cases the minimum number of samples needed to recover equations that were close to the true models, was 20 samples. When noise levels became too high, even 25 samples were not enough to recover the true model. When visually comparing the artificial dataset sequences with those of the Ngodwana dataset earlier in the Chapter, it can be noted that the Ngodwana dataset seems to contain noise with a standard deviation that is lower than 0.005. Therefore, it should be possible for the SINDy algorithm to find the true underlying models if the complexity of the models is not too high. It was also noted that the higher-order models were very susceptible to divergence when an ODE integrator was applied to them. This was especially seen with the second-order polynomial models, which makes sense, as a sudden deviation in the model input could lead to the squared fouling factor term diverging. This shows that a more reliable model to potentially fit to the Ngodwana dataset would be a first or zero-order model.



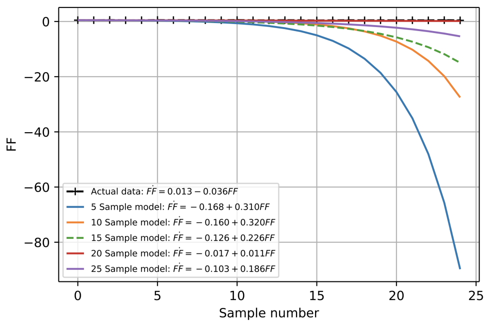
(a) Laplacian noise with a standard deviation of 0.001



(b) Laplacian noise with a standard deviation of 0.005



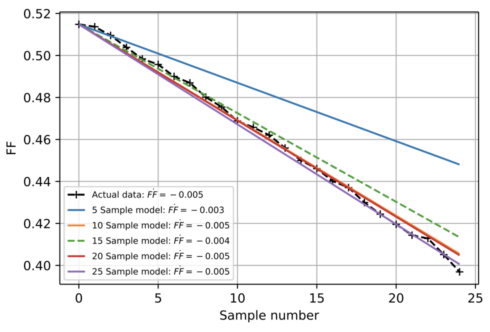
(c) Laplacian noise with a standard deviation of 0.01



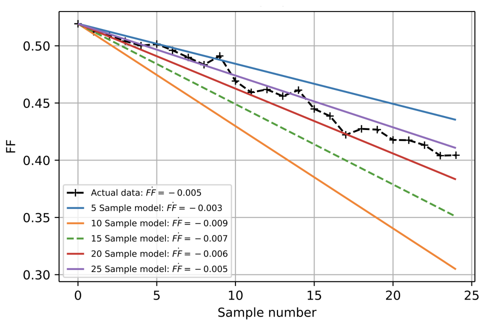
(d) Laplacian noise with a standard deviation of 0.05

Figure 3.11: SINDy algorithm's recovery ability under different noise levels and varying training sequence lengths for a first-order physics model.

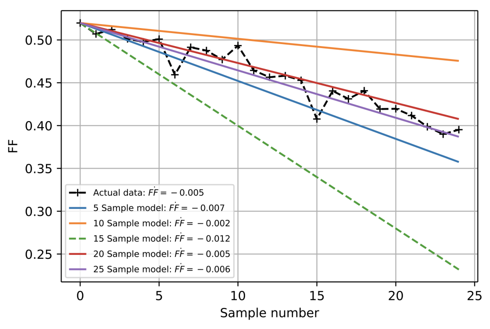
The SINDy package implementation on Python has another built-in differentiator, namely the smooth finite difference method. The difference between this differentiator and the basic finite difference method is the fact that the input dataset is filtered and 'smoothed' before being differentiated. The filter is designed to reduce the effect of noise when differentiating. The finite difference method is known to deliver extremely noisy derivative values when a noisy dataset is sent through the algorithm. The smoothed finite difference method attempts to counter this. This differentiator was tested on the second and first-order polynomials for noise levels with a standard deviation of 0.005. The results can be seen in figure 3.13. The results are not much improved from the basic finite difference method. The second-order model fits the actual data better, however it has an equation that is further from the true underlying model. The extracted second-order model also has a very unexpected shape and seems to jump to erroneous values in some cases. It almost seems as if the function over-fits the data. The same result is obtained for the first-order polynomial where the extracted model with



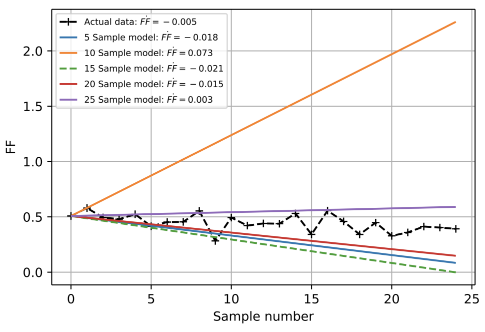
(a) Laplacian noise with a standard deviation of 0.001



(b) Laplacian noise with a standard deviation of 0.005



(c) Laplacian noise with a standard deviation of 0.01

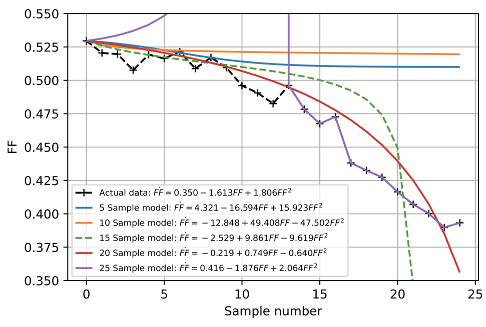


(d) Laplacian noise with a standard deviation of 0.05

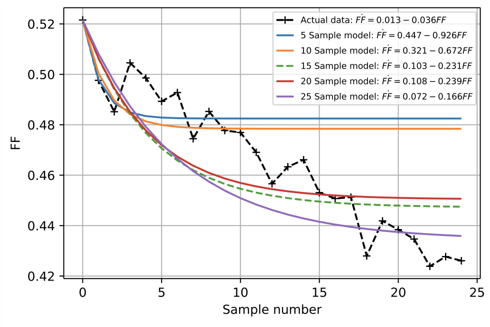
Figure 3.12: SINDy algorithm's recovery ability under different noise levels and varying training sequence lengths for a zero-order physics model.

the smoothed finite difference method fits the noisy data better, however the equation of this model is further from the ground truth. It therefore, seems that the smooth finite difference method may aid in fitting the given data better, however it hinders the discovery of the true underlying model in some cases and will thus not be used for this project but may be valuable for future research projects.

The addition of noise to the artificial datasets has proven that noise can influence the recovered model coefficients severely, and if there is a lot of noise present in the Ngodwana dataset, one can expect fluctuations in the recovered models' coefficients. During the experiments so far, it was seen that fluctuations in the initial conditions could potentially impact the coefficients of the recovered models. Therefore, another recovery ability experiment is set up, to test the impact that changes in the initial sequence fouling 'level' could have on the model coefficients.



(a) Second-order model



(b) First-order model

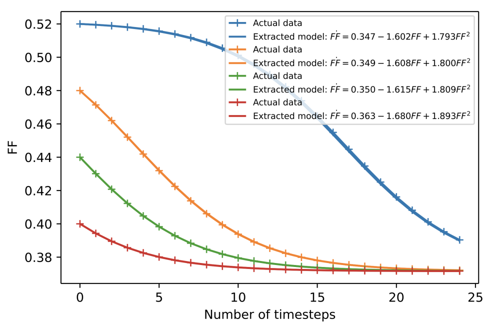
Figure 3.13: Testing the smooth finite difference method on noisy data.

SINDy's recovery ability when initial conditions are varied

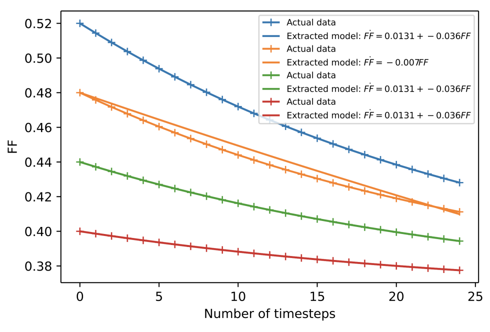
To test the influence that different initial conditions have on the recovered models, the same polynomial models are used as before. For each polynomial model type four different initial conditions are selected and the artificial sequences are generated. The initial conditions are chosen arbitrarily as the following: 0.52, 0.48, 0.44, 0.40. No noise is added to these sequences to purely test the influence of changing the initial conditions. The SINDy algorithm setup is not changed and the recovered models are noted. The results for each polynomial model can be seen in figure 3.14.

Different initial values have a slight influence on the extracted models from the SINDy algorithm when investigating figure 3.14. This is more likely to be the case when the polynomial order of the underlying models is higher as is highlighted by the figure. The fluctuations in the model coefficients, however, are relatively small, and do not cause the extracted models to be far off target. It may cause more fluctuation, however, when a real dataset is used that contains noise. If noise is present in the system and there are changes in the initial fouling values, one may start seeing significant fluctuations in the model coefficients as it was seen that noise can impact the SINDy algorithm's recovery ability significantly. If the influence of noise is added to the influence of initial value changes, one may not be able to extract identical models from one sequence to the next, even when the underlying model is the same and this should be kept in mind when applying the SINDy algorithm to the Ngodwana dataset.

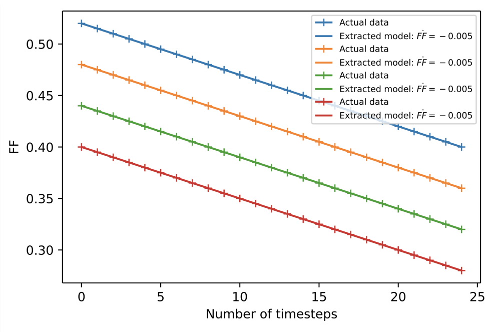
An additional test regarding initial values is also performed where the curve shapes of the sequences are kept identical but the initial values from which these curves originate are changed. This is done, as it was seen that there are many curves in the Ngodwana dataset that have near-identical shapes, but have different initial values. The same polynomial



(a) Second-order model



(b) First-order model

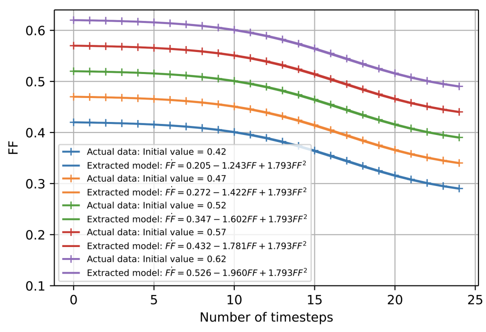


(c) Zero-order model

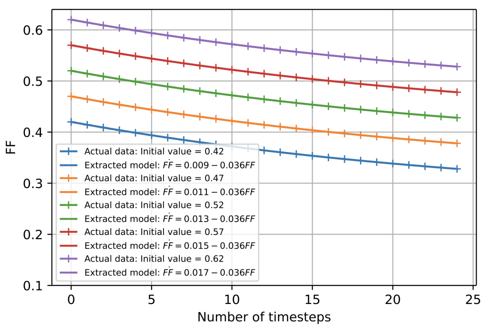
Figure 3.14: Influence of varying the initial conditions on the recovered models from the SINDy algorithm.

models are used in this experiment and a baseline curve is generated. Additional curves are generated by shifting the baseline curve up or down. The baseline curves used are the same curves that were seen in figure 3.7 for the respective polynomial order models. The results of this experiment can be seen in figure 3.15 for these respective polynomial order models. The equations of the extracted models can also be seen in figure 3.15.

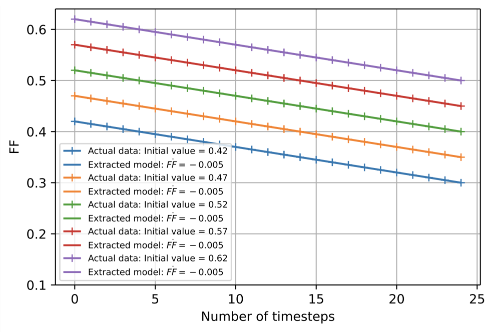
It can be seen in figure 3.15 that the equations extracted from the curves are not identical for the first and second-order polynomials, even though the curve shapes are. This is important to note, since it shows that the initial value of the fouling factor sequence is a critical value that influences the model coefficients. Furthermore, the zero-order polynomial model is always the same, which makes sense as it is a linear model and the gradient does not change. The first-order polynomial has a consistent coefficient for the fouling factor term while the constant term varies. The second-order polynomial on the other hand has a consistent coefficient for the squared fouling factor term, while the other two coefficients vary. It, therefore, seems that one could expect more consistency in lower-order models when curves are similar but do not start at the same point, whereas higher-order models will have fluctuating coefficients. One can, therefore, not expect completely consistent models in the Ngodwana dataset, when curves are visually similar but originate



(a) second-order model



(b) First-order model



(c) Zero-order model

Figure 3.15: Influence of varying the initial conditions on the recovered models from the SINDy algorithm.

from different values and is an important finding pertaining to the implementation of the algorithm on the Ngodwana dataset .

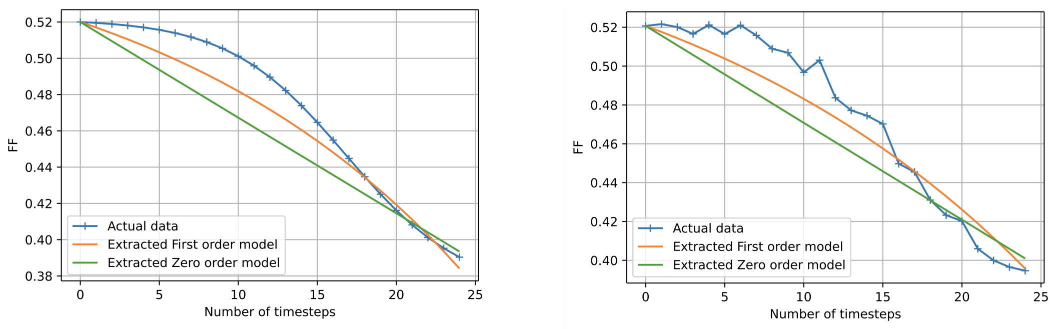
3.4.3 Fitting lower-order models into higher-order physics models

The next verification problem that is set up, is an experiment where a lower-order polynomial is fitted to an artificial dataset that was generated with a higher-order model. This is done, to determine whether the SINDy algorithm will be able to extract logical models, even though the exact polynomial order is not matched. When fitting real measurement data, there is a distinct possibility that one might try to fit lower-order models to data generated by higher-order physics models. For example, this may happen when one implements the SINDy algorithm on the Ngodwana dataset and lower-order polynomials are chosen for the feature library than is needed. This experiment will thus test, whether one would obtain models that are still useful to some extent, even though the exact physics model is not extracted.

Table 3.3: Extracted model equations and their RMSE's when no noise is added vs when noise is added.

	Extracted model equation	**RMSE of model prediction**
No noise	$\dot{FF} = -0.029 + 0.050FF$	0.011341
	$\dot{FF} = -0.005$	0.021015
With noise	$\dot{FF} = -0.025 + 0.042FF$	0.012223
	$\dot{FF} = -0.005$	0.019379

The experiment is set up as follows: The second-order polynomial model, used to generate the curve in figure 3.7 and the previous recovery experiments, is used to generate an artificial fouling factor sequence. A first-order and zero-order model is fitted to this sequence. Initially, no noise is added to the artificial sequence. Once the models have been fit, a small amount of noise (Laplacian noise with a standard deviation of 0.005) is added to the sequence and the models are fitted to the data again. One can then compare the effect of fitting lower-order models on clean and noisy data respectively, and evaluate whether plausible answers are obtained. The result of the experiment before noise is added can be seen in figure 3.16a, while the results when noise is added can be seen in figure 3.16b. Table 3.3 shows the equations of the extracted models as well as the root-mean-square-error of the respective models.



(a) Data without noise (b) Data with noise

Figure 3.16: Fitting lower-order models to data, with and without noise, generated by higher-order physics models.

Figure 3.16a shows that the SINDy algorithm extracts lower-order models that fit the higher-order curve relatively well if only considering the initial and final fouling factor values. While the exact shape of the curve is not matched, the endpoints of the extracted models seem to match up well with that of the higher-order curve. This is important, as the endpoints of a soot blowing sequence are a measure of how successful a soot blower pair was. Even though a lower-order model is used one would still be able to determine how efficient the soot blowing was and one could also roughly determine the energy saved because of soot blowing. This would only be true however, when the duration of soot blowing does not change drastically. The further the predictions are extended, the more inaccurate the lower-order models would become. In figure 3.16b it can be seen that the lower-order models still fit the artificial measurement data well, despite the noise being added. Once again, the endpoints of the soot blowing sequence and the model predictions match up and the general slope of the curve is captured by the lower-order models. This shows that one at least recovers some interpretability from the data as one could estimate the success of the soot blowing sequences should they have the same duration and very roughly determine the energy saved. It is however clearly not optimal as the rate of decay is not captured. When investigating table 3.3, one can see that the performance of the extracted models has not dramatically decreased once the noise was added, as the RMSE values are within the same range. Overall, the RMSEs are not extremely large, relative to the range of the fouling factor curve. This proves that fitting higher-order data with lower-order models is a feasible option, that will not result in illogical answers being obtained. While it is not optimal to fit lower-order models to higher-order physics, this experiment proves that the models one obtains should still be usable in a broad sense and can be interpreted to some degree.

3.4.4 SINDy's recovery ability for combined soot blowing sequences

The final verification problem is set up to test the SINDy algorithm's recovery ability when two soot blowing models are entangled. This is tested, as it could be a possibility that two soot blower pairs could activate in the boiler in approximately the same period, and have combined effects on the fouling level in the boiler. So far only pure, single soot blower curves have been tested on the SINDy algorithm and have been shown to work relatively well, except when substantial noise is present. Now a combination of soot blower models is tested to see what performance one can expect, from the SINDy algorithm, if the models are entangled.

To set up the verification problem, the first and second-order models, used in the previous verification problems are used as the two individual soot blowing models. The SINDy algorithm is applied to each model's artificial dataset individually, for the cases where a pure soot blowing sequence is present. The two artificial datasets are then combined by taking the average fouling factor value of the two curves for each sample, to entangle them to some extent. The SINDy algorithm is then applied to this dataset, and the recovered model is noted. It should be noted that all datasets will have added laplacian noise with a standard deviation of 0.005 to simulate more realistic 'measurement' data. The results of this experiment can be seen in figure 3.17 where the individual soot blowing sequences are shown, as well as the combined sequence with their corresponding extracted models.

The SINDy algorithm extracts physics models very close to the actual models

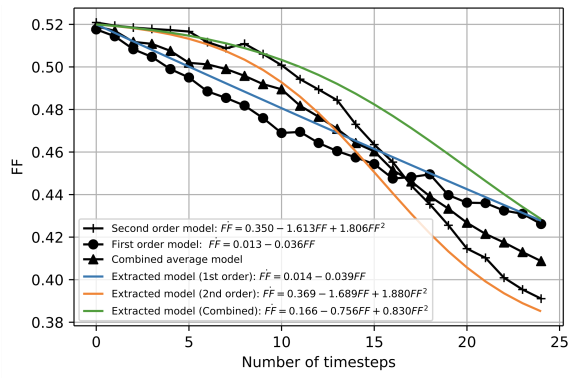


Figure 3.17: SINDy algorithm recovery ability when models are entangled.

when the pure soot blowing sequences are used, despite low levels of noise being present. This is expected, as it has already been shown that pure soot blowing models can be extracted fairly well with the SINDy algorithm. The combined model has a curve shape that is very similar to that of the second-order model and has the same initial value. Therefore, one would expect the coefficients of the extracted models to be similar, since the curves are very similar. When investigating the coefficients of the extracted models in figure 3.17, however, one can see that the combined curve has drastically different coefficients from that of the pure second-order model. The combined model that is extracted, does not have similar coefficients to the second-order model at all, despite the curve shapes being similar. This highlights a potential problem one may encounter when applying the SINDy algorithm to the actual dataset. If more than one soot blower influence is combined in the real dataset, one cannot expect the same model to be extracted as would be extracted if it were a pure soot blower influence, even though the curves may look similar. One would therefore have to be aware that, while curves may look visually similar, the slight variation between them will likely result in substantial model coefficient differences.

Furthermore, this experiment has shown that curves that look the same may not always have the same soot blower influences. One curve could be a pure soot blower influence while the other may be a combination of more than one soot blower's influence. It would therefore be very difficult to distinguish between them visually. This potential problem is minimised, by selecting soot blowing sequences that are as isolated as possible.

3.4.5 Discussion

Several verification problems and experiments have been completed to test the SINDy algorithm's capabilities and shortcomings. Initially, the algorithm was tested on a verification problem that was taken from a paper by Shi et al. (2019). It was seen that the SINDy algorithm can extract underlying physics models, similar to what one would expect to be present in the Ngodwana dataset, even when noise is present. Some basic model forms were extracted from the Ngodwana dataset to obtain representative soot blowing curves and models, on which the SINDy algorithm could be tested more thoroughly. It was noted that the Ngodwana dataset contained what seemed to be zero, first and second-order polynomial soot blowing curves, and a basic model of each type was selected for further testing.

The following experiment investigated the influence that the respective polynomial model coefficients had on the shape of the prediction curve. Furthermore, it was also noted that it may become difficult for the SINDy algorithm to distinguish between curves with certain coefficient ranges. It was noted, that curves containing fewer than 10 samples would likely be difficult to distinguish and hence identify with the SINDy algorithm. Most of the soot blowing sequences in the Ngodwana dataset, contains more than 25 samples however, and thus the curves should be recoverable since the sampling rate and the general sootblowing curve shapes remain the same in the real dataset. Furthermore, when some coefficients approached certain values, the curves would become almost indistinguishable. Should the Ngodwana dataset have curves that lie within such regions, one would likely see coefficient fluctuations as the physics is ill-posed in such a region and multiple models would result in approximately the same curve. This should be kept in mind when implementing the SINDy algorithm on the Ngodwana dataset especially when seeing coefficient fluctuations for similar curves, however, it would be a difficult problem to solve. It was further seen that lower-order polynomial models were more identifiable, even in the lower sample regions as the curves did not tend to converge as the coefficients approached certain values.

Once the coefficient influences on the shape of the soot blowing curves were determined, the SINDy algorithm's recovery ability was put to the test under different conditions. The first tests ensured that the SINDy algorithm was able to extract consistent models despite the samples from the same model not being the same to ensure that one would extract the correct models, irrespective of the type of samples used. Subsequently, the influence of the number of training samples given to the algorithm was tested. It was found that the algorithm could extract the correct model, even when as few as 10 samples were used for most cases, and that the lower-order polynomial models could be extracted with even fewer samples.

No noise had been added to the artificial problems up to this point and the influence of different noise levels were hence tested. High levels of noise influenced the recovery ability of the SINDy algorithm significantly, especially for the second-order models. It was noted, that high levels of noise drastically increased the number of samples one would need to obtain reliable models, and even if sufficient samples were available, the model coefficients would likely fluctuate. When visually comparing the Ngodwana soot blowing curves, in figure 3.3, to artificial noise corrupted curves in figures 3.10, it was seen that the Ngodwana data seemed to contain only small amounts of noise and should not negatively impact the SINDy algorithm's recovery ability.

The next verification experiment, tested the influence of the initial values of the soot blowing curves on the recovered models. It was seen that curves with the same shape but different initial values resulted in different models being recovered. This should be kept in mind for the Ngodwana dataset, as there are many soot blowing curves with the same shape but with different initial values. Further,it was tested what the result would be if lower-order models were fitted to data that is generated with a higher-order physics model. It was seen that, while the results were not optimal, the resulting models could still be interpreted and one obtained plausible answers. This is good news should one accidentally fit lower-order models than necessary on the Ngodwana data. Finally, the algorithms recovery ability was tested, when more than one soot blower model influences a soot blowing curve. It was seen that two models, that were active simultaneously, would result in a completely different model being extracted. This should be kept in mind in the real dataset if curves, that look visually similar, obtain completely different model coefficients.

Now that the SINDy algorithms strengths and shortcomings have been evaluated, it is necessary to implement the algorithm on the real Ngodwana dataset. From the experiments above the following results can be expected: There will most likely be significant model coefficient fluctuations, as there are many different curve shapes as well as curves with similar shapes but different initial values. Noise could affect the performance of the algorithm, however most sequences contain 25 samples and more and visually, the noise levels seem to be low enough to not severely impact performance. Therefore, one can expect to extract realistic model equations from the dataset. Even when lower-order models are fitted to higher-order data, one should still be able to use the resulting models for basic fouling factor development predictions. The next chapter will test these theories, as well as test additional inputs and methods to try and obtain model consistency, for a limited time interval in the boiler, for a specific soot blower pair.

CHAPTER 4

SINDy algorithm implementation on DCS data

This chapter investigates the performance of the SINDy algorithm when it is implemented on the measurement data from the Ngodwana mill. The results found in this chapter are also interpreted with regards to the work done in chapter 3 to determine whether the algorithm performs as expected. It was seen in chapter 3, that significant changes in the data due to noise, or changes to the soot blowing curve shapes and initial values, could potentially cause model coefficient fluctuation. Several methods are tested in this chapter to try and circumvent these issues and to try and extract more consistent model coefficients. These methods include adding inputs to the SINDy algorithm, optimising the algorithm's threshold parameter sequentially and finally, scaling and normalising the data itself. In a final experiment, several soot blowing sequences are manually selected for the SINDy algorithm, to illustrate the complexities of the dataset and to understand how model consistency can be achieved in future research.

4.1 The Case Study

As has been discussed previously, the focus of this study is on the second recovery boiler at the Ngodwana plant. The boiler is used to generate steam at high pressure that is sent to turbines in the plant that generates electricity for the plant processes. The boiler fuel used is black liquor, that is injected using 8 nozzles. The nozzles are designed to particularize the fuel into droplets for optimal combustion and to allow for more effective control of the combustion process. The nozzles are situated in the lower part of the boiler. The net calorific value of the black liquor is approximately constant at 8.96 MJ/kg although this value can change depending on the constituents of the fuel. The boiler has 4 air intakes to ensure optimal combustion and to control the flame height in

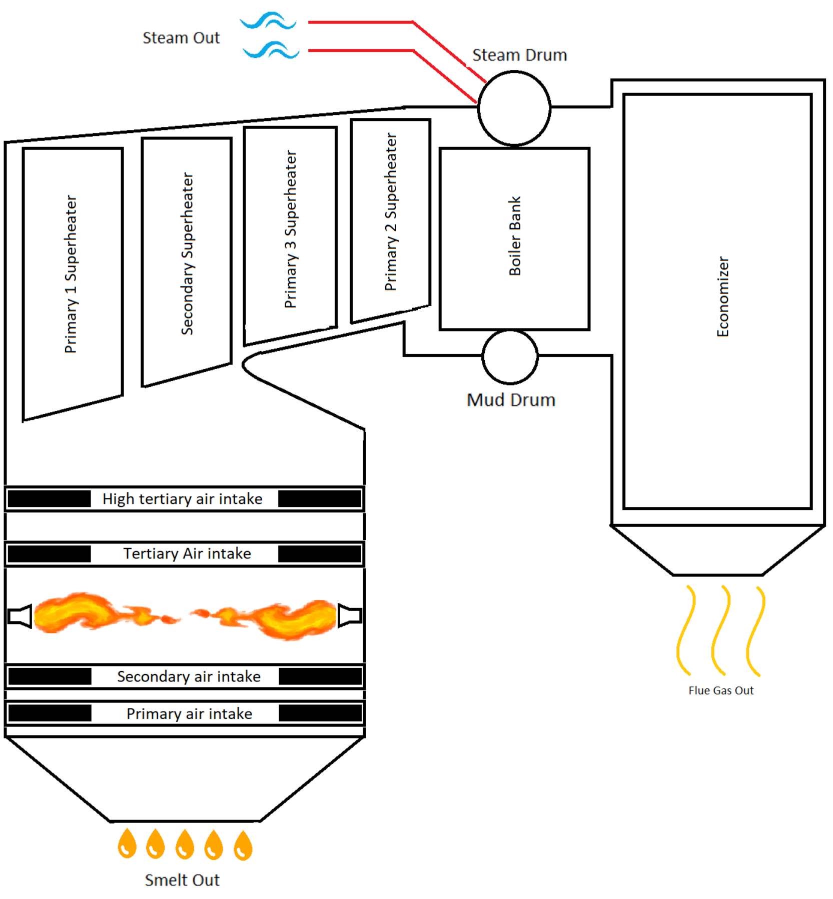


Figure 4.1: Schematic of SAPPI Recovery boiler.

the boiler. The primary and secondary air intakes are situated beneath the black liquor guns and ensure that an optimal air-fuel ratio is achieved, and no incomplete combustion occurs. The tertiary and high tertiary air intakes ensure that there is excess oxygen for the combustion process. Additionally, these two air ducts form a 'blanket' of air to try and block carryover deposits as much as possible to prevent deposit formation to some extent. The boiler consists of superheaters, followed by a boiler bank and finally an economiser.

There are 4 superheaters present in the boiler. A bullnose in the lower part of the boiler ensures that the flue gas is forced to flow through the superheaters at all times. Steam exits the boiler at approximately 8965 kPa and 480 ^0C. The boiler produces steam at a rate of approximately 114 kg/s. From these parameters the power output of the boiler can be calculated. The boiler is approximately a 310 MW boiler. A schematic of the boiler can be seen in Figure 4.1.

The DCS system, from which the dataset for this work is drawn, interfaces with the sensors that are installed in the boiler. In general, there are pressure and temperature sensors installed before and after each section of the boiler. Currently, there are no sensor measurements installed within sections of the boiler which is one of the reasons the thermodynamic model was built per section. Some sections, also did not have sufficient measurement sensors to apply the current thermodynamic model to and were left out, as mentioned before. Other sensor measurements in the boiler include airflow sensors, which measure the amount of air entering the boiler, as well as flow sensors measuring the amount of fuel entering the boiler, to name but a few. All the necessary sensor measurements were used for the determination of the fouling factor via the thermodynamics models and the results were shown in Chapter 2.

There are 82 soot blowers installed in the boiler starting from the superheaters through to the economizer. The soot blowers utilise steam at 2800 kPa and 390 ^0C to clean the heat transfer surfaces. Currently, the system is set up as a dual soot blowing system meaning 2 pairs of soot blowers can be utilised at a time. The pair of soot blowers are opposite each other and clean the same region of the boiler from different sides. As mentioned before soot blowing takes place on a predetermined schedule. In areas where deposit build-up is substantial, soot blowers blow for 3 min 40 seconds at a time. In areas where soot blowing is not as critical, the soot blowing operation typically lasts 2 min 40 seconds. These soot blowing sequences' measurement data is extracted from the DCS dataset, including the fouling factor calculation, which is then used to fit the SINDy algorithm to. In other words, the SINDy algorithm in this chapter, is fitted to data that was drawn during periods when the soot blowers are active, where the assumption was made that the physics should remain relatively unchanged. The differential equations, that are extracted by the SINDy algorithm, describe the gradient development of these soot blowing sequences. More specifically, the equations describe the fouling development curve's gradient during a soot blowing operation. The equations are integrated in this Chapter, to obtain the fouling factor development prediction.

To start the SINDy algorithm implementation on the measurement dataset and the above-mentioned sequences, the optimal polynomial order for the feature library of the algorithm is determined first. In the previous chapter, three polynomial order models were tested and the first-order model seemed the logical best choice. This is investigated further in this chapter in terms of prediction performance and consistency. Determining a polynomial order allows one to establish a baseline algorithm setup to extract initial models with. The performance of these models is then be compared to more complex models further on in the chapter to determine whether improvements are made or not.

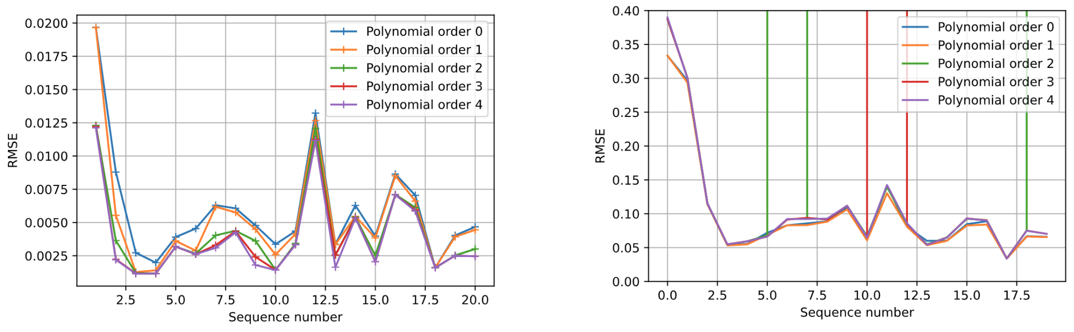
4.2 Determining the optimal polynomial order for the SINDy models

It was seen in the previous chapter, that there are a few possible model orders that can be used to fit the soot blowing curves in the Ngodwana dataset. It was shown that the second-order models were complex enough to fit more complex soot blowing curves very well, while the first and zero-order models were more easily identifiable, but lacked the complexity of the second-order model. It was also shown in the verification problems, that the second-order model coefficients were more susceptible to fluctuations when conditions, regarding the soot blowing curves, changed. High levels of noise, changing initial conditions and fouling factor curve changes all caused the second-order model coefficients to fluctuate more than the lower-order models. The zero-order model was the most consistent under changing conditions as it is the simplest model, while the first-order model consistency was somewhere between the second and zero-order model and seemed to be the happy middle ground. To choose an optimal model order for the establishment of a baseline model, it is necessary to investigate which polynomial order model best fits the soot blowing sequences of the real dataset. To determine which polynomial order works best with these models, the model order of the feature library will be increased from 0 to 4, where a polynomial of order 0 refers to a constant value and a polynomial of order 4 refers to a polynomial containing terms up to x^4. The reason for testing higher-order polynomials than the second-order, is to ensure that no assumption errors are made regarding the shape of the soot blowing curves.

The SINDy algorithm setup is as follows: The finite difference method is used for differentiation and the STLSQ optimizer is chosen for finding the model coefficients. The threshold value is set to the same value as was used in the verification problems namely, 0.0001. The feature library polynomial order is changed after 20 soot blowing sequences has been fitted and the RMSE's for them have been determined. For each polynomial order, the first 20 sequences for soot blower pair 3 and 4 from the Ngodwana dataset are fitted one at a time. These soot blowers are located in the secondary superheater, which is an area prone to fouling, as mentioned before. Once the algorithm has extracted the respective models for each soot blowing sequence individually, the derivative is predicted using the extracted model, and the actual derivative is computed using the central finite difference method. The root mean squared error of the predicted derivative and the computed derivative is calculated for each sequence and the results are shown in figure 4.2a. The extracted models are also integrated using an ODE integrator from the python math package, to obtain the fouling factor development curve prediction. Once again, the root mean squared error is calculated for the actual fouling factor development sequence and the predicted sequence. The results can be seen in figure 4.2b. It should be noted that one would normally test the model fit on test data and not the training data once again. However, it was seen that the models that were extracted at this point, differed too much from one sequence to the next and hence, a test dataset would just confirm that the model from one sequence does not fit the following sequence. This was most likely due to the inconsistency seen in the data.

In the figures, it is seen that the model that can predict the sequence derivatives the best, is the fourth-order polynomial model, however SINDy often eliminated the third and fourth-order terms to simplify the equation into a second-order term. It also often set the equation to a first-order model, hence the second-order term coefficient was also zero. This proves that most of the soot blowing sequences have shapes resembling second or first-order models. The other models also perform very well and their RMSEs do not increase substantially from the fourth-order model's RMSE, due to most sequences being relatively simple. In many cases, the first-order polynomial model performs just as well as the more complex models. This once again makes sense, as some sequences are nearly linear and the first-order model would be able to fit them well. It was seen that most of the extracted models' predictions resembled a second or first-order model and not a more complex curve indicating that higher-order models were unnecessary. When looking at figure 4.2b it can be seen that the zero and first-order polynomial models never diverge when they are integrated, as their RMSEs do not jump to a very large values for some of the sequences. It is also interesting to note that their root mean squared errors, for the actual time series data, are almost exactly the same as the more complex models. The higher-order polynomial models are seen to diverge, specifically the second and third-order models. This diverging behaviour was also seen in the verification problem, when noise was added to the artificial sequences. The fourth-order model does not diverge in these sequences, however, its performance is not much better than that of the first-order model and, if more sequences are predicted, the fourth-order model was seen to diverge often. The fourth-order model is also far more complex than necessary to predict the soot blowing curves, as these curves visually seem to be mostly first or second-order curves as can be confirmed in the examples of the previous chapter in figure 3.3

Thus, looking at the performance of these models, as well as their overall complexity, the first-order model seems to be the logical best choice. It has a very low level of complexity, while not simply being a straight line as with the zero-order model. It will therefore be able to fit more complex curve shapes. Furthermore, it performs nearly as well as the much more complex fourth-order model, when the RMSEs are compared. The first-order model also does not diverge, while the more complex models do, especially when there is substantial noise present in the data or when the initial



(a) Derivative RMSEs (b) Soot blowing curve RMSEs

Figure 4.2: Root mean squared error of the predicted and computed sequence derivatives for the first twenty sequences of soot blower pair 3 and 4 (a) and the root mean squared error of the actual sequences and the predicted sequences (b) for different polynomial orders.

values change too much from what the models had been trained on. This significantly affects the higher-order models' performance. Therefore, it is decided that the baseline models will be built using a first-order polynomial feature library.

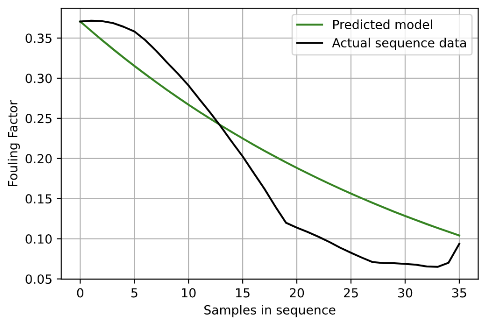
4.3 Establishing a baseline algorithm setup for model extraction

It has been established, that a first-order polynomial for the SINDy feature library, is a logical best choice when the performance, complexity and consistency of these models are compared. This is because most soot blowing curves are either linear or parabolic, and second-order models had the tendency to diverge more often. From the verification problem it was seen that the coefficients tend to fluctuate less with the first-order model than with the second-order one and fewer sequence predictions diverge when the ODE is integrated. These models are also more complex than the zero-order models and should therefore fit the data better.

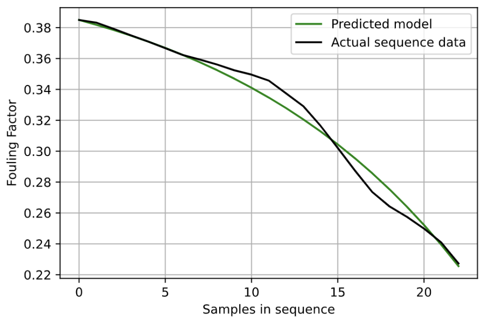
To establish a baseline of model performance, the only parameter that has yet to be determined in-depth, is the threshold parameter of the optimizer. So far, the threshold has simply been chosen to ensure that no trivial answers are obtained for the models. To optimise the threshold, a range of threshold values is tested. The sequences are fitted using the SINDy algorithm, and the extracted models are used to predict the fouling factor development from an initial value. RMSEs for the actual and predicted fouling factors are then determined and averaged for all the sequences. A threshold value with the lowest average RMSE for the sequences is selected to establish the baseline model. The threshold range was chosen to start at 1×10^{-6} and to end at 1×10^{-1}. The best threshold for the baseline model, which takes only the fouling factor as input, is found to be 0.0011 and is selected for the baseline model.

The SINDy algorithm is set up as follows for the baseline model: The feature library is chosen as the polynomial feature library with first-degree polynomials. The differentiation method used is the finite difference method and the optimizer of choice is once again the STLSQ optimizer. The sequences were fitted one at a time using the SINDy algorithm. Figure 4.3 shows 4 examples of the actual and predicted sequences once the SINDy algorithm has extracted models from the data set. The predicted models seem to fit the actual sequences very well and the noise in the sequences are not fitted, as a general model is extracted from the soot blowing sequence.

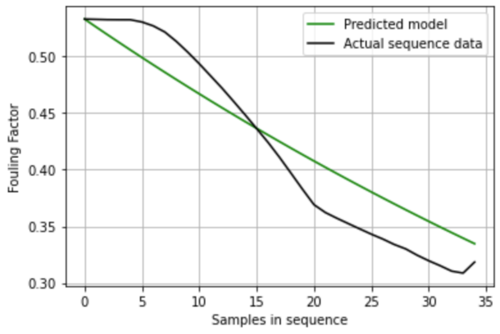
Once the sequences are fitted using the SINDy algorithm, the RMSE of each sequence is determined, for both the prediction of the sequence derivative and the time-dependent sequence itself. The RMSEs for the predicted and actual sequence derivatives can be seen in figure 4.4a for all extracted sequences. The RMSEs, for the actual and predicted sequences, are also shown in figure 4.4b. By investigating the two plots shown in figure 4.4, it can be noted that the baseline model performs well, as the RMSEs for both the predicted derivatives and the predicted sequences are relatively low. This is a positive result, as it shows that the models that are being extracted are plausible, that most likely correctly describe the underlying fouling factor development trend during a soot blowing sequence.



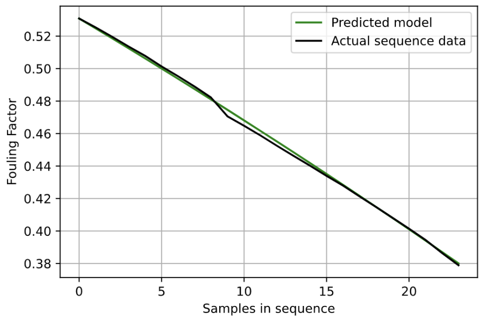
(a) Example 1



(b) Example 2

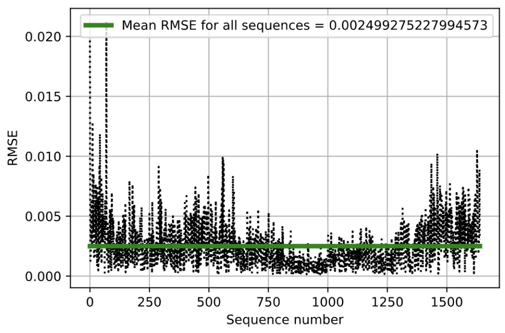


(c) Example 3

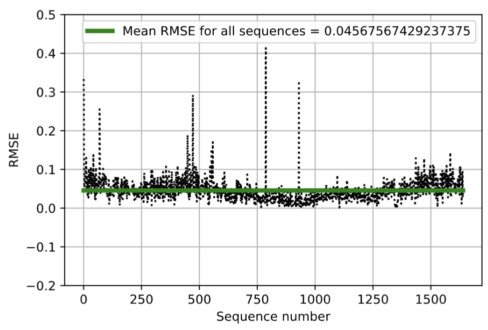


(d) Example 4

Figure 4.3: Examples of simulations of the models found by the SINDy algorithm when only the FF is used as input to the system.



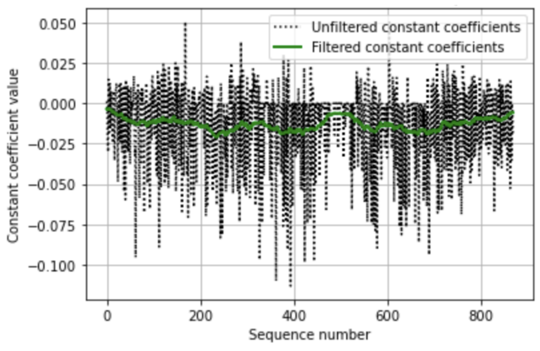
(a) RMSE for derivative prediction



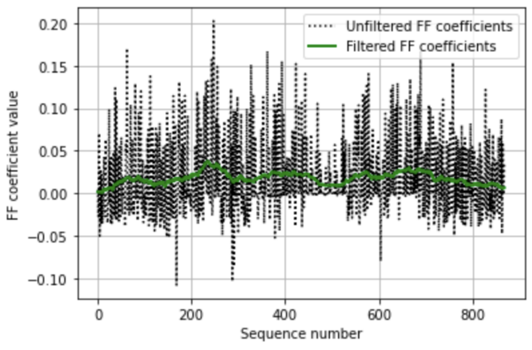
(b) RMSE for sequence prediction

Figure 4.4: Baseline model performance in terms of the RMSEs for the predicted and actual derivatives as well as the predicted and actual soot blowing sequences for sequences from soot blowers 3 and 4.

While the prediction accuracy of the models is important, consistency, in terms of the extracted models, is also necessary if robust predictions are to be made in the future. If



(a) Constant coefficient development



(b) FF coefficient development

Figure 4.5: Coefficient value development for the different soot blowing sequences over time, for the model that takes only the fouling factor as input.

the SINDy models perform very well in terms of their predictions, but all the models are different for each of the sequences, one cannot make justifiable assumptions regarding the underlying physics in the data. This simply means that one already requires the data to build a model, which is not very useful in practice, as one would want to predict the data. Looking at figure 4.5, it can be noted that the model coefficients are not consistent at all. This is expected to some extent, as it was shown in chapter 3 that the model coefficients are highly dependent on the shape of the curves, their initial values and noise in the system. From the examples shown in figure 4.4, one can see that neither the shape nor the initial values of the curves stay consistent. Thus, one sees significant fluctuations in the model coefficients. There is also likely some degree of noise present in the data, since it was derived from measurement data. This could also affect the model consistency as was shown in Chapter 3.

The SINDy model can therefore not extract a consistent underlying model so far, since almost every sequence has its own unique model. This makes the models difficult to analyse and makes it difficult to extract a general model, even for short time intervals in the boiler. While the results are not optimal thus far, a positive aspect, is the fact that the models that are recovered, are very interpretable and seem to fit the data very well. The SINDy algorithm is also performing as one would expect, which is a step in the right direction. A possible method of improving the model consistency, would be to investigate additional inputs to the SINDy algorithm. If more inputs are added to the system, one might provide the algorithm with more information and hence extract a more general model, since the complexity of the model will be increased without increasing the polynomial order explicitly.

Since a baseline has now been established, the models can be developed further to include additional inputs and to potentially incorporate more complexity. The newer models can then be compared against the baseline model to determine whether any improvement can be seen, not only in terms of performance but also in terms of model consistency.

4.4 Adding more inputs to the SINDy models

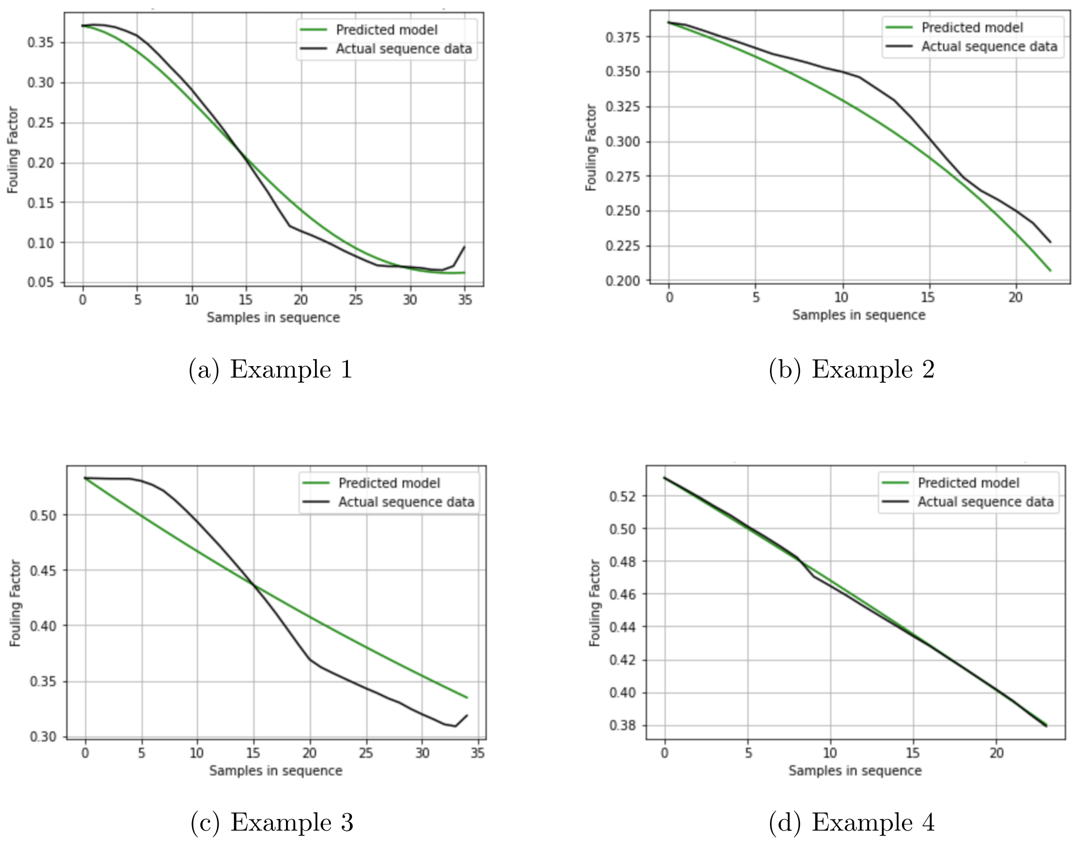
It has been seen that the SINDy algorithm can extract the underlying equations from the sequence data efficiently and that these models capture the trend in the soot blowing sequences. Furthermore, the best polynomial order was shown to be a first-order polynomial, as its performance was comparable to that of higher-order models, while the integrated sequences did not diverge due to the low level of complexity of the models. These changes were implemented and a baseline algorithm setup and model performance was established.

To further try to improve the model performance, as well as the coefficient consistency of the extracted models, additional inputs to the SINDy algorithm are investigated. As most thermodynamic models depend on some form of temperature difference to estimate the amount of heat being transferred, these types of inputs might add significant value to the SINDy models, since they can indirectly give one an indication of the level of fouling in the system. The higher the fouling, the less heat is transferred and vice versa ,therefore, temperature differences seem to be a logical input choice. The first input that is added to the SINDy algorithm is the flue gas temperature difference, defined as the difference between the flue gas exit temperature and the flue gas inlet temperature. The flue gas temperature difference is chosen, as this temperature change could give one an indication of the amount of heat that could be transferred to the steam and hence it can also be linked to the level of fouling present in the boiler. Initially, the input will be given to the SINDy algorithm as a control input, as described in section 1.2.5 and first explained in the paper by Kaiser et al. (2018). The SINDy algorithm will thus not have to estimate the temperature difference model in addition to the fouling development model. This is done purely to investigate whether this additional input provides any valuable information that can improve model performance and consistency so that this input type model can be compared directly to the baseline model.

4.4.1 Using the flue gas temperature difference as control input

The algorithm setup is similar to what has been done before. The finite difference method is used for the differentiation of the input dataset, the optimizer of choice is the STLSQ optimizer and the polynomial feature library is populated with polynomials up to order 1. Similar to what had been done with the baseline model, the threshold is once again optimized by finding a threshold value that results in the lowest average RMSE for all the sequences. The same range of values is tested, as was mentioned in the section on the baseline model and the best general threshold was found to be 0.00225. The models are trained using the fouling factor sequences as system input and the flue gas temperature difference sequences as control inputs. Examples of the model predictions of the fouling factor can be seen in figure 4.6. It was noted that most of the models were first-order models, while some were set to be 0 order models by the SINDy algorithm.

It can once again be noted that the models fit the data very well, as the trends in each soot blowing sequence are captured by the models that were extracted by the SINDy algorithm for the respective sequences. When one compares the results to those of the baseline model, shown in figure 4.3, it can be seen that the results look fairly similar and no significant changes have occurred regarding the general shape of the predicted models.

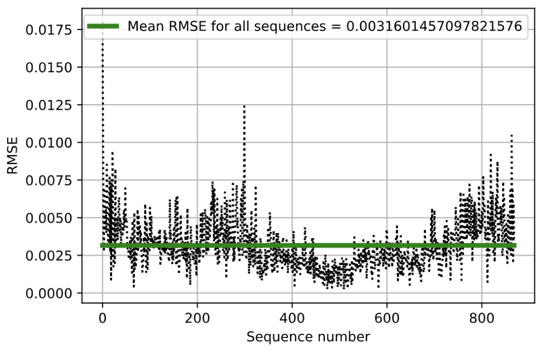


(a) Example 1 (b) Example 2

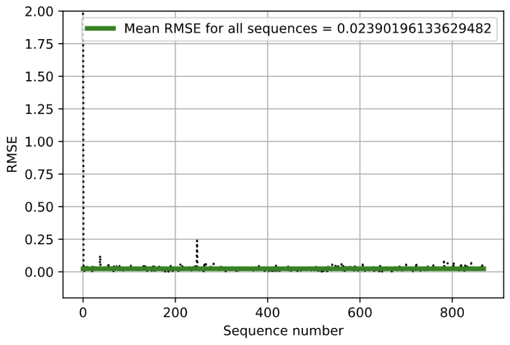
(c) Example 3 (d) Example 4

Figure 4.6: Examples of simulations of the models found by the SINDy algorithm when ΔT_f is used as control input to the algorithm.

To properly evaluate the performance of the model compared to the baseline model, the



(a) RMSE for derivative prediction



(b) RMSE for sequence prediction

Figure 4.7: Model performance, in terms of the RMSE's for the predicted and actual derivatives as well as the predicted and actual soot blowing sequences, when ΔT_{flue} is added as control input.

RMSE of the predicted and computed derivatives, as well as the RMSE of the predicted and actual soot blowing sequence data, is once again plotted. The results can be seen in figure 4.7. When comparing the average RMSEs of the predicted and actual derivatives of the models in figures 4.4a and 4.7a it seems that the model has not improved. However, when comparing the actual sequence and predicted sequence RMSEs of the two models in figures 4.4b and 4.7b one can note that the model with the added flue gas temperature input performs significantly better on average, with the average RMSE of sequence prediction dropping from 0.0457 to 0.0239. Its average RMSE is almost half that of the baseline model. It, therefore, shows that, by adding the flue gas temperature difference, one has incorporated extra information regarding the fouling factor into the model, and this has led to the models being able to better predict the fouling factor development.

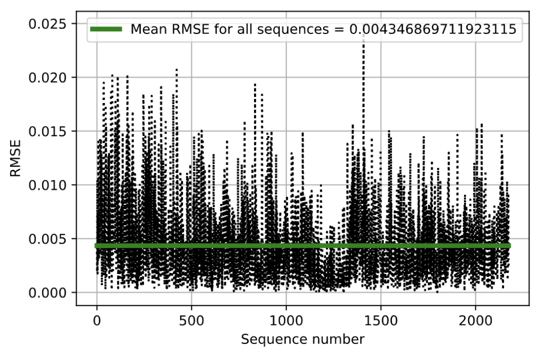
It should be noted, however, that this model is not feasible to be used in practice, if predictions need to be made. This is because the flue gas temperature difference will not be known beforehand and cannot be used as a control input to the model when simulated for prediction. To therefore see if a model, that takes the flue gas temperature difference into account, will be useful in practice, the SINDy algorithm must be altered so that the flue gas temperature difference is no longer a control input, but rather a system input that also has to be predicted. This model can then be compared against the previous ones to determine whether adding this as a system input would be beneficial for prediction accuracy purposes and whether the model coefficients become more or less consistent.

4.4.2 Using the flue gas temperature difference as a system input

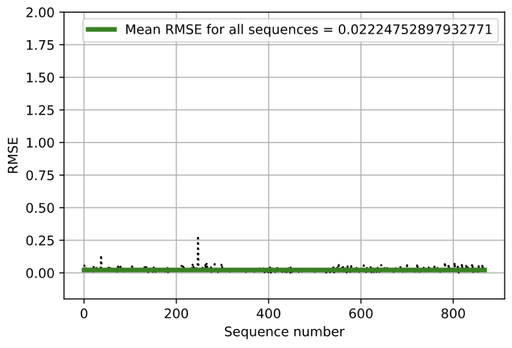
A model, that estimates the difference in the flue gas temperature as well as the fouling factor, is built using the same parameters as the model where the flue gas temperature difference was used as a control input. Now the flue gas temperature difference model is also extracted and the temperature difference model is not just used as an external control input to the system. Similar to the previous models, the threshold parameter is optimized and found to be 0.00065, which differs slightly from the model that takes the flue gas temperature difference as a control input. The model is fitted to the training data and it is seen that the plots of the predicted and actual sequences are nearly identical to those seen in figure 4.6. Once again, the RMSEs for the sequences, for both the predicted derivative and the predicted sequences of the fouling factor, are plotted, and can be seen in figure 4.8.

Using the flue gas temperature difference as a system input rather than a control input, does not negatively affect the model performance and delivers similar results. The average RMSE for the predicted and actual sequences is slightly lower than the control input model, showing that this model makes slightly better predictions on average. It can also be noted that there are fewer spikes in the predicted sequence RMSEs, so the models are also slightly more consistent when integrated. When evaluating the overall performance and comparing it to that of the baseline model, this model outperforms the base model in terms of prediction accuracy, even though the derivative RMSE is slightly higher. Additionally, this model may perform better than the one with the control input, because the noise in the flue gas temperature measurement is not incorporated directly into the model. Instead, the flue gas temperature difference is estimated using a ‘smoother’ model that does not introduce noise to the model system. This then results in better predictions. The model that uses the flue gas temperature difference as input directly, directly incorporates the noise in the measurements into the model and the model has no way of compensating for it, which may result in less accurate predictions.

This model makes better predictions than that of the baseline model, even when having to predict more than one input. However, as mentioned before, model prediction accuracy is not the only important requirement and model consistency, in terms of the coefficients, must also be investigated. Figure 4.9 shows the coefficient development over the different sequences, where there are now 3 coefficients namely the constant coefficient, the fouling factor (FF) coefficient and the temperature difference coefficient (ΔT_f). The basic form



(a) RMSE for derivative prediction



(b) RMSE for sequence prediction

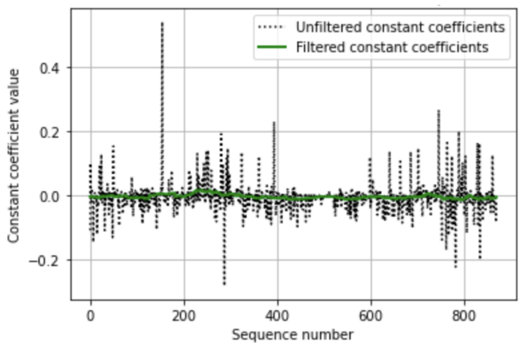
Figure 4.8: Model performance, in terms of the RMSEs for the predicted and actual derivatives as well as the predicted and actual soot blowing sequences,when ΔT_{flue} is added as an additional system input.

of the extracted models looks as follows:

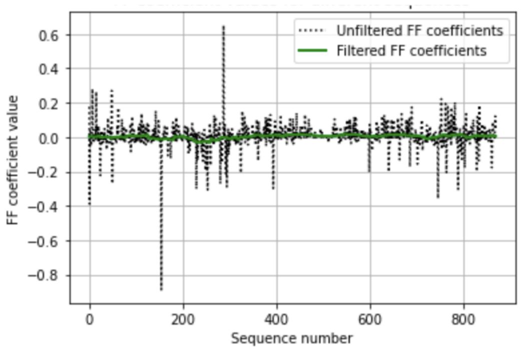
$$FF' = C1 + C2 * FF + C3 * \Delta T_f \tag{4.1}$$

where C1, C2 and C3 correspond to the constant, fouling factor and temperature difference coefficients, respectively.

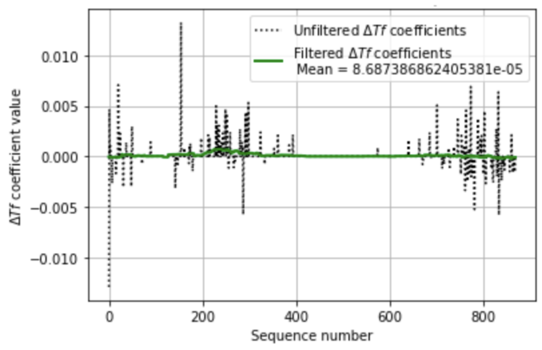
When comparing figures 4.5 and 4.9, it seems as if the model coefficients have stabilised slightly. The range of the values of the coefficients are very similar between the baseline model and the model with the flue gas temperature difference as input, however, the baseline model coefficients are noisier. The model with the flue gas temperature difference has fewer irregularities in the coefficient, and noise in the coefficient development graphs seems to be visually less. This can be seen in both the constant coefficient and the fouling factor coefficient graphs. It should be noted that the values for the flue gas temperature difference coefficients are very small. This is because the flue gas temperature difference is a large input value and the model compensates for this by making its coefficient small. Therefore, a smaller coefficient, in this case, does not mean that the input is less important. It was noticed that the temperature difference input coefficient was slightly less noisy than that of the other coefficient development plots, with most of the irregularity occurring near the beginning and end sequences in the dataset. This is promising, as one would expect the physics to change rapidly after the boiler had just been water washed, due to deposits forming quickly on the clean surfaces of the boiler. Similarly, when the boiler is due for another water wash, the flue gas temperature most likely lies close to the sticky temperature range, mentioned in chapter 1. Deposits will therefore accumulate fairly quickly on the heat transfer surfaces, which most likely changes the



(a) Constant coefficient development



(b) FF coefficient development



(c) ΔT_f coefficient development

Figure 4.9: Coefficient development for the different soot blowing sequences over time, for the model that takes the flue temperature difference as a system input.

physics rapidly. A change in physics will result in a change in the coefficients. Therefore, the irregularities in the flue gas temperature coefficient can be interpreted to some degree.

It can thus be confirmed that adding the flue gas temperature difference to the model seems to have improved the overall model performance, not only in terms of prediction accuracy, but also in terms of model consistency to some extent. The coefficients have become slightly more consistent and the average RMSE has halved. Adding inputs to the SINDy algorithm, therefore, seems to improve the model, as the underlying physics can most likely be better extracted by the algorithm or the added complexity of the models can better describe the soot blowing sequences compared to the very simple baseline model. The fluctuations in the model coefficients are still significant and more inputs may further reduce the fluctuation. Since the flue gas temperature difference improved the model performance, adding another temperature difference metric, namely the steam temperature difference, might also improve the model performance, especially considering that it was a key metric used in the calculation of the fouling factor.

4.4.3 Using both the flue gas and steam temperature differences as system inputs

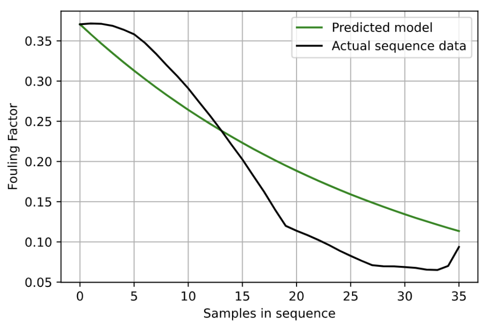
As was seen in the previous model, adding the flue gas temperature difference as an input allowed the model to not only be more consistent, but also improved the average RMSE of the predicted soot blowing sequences. Therefore, the steam temperature difference might also add valuable information to the model if it is used as a system input. This is because the temperature change of the steam correlates directly to the amount of heat transferred to it and, thus, directly correlates to the amount of fouling present in the boiler. Three ODEs are extracted by the SINDy algorithm once the steam temperature difference is added, as there are now three system inputs. ODEs are found for the fouling factor, the flue gas temperature difference and the steam temperature difference, respectively. A generic system can be seen in equation 4.2

$$\begin{aligned} FF' &= C1 + C2 \times FF + C3 \times \Delta T_f + C4 \times \Delta T_s \\ \Delta T_f' &= C5 + C6 \times FF + C7 \times \Delta T_f + C8 \times \Delta T_s \\ \Delta T_s' &= C9 + C10 \times FF + C11 \times \Delta T_f + C12 \times \Delta T_s \end{aligned} \tag{4.2}$$

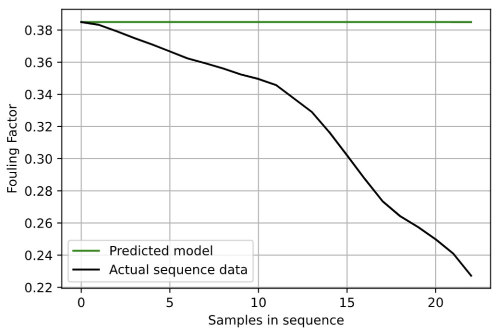
Once again, the SINDy algorithm is set up the same way as with the previous experiments. The threshold value is optimized to ensure the lowest average RMSE value is obtained for the sequence predictions and is found to be 0.01. The four example sequences, together with the predicted sequences from the new model, can be seen in figure 4.10, while the RMSEs of the predicted derivatives and predicted sequences can be seen in Figure 4.11.

The models fit the actual sequence data poorly when investigating figure 4.10. Many of the models extracted by the SINDy algorithm are trivial answers (the fouling factor derivative equation was simply zero), which is mostly due to the threshold being set to a large value. However, if the threshold is adjusted to be smaller, the resulting models also do not fit the sequences very well, with many of the predictions diverging to very large values once the model is integrated. This is why 0.01 is found to be the best threshold when optimising this value. Adding the steam temperature difference as an additional input seems to have over-complicated the models and results in trivial or divergent sequences once these models are integrated.

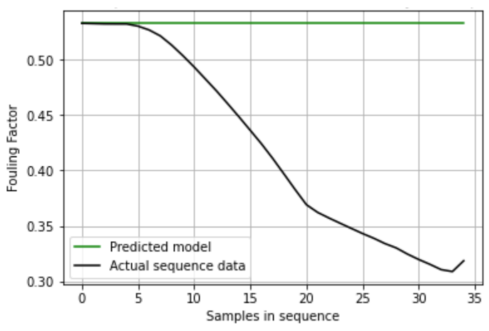
The RMSE plots in figure 4.11 further show that the model performance has decreased. The average RMSE for the predicted derivative has increased substantially and the average RMSE for the sequence prediction shows the worst performing model yet. The decrease in performance is also noted when looking at the model coefficients for the FF ODE. The fact that the algorithm often extracted trivial models can be noted when looking at the coefficient development plots in figure 4.12. It can be seen that both temperature differences were never regarded as important inputs and their coefficients were always zero. The constant coefficient shows the same trend, except for the last sequence where a value was found. Finally, the fouling factor coefficient was the only coefficient that was not always found to be 0 by the algorithm. However, in many sequences, a trivial answer was still obtained and the entire model was equal to zero.



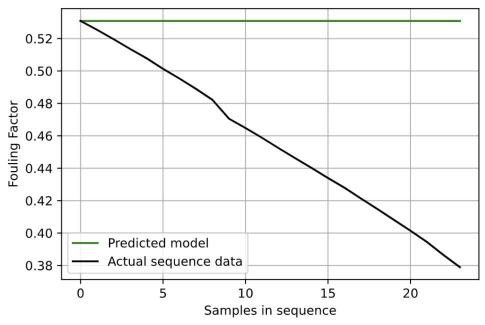
(a) Example 1



(b) Example 2

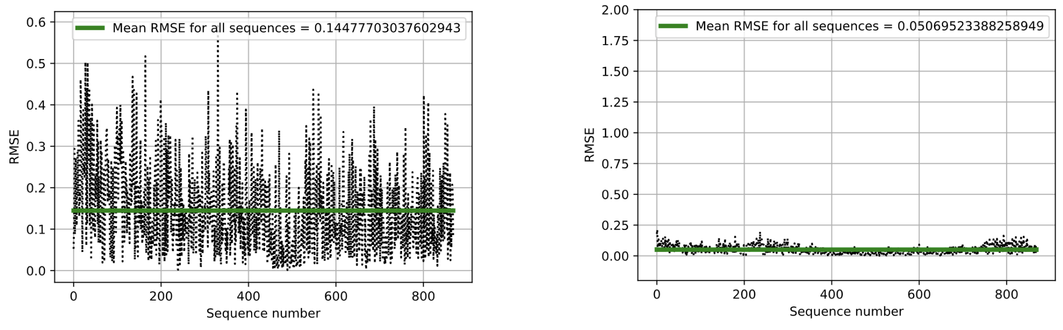


(c) Example 3



(d) Example 4

Figure 4.10: Examples of simulations of the models found by SINDy algorithm when ΔT_{flue} and ΔT_{steam} are used as system inputs to the algorithm.

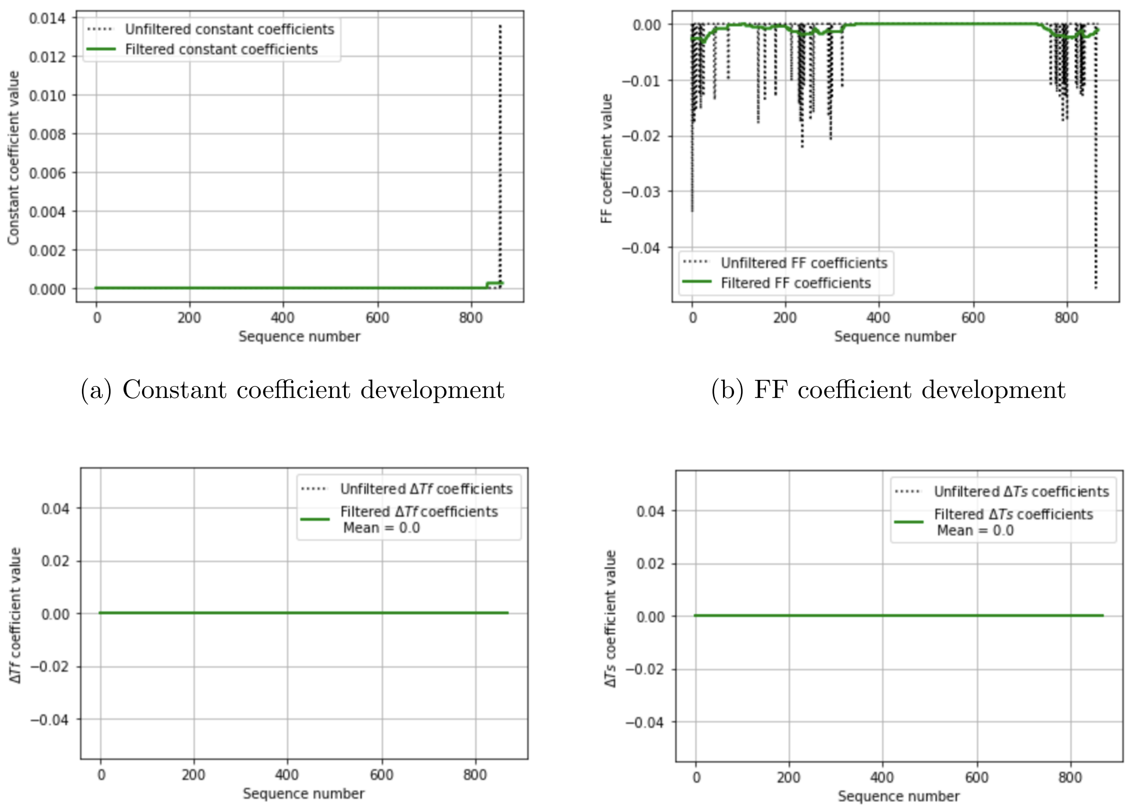


(a) RMSE for derivative prediction (b) RMSE for sequence prediction

Figure 4.11: Model performance, in terms of the RMSEs for the predicted sequence derivatives as well as the predicted sequences,when ΔT_{flue} and ΔT_{steam} are added as additional system inputs.

Adding the steam temperature difference has not improved the model performance, and the extracted physics is not interpretable due to trivial answers being extracted or answers that cannot be used for prediction. It therefore seems that if the wrong input combination is given to the algorithm, the resulting models can be a very poor representation of the underlying physics. An explanation for this substantial decrease in performance might be the fact that the steam temperature is regulated significantly in the boiler. Since the boiler is delivering steam to a turbine on the plant, the steam has to leave the boiler at a specific pressure and temperature and adjustments are made in the boiler regarding fuel and air that ensure the steam is at the correct temperature. It therefore, makes sense why the SINDy algorithm is struggling to extract the underlying physics when the steam temperature difference is used in such a raw form. It was also noted, when plotting the steam temperature difference over the sequences, that the value does not change much when soot blowing is active. Therefore the steam temperature difference input is not supplying a significant amount of information to the algorithm in its current form. If the steam temperature is also being controlled in the boiler, the SINDy algorithm is trying to extract not only the physics in the boiler, but also the changes that control systems are making to the physics. This may cause a discrepancy in the models, where the physics and control system dynamics are contradicting each other, resulting in trivial models being extracted.

Another possible explanation for the poor performance might be the fact that the definition of the steam and flue temperature differences is not well suited to the physics in the boiler. If one recalls the thermodynamic model that was built in chapter 2, one should keep in mind that the log mean temperature difference (LMTD) was used in the calculation of the actual heat transfer coefficient. The temperature differences



(a) Constant coefficient development (b) FF coefficient development

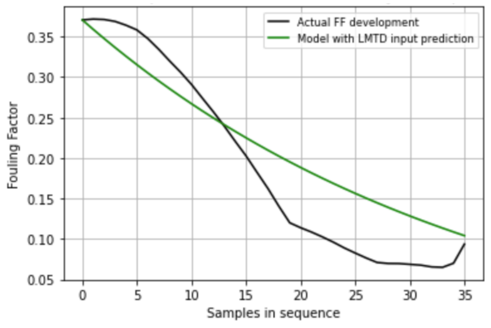
(c) ΔT_f coefficient development (d) ΔT_s coefficient development

Figure 4.12: Coefficient development for the different soot blowing sequences over time, for the model that takes the flue and steam temperature differences as system inputs.

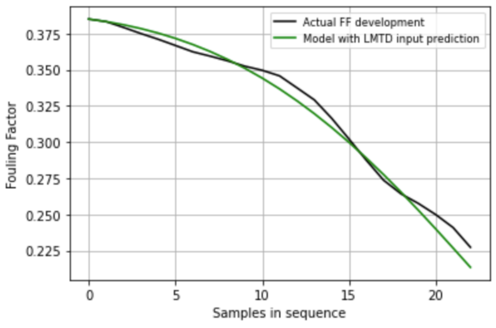
were then not defined as the inlet and outlet temperature difference of the steam and flue gases, but rather as the temperature difference between the hot sides of the fluids for one temperature difference and the cold sides for the other. Additionally, the LMTD is non-linear, which the SINDy algorithm might not be able to capture with its polynomial feature libraries. Hence, a model is built that will not take the flue and steam temperature differences as separate inputs, but will take the LMTD as a single input to test whether a performance improvement can be seen.

4.4.4 Using the log-mean temperature difference as system input

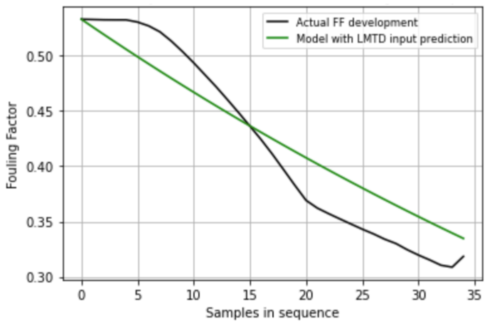
Since the LMTD was used in the calculation of the theoretical heat transfer coefficient, adding it as a system input to the model might improve the model performance. As was seen previously, adding the steam temperature difference as input to the model degrades the model performance and results in trivial answers from the SINDy algorithm. The LMTD, on the other hand, still incorporates the steam temperature difference, but in a different configuration that may provide more useful information to the models, and hence improve their prediction capability. Using the same algorithm setup as before and optimising for the threshold of the STLSQ optimizer, the best threshold value was found to be 0.000075. Once again, the four example sequences are shown in figure 4.13.



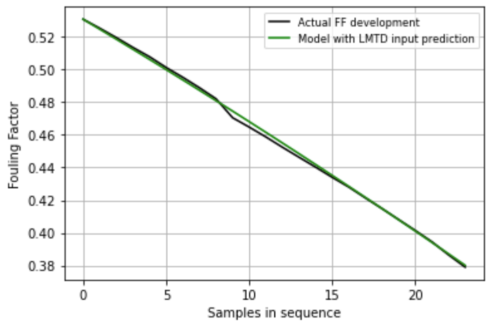
(a) Example 1



(b) Example 2



(c) Example 3



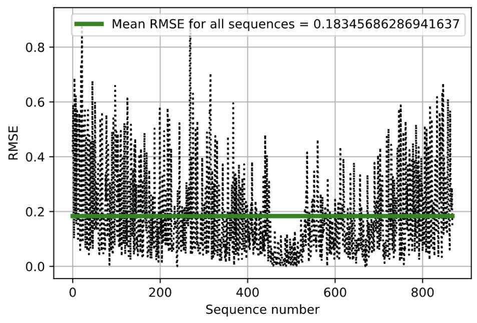
(d) Example 4

Figure 4.13: Examples of simulations of the models found by the SINDy algorithm when the LMTD is used as a system input to the algorithm.

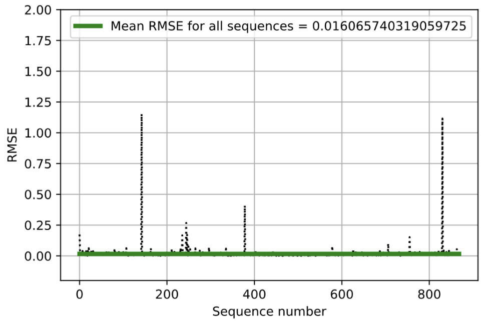
It can be seen that the predicted sequences look much better than that of the previous model, where the temperature differences were added as separate inputs, since no trivial answers are obtained. The example sequences closely resemble those of the model where the flue gas temperature difference was used as input to the system. The generic model form that was extracted can be seen in equation 4.3. The improved model performance can be confirmed when investigating figure 4.14

$$\begin{aligned} FF' &= C1 + C2 \times FF + C3 \times \Delta LMTD \\ LMTD' &= C4 + C5 \times FF + C6 \times \Delta LMTD \end{aligned} \tag{4.3}$$

The predicted sequence average RMSE is the lowest out of all the models thus far. This shows that the LMTD has provided valuable information to the algorithm, which allows



(a) RMSE for derivative prediction

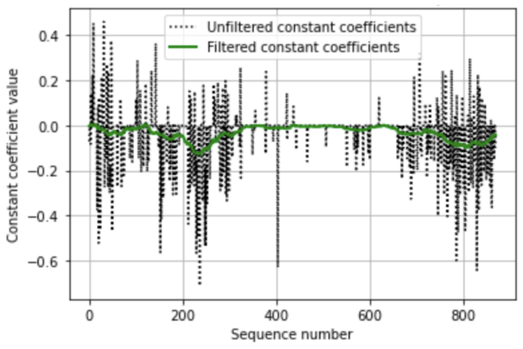


(b) RMSE for sequence prediction

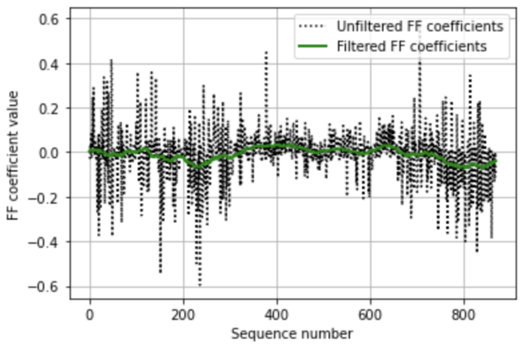
Figure 4.14: Model performance, in terms of the RMSEs for the predicted sequence derivatives as well as the predicted sequences,when LMTD is added as system input.

it to extract improved models. It is interesting to note that the RMSE of the derivative prediction is comparable to that of the model where the steam temperature difference was added, as the RMSEs are within the same order of magnitude. However, the time-series sequence prediction of the model with the LMTD as input is far more accurate than that of the previous one. This could indicate that the model generalises well and does not fit the noise, which is inevitably present within the derivative. The model possibly captures the true underlying physics better. Since the derivative is computed using the finite difference method, the noise in the measurement data is amplified, as noise gets amplified when numerical differentiation is applied to a dataset (Chartrand 2011). The fact that this model does not fit the derivative well, but can predict the fouling factor development curves the best out of all the models so far, gives one an indication that the correct physics is likely being extracted. The model prediction consistency however, is still fluctuating too much to test on an out of sample dataset, since the fit will still likely be poor in most cases.
Figure 4.15 once again, shows the coefficient development over the different sequences. The model consistency is not as good as that of the models where only the flue gas temperature difference was used, but the consistency has not decreased dramatically. The models also fluctuate slightly less than that of the baseline model, showing that the LMTD does at least improve the model consistency somewhat. It can be seen that the coefficients stabilise marginally around the middle 200 sequences, which corresponds to sequences approximately halfway through the dataset of 4 months. This might be due to the boiler physics not changing as rapidly when compared to the initial and final parts of the boiler cycle.

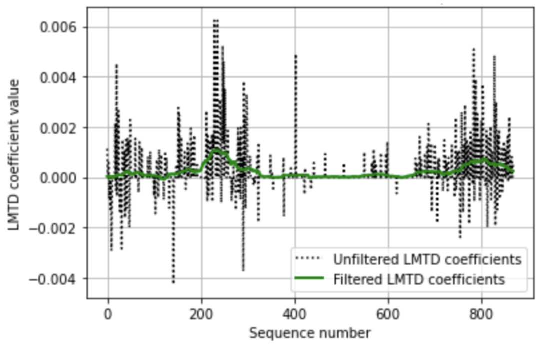
The boiler fouls rapidly right after it has been water-washed and the physics hence changes rapidly as was discussed previously. The conditions in the boiler may stabilise slightly when the temperature reaches the radical deformation temperature of the deposits. This causes them to melt and fall off due to gravity alone (Tran 2007). At this point the combination of soot blowing and deposits melting off, may be causing a state of equilibrium in certain sections of the boiler and the physics do not change as rapidly. Once the fouling reaches a critical level in other sections, the accumulation rate increases rapidly and the physics start to change rapidly again near the end of the dataset, which was also discussed earlier. One thus sees significant coefficient fluctuation in the first and final sequences from the dataset.



(a) Constant coefficient development



(b) FF coefficient development



(c) LMTD coefficient development

Figure 4.15: Coefficient development for the different soot blowing sequences over time, for the model that takes the LMTD as system input.

Since the model predictions are the most accurate when the LMTD is used, this model seems to be a step in the right direction and slightly improves upon the model that only uses the flue gas temperature difference as input. So far, adding inputs to the SINDy algorithm has not resolved the fact, that changes in the soot blowing curves and initial values cause fluctuations in the extracted model coefficients. While some improvement in consistency has been seen once inputs are added, there has not been a way to extract a single physics model, over a fixed period in which the underlying physics should not change, for a specific soot blower pair. The initial values and soot blowing curves still change too much in a space amount of time to achieve this. A positive result, is the fact that the models are very accurate when certain inputs are added and the physics models can be interpreted easily, to obtain a general idea of what we expect the physics to look like. The models that are being obtained, as well as their prediction accuracy is therefore proving, that the SINDy algorithm is a viable method of model extraction for the boiler. The inconsistency that is being seen, seems to merely be due to a complex dataset with many inconsistencies and not due to a flaw or shortcoming in the SINDy algorithm. To further try and overcome this data problem, the threshold parameter of the STLSQ optimiser will be investigated further, since this has not been done in the study so far, even though the parameter plays a significant role in the models that are extracted.

4.5 Optimising the SINDy threshold parameter sequentially

It was seen in the initial implementation of the SINDy algorithm on the Ngodwana data, that optimising the threshold parameter of the STLSQ optimizer, resulted in more accurate predictions and slightly improved the consistency of the model coefficients. It was also noted throughout the previous experiments that the threshold parameter played a significant role in the general model form that is extracted, i.e. the number of terms in the ODE that are kept. The threshold parameter could also lead to trivial answers when the value was set too high. This parameter, therefore, has a significant impact on the results one obtains and must be investigated further to try to overcome the fluctuations in the model coefficients due to inconsistent data.

An experiment is set up, where the threshold parameter is optimized for each individual sequence instead of selecting one value that fits all sequences the best. The SINDy algorithm may not be extracting the best model for every sequence when a general threshold parameter is set. Optimising the threshold parameter for every sequence may result in more consistent model coefficients, as the best performing model possible (which is likely the model that is closest to the ground truth) will be extracted for each sequence. To explain this concept, one might imagine that every sequence is a tree in a forest that has many hills and valleys. If one chops off all the trees at the same height, some trees may be cut off correctly, while others will only be cut at the top and some may be cut off beneath the roots. If each tree is chopped individually, the overall consistency of the cut will be better and all trees will be cut off at the same height. Similarly, optimising the threshold for every sequence may result in more consistent models being extracted that all have similar forms and coefficients. Therefore, it might overcome the data inconsistency that is causing fluctuations.

The SINDy algorithm is set up, as has been done in the previous experiments. The exception for this test is the fact that the threshold value is changed iteratively for every sequence and the best threshold value is extracted. This best threshold value is determined by the prediction error of the model on the training data of the sequence. The threshold value, that results in a model with the lowest prediction error is seen as the optimal threshold for the specific sequence. The best performing model with the optimal threshold is saved, and the coefficients are noted for each sequence. This is done for every input type model that was built in the previous section, up to and including the model with the LMTD as input. Table 4.1 shows the average prediction RMSEs and the standard deviation (STD) of the RMSEs for each model type, when a general threshold was used and when the threshold is optimized for each sequence.

Table 4.1: Average RMSE and RMSE STD for models when a general threshold is used vs when the threshold is optimized for every sequence.

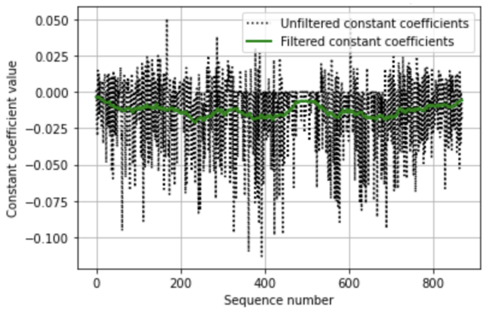
	General threshold		**Threshold optimized**	
Model Input Type	**Mean RMSE**	**RMSE STD**	**Mean RMSE**	**RMSE STD**
Only FF	0.0457	0.0285	0.0130	0.00012
ΔT_finput	0.0222	0.0373	0.0093	5.6548e-5
$\Delta T_f + \Delta T_s$ input	0.0506	0.02996	0.01695	0.0017
LMTD input	0.0161	0.0581	0.00871	9.3650e-5

Clearly, optimising the threshold for every sequence improved the prediction accuracy of the respective models for every sequence, as the average RMSEs dropped notably. While this is a good indication, one still has to determine whether the models have become more or less consistent when the threshold is optimized. To do this, one has to look at the coefficients that are extracted once every sequence threshold is optimized and evaluate their consistency. For comparative purposes, the constant coefficient development for every input type model is shown in figure 4.16 for the case where a general threshold was used and for the case where every sequence threshold was optimized.

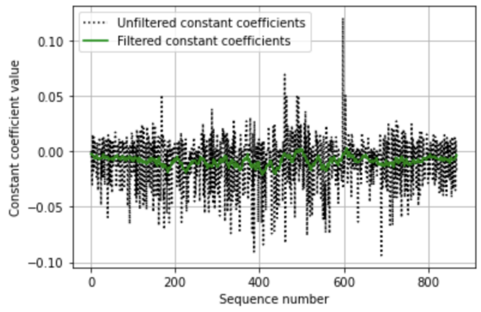
When investigating figure 4.16 it can be noted that the coefficient fluctuations have either stayed visually similar or increased for the case where the threshold is optimized for every sequence versus the case where a constant threshold was chosen. The fouling factor input only model has a smaller range of coefficient values, however the same amount of fluctuation is present. The ΔT_f input models have more coefficient fluctuations and a wider range of possible coefficient values, when the threshold is sequentially optimized. For the LMTD input model, the range of the coefficient value fluctuation has decreased slightly. However, the more consistent coefficient values that were previously extracted and seen in the middle of the dataset in figure 4.15 has also now started to fluctuate. The model that takes both the flue gas and steam temperature differences into account sees the least improvement, with the coefficients becoming more sporadic and coefficient values becoming very large in some cases which lead to diverging sequences once the models were integrated. Most of the coefficients from this model input type are otherwise still trivial (equal to zero).

Optimising the threshold for every sequence has thus not solved the problem of inconsistent models due to inconsistent data, but has rather increased the fluctuations in the model coefficients in most cases. This does make sense as the models are fitted even more specifically to each sequence and small changes between sequences result in further changes in the extracted model coefficients. Figure 4.16 only shows the constant coefficient development for the different input type models, however the other coefficients showed the same behaviour, where the fluctuations were increased rather than improved when optimising the threshold. It is therefore clear that optimising the threshold for every sequence individually, does not resolve the problem of inconsistent data sequences and proves that the fluctuations are not due to incorrect threshold parameters, but rather due to inherent data inconsistency.

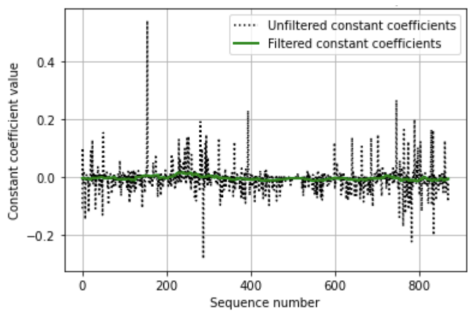
To try and introduce some consistency and standardisation in the data sequences, and hence the extracted models, scaling of the fouling factor and normalisation of the dataset will be performed. The scaling of the fouling factor is done to try and manipulate the extracted model coefficients to lie within a different range. This is done, as it was seen in the verification problems in Chapter 3, that some ranges of coefficient values make it difficult to distinguish between respective soot blowing curves. Hence, forcing the coefficients to be in a different value range might allow the algorithm to more easily distinguish between respective soot blowing curves. The normalisation of the dataset is performed to attempt to make input values more uniform and standardized and hence, one might extract more consistent models in different sections of the dataset.



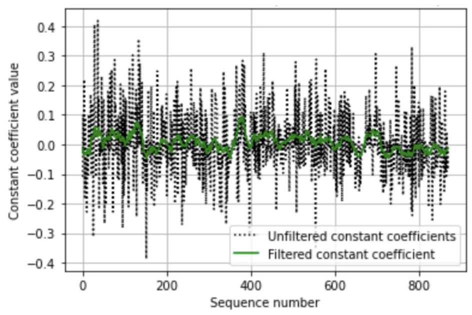
(a) General threshold (FF input only)



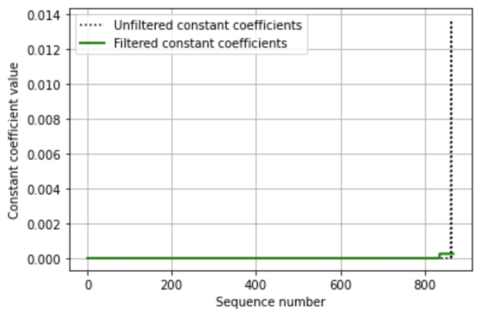
(b) Individually optimized threshold (FF input only)



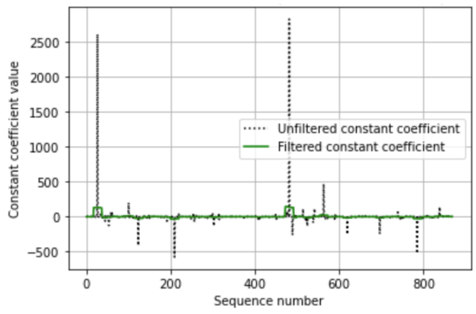
(c) General threshold (ΔT_f input)



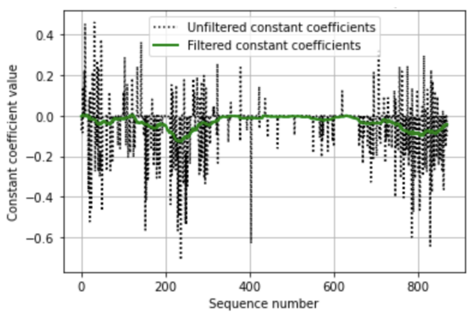
(d) Individually optimized threshold (ΔT_f input)



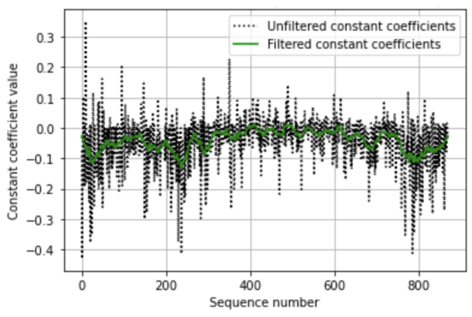
(e) General threshold (ΔT_f and ΔT_s input)



(f) Individually optimized threshold (ΔT_f and ΔT_s input added)



(g) General threshold (LMTD input)



(h) Individually optimized threshold (LMTD input)

Figure 4.16: Comparison of constant coefficient development for different input type models when a general threshold is used vs optimising each sequence's threshold.

4.6 Scaling and normalising the data

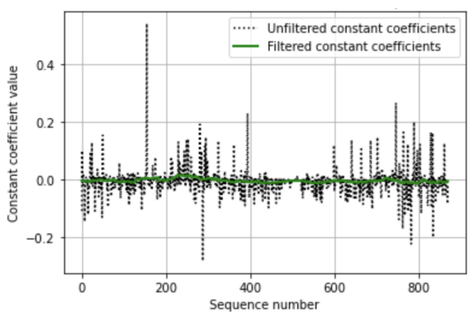
It was seen in the verification problems in Chapter 3, that some ranges of model coefficient values made it difficult to distinguish between different soot blowing curves and could cause a manner of ill-posedness in the model coefficients. In other words, there were potentially more than one coefficient or coefficient combination that could result in the same general curve shape, which could lead to coefficient fluctuations. It could be possible that the curve location and coefficient ranges that are being extracted from the Ngodwana dataset lie in such a range and could be causing some of the fluctuations that are being seen in the model coefficients. An experiment is therefore set up, to test whether indirect scaling of the model coefficients could improve model consistency

Changing the scale of the coefficients is easy when one has a verification problem model where you can simply change their values. However, changing the extracted model coefficients directly for the dataset is not as easy. To change the size of the model coefficients, one can do one of two things. The dataset can be normalized and, by doing so, the input values to the model are made smaller or larger, depending on their initial range. This will force the SINDy algorithm to compensate by making the model coefficients larger or smaller to still obtain the same prediction curve shape. Hence one can indirectly adjust the coefficient scales by normalising the data. Another way of artificially increasing the coefficient values is to increase the target parameter's value. Similar to the normalisation, the coefficients have to increase, assuming the inputs are kept the same, when the target is enlarged, to ensure the model obtains the larger target value.

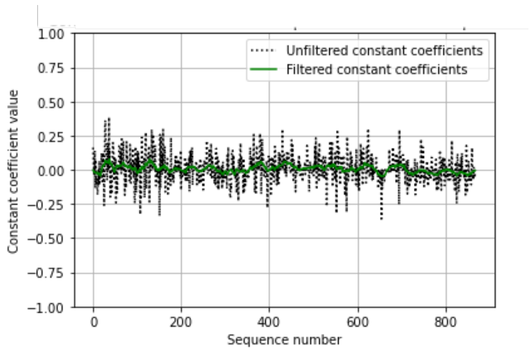
Normalisation is first tested to try to improve the model consistency, as it is a fairly standard data handling procedure and has been shown to work well for many different machine learning models. The entire dataset is normalised except for the target variable, namely the fouling factor. The normalisation method used is the z-normalisation method and the formula can be seen in equation 4.4

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

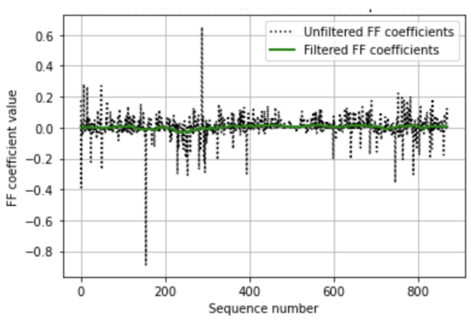
where $\bar{x}$ refers to the mean of the parameters being normalised and σ to the standard deviation. Normalising the data has the additional benefit of forcing the inputs to all lie within the same range. This allows for the coefficients of each input to be within the same range, which could further improve model consistency. Once the data has been normalised using equation 4.4, the same procedure is followed for setting up the SINDy algorithm, as was done previously. The finite difference method is used, a polynomial feature library is chosen and the STLSQ optimizer is once again used. The threshold is optimized for each sequence individually to ensure optimal results are achieved. For comparative purposes, the results of the coefficient development, for the model with the flue gas temperature difference as input before and after the normalisation of the dataset, are shown in figure 4.17. This model input type is chosen as it was the one that showed the most distinct results.



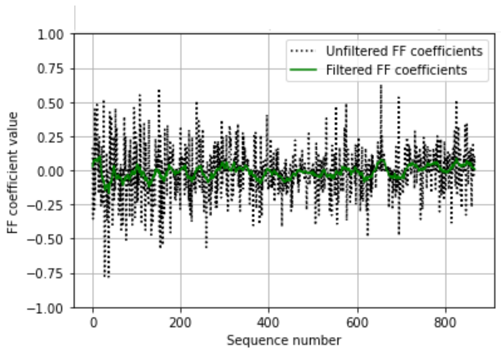
(a) Constant coefficient (No normalisation)



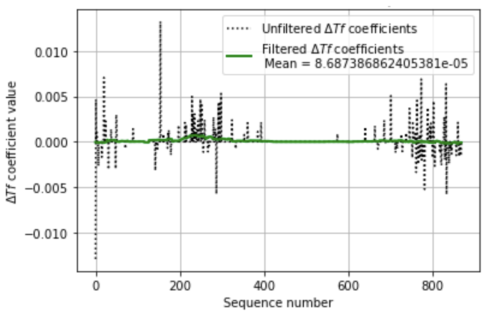
(b) Constant coefficient (With normalisation)



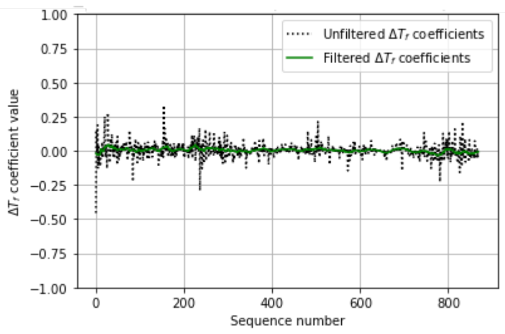
(c) FF coefficient (No normalisation)



(d) FF coefficient (With normalisation)



(e) ΔT_f coefficient (No normalisation)



(f) ΔT_f coefficient (With normalisation)

Figure 4.17: Coefficient development for models with ΔT_f as input when the dataset is normalised vs when the dataset is kept as is.

It can be seen in figure 4.17 that normalising has not drastically improved the model consistency. The constant coefficients have slightly improved, as there are fewer spikes visible in the coefficient development. However, the fouling factor coefficient has become more chaotic once the data has been normalised. What can be noted is that the ΔT_f coefficient values have become much larger as a result of the normalisation. This was expected, as previously the flue gas input values were larger than the other inputs and hence the coefficient had to be small to compensate. With normalisation, the input values are more on par and hence the coefficient was increased. The normalisation, therefore, managed to scale the model coefficients, however, it did not improve model consistency. Normalisation alone is, therefore, not enough to make the models more consistent and to resolve the inconsistent soot blowing sequence data problem.

A more aggressive method of scaling the model coefficients is thus tested, where the target variable is scaled directly. This ensures that the model coefficients have to become larger as well to obtain the larger fouling factor values. The larger model coefficients might then be in a range of values that enables SINDy to more effectively distinguish between respective soot blowing curves, should the model coefficient ranges be the problem.

To set up this experiment, the fouling factor was scaled to different values to determine the effect that scaling has on the coefficient distributions. The scales used were 0.1, 1, 10, 100, 1 000 and 10 000. The SINDy algorithm was set up, as has been done in previous experiments. These scaled fouling factor values were sent through the SINDy algorithm for the different input type models and the coefficient distributions were noted for each fouling factor scale. The distributions of the coefficients are plotted in figures 4.18, 4.19, 4.20 and 4.21, as opposed to the development curves seen before. This allows one to easily compare whether any true change in the coefficient distribution has occurred.

To ensure that the distributions are comparable in terms of their value ranges, the coefficient values are divided by the scaling factor that was used in each case. For example, the constant coefficient values, found when the fouling factor was scaled by 100, are divided by 100 to ensure that the distribution range of these coefficients are in the same range as the coefficients when the fouling factor was not scaled. This ensures that the variance of the coefficients is directly comparable, as their values are within the same order of magnitude. This makes it easier to see whether a specific scaling value reduced the variance in the coefficients significantly. The division of the coefficient by their scaled value is done for all distributions except the fouling factor coefficient distribution, since it was seen that this was already built into the coefficients as the fouling factor is scaled directly. The Gaussian curve fit of the coefficient distribution is also shown for each scaling value.

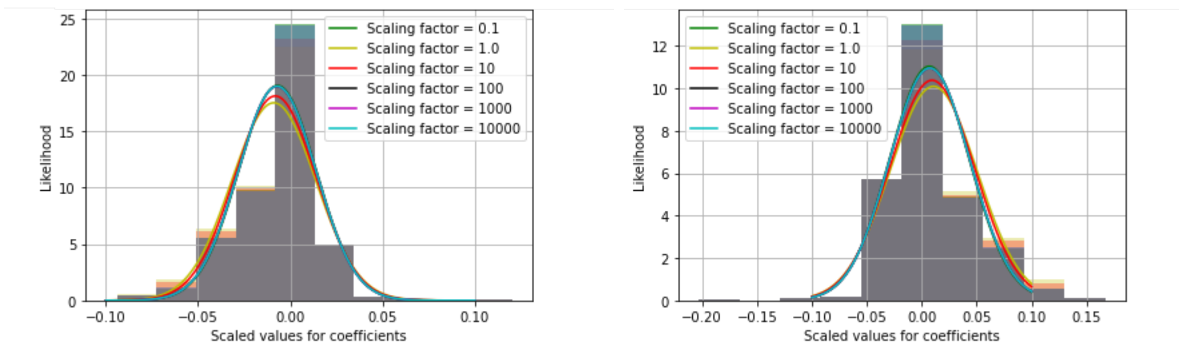


Figure 4.18: FF input model coefficient distributions for different scaling values.

Figure 4.18 shows the coefficient distributions for the model that takes only the fouling factor as input. There is no significant change in the coefficient distributions when the fouling factor is scaled. The variances for all the scaling values are very similar and do not change significantly. This shows that the coefficients have not become more consistent and are still fluctuating as before.

Figure 4.19 shows the same trends for the model that takes the flue gas temperature difference into account. The coefficient distributions change little with scaling and the coefficient values are merely adjusted higher or lower, depending on the scaling factor used. Any variations in the distributions are simply individual sequences that obtain slightly different values, however the general mean and variance of the coefficient distributions do not change, showing that no improvement is being seen regarding model consistency.

Figures 4.20 and 4.21 show the coefficient distributions for the remaining two model input types. It can be seen in Figure 4.20 that most of the variances of the coefficients are still the same, despite scaling being performed. The very large variance seen in the constant coefficient distribution for this model input type, is because this coefficient is sometimes extracted to be extremely large values, even when the majority of values are still 0. This model therefore clearly does not seem to work well despite the data being scaled or adjusted and mostly trivial or extreme values for the model coefficients are observed.

The final LMTD input type model shows some variation in the model coefficient distributions, however the change from one scaling value to the next is very slight. In most cases, the model with no scaling (or a scaling value of 1) has the least variance, therefore proving that the scaling did not have a positive impact on the model consistency.

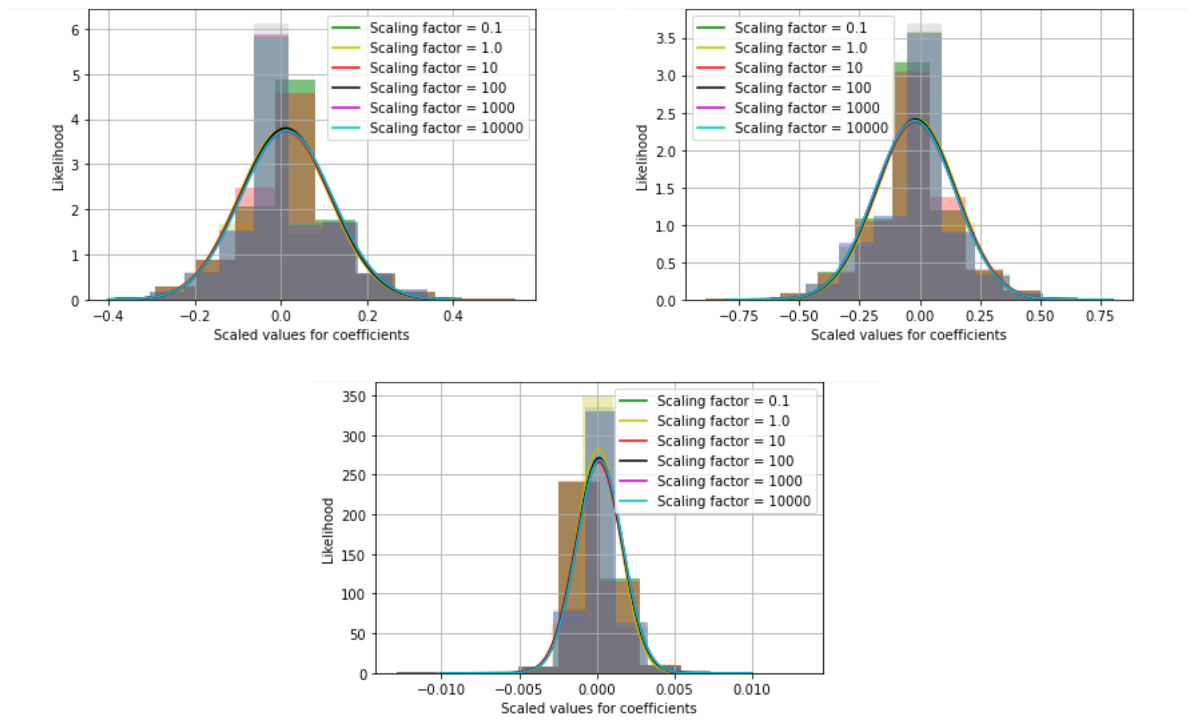


Figure 4.19: ΔT_f input model coefficient distributions for different scaling values.

To summarise, the normalisation of the dataset and scaling of the fouling factor has not improved the model consistency and, in some cases, made it worse. While the verification problems in chapter 3 showed that certain coefficient ranges could make it difficult to distinguish between the respective soot blowing curves, these experiments prove to some extent, that it is likely not the cause of the coefficient fluctuations for the Ngodwana dataset. A few methods have been tested to try and circumvent the inconsistent data with little success, and the other possible causes of coefficient fluctuation have been ruled out. Hence, the most likely cause of the severe coefficient fluctuations is due to inconsistent data. The curve shapes and initial values in the dataset have been seen to vary significantly and it was shown in the verification problems that this would likely cause significant coefficient fluctuation. To test whether this is definitively the problem, an experiment is set up where soot blowing curves are handpicked to have the same general shape and initial values. Their respective extracted models are compared, as it is then expected that they should have similar equations.

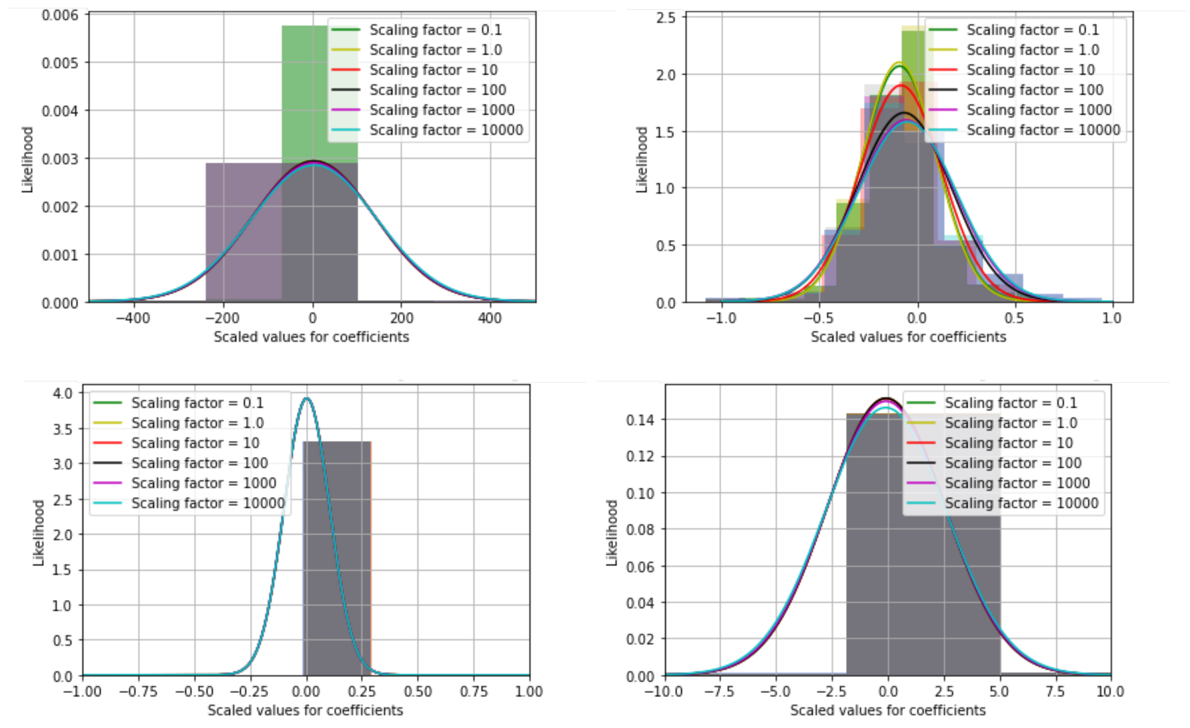


Figure 4.20: ΔT_f and ΔT_s input model coefficient distributions for different scaling values.

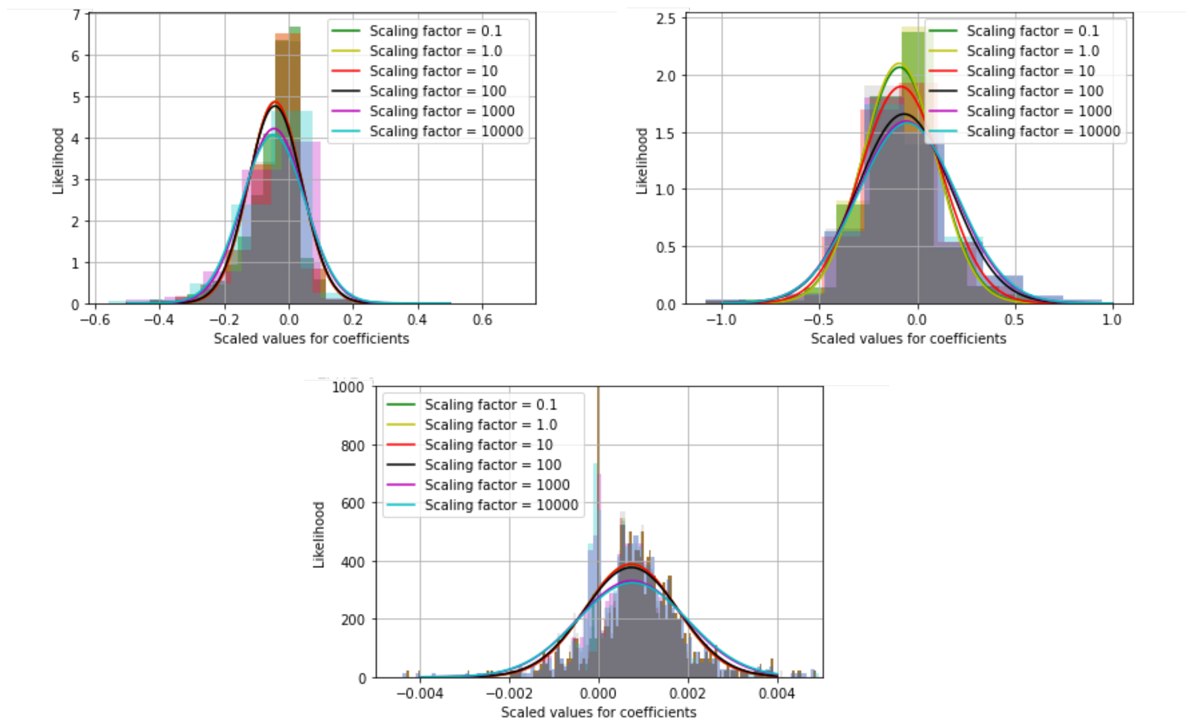


Figure 4.21: LMTD input model coefficient distributions for different scaling values.

4.7 Manually selecting soot blowing sequences for model extraction

Thus far, it has been seen that significant variation in soot blowing curves and their initial values are most likely the cause of the coefficient fluctuation being seen when the SINDy algorithm is applied to all the soot blowing sequences in the Ngodwana dataset. To determine whether one would have to more carefully select soot blowing curves to obtain interpretable answers, sequences will be selected from the dataset manually. These sequences will be selected to have the same general curve shape and initial values. The SINDy algorithm will be applied to these sequences and their coefficients will be compared, since it is expected that they would have similar model equations and coefficients if the soot blowing sequences are not combined as was shown in Chapter 3 in the final experiment.

To select the soot blowing sequences for this experiment, it was noted which curve shapes and initial values occurred frequently in the dataset. It was seen that a curve starting at a fouling level of approximately 0.53 and sloping downwards towards 0.37 was fairly common in this dataset. From the first 100 sequences in the dataset, 10 sequences were selected that had similar shapes and initial values. The 10 selected sequences can be seen in figure 4.22. These sequences still differ from one another despite being carefully selected. This gives one an indication of the complex nature of the Ngodwana dataset as even the 10 most similar curves, out of the first 100 sequences, have significant variation in their shapes. It is therefore expected that the coefficient values of the extracted models will still fluctuate somewhat, however, the fluctuation should be less than what was seen in previous SINDy implementations.

The SINDy algorithm setup is the same as has been used in the study so far. The threshold parameter is set to 0.0001 as this value was seen to work well for the selected sequences and resulted in the most accurate models. Different input type models are tested. A model type with only the FF as input is tested as well as models with the flue gas temperature difference and the LMTD as respective inputs. These models are chosen, as they are the best performing and most consistent model types thus far. For each selected sequence, the best fitting model of each input type is extracted and plotted together with the original sequence. Figure 4.23 shows the extracted models when only the fouling factor is used, while figures 4.24 and 4.25 shows the extracted models when the flue gas temperature difference and the LMTD are added respectively.

When investigating figure 4.23 it can be seen that the model coefficients still fluctuate, which is expected, since the soot blowing curves are not the same. The model coefficients

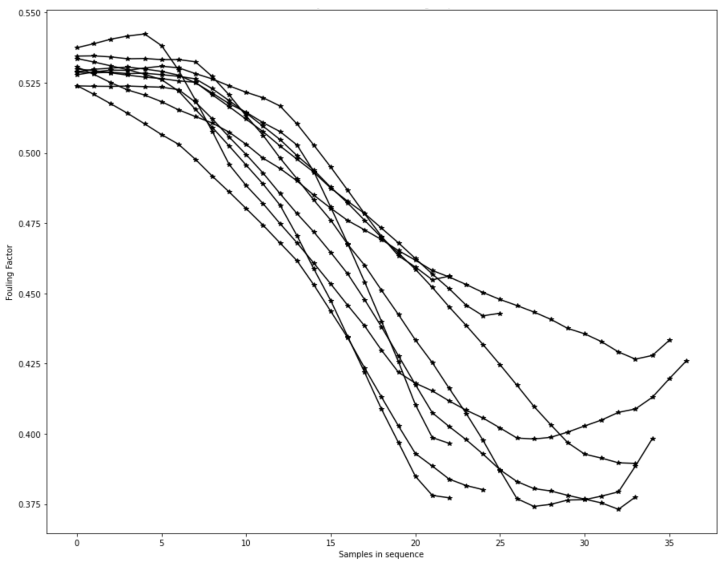


Figure 4.22: Manually selected soot blowing curves with similar shapes and initial values.

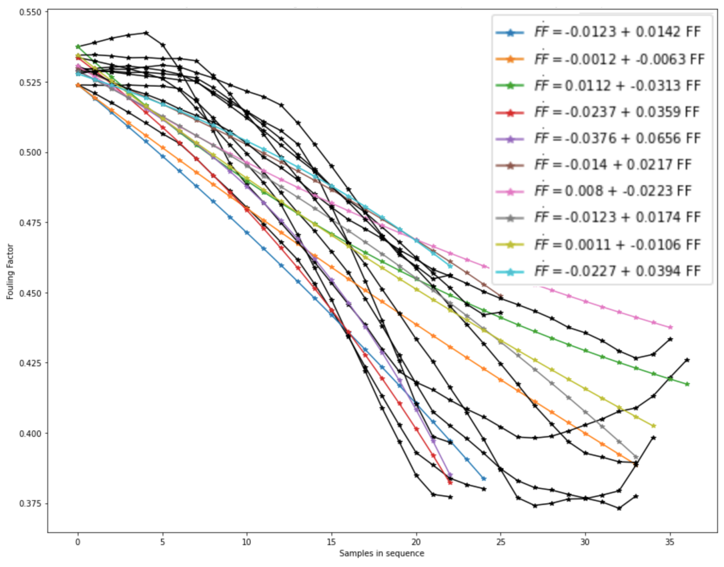


Figure 4.23: Manually selected soot blowing curves and the extracted SINDy models when only the FF is used as input.

are, however, in similar ranges and are comparable to one another. There are a few exceptions in the extracted models where a curve shape that is concave upwards is extracted, instead of the usual concave downwards shape. This shows that even the slightest variation in curve shape could result in significant model coefficient fluctuation and explains why no consistent models are extracted when the SINDy algorithm is applied to the dataset of sequences as a whole.

Another explanation for the coefficient fluctuation, could be the fact that more than one soot blower pair's influence is being seen for some of these curves. It was shown in the final verification problem in chapter 3, that curves may look similar but have significantly different underlying models, especially when it is a combination of models. This may be why some models that are extracted that do not conform to the general underlying model trend for these sequences. The chance is slim, however, since the sequences are selected to be as isolated as possible, but influences from nearby sections in the boiler could result in a combination of soot blower models that has not been accounted for in this dataset. The same trends can be seen in the other model input types in figures 4.24 and 4.25, where there are also significant model differences due to slight variations in the soot blowing curves or potential soot blower influence combinations.

When investigating the respective measurement data curves, some curves start to curve upwards again, indicating an increase in the fouling factor. This is an unexpected result, since the fouling factor should not increase when a soot blower pair is active. A possible explanation for this, could be the fact that there is an error in the limit switch readings that indicate whether a soot blower pair is active or not. The soot blower pair may already have finished the soot blowing procedure, but the limit switch might not have engaged properly. Therefore, the switch reading that is sent to the DCS system, still indicates that the soot blower is active, while it has finished the soot blowing procedure already. This would also explain why the soot blowing sequences are not all the same length and why some sequences stop after 25 time steps but others continue past 30. Before this theory can be confirmed, however, one would have to investigate this at the plant itself, which is not currently possible. It should however be considered in future research regarding this topic, since more consistent sequence lengths, as well as sequences where the fouling factor does not start to rise at the end, will most likely make the extracted SINDy models more consistent. Once again the complexity of the dataset that is being worked with is highlighted as there are likely errors like this present.

To truly determine whether manually selecting soot blowing curves has improved model consistency, one must plot the coefficient development for the different curves for each

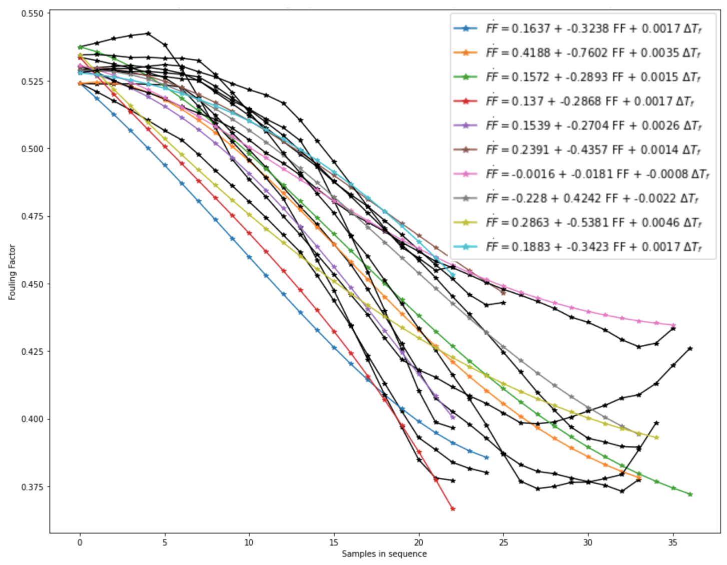


Figure 4.24: Manually selected soot blowing curves and the extracted SINDy models when the FF and flue gas temperature difference are used as inputs.

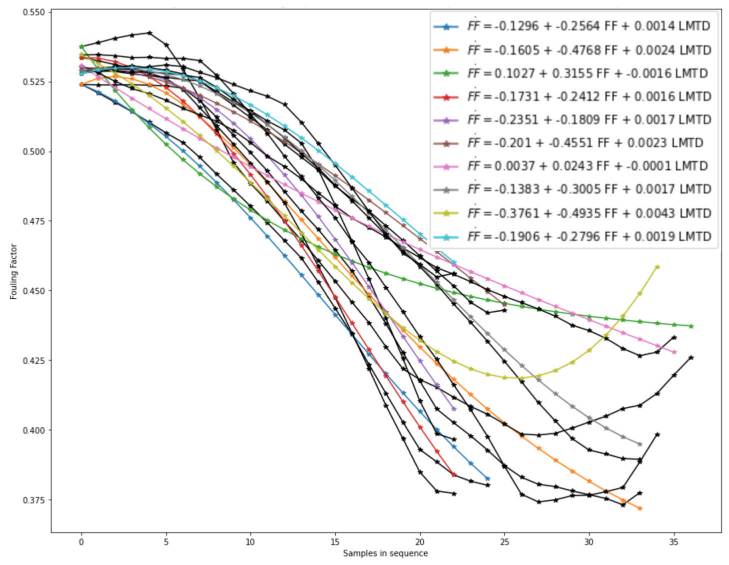
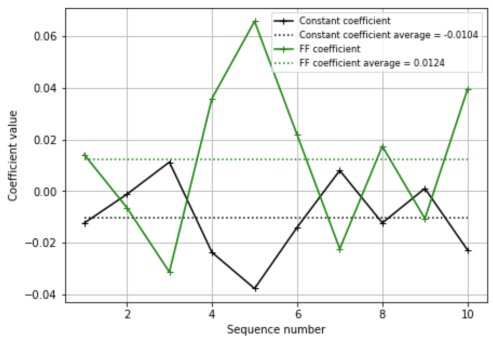
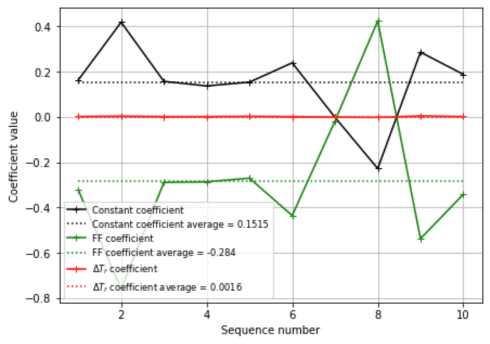


Figure 4.25: Manually selected soot blowing curves and the extracted SINDy models when the FF and LMTD are used as inputs.

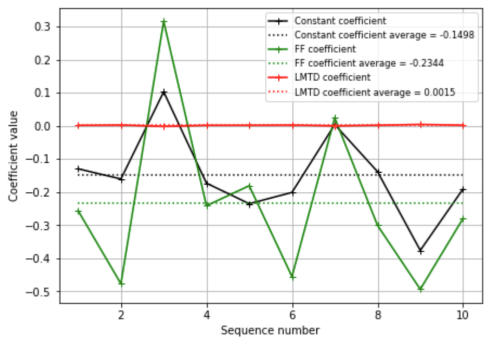
model type. If the coefficients fluctuate about zero, as was often the case with the entire dataset being fitted, the manual selection did not improve consistency. If, however, the coefficients generally fluctuate about a a non-zero point, one can extract a mean underlying model that averages all the models' coefficients, which has not been possible before. Figures 4.26 shows the coefficient development of these ten selected sequences for the different input type models.



(a) FF only input model



(b) ΔT_f input model



(c) LMTD input model

Figure 4.26: Model coefficient development for manually selected soot blowing sequences for different input type models.

Most of the model coefficients are consistently either negative or positive with the average values of the ten coefficients proving this. There are some cases where the coefficients jump from a usual positive value to a negative or vice versa, however, it is much less frequent than it was when the SINDy algorithm is applied to the entire Ngodwana dataset sequences. These outlier coefficients once again illustrate the complexity of the problem we are dealing with, as curves that look visually similar, with only slight shape and initial value variations, may extract completely different coefficients. This fluctuation may be due to soot bowers in other sections of the boiler combining their influence with this specific soot blower pair's influence as mentioned before,or simply due to too much variation in the data. It is also interesting to note that the constant and fouling factor coefficient development plots seem to resemble mirror images. If one coefficient is more negative the other is more positive and vice versa. This indicates that there is a possible correlation between the model coefficients. The only model input type that does not have this is the LMTD model where the these two coefficients rather seem to follow one another. If one rises the other does as well which also indicates that there is a correlation between them. This correlation may be a result of the SINDy model extracting some underlying physics coupling between the two terms. Alternatively it could simply be, because the curves are similar to one another and hence the coefficients relate to each other in similar ways.

It can be confirmed that even though the soot blow curves are similar in shape and initial values, the slight variation present, results in coefficient fluctuation regardless. There does seem to be some improvement in consistency as one can find average coefficient values that are non-trivial, where it was not possible before with the entire dataset. The average models for the entire dataset were often very close to zero or zero and did not fit the sequences well. This result in itself is therefore already a very positive finding and indicates that sequence filtering is another step in the right direction.

The average models found for this experiment was simulated and plotted together with the sequences that were selected and can be seen in figure 4.27. It can be seen that the average model of the FF only input models, fits the selected sequences fairly well and captures the trend of the soot blowing curves. The other input models, however, do not fit the sequences well at all. Both the ΔT_f and LMTD input models start to diverge upward instead of following a downwards trend. This shows that the more complex models are very sensitive to coefficient variations (which may be due to underlying variations in the physics) and to find a working average model, the coefficients should be significantly more consistent. The simpler fouling factor input model seems to work well when averaged and the model would make fairly accurate predictions. It makes sense that the less complex model is the more stable one, as it was already seen in chapter 3 that the complex models experienced more coefficient fluctuation when conditions regarding the soot blowing curve changed, while the simpler models were more robust to slight variations in the data.

This experiment has thus illustrated the complexity of the Ngodwana dataset and shown, that if sequences are manually selected, one can start to see a slight improvement in model consistency. Further, this experiment has shown that the dataset would have to be investigated carefully, to identify potential errors in measurement sensors. Carefully filtering sequences and double-checking sensor readings should allow one to find more general and consistent models.

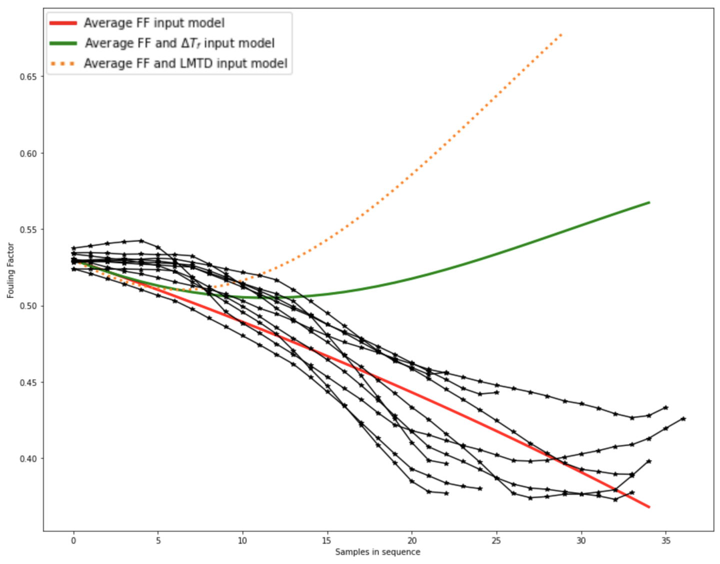


Figure 4.27: Manually selected soot blowing curves and the average model of the extracted SINDy models for the different input types.

Even though the results are not as clear cut as one would want, the overall study has proven that the SINDy algorithm is more than capable of extracting interpretable models, which would be excellent for generalisation outside the domain of the training data. The study has also shown how one would have to treat future measurement datasets (in terms of pre-processing) to obtain the results necessary to make robust predictions. The final experiment has shown that a promising direction for future research would be, to further subdivide the data into collections of similar soot blowing curves and fit the SINDy algorithm to them. However, one should keep in mind that one must then find a way to determine when certain models must be used and how to switch between the different models during the boiler life cycle. The work in Chapter 4 has highlighted which measurement sensors and inputs are important, to estimate the fouling in a boiler, and it has been shown how these measurements improve the SINDy algorithm's ability to extract models, that can predict the fouling factor development during soot blowing, accurately. This concludes the work done for this study.

CHAPTER 5

Conclusion and recommendations

5.1 Conclusion

In this dissertation, a methodology is proposed that attempts to extract the underlying physics from measurement data in, a Kraft recovery boiler, at the SAPPI Ngodwana paper and pulp mill. The recovery boiler at the plant is prone to significant soot accumulation on the heat transfer surfaces, which eventually leads to plugging. This results in frequent shutdowns and water washes being required at the plant, which wastes production time and thus money. The proposed method, incorporated the sparse identification of non-linear system dynamics (SINDy) algorithm, to discover the underlying boiler physics using basic measurement data. The extracted models could potentially be used for the prediction of the fouling change during soot blowing. This would allow one to potentially optimise the soot blowing schedule in the boiler to increase the efficiency and to ensure that fewer water washes are needed per year. This would save a lot of money in the long run for the company.

A method of estimating the amount of fouling in the boiler was developed first, where simple measurements, such as temperature and pressure, as well as fixed boiler design parameters, were used to build a thermodynamic model. This thermodynamic model was able to obtain an approximate level of fouling in the boiler, where previously, no such metric existed. This was already a big step towards soot blowing optimisation at the plant. It was shown that the thermodynamic model performed very well, as the fouling factor decreased when a soot blower was activated and increased when no soot blowers in specific sections were active. This confirmed that the thermodynamic model's fouling factor values were plausible and could be applied with relative confidence to machine learning models. The complexity of the dataset that would be used, was also highlighted with the thermodynamic model to some extent. The labelled fouling factor dataset could subsequently be used to build different predictive models and be used specifically in this study for the SINDy algorithm model extraction. Once the fouling factor was calculated for approximately 4 months, corresponding to the boiler life cycle after it had just been washed until it was completely fouled, an initial investigation with the SINDy algorithm was performed.

A verification problem was set up to test the algorithm's ability to identify a valid physics model, given only the fouling factor as input. Once it was confirmed that the algorithm could identify interpretable models from basic data, the SINDy algorithm was briefly applied to the real measurement dataset to establish basic model forms for use in additional verification problems. The verification problem that followed, tested the influence of the model coefficients on the shape of the predicted soot blowing curves, for models that were typically extracted from the real measurement dataset. Different polynomial order models were tested to also determine which polynomial order model was more robust and which was more accurate. The experiment showed that the first order polynomial model seemed to be both accurate and robust and there were no obvious coefficient value ranges in which the identifiability of the models was obscured.

Subsequently, the different polynomial models were used to generate artificial soot blowing sequences. These sequences were used to test the SINDy algorithm's recovery ability. The SINDy algorithm's model recovery was tested for different noise conditions as well as conditions where the sequence's initial values changed. The experiments showed that the algorithm was sensitive to noise, as the extracted model coefficients started to fluctuate significantly when more than 5% noise was present. Changes in the initial values while the curve shapes remained the same also proved to result in different model equations. Furthermore, the verification experiments proved that different curve shapes resulted in notably different model equations, and that at least 15 measurement samples were needed to extract consistent models when low levels of noise are present. In the final verification problem, lower order models were fitted to data that was generated with higher order models. It was shown that the resulting models did not fit the data optimally, however, the results were still interpretable and could potentially be used for broad optimisation of the soot blowing schedule.

Once the verification problems were completed and the SINDy algorithm's capabilities were determined, the algorithm was implemented on the Ngodwana measurement dataset. Initially the optimal polynomial order was once again investigated, while keeping in mind the verification problems' results. The optimal polynomial order was shown to be a first order model. Once the model order was determined, a baseline algorithm setup was established and simple physics models were extracted from the dataset. These models described the fouling factor development during a soot blowing sequence for a specific soot blower pair in a section of the boiler. These models had fairly good predictive capabilities despite their simplicity and were very interpretable. This was a positive result that indicated that the SINDy algorithm was more than capable of extracting plausible models from the measurement dataset that could be used to discover the underlying boiler physics. It was noted, however, that the models that were extracted were not particularly consistent, since their coefficients fluctuated significantly, depending on which soot blowing sequence the SINDy algorithm was applied to. This made sense as the soot blowing curves had different initial values and shapes, which was shown in the verification problems to cause fluctuations in the coefficients of extracted models.

In an attempt to circumvent the inconsistency in the data, additional inputs were given to the SINDy algorithm, to increase the complexity of the models that are extracted and hence, potentially improve model coefficient consistency. Inputs that were added included the flue gas temperature difference, the steam temperature difference and lastly, the log mean temperature difference for a specific section in the boiler. It was seen that the additional inputs increased the prediction accuracy of the extracted models as a result of added complexity. The experiments therefore, also highlighted which sensor measurements were important for the prediction of the fouling factor and how they interacted with the thermodynamic model's fouling factor estimation. The additional inputs also increased the consistency of the models that were extracted slightly and made the coefficients interpretable in some cases, however, fluctuations were still too severe to extract more general models.

To try and improve model consistency further, the optimizer algorithm's threshold parameter was investigated. The threshold value was optimised for each data sample sequence individually to allow the algorithm to extract the best fitting model possible for each sequence. This improved the model prediction accuracy once again resulting in very accurate predictions regarding the fouling factor curve. This was a positive result as it illustrated that the SINDy algorithm could extract very accurate models from relatively complex measurement data. The consistency of the models was not improved much by optimising the threshold however.

Data manipulation was hence investigated, to once more try and improve model consistency. The data was first normalised before the fouling factor was scaled. Both attempts, however, proved to have no significant effect on the model consistency as there was simply too much variation in the data. Thus, it was decided that sequences with similar curve shapes and initial values, would be selected manually in an attempt to introduce some consistency into the data being sent to the SINDy algorithm. 10 Sequences, out of the first 100 soot blowing sequences for soot blowers 3 and 4 were selected. It was shown that manually selecting the sequences did indeed improve model consistency somewhat. There were still coefficient fluctuations present, however, they were less sporadic compared to applying the algorithm to the entire dataset and an average model could be extracted for the case where only the fouling factor was used as input to the SINDy algorithm. The more complex models, with additional inputs, however, still had too much coefficient fluctuation to extract a working average model with. The experiment also highlighted that one would have to investigate potential errors in the sensor readings at the plant, and proved that careful filtering and selection of sequences might result in very consistent models, that one would be able to use for fouling factor development prediction and soot blowing optimisation.

Overall, the study has enabled one to estimate the fouling factor in the boiler, which had not been possible before and is a very useful metric for different machine learning applications. The study has also shown, that the SINDy algorithm is a valuable tool for the extraction of simple and interpretable physics models from measurement data, and that these models can be used to accurately predict a soot blowing procedure's fouling factor development. Finally, the study has shown how one would have to treat future measurement datasets from this boiler, to extract not only accurate, but consistent physics models and has highlighted where potential areas of focus should be regarding sensor measurement errors and sequence filtering. Overall the study has laid a strong foundation for future research regarding this topic and has ironed out some of the potential kinks one would encounter.

5.2 Recommendations

The following recommendations are made for future work

- Data consistency should be investigated, since it severely affects the SINDy algorithm's recovery ability when there is too much variation in the data. Thus methods of filtering the data, selecting sequences and improving consistency should be investigated as well as methods of identifying faulty measurement sensors.

- Different feature library functions for the SINDy algorithm need to be investigated as there could be non-linear basis functions that better describe the fouling accumulation process.

- Data augmentation, in regions where sensor measurements are scarce, need be investigated to separate the boiler into smaller sections, where the physics should be less likely to fluctuate. This will also improve the SINDy algorithms extraction capability.

- In-depth boiler physics experiments should be conducted, for example with CFD simulations, to better understand how the respective soot blowers influence each other and to determine for which sequences of a soot blower pair one would expect similar physics equations. A better understanding of boiler physics could also improve the thermodynamic model used in this study.

- Investigate the possibility of combining the SINDy algorithm with more traditional and complex machine learning algorithms, to generate models that are not only accurate and complex, but also interpretable.

Bibliography

Anitha Kumari, S. & Srinivasan, S. (2019), 'Ash fouling monitoring and soot-blow optimization for reheater in thermal power plant', *Applied Thermal Engineering* **149**(November 2018), 62–72.
URL: *https://doi.org/10.1016/j.applthermaleng.2018.12.031*

Bernath, P., Sinquefield, S. A., Baxter, L. L., Sclippa, G., Rohlfing, C. M. & Barfield, M. (1998), 'In situ analysis of ash deposits from black liquor combustion', *Vibrational Spectroscopy* **16**(2), 95–103.

Brunton, S. L., Proctor, J. L., Kutz, J. N. & Bialek, W. (2016), 'Discovering governing equations from data by sparse identification of nonlinear dynamical systems', *Proceedings of the National Academy of Sciences of the United States of America* **113**(15), 3932–3937.

Cengel, Y. A. & Ghajar, A. J. (2015), *Heat ad Mass Transfer Fundamentals Applications*, 5th edition edn, McGraw-Hill, New York.

Chartrand, R. (2011), 'Numerical Differentiation of Noisy, Nonsmooth Data', *ISRN Applied Mathematics* **2011**, 164564.
URL: *https://doi.org/10.5402/2011/164564*

Cioffi, R., Travaglioni, M., Piscitelli, G., Petrillo, A. & De Felice, F. (2020), 'Artificial Intelligence and Machine Learning Applications in Smart Production: Progress, Trends, and Directions', *Sustainability* **12**(2).
URL: *https://www.mdpi.com/2071-1050/12/2/492*

de Silva, B., Champion, K., Quade, M., Loiseau, J.-C., Kutz, J. & Brunton, S. (2020), 'PySINDy: A Python package for the sparse identification of nonlinear dynamical systems from data', *Journal of Open Source Software* **5**(49), 2104.

Gunasekaran, A. & Spalanzani, A. (2012), 'Sustainability of manufacturing and services: Investigations for research and applications', *International Journal of Production Economics* **140**(1), 35–47.
URL: *http://dx.doi.org/10.1016/j.ijpe.2011.05.011*

Kaiser, E., Kutz, J. N. & Brunton, S. L. (2018), 'Sparse identification of nonlinear dynamics for model predictive control in the low-data limit', *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **474**(2219).

Keith Mills, K. M. P. (2019), 'Artificial intelligence to drive intelligent production'.
URL: *https://metrology.news/artificial-intelligence-to-drive-intelligent-production/*

Kumar, S. (2020), '7 ways to handle missing values in machine learning'.
URL: *https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e*

Leppänen, A., Tran, H., Taipale, R., Välimäki, E. & Oksanen, A. (2014), 'Numerical modeling of fine particle and deposit formation in a recovery boiler', *Fuel* **129**, 45–53.

Mishra, S. (2017), 'Unsupervised learning and data clustering'.
URL: *https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a*

Parente, A. P., de Souza, M. B., Valdman, A. & Mattos Folly, R. O. (2019), 'Data augmentation applied to machine learning-based monitoring of a pulp and paper process', *Processes* **7**(12).

Peña, B., Teruel, E. & Díez, L. I. (2011), 'Soft-computing models for soot-blowing optimization in coal-fired utility boilers', *Applied Soft Computing Journal* **11**(2), 1657–1668.

Peña, B., Teruel, E. & Díez, L. I. (2013), 'Towards soot-blowing optimization in superheaters', *Applied Thermal Engineering* **61**(2), 737–746.
URL: *http://dx.doi.org/10.1016/j.applthermaleng.2013.08.047*

Prem, S., Ayyagari, K., Singh, R., Purakasthya, P. & Deeskow, P. (2014), 'Estimation of soot deposition in the boilers of coal fired power plant', (1), 1–4.

Radhakrishnan, V. R., Ramasamy, M., Zabiri, H., Do Thanh, V., Tahir, N. M., Mukhtar, H., Hamdi, M. R. & Ramli, N. (2007), 'Heat exchanger fouling model and preventive maintenance scheduling tool', *Applied Thermal Engineering* **27**(17-18), 2791–2802.

Ramirez, B. A. (2000), 'Determination of Cleanliness Status of Boiler Heat Transfer Surfaces by Using Artificial Intelligent techniques', (Theses and Dissertations, Lehigh University).
URL: *https://preserve.lehigh.edu/cgi/viewcontent.cgi?article=1671context=etd*

Robbins, B. (2017), ‘Machine learning: How black is this beautiful black box’.
URL: *https://towardsdatascience.com/machine-learning-how-black-is-this-black-box-f11e4031fdf*

Rudin, C. (2019), ‘Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead’, *Nature Machine Intelligence* **1**(5), 206–215.
URL: *https://doi.org/10.1038/s42256-019-0048-x*

Sharpe, P. K. & Solly, R. J. (1995), ‘Dealing with missing values in neural network-based diagnostic systems’, *Neural Computing Applications* **3**(2), 73–77.
URL: *https://doi.org/10.1007/BF01421959*

Shi, Y., Li, Q., Wen, J., Cui, F., Pang, X., Jia, J., Zeng, J. & Wang, J. (2019), ‘Soot blowing optimization for frequency in economizers to improve boiler performance in coal-fired power plant’, *Energies* **12**(15).

Shi, Y., Wang, J. & Liu, Z. (2015), ‘On-line monitoring of ash fouling and soot-blowing optimization for convective heat exchanger in coal-fired power plant boiler’, *Applied Thermal Engineering* **78**, 39–50.
URL: *http://dx.doi.org/10.1016/j.applthermaleng.2014.12.002*

Shuiguang Tong, X. Z. (2019), ‘Online Ash Fouling Prediction for Boiler Heating’, (July 2017).

Smrekar, J., Assadi, M., Fast, M., Kuštrin, I. & De, S. (2009), ‘Development of artificial neural network model for a coal-fired boiler using real plant data’, *Energy* **34**(2), 144–152.

Teruel, E., Cortés, C., Ignacio Díez, L. & Arauzo, I. (2005), ‘Monitoring and prediction of fouling in coal-fired utility boilers using neural networks’, *Chemical Engineering Science* **60**(18), 5035–5048.

Tran, H. (2007), ‘Recovery boiler fireside deposits and plugging prevention’, *TAPPI Kraft Recovery Course 2007* **2**, 537–572.

Verweij, G. & Rao, A. (2017), ‘Sizing the prize: What’s the real value of AI for your business and how can you capitalise?’, *PwC* p. 32.

Wassiliadis, N., Adermann, J., Frericks, A., Pak, M., Reiter, C., Lohmann, B. & Lienkamp, M. (2018), ‘Revisiting the dual extended Kalman filter for battery state-of-charge and state-of-health estimation: A use-case life cycle analysis’, *Journal of Energy Storage* **19**(June), 73–87.
URL: *https://doi.org/10.1016/j.est.2018.07.006*

Wiley, K. (2020), 'UC Berkeley UC Berkeley Electronic Theses and Dissertations', *DNA Mediated Assembly of Protein Heterodimers on Membrane Surfaces* p. 67.
URL: *https://escholarship.org/uc/item/98384265*

Zhang, X., Yuan, J., Chen, Z., Tian, Z. & Wang, J. (2018), 'A dynamic heat transfer model to estimate the flue gas temperature in the horizontal flue of the coal-fired utility boiler', *Applied Thermal Engineering* **135**(February), 368–378.
URL: *https://doi.org/10.1016/j.applthermaleng.2018.02.067*