A neural network surrogate modeling routine for simulation-based analysis of ship energy systems

Joonas Riekkinen

School of Science

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 31.7.2023

Supervisor

Prof. Fabricio Oliveira

Advisor

MSc Mika Vuorinen

Aalto University
School of Science

Aalto University
School of Science

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the master's thesis

Author Joonas Riekkinen

Title A neural network surrogate modeling routine for simulation-based analysis of ship energy systems

Degree programme Mathematics and Operations Research

Major Systems and Operations Research **Code of major** SCI3055

Supervisor Prof. Fabricio Oliveira

Advisor MSc Mika Vuorinen

Date 31.7.2023 **Number of pages** 48 **Language** English

Abstract

Simulation can be used as a tool in the design of ship energy systems, but the computational cost of evaluating simulation models can be high, rendering the analysis and optimization of simulation-based designs challenging.

In this thesis, a neural network surrogate modeling routine is developed to facilitate the analysis of ship energy systems simulation models. The routine employs active learning and space-filling sampling algorithms to decrease the number of required simulation model evaluations. The routine is designed to be generalizable and scalable, to enable its use in various settings.

The use of active learning is seen to decrease the number of simulation model evaluations required to train a robust surrogate model, in comparison to a setting, where the model is trained using a single batch. Sampling routine selection is also seen to have an effect on the performance of the surrogate model.

Keywords ship energy systems, simulation-based analysis, surrogate modeling, active learning

Aalto University
School of Science

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Diplomityön tiivistelmä

Tekijä Joonas Riekkinen

Työn nimi Neuroverkkovastepintamalli laivojen energiajärjestelmien simulointipohjaiseen analysointiin

Koulutusohjelma Matematiikka ja Systeemianalyysi

Pääaine Systeemi- ja operaatiotutkimus **Pääaineen koodi** SCI3055

Työn valvoja Prof. Fabricio Oliveira

Työn ohjaaja DI Mika Vuorinen

Päivämäärä 31.7.2023 **Sivumäärä** 48 **Kieli** Englanti

Tiivistelmä

Simulointimallinnusta voidaan hyödyntää laivojen energiajärjestelmien suunnittelussa, mutta simulointimallien evaluointi voi vaatia merkittäviä laskennallisia resursseja, mikä tekee simulointipohjaisesta analyysista ja optimoinnista haasteellista.

Tässä diplomityössä kehitetään neuroverkkovastepintamalli laivan energiajärjestelmien analysoimisen tueksi. Mallin kehittämisessä hyödynnetään aktiivista oppimista ja tilaa täyttäviä otanta-algoritmeja simulointimallin evaluointikertojen minimoimiseksi. Vastepintamallista pyritään tekemään yleistetty ja skaalautuva, jotta sitä voitaisiin hyödyntää mahdollisimman monessa asetelmassa.

Aktiivisen oppimisen havaitaan laskevan robustin vastepintamallin kouluttamiseen vaadittavaa aineistomäärää verrattuna tilanteeseen, jossa malli koulutetaan käyttäen yksittäistä otantaa. Myös otanta-algoritmin valinnalla todetaan olevan vaikutusta mallin suorituskykyyn.

Avainsanat laivojen energiajärjestelmät, simulointipohjainen analyysi, vastepintamallinnus, aktiivinen oppiminen

Preface

I want to thank Professor Fabricio Oliveira for his guidance and his contribution in the inception of the concept of this thesis. I would also like to thank my advisor Mika Vuorinen, and the rest of my colleagues at Elomatic, for lending their expertize regarding the marine industry, and their support in the thesis writing process.

Otaniemi, 31.7.2023

Joonas Riekkinen

Contents

1 Introduction

Measured by volume, more than 80% of global trade is transported by sea on Trade and Development [2022], and shipping is responsible for around 3% of global greenhouse gas emissions, with the figure estimated to rise in the coming decades King [2022]. As of 2023, the International Maritime Organization (IMO), requires the monitoring of ship energy efficiency for both new and existing ships. The energy efficiency indices, called EEDI and EEXI for new and existing ships respectively, are used to monitor compliance with IMO's energy efficiency standards, which are set to tighten periodically. Shipowners are thus under increasing pressure to improve the energy efficiency of both new and existing vessels.

During operation, a ship's greenhouse gas emissions are mostly created by the burning of fossil fuels. As such, a considerable share of the research in the field of marine engineering focuses on alternative fuels and the efficiency of the main fuel consumers, namely engines. However, optimized energy system design can also yield significant efficiency improvements. The challenges in the optimization of energy system design and operation can be attributed to both the complexity of modern ship energy systems and the dynamic operating conditions and demands of a ship. Simulation is a commonly utilized tool in the design of such complex systems, but the computational cost of running a simulation model and the complex structure of the models can make simulation-based analysis challenging. This thesis aims to develop a proof-of-concept level surrogate modeling routine to assist in efficient simulation-based analysis of ship energy system designs.

Section 2 gives a brief overview of ship energy system design and looks at similar studies. The objectives of the thesis are discussed in Section 3, and an overview of the developed system is presented. Section 4 looks at the mathematical methods used in the study and justifies the use of such tools in the process of ship energy system design. Section 5 describes the implementation and development of the routine using a case study. Section 6 reviews the implementation and discusses possible future development. The results of the thesis are summarized in Section 7.

2 Ship power plant design

2.1 Ships as energy flow systems

From a systems research perspective, a ship can be treated as a set of interconnected components, which consume and/or produce energy to fulfil their individual duties in order to help complete the ship's mission. Two typical missions for a ship are the transportation of goods, and the transportation of passengers. Ships designed for such purposes are referred to as cargo ships and cruise ships, respectively. The composition of a ship's power plant is heavily influenced by its mission. As ships typically spend most of their lifespan operating a single route, the demands of the planned operation dictate the design of the ship and its power plant. The mission of a ship has a major influence on all of its major characteristics, such as hull size and shape, operating velocity, and the location and size of the machinery.

The energy flow onboard a ship is generally divided into three types of energy: mechanical, electrical, and thermal. Mechanical energy is mostly required for propulsion, although in many modern ships propulsion is powered by electric motors. In such a setting, the energy required for propulsion can be viewed as a part of the electrical energy demand. Electrical energy demand used for purposes other than propulsion is referred to as the *hotel load*. The hotel load depends heavily on the mission of the ship: for cargo ships the hotel load tends to be fairly minor apart from some cargo moving operations in ports, whereas for cruise ships the electrical energy used for purposes such as lighting, entertainment, and food preparation can be equal or larger than the energy used for propulsion. Thermal energy is typically required for heating the passenger and crew cabins, providing warm water, and fuel heating. Similarly to electrical energy, thermal energy demand depends heavily on the type of the ship, typically constituting a larger share of the total energy demand on a cruise ship. In two case studies, the distribution of energy consumption between propulsion, electrical power, and heat was determined to be 46-27-27 for a studied cruise ship Baldi et al. [2018], and 70-16-14 for a chemical tanker Baldi et al. [2015].

2.2 Power plant design methods

In comparison to many other seemingly similar systems, such as the power plant of an industrial production line, the design of a ship's power plant is a fairly complicated task. Several factors complicate the design, and especially the optimization of its energy efficiency. The operational conditions of a ship are dynamic and difficult to accurately predict, since sea conditions can have a large influence on the energy demand of an operating ship. The fairly isolated nature of most ship operations also means that external power can be difficult or impossible to obtain, meaning that a ship must contain sufficient power reserves. The energy density of a ship's power source, typically a liquid fuel or a combination of fuels, can have a large influence on the profitability of the operation, especially in the case of passenger ships, since a large fuel tank can decrease the amount of available cabins.

Because ships are typically designed with a specific mission in mind, the system-

level optimization of ship power plants has not been a topic of particularly wide interest. A considerable share of the research regarding marine transportation energy efficiency has instead been focused on specific components, such as engines, since they are mass produced and less mission-dependent than the system design. Recently, the development and analysis of novel energy efficiency and emission reduction technologies, such as waste heat recovery systems Shu et al. [2013], batteries Lan et al. [2015], and alternative fuels Al-Enazi et al. [2021], has been a topic of considerable interest. It has however been estimated that significant efficiency improvements could be achieved with system-level optimization.

Traditionally the design of a ship's power plant has been performed using what is known as the *design point approach*, in which the power producing system is designed to optimize the fuel consumption for the most common operating mode of the ship. For cargo ships, this usually means sailing at open sea with a full cargo load. While the design point approach leads to an optimized consumption for large parts of the operation, the resulting design can be far from optimal when considering the entire life cycle of the ship Frangopoulos [2020]. Including the dynamic nature of the operating conditions and energy demands in the design problem can be achieved in two ways. The more commonly used one utilizes the ship's *operational profile*. In the context of this thesis, the operational profile of a ship can be defined as an estimation of all distinct energy consumption levels and their frequencies during the ship's operation. The other way is using data from a reference vessel, although this approach has been implemented seldomly, according to Baldi [2016].

Baldi [2016] presents a comprehensive analysis of ship energy systems, employing a systems approach and using tools such as energy and exergy analysis. In this work, the energy system of a ship is identified as a *complex system*. Common features of complex systems are a high number of parts, nonlinear and nontrivial interactions, and common objectives, which require the involvement of multiple components Yates [1978]. While there is no consensus regarding a concrete definition of a complex system Ladyman et al. [2013], the aforementioned features can certainly be identified in the power plant of a ship. As such, some general ideas concerning the handling of complex systems can be applied in their design and analysis. A common way of analyzing and optimizing such systems is using computer simulations. Papanikolaou [2019] presents a holistic optimization approach for the design of a ship, and includes several simulation-based design problems. Lappalainen et al. [2019] discusses the utilization of cloud-based computing in simulation-based optimization of ship energy systems, noting that such parallelization of simulations can mitigate the challenges related to the long running times of simulation-based analysis. Trivyza et al. [2018] presents a simulation-based multi-objective optimization methodology for ship energy system design, combining environmental and economic objectives. Trivyza et al. [2019] uses the same methodology to analyze the impact of future carbon pricing on optimal designs, using a scenario-based method. Baldasso et al. [2019] compares linear and nonlinear approximations in the optimization of a ship's power plant layout and unit scheduling against multiple objectives. Both models use the operational profile of a ship's estimated voyage.

On top of the difficulty of efficiently utilizing the operational profile of a ship,

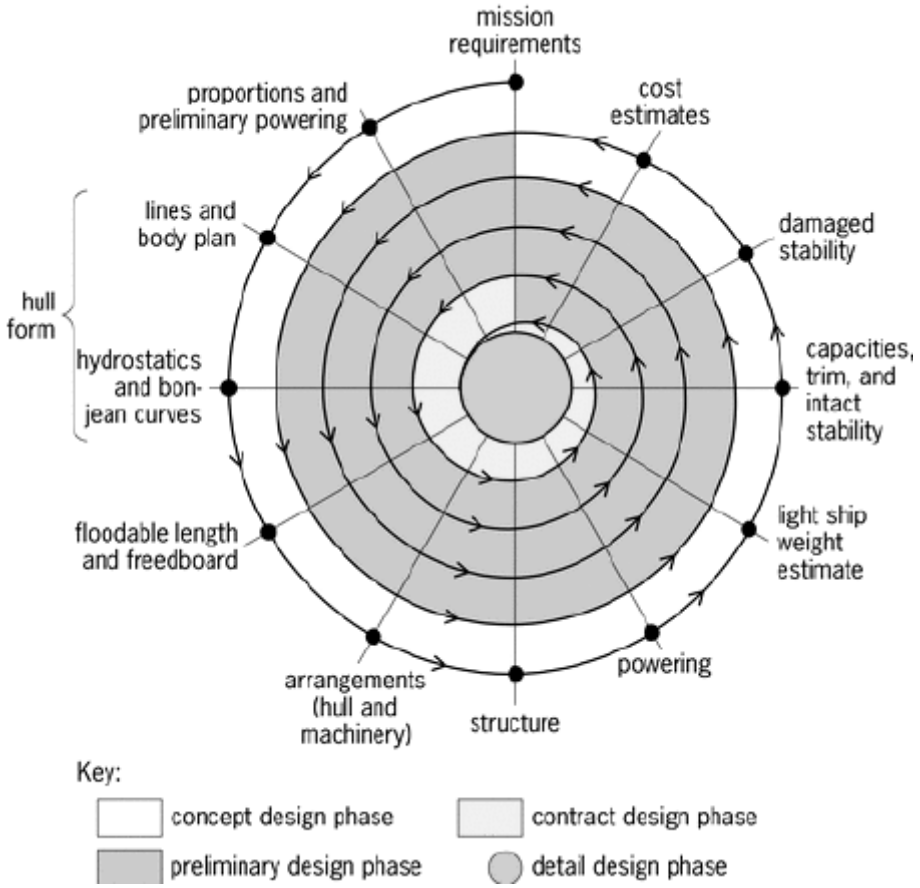

Figure 1: A ship design spiral Evans [1959]

the design point approach has other limiting factors. The effects of uncertainty are often accounted for by increasing the engine sizes by a constant factor, such as 10%, for the final design. While basic stochastic/robust optimization techniques might be applicable, at least when a closed-form representation of the energy demand is used, it can be difficult to include multiple sources of uncertainty, as well as perform proper sensitivity analysis.

In ship design projects, the power plant design needs to fulfil several criteria, and the requirements can change multiple times during the project. As such, it is rarely feasible to only optimize one aspect of the power plant, while ignoring other variables, meaning that the energy system has to be designed holistically. Ship design projects follow an iterative form and consist of several phases Evans [1959], as well as requiring the input of several stakeholders. The process is often depicted using a *design spiral*, as illustrated in Figure 1. Due to these requirements and the complexity of a ship, the power plant design process needs to be flexible and have the ability to compare different designs without the need to change the analysis system.

While simulation has been used in ship energy system design, the analysis of simulation models remains a challenging task, both in terms of computational burden and the interpretability of the models. This thesis aims to develop a neural network surrogate modeling routine to decrease the number of model evaluations required

for efficient analysis. The benefit of the suggested approach is the scalability of the solution due to the representative power of neural networks, and the ability to insert the surrogate model to gradient-based optimization problems, provided the neural network architecture fulfils certain requirements. The use of the neural network surrogate model in optimization is expanded upon in Section 5.3.

3 Project overview

As discussed in the previous section, designing ship energy systems is a complicated and often unoptimized process. With the emergence of efficient gradient-free optimization algorithms such as NSGA-II Deb et al. [2002], and the wide availability of cloud computing resources, partially automated routines have been developed to enable the evaluation of a wide variety of energy system designs using simulation models. The objective of this project is to develop a routine allowing the analysis and optimization of power plant simulation models without the need to use gradient-free algorithms.

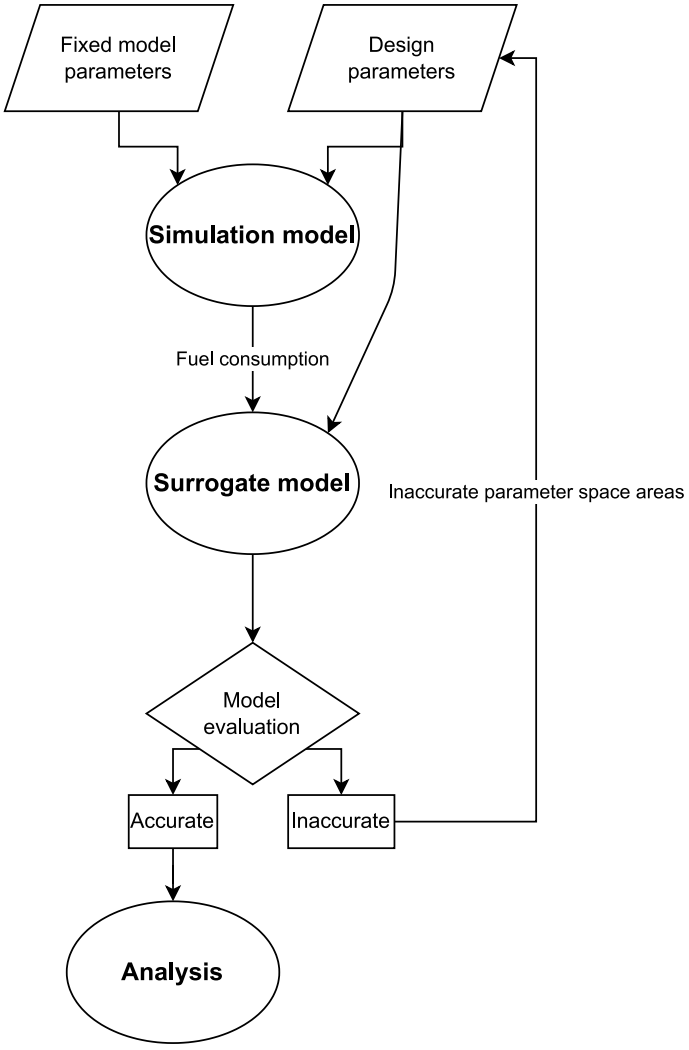

Figure 2: Flowchart of the routine

A simplified parametric energy system simulation model using mock operational data is created and used to model the fuel consumption of a cargo ship on its typical route. In order to minimize the number of simulations required for model analysis, a neural network surrogate model is trained to approximate the input/output structure of the simulation model. As the study is meant as a proof of concept for the design

of an accurate surrogate model, actual analysis of the simulation model is outside the scope of this thesis. The realism of the used simulation model is also not evaluated, and its complexity is kept to a minimum to avoid long computation times. A flowchart of the routine is shown in Figure 2.

The routine is developed with two important features in mind. Firstly, as the used simulation model is a simplified one and thus has fairly few parameters, scalability of the routine is considered to be of vital importance. This enables the use of the routine and its parts for a considerably larger model. Secondly, assumptions about the simulation model and its parameter structure are minimized when developing the routine. This aims to improve the applicability of the results of the study in more complicated ship energy system design projects, as well as projects in other fields.

To improve generalizability, an attempt is made to implement the routine modularly. This should enable the altering of parts of the routine, such as the simulation software or the surrogate model type, without the need to remake the entire system. The use of a surrogate model enables the development and adoption of analysis and optimization tools that should work with a wide variety of simulation models. The routine is developed so that the two main parts of it, the surrogate model and the simulation model, make no assumptions about the structure of each other. This means that the surrogate model treats the simulation model as a completely black-box model, only considering it as an input/output structure, and no information about the model subsystems is included in the surrogate model The same parameter inputs are used for both the simulation model and the surrogate model, as visualized in Figure 2.

A substantial part of the thesis focuses on how to select which points to use for the training of the surrogate model. As one of the main interests in the development of the routine is the potential to reduce the number of simulations, a significant bottleneck in simulation-based analysis, being able to construct an accurate surrogate model using as small of a sample as possible is desirable. While the optimal sampling routine is dependent on both the simulation model and the surrogate model, as well as the analysis task, some basic principles can be detected. Sampling is discussed in Section 5.4, and a test comparing different sampling algorithms is conducted.

Another technique used to reduce the number of simulation model evaluations required for accurate surrogate performance is *active learning*. Active learning is a machine learning approach, in which a labeled sample is used to train a model, whose performance is then analyzed to select a subset of a larger set of unlabeled points. These points are then labeled, in this routine by running the simulation model with the selected inputs, and the labeled points are added to the training set. This approach is visualized in Figure 2 as the link between the *Model evaluation* block and the *Design parameters* block. Active learning is discussed and an experiment is conducted as a part of the case study in Section 5.5.

4 Methods

4.1 System simulation

Mathematical models can be used to approximate the performance of complex systems, such as the energy systems of a ship. Models can be roughly divided into three categories: white-box, black-box, and gray-box models Driscoll et al. [2022]. In a white-box model, all internal interactions and formulas are known, and the input-output structure of the model can be stated using analytical expressions. A black-box model is an abstraction where only the input-output structure is available, and no inner workings are visible. In analyzing a black-box model, information can only be acquired by evaluating the model at different points of its domain. In reality, a large portion of models can be classified into the third category: gray-box models. A gray-box model uses available domain knowledge in subsystems of the model, while also having parts defined using information derived from data of the modeled process.

An alternative classification of models into white-box and black-box concerns the models' interpretability. This definition, especially prevalent in the context of artificial intelligence, defines a white-box model as one whose inner logic is explainable and meaningful, at least to the creators of the model. Such AI models are referred to as *Explainable Artificial Intelligence*, or XAI, and have been a topic of great interest in recent years Došilović et al. [2018]. In contrast, the internal logic of a black-box model, such as an artificial neural network, cannot be explained even by its creators Loyola-Gonzalez [2019]. While black-box models tend to achieve better accuracy, white-box models have the advantage of being interpretable and thus generally easier to modify. Analysis of white-box models can be easier due to a clearer understanding of relationship between parameters and subsystems, as well as the possible availability of model gradients.

Simulation models are a class of mathematical models depicting the dynamics and behaviour of a real-life system. As simulation models apply domain knowledge, and often all interactions between subsystems are explicitly defined, they could be classified as white-box models. However, practical simulation models tend to be complex, and the effect of model parameters can be nearly impossible to predict without running the model. Crucially, in the context of systems analysis and optimization, gradient information regarding a simulation model is often unattainable. Therefore simulation models can be classified as gray-box models.

There are various ways to classify simulation models, such as dynamic versus steady-state, stochastic versus deterministic, and discrete-event versus continuous. In this thesis, only deterministic dynamic models are considered. Such models can be thought of as mappings $S : \mathcal{X} \to \mathcal{Y}$. It is presumed that any point $\mathbf{x} \in \mathcal{X}$ can be evaluated, and that $a = b \to S(a) = S(b)$. The assumption of the models being dynamic, meaning that time is one of the model variables, does not necessarily have an impact on the routine. Steady-state models could also be used, as long as an explicit input-output structure can be defined. While in the case study conducted in this thesis the duration of the simulation is fixed, time could be added as an input

parameter without the need to majorly change the structure of the routine.

Computer simulations have the advantage over closed-form models of being able to model the complex interactions between different subsystems without the need to simplify these subsystems. As such, the impact of adding or modifying components can be studied without the need to completely reconstruct the model. The interactions between different subsystems can also reveal interesting effects not directly stated in the model definition. This phenomenon is known as *emergence* Goldstein [2011], and is observed in a wide variety of scientific fields. The main downside of simulation-based analysis is the computational cost of evaluating such models. Unlike closed-form expressions, which can usually be evaluated effectively instantly, simulation models can take up to weeks to run. Long evaluation times combined with the unavailability of gradient information make systematic simulation-based analysis a challenging task. The development of simulation models is also a laborious process, especially if real-world data needed for model verification is scarce.

While the construction of simulation models is an interesting subject, it is not covered in detail in this thesis. Instead, the routine is developed with the assumption of the existence of an accurate model. As such, the focus is on effectively using the simulation model for analysis. Some guidelines are given by a widely adopted framework of experiment design known as *Design of Experiments (DoE)*, first introduced in Fisher [1935]. The application of DoE in simulation-based experiments is discussed in Law [2017]. As noted in Law [2017], a problem with using the common approach in DoE known as 2^k factorial designs is the *curse of dimensionality*. 2^k factorial designs are used for analyzing the effects of variables on the output, and the interactions between variables. It consists of setting two possible values for each of the k model variables, and running the simulation for each of the resulting 2^k configurations. Clearly for a model with a large number of variables and non-negligible runtimes, the approach can be infeasible. A more general approach to simulation experiment planning is provided in Kelton [2000]. Santner et al. [2003] provides a detailed review of topics relevant for simulation experiment planning, including the use of space-filling sampling algorithms, a topic which is discussed in Section 5.4 of this thesis. For a comprehensive outlook of simulation experiments, Banks [1998] is recommended.

4.2 Surrogate modeling

Evaluation of simulation models can be computationally expensive, rendering an exhaustive collection of simulation-based data infeasible. One way to analyze such a model is constructing an easily evaluable approximation. The desired properties of the approximation model depend on the objectives of the analysis project. Some tasks require a differentiable representation, whereas others need the model to be simple to evaluate while still providing an accurate approximation everywhere in the simulation model's domain. The process of using one mathematical model to approximate another one, such as a simulation model, is called surrogate modeling. The function approximating the other model is called a *surrogate model* or a *metamodel*.

The task of finding an optimal surrogate model $\hat{f}$ for a model f can be formulated

as an optimization problem:

$$\min_{\hat{f}}. \; L(f, \hat{f}), \tag{1}$$

with L being a loss function. The selected loss function depends on the task at hand. Some problems require a good approximation for all possible inputs, whereas others may emphasize the mean accuracy. For the fist task, the maximum absolute error loss could be used. For a dataset $\mathcal{S}$, it is defined as:

$$L^{\infty}(f, \hat{f}) \coloneqq \max(|f(x) - \hat{f}(x)|), x \in \mathcal{S} \tag{2}$$

For the second task, a loss function based on mean absolute error could be used. For a dataset $\mathcal{S}$ it is defined as:

$$L^{1}(f, \hat{f}) \coloneqq \frac{1}{|\mathcal{S}|} \sum_{\mathcal{S}} |f(x) - \hat{f}(x)| \tag{3}$$

The most commonly used loss function for regression tasks, both due to the easy calculation of gradients and the tendency to react to outliers, is the mean squared error, or MSE. For a dataset $\mathcal{S}$ it is defined as:

$$L^{2}(f, \hat{f}) \coloneqq \frac{1}{|\mathcal{S}|} \sum_{\mathcal{S}} (f(x) - \hat{f}(x))^2 \tag{4}$$

Surrogate modeling is used for many purposes, but in the context of simulation-based analysis and optimization, the main benefits are the reduction in computing time and the possible availability of gradient information. The reduction in computation time is highly dependent on both the original process and the selected surrogate model. Using a surrogate can reduce total computation time, if training it requires significantly fewer simulations than using the simulation model to gather the necessary amount of data. The availability of gradient information depends on the function class of the surrogate model.

Surrogate models range in complexity from simple linear models to massive deep neural networks. Bhosekar and Ierapetritou [2018] offers an introduction into frequently used surrogate models for regression problems, as well as a review of recent advances in surrogate modeling. The use of simple functions, especially polynomials, has several benefits. They are continuously differentiable, and are easily interpretable. Polynomials have easily calculable gradients, a desirable quality for optimization problems. However, the assumptions that justify the use of polynomials can be difficult to verify, and the quality of predictions can vary considerably within the domain of the model. High order polynomials have a tendency to overfit, especially if no regularization is used. While some of the more complicated surrogate models may achieve a high accuracy, their interpretability can be lower than that of polynomials.

In accordance with the project objectives of generalizability and scalability to more complex designs, artificial neural networks (ANN) are selected as the function type of the surrogate model. ANNs are known to be universal approximators, meaning that they can approximate any continuous bounded function with a finite domain with arbitrary accuracy Hornik et al. [1989], using as few as one hidden layer. ANNs

have been succesfully used as surrogate models for agent-based models in Angione et al. [2022] and Lamperti et al. [2018], achieving a better performance than other surrogate model types in both studies. Lamperti et al. [2018] also employed an active learning approach similar to the one used in this routine.

While the use of ANN surrogate models in optimization problems is challenging, it is not impossible. Mixed-integer programming formulations for ANNs are discussed in Anderson et al. [2020], and enable the use of ANNs as objective functions in optimization problems. The use of the ANN surrogate developed in this thesis in an optimization setting is discussed further in Section 5.3.

5 Implementation and case study

This section describes the development of the modeling routine using a case study. Due to the lack of available real world data, mock data is created. The use of mock data is not seen as a problem, because the focus of the thesis is on exploring the development of a surrogate model building routine, not ship energy system analysis.

5.1 Case study

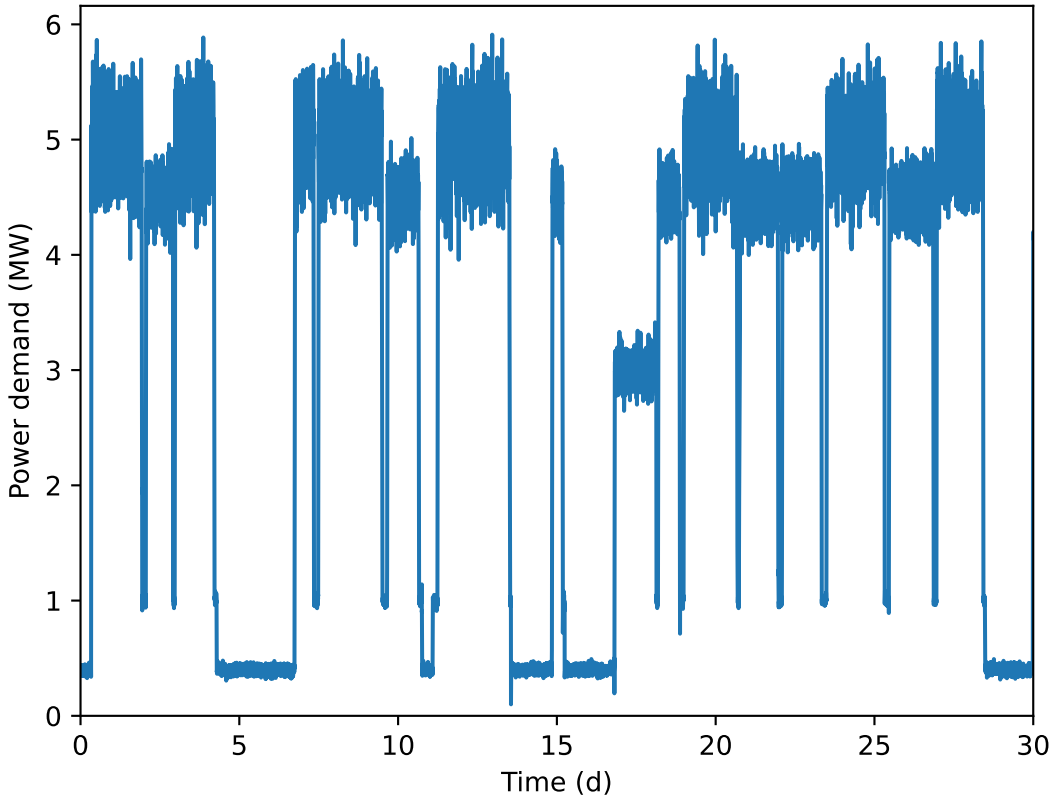

Figure 3: Total power demand of the studied ship

The case study constitutes a simplified ship power plant design problem in the concept phase of a ship design project. The considered ship is a medium-sized cargo ship operating on the Baltic Sea. From the power plant design viewpoint, the ship is defined by its power consumers and power producers. In the simulation model, the power consumption is represented by a single time series. The power demand is not split into the three categories discussed in Section 2.1. In fact, all power demand is represented by a single component in the model. The power demand time series, as

well as the simulated voyage, has a duration of 30 days and a sampling period of 60 seconds. The series, consisting of a total of 43201 points, is shown in Figure 3. The series is based an operational profile described in Table 1, where the average power and white noise are given in megawatts, and the durations in hours. The series was created by randomly selecting an operating phase, and creating a time series slice with the duration randomly selected between the corresponding minimum and maximum durations. The time series values P_m in operation mode m are created using the following formula:

$$P_m(t) = \mu_m + \sum_{i=1}^{min(t_m,30)} \theta_i \epsilon_{t-i} + \epsilon_t, \tag{5}$$

when the process is not transitioning from one operation mode to another. The mode changes are selected randomly, and transitions between modes are made with a constant slope, corresponding to a rate of change from stationary to full load in 30 minutes, or the opposite, in the case of decreasing power demand. In Equation (5), t_m refers to the number of time points spent in the current phase, μ_m is the phase mean power demand, ϵ_t is the random perturbation at time point t, drawn from a normal distribution with variance corresponding to the field *white noise* in Table 1. The coefficients θ_i are defined as $\theta_i = 0.002 \cdot (30 - i)$. The formula is that of a moving-average model, and is used in the time series simulation to replicate a stationary process with random impulses propagating to future values, as would be characteristic for an operating ship. While the profile and the resulting time series are not necessarily realistic, the time series visually resembles real operating data well enough to be used in a proof of concept study.

Mode	Average power	White noise	Min duration	Max duration
Full load	5	0.5	7	60
Partial load	4.5	0.3	7	36
Empty	3	0.2	7	72
Loading	1	0.05	1	4
Stationary	0.4	0.05	1	72

Table 1: Operation modes

Only diesel-electric concepts are considered in the design problem. This justifies the decision to use a single time series to represent both the energy required for propulsion and the hotel load, as all energy demand can be treated as electrical power. The decision also simplifies the parameter space, since all power producers can be treated equally, instead of having to split them to electric and mechanical power producers. Thermal energy is not included in the model.

The simulation models are created using *SimulationX*, a system simulation software developed by ESI Group. SimulationX models are based on *Modelica*. Modelica Fritzson and Engelson [1998], is an object-oriented multidomain physical simulation modeling language. Modelica-based models are evaluated by solving differential equations defined by equality declarations in model components and their

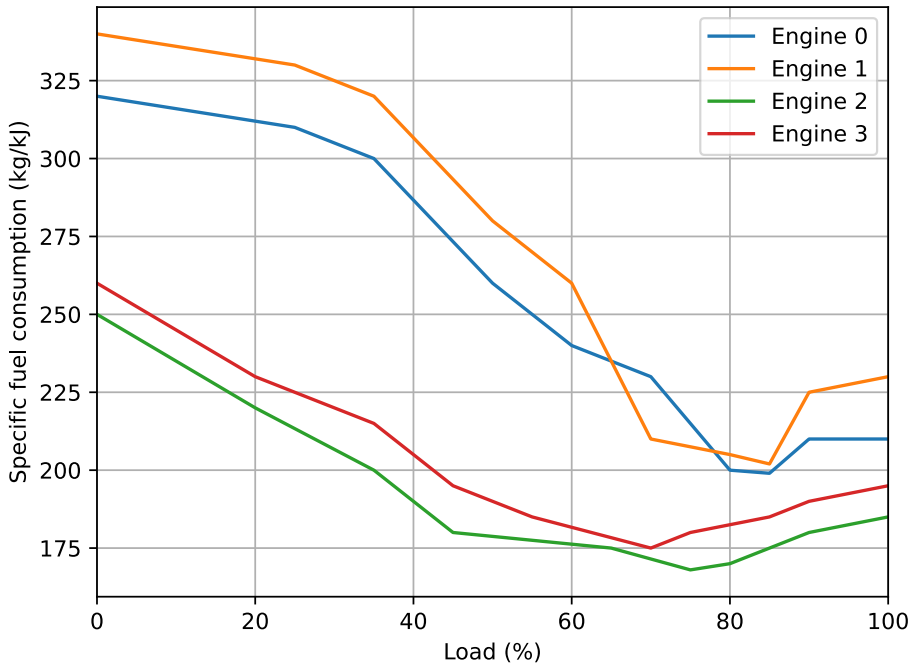

Figure 4: Engine specific fuel consumption curves

connections. Modelica, unlike many other modeling languages, is considered *acausal*. Acausal modeling enables relatively simple multidomain simulation, since the models are not signal based. According to Schweiger et al. [2020], acausal modeling is considered to be more suitable for the modeling of large physical systems than causal modeling.

In the case study, the *Ship Energy Systems* component library of SimulationX is used. The library components included in models are *electrical usage*, a varying number of *engines*, a *fuel tank*, and the *operating conditions*. The electrical usage component represents the total power demand at each time point, as defined by the time series in Figure 3, and must be satisfied exactly at each time point.

The engines are a slight modification of the component *Generator Set* provided in the Ship Energy Systems library. In this system, an engine is defined by its fuel, the specific fuel consumption curve, and its maximum continuous rating (MCR). The specific fuel consumption curve defines the efficiency of the engine in $\frac{kg}{kJ}$ as a variable of the engine's *load*, the output relative to its MCR. The specific fuel consumption curves of the available engine types are shown in Figure 4. The MCR is the maximum power an engine can produce continuously, although in practice engines are rarely operated at 100% load for engine health and fuel economy reasons.

Parameters of the operating conditions and the fuel tank are fixed between models, and are only included in the simulation models because SimulationX requires them to be present. The operating conditions component enables the setting of ambient conditions such as seawater temperature, air temperature, and air pressure,

but as thermodynamics are excluded from the model, the conditions do not affect the model output. The fuel tank component controls the available fuels and their parameters, such as energy density. In this study, engines use either heavy fuel oil (HFO) or liquefied natural gas (LNG), with lower heating values of $42700\frac{kJ}{kg}$ and $49620\frac{kJ}{kg}$, respectively. No dual-fuel engines are considered, but configurations including engines running on different fuels are allowed, meaning that the output of the simulation model, the total fuel consumption, is multidimensional.

Engine	Fuel	m_{min}	m_{max}	l_{opt}	l_{max}
1	HFO	2.5	4.0	0.85	0.95
2	HFO	0.2	1.0	0.85	0.96
3	LNG	2.5	3.5	0.75	0.95
4	LNG	0.2	1.2	0.70	0.97

Table 2: Engine types

All power production in the model is provided by engines, which are connected to generators, since all power demand is considered to be electrical. All generators have an efficiency of 85%, so they do not need to be considered separate components in terms of the model inputs. The engines are selected from a predefined dataset, described in Table 2, where m_{min} and m_{max} refer to the minimum and maximum MCR of the engine type, and l_{opt} and l_{max} to the optimal and maximum allowed load of the engine type, as a fraction of the engine's MCR.

A custom-made component called the *load splitter* is also used. At each time point, the load splitter reads the power demand and determines the load of each connected engine. There is no delay and the splitter cannot anticipate future demand. The load sharing is controlled one engine at a time, so that at any time point only one engine can be turned on or off. Therefore, the slope of the power demand is assumed to be constrained in magnitude. The load of engines are set according to the following rules, where P_{tot} refers to the total energy demand at the current time point, and P_i is the power production of engine i ant the time point:

1. If $0 < P_i, P_j = P_j^{opt} \ \forall \ j < i$.
2. If $P_i < P_i^{max}, P_k = 0 \ \forall \ k > i$.
3. If P_{tot} goes above $\sum_{n=1}^{i-1} P_n + P_i^{max}$, set $P_i \leftarrow P_i^{opt}$, and $P_{i+1} \leftarrow P_{tot} - \sum_{n=1}^{i} P_n^{opt}$.
4. If $i > 1$ and P_{tot} goes below $\sum_{n=1}^{i-1} P_n^{opt}$, set $P_i \leftarrow 0$ and $P_{i-1} \leftarrow P_{tot} - \sum_{n=1}^{i-2} P_n^{opt}$

In the last rule, $P_0 = 0$. The values of P_i^{max} and P_i^{opt} are defined by multiplying p_i, the MCR of engine i, with l_{max} and ł$_{opt}$ of the corresponding engine type, using the values in Table 2. P_i^{max} is the maximum acceptable continuous power output of engine i, and models the principle of not running an engine at its maximum power for non-negligible durations. P_i^{opt} is the power at which the specific fuel consumption of engine i is minimized, and is derived from Figure 4.

After consulting marine engineering experts, some constraints have seen set for the design problem. Firstly, the number of engines in a design must be between two and four. While one engine might be able to provide enough power to fulfil the power requirements at all times, an additional engine is required in a ship to provide redundancy in the case of engine failure. The maximum number of engines is set to limit the size of the modeling problem. Additionally, ships similar to the one modeled in the case study rarely have more than three engines. Secondly, at most two different engine types are allowed to be present in a design. This constraint is set because it simplifies the maintenance of engines onboard a ship, as different engine types would require different spare parts as well as increasing the amount of training required for the maintenance crew.

5.2 Model construction

In order to efficiently explore different designs, a parameterization of the models is defined. Since all power is produced by the engines, and both the operating logic and energy consumption are not altered between models, the parameterization only needs to define the engine layout. The layout is defined by the number of selected engines, the engines' types, the order of the engines, and their MCR. The parametrization should be constructed so that the input/output structure of the model is interpretable, and efficiently readable by the surrogate model training routine. The selected parameterization consists of a matrix T and a vector $\mathbf{p}$.

$T \in \{0,1\}^{4\times4}$ is a one-hot matrix describing the selected engine types and their order. Each column contains at most one non-zero element, with the row of the non-zero element of column i indicating the type of the i^{th} engine. The matrix must be filled from left to right, meaning that if column i contains only zeros, so must all columns j where $j > i$. As stated in the previous section, at least two engines must be present in each design, meaning that columns 1 and 2 must always contain one non-zero element.

$\mathbf{p} \in \mathbb{R}^4_+$ represents the MCRs of the engines. In this system, the value of p_i is the MCR of engine i in megawatts. The value of p_i must be between m_{min} and m_{max} for the engine type, as given in Table 2.

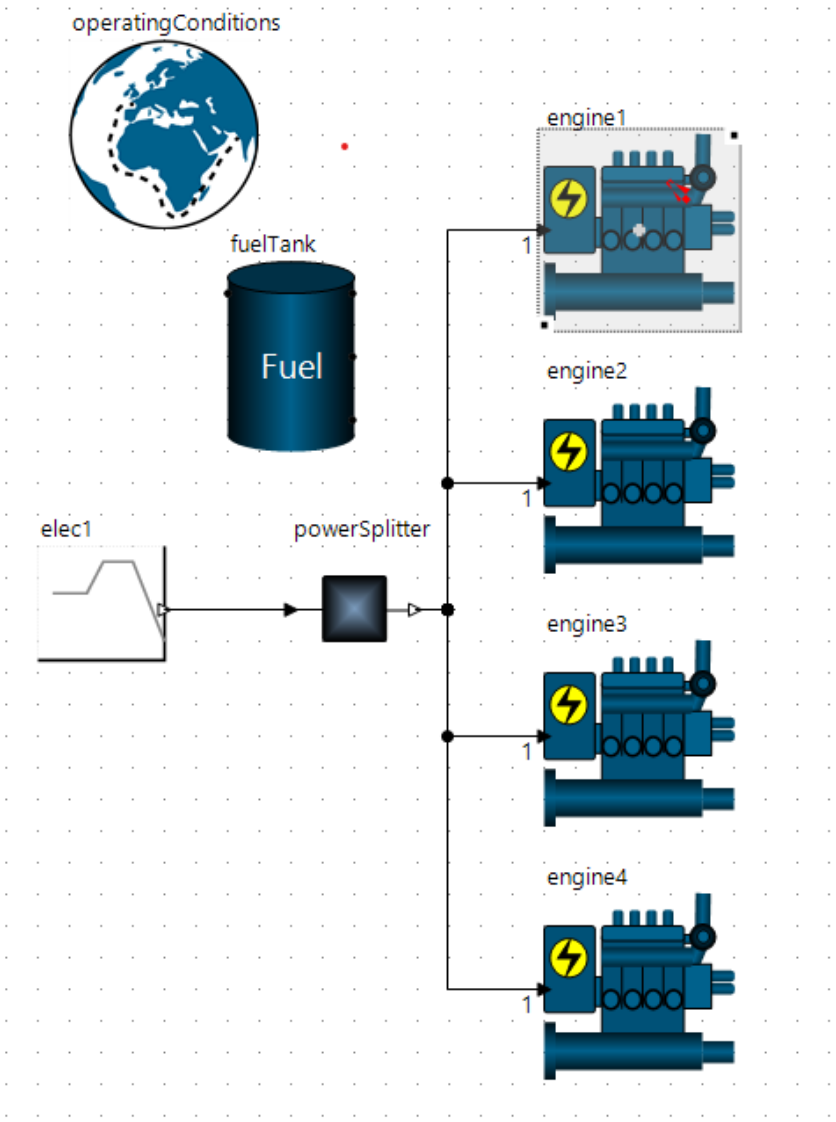

Figure 5: A SimulationX model

Mathematically, the parameter space is set by the following constraints:

$$\sum_{i=1}^{4} T_{ij} = 1 \ \forall \ j \in \{1, 2\}$$

$$\sum_{i=1}^{4} T_{i3} \leq 1$$

$$\sum_{i=1}^{4} T_{i4} \leq \sum_{i=1}^{4} T_{i3}$$

$$\sum_{i=1}^{4} [1 - \prod_{j=1}^{4} (1 - T_{ij})] \leq 2$$

$$T^{\mathrm{T}} \mathbf{m}_{min} \leq \mathbf{p} \leq T^{\mathrm{T}} \mathbf{m}_{max}$$

$$6.45 \leq \sum_{i=1}^{4} p_i \leq 12$$

$$T \in \{ 0, 1\}^{4 \times 4}, \mathbf{p} \in \mathbb{R}_{+}^{4}.$$

The last constraint ensures that the total power production capacity is larger than the maximum energy demand of the planned route multiplied by a security factor. The maximum capacity is also limited, to avoid an overly expensive design.

SimulationX models are constructed algorithmically using the defined parameterization. The models are initialized, ran, and their output recorded with VBScript files, using SimulationX's COM interface. In this implementation, the VBScript files are created using Python. Python scripts are also used to connect the model outputs to the inputs. A diagram view of a constructed SimulationX model with four engines in shown in Figure 5.

5.3 Surrogate model

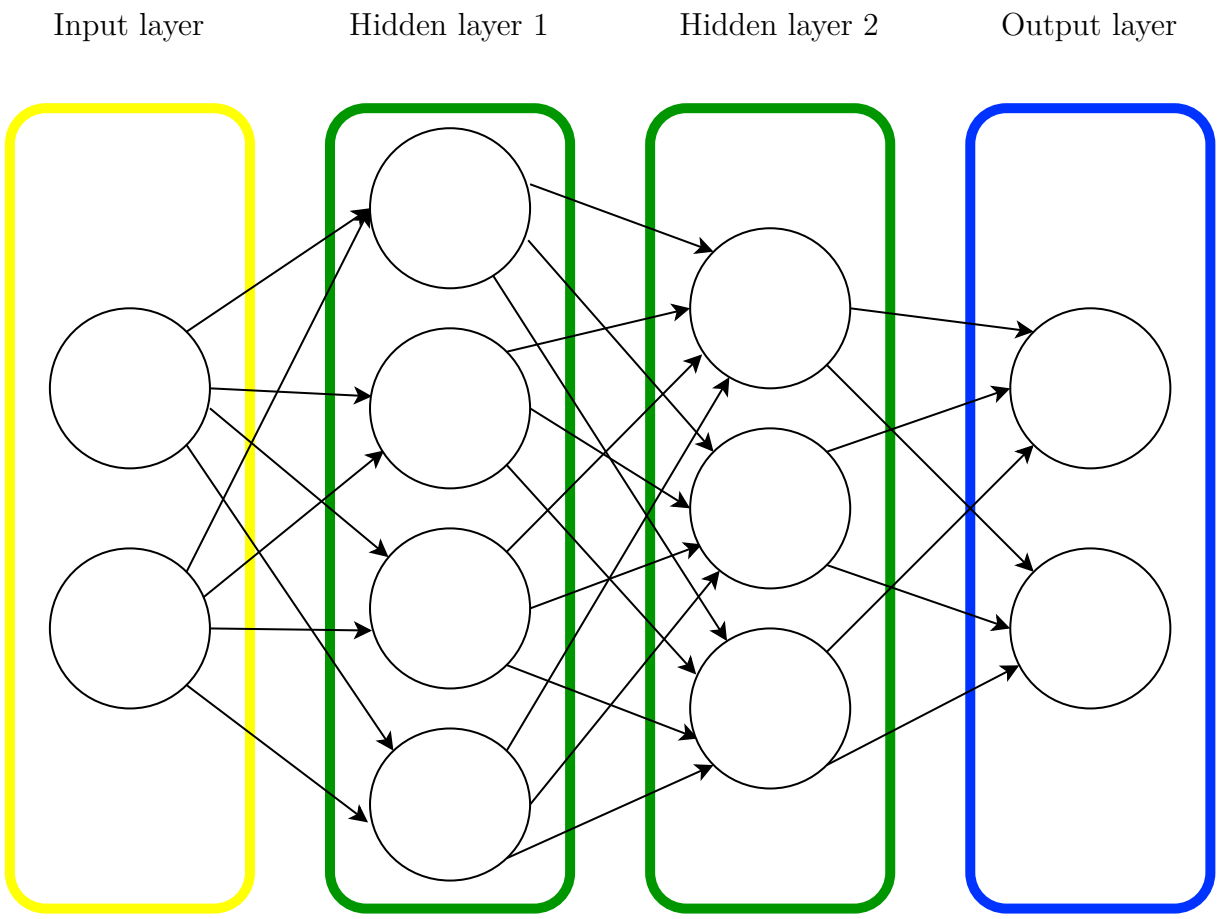

Figure 6: ANN architecture

As discussed in Section 4.2, artificial neural networks are used as surrogate models in the routine. To be exact, the types of models used are fully connected multilayer perceptrons (MLP), a subclass of feedforward artificial neural networks. A basic

implementation of an MLP consists of an input layer, one or more hidden layers, and an output layer. Figure 6 shows an illustration of a fully connected MLP with three inputs, two hidden layers with four and three neurons respectively, and two outputs. Each neuron performs a linear transformation on its inputs, followed by a nonlinear *activation function*. The output of the activation function, along with the output of all other neurons in the layer, are then used as the input of the following layer's neurons. MLPs generalize and scale well. It is simple to change the number of both input and output variables, although the model architecture may need to be changed and the model needs to be retrained for each separate problem. One challenge with using MLPs in the case study is that they only accept one-dimensional input data, meaning that the engine type matrix T has to be transformed. In the routine, T is flattened from a 4×4 matrix to a vector of length 16. That vector is then concatenated with $\mathbf{p}$ to form the input vector $\mathbf{x}$ of the ANN, with $x_i \in \{0, 1\}, 1 \leq i \leq 16$, and $x_i \in \mathbb{R}_+, 17 \leq i \leq 20$. The output of the surrogate model is a vector $\mathbf{y} \in \mathbb{R}_+^2$, whose elements are the predictions for the total consumption of HFO and LNG, respectively.

There are no universal rules for the design of regression MLP architectures. In the methodology introduced in Xu and Chen [2008], the optimal number of hidden nodes is between $\sqrt{N/(d \log N)}$ and N/d, where N is the number of training pairs in the dataset and d is the dimension of the model input. Others place the number of hidden nodes between the dimension of inputs and outputs Blum [1992]. Generally, having considerably more hidden nodes than input variables leads to overfitting, although in practice large networks are often observed to generalize better than smaller ones Novak et al. [2018]. Both large and small networks are tested in this study, although major overfitting is avoided. The regularization techniques implemented in the model selection process include limiting the model size, L1-regularization, and early stopping.

Hyperparameter selection

architecture	layer 1	layer 2	layer 3	layer 4
1	10			
2	22			
3	100			
4	1000			
5	6	4		
6	14	8		
7	30	10		
8	10	10	10	10
9	50	50	50	20

Table 3: Grid search architectures

Before the active learning routine, the architecture of the model must be selected. As it is difficult to determine the optimal architecture theoretically, several alternatives are tested and their performance is compared. The variables dictating the model

architecture are called *hyperparameters*. In the comparison, the altered hyperparameters are the number of hidden layers and the number of neurons in them, collectively called the model *architecture*, and the L1-regularization weight. The candidate architectures are listed in Table 3. L1-regularization is tested with weight coefficients $10^{-3}, 10^{-4}, 10^{-5}$ and 0, with the latter being equivalent to no regularization. L1-regularization is implemented by adding the L1-norm of the MLP weights multiplied by the L1-regularization weight coefficient to the model loss. The total loss during the training procedure for a training batch with n samples and a model with a total of b weight parameters is then:

$$L(\mathbf{X}, \mathbf{Y}, \hat{f}) = \frac{1}{2n} \sum_{i=1}^{n} [(\mathbf{y}_i[1] - \hat{f}(\mathbf{x}_i)[1])^2 + (\mathbf{y}_i[2] - \hat{f}(\mathbf{x}_i)[2])^2] + \lambda \sum_{j=1}^{b} \beta_j \tag{6}$$

In Equation (6), β are the model weight parameters. L1-regularization, also known as *LASSO*, is chosen over L2-regularization, because it has the feature of setting some parameter values to exactly zero, which facilitates the use of the model in optimization problems. One way of using a neural network in an optimization model is through the use of the Python library OMLT Ceccon et al. [2022]. OMLT also requires the use of ReLU as the activation function, which is why no other activation functions are tested.

While there are several algorithms for hyperparameter tuning, in this study an exhaustive hyperparameter space search is performed. This approach is known as *grid search*. It is fairly simple to understand in comparison to techniques such as evolutionary algorithms, and has the benefit of testing each hyperparameter combination. The downside is that the hyperparameter space must be discrete, and for a large number of hyperparameters and their values, the procedure can be very time-consuming.

In machine learning tasks, input and output normalization can help with the convergence of the model. While the effect of normalization depends on the used optimizer and the structure of the model, having input values be of similar magnitude is usually helpful. Scaling the training data outputs, also called *labels* can also be useful, since a mapping from small inputs to significantly larger outputs requires large weights, which can slow down the convergence of the model training, since MLPs are usually initialized with weights close to zero. In training our model, dividing the labels $\mathbf{y}$ by their maximum values in the validation set lead to faster convergence and better performance. Every training set is scaled using the validation set values, since the same validation set is used to evaluate the performance of each model, leading to a consistent scaling. In some settings, labels are transformed to zero mean and unit variance at Stanford University [2018]. In the case study, this approach is not used due to the outputs having a guaranteed value of zero, if no engines using the respective fuel are present; an important property for the model to learn. A set with labels on both sides of zero might not support this objective. It is unclear whether linear scaling is the optimal transformation for data in which labels often have a value of zero.

Each model is trained in PyTorch using MSE loss, and 1000 epochs. Adam Kingma and Ba [2014] is used as the optimizer, as it was shown to outperform

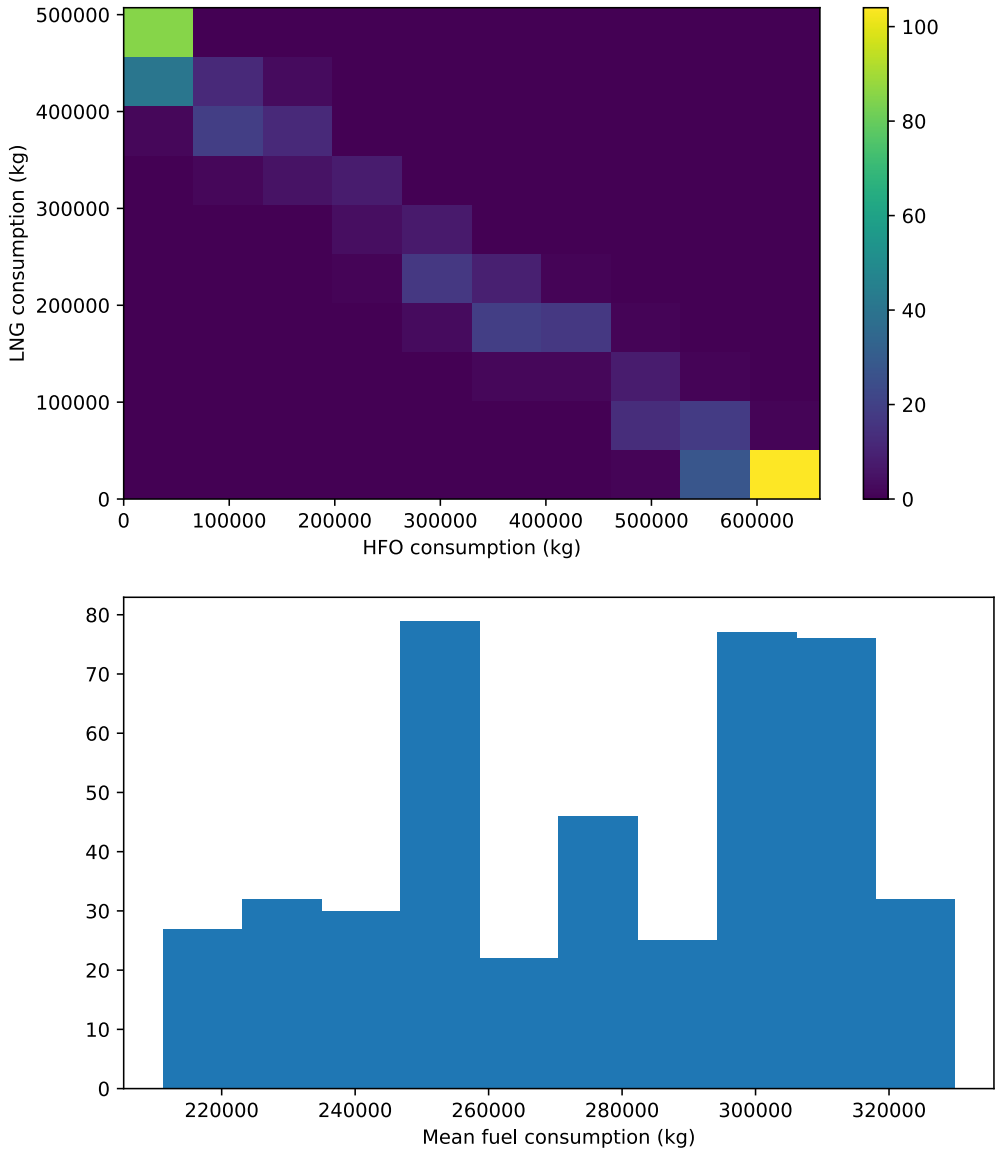

Figure 7: Label distribution

stochastic gradient descent both in terms of convergence time and accuracy. Different learning rates were tried at first, but a learning rate of 0.001 converged better than the tested alternatives, and was thus selected for the grid search. As the training sets are fairly small, the training is conducted without needing to use minibatches. Each model is evaluated both by the final test loss and the ratio between final training and

test loss. The test is conducted using a validation set of 445 points, which includes 5 points from all 89 feasible values of T. Some Ts are infeasible for all values of $\mathbf{p}$ due to the design problem constraints, namely the one concerning the minimum and maximum total production capacity. The distribution of the outputs of the simulations are visualized in Figure 7, with the latter showing the mean consumption of a single type of fuel. That is, not the total fuel consumption, but roughly half of the mean total consumption. This is because the used MSE loss is calculated in the same manner.

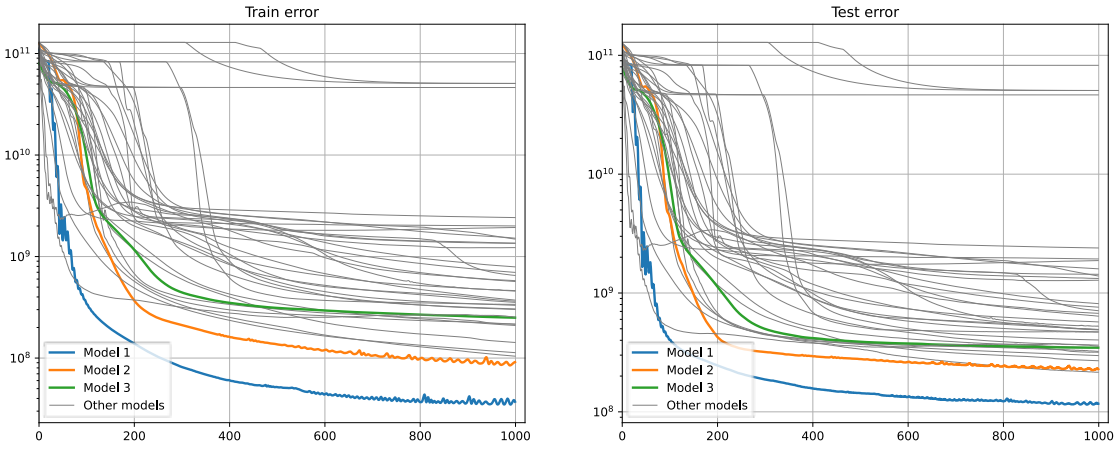

Figure 8: Grid search model convergence

Following the grid search, the three models listed in Table 4 are selected. The models are used in the sampling routine comparison described in the next section. The architectures are selected based on diversity, test error, and the ratio between test and training error, the latter of which describes how much the model overfits. Model

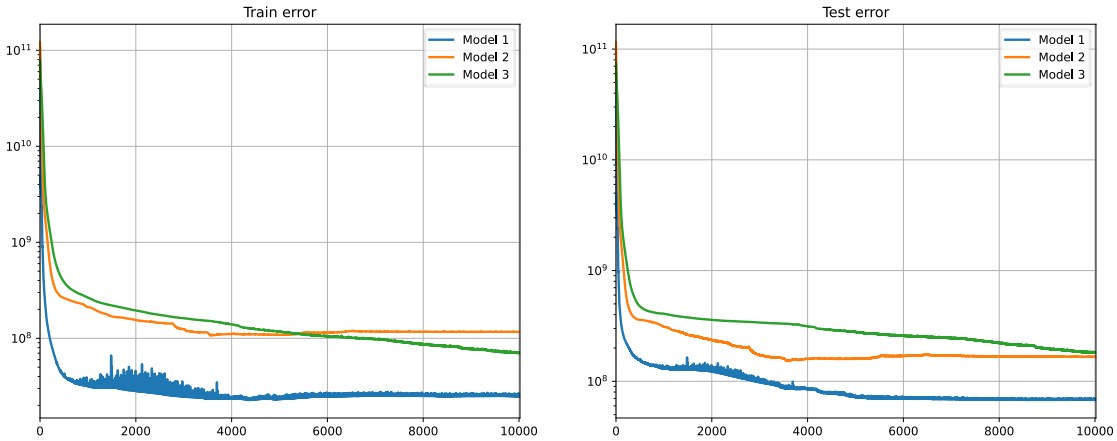

Figure 9: Convergence of the selected models in a longer routine

Model	Hidden nodes	L1 Regularization weight
1	1×1000	10^{-5}
2	1×100	10^{-4}
3	1×22	10^{-5}

Table 4: Selected architectures

convergence is inspected visually, and is shown in Figure 8. The visualized training and test errors are calculated using the unscaled labels and predictions, although the error used in the training is calculated using scaled labels. The best models converge to an MSE of 10^8, corresponding to a mean error of around 10 000 kg per prediction for both fuels. This is an error of roughly 3-4%. Models with a large number of nodes seem to perform better than those with less than 20 nodes. Interestingly, the models with the smallest test errors are ones which seemingly overfit the training data significantly, as exemplified by Model 1. It is unclear whether a less overfitting model performs better for different sample sizes, which is why both overfitting and non-overfitting models are selected. Figure 8 shows that none of the architectures has a problem with growing test errors, indicating that a longer training routine might improve the models. The three selected models are therefore also trained with 10000 epochs. The results of the longer training routine are shown in Figure 9. The models with a smaller L1-regularization coefficients seem to achieve optimal performance at around epochs 3000-4000, whereas the model with tighter regularization still seems to improve towards the end of the training. Interestingly, the training error of the smallest model starts to temporarily increase in the middle, suggesting that the use of an optimizer with an adaptive learning rate, such as Adadelta Zeiler [2012], or manually changing the learning rate during training, might be reasonable.

5.4 Sampling

In order to construct a surrogate model that performs accurately in the entire domain of the simulation model, a comprehensive training dataset is required. The size of the required dataset depends on both the complexity of the modeled process, and the type of function used as the surrogate model. In studying the relationship between simulation sample size and population statistics, sampling strategies producing low-discrepancy sequences have been shown to converge faster than techniques based on random sampling Burhenne et al. [2011]. While the task of surrogate model training differs from that of determining population statistics, both tasks require the dataset to have similar space-filling qualities.

Latin hypercube sampling (LHS) is selected as the space-filling sampling algorithm McKay et al. [2000]. Deterministic sampling strategies, such as ones based on Sobol' sequences Sobol' [1967], could provide samples with lower discrepancy than LHS sampling. However, LHS is fairly easy to understand, and has the advantage over Sobol' sequence-based sampling of having non-uniform distances for the projections of the samples onto each axis, which might lead to a more robust surrogate model, although this assumption is not tested. Another commonly cited benefit of Sobol'

sequence-based sampling, the ability to increase the sample size without the need to recompute the entire sample, is inconsequential in this routine, since in the implemented active learning scheme, each new set is sampled in a subspace of the parameter space, defined by the performance of the surrogate model. The sampling of these new datasets is described in Section 5.5.

In this project, an optimal sampling strategy is one which enables the construction of a globally accurate surrogate model with as few simulations as possible. Simulation-based studies often employ a *simulation budget*, typically defined as a maximum number of simulations allowed for either the entire study or a part of it. While no exact budget is set for this study, the number of simulations is assumed to be limited so that analysis of the simulation model by conducting an exhaustive search of the parameter space by evaluating a large number of simulations is infeasible.

In this section, several sampling strategies are compared in terms of scalability, generalizability, and ANN convergence. Due to the fact that an active learning scheme is employed, a smaller initial dataset can be used, in contrast with a setting where no new points can be evaluated. However, a comprehensive validation set is required to enable the identification of parameter space areas where the surrogate model performs poorly. If the initial dataset is too sparse, some areas of the parameter space might be unrepresented, and the surrogate performance cannot be evaluated well. When using a small sample size, the training set might also be sparse in some areas, but the possible lacking performance in such areas can be reacted to within the active learning scheme.

The main challenge in implementing commonly used space-filling sampling strategies is that they tend to assume a constant number of parameters, which take values from continuous intervals. In such a setting, the sampling can then be formulated as selecting points from a unit hypercube $[0, 1]^k$, with k being the number of parameters in the model. After sampling, the sampled values are scaled to their respective value ranges. In this study, several qualitative variables are present, and the feasible value ranges for some of the parameters are dependent on the values of other parameters. In such a setting, hypercube-based sampling strategies cannot be implemented without modifications. In this study, two approaches in dealing with qualitative parameters, represented by T, are implemented. The first one is to enumerate all feasible permutations of qualitative variable values, and sample the numeric variables using a separate sampling strategy. This sampling strategy shall be referred to as *enumerative sampling*. The second approach is to parameterize the qualitative variable values, referred to from now on as configurations. This approach shall be called *parameterized sampling*. In the following section the approaches are described for both the case study and a general power system design, since the goal of the thesis is to develop a scalable routine.

5.4.1 Enumerative sampling

Basic idea

In the enumerative sampling approach, a list of feasible configurations is enumerated, and a space-filling sampling strategy is used for the numeric variables. For a model

with n qualitative variables with each having $m_i, 1 \leq i \leq n$ levels, this means that at least $\prod_{i=1}^{n} m_i$ points have to be sampled for a completely representative dataset, assuming all permutations are feasible. If the number of variables is not constant, the omission of a variable can be represented as an additional parameter value.

The simplest way to implement enumerative sampling is to list all qualitative parameter value combinations, and perform a sampling routine for the numeric parameters separately for each configuration. This method guarantees that each configuration is represented evenly, and it is easy to interpret. If the sample size is large compared to the number of configurations, a space-filling routine can be implemented for each configuration, leading to a very comprehensive dataset. If the values of the qualitative parameters do not influence the values of numeric parameters, the latter can be sampled in a single routine, and then assigned randomly to the enumerated configurations. This ensures that the projections of the parameter space to both the numeric and qualitative parameters are sampled in a space-filling way. If the value range of one or more numeric parameters is dependent on the value of the qualitative parameters, separate sampling routines might have to be implemented for each configuration. In such a setting a large sample size might be necessary to capture the relationship between parameters.

Case study implementation

In the case study, the engine type matrix T represents qualitative parameters and the engine power vector $\mathbf{p}$ the numeric ones. For sampling purposes, the one-hot representation of T can be changed to a vector of integers $\hat{\mathbf{t}}$, each element representing the row index of the non-zero element of a column. Since the number of engines varies between two and four, the possible values of the parameters are between one and four for the first two elements of $\hat{\mathbf{t}}$, and between zero and four for the latter two elements, with the value zero corresponding to no engines in the respective engine slot. The possible values of $\hat{t}_3$ depend on the values of $\hat{t}_1$ and $\hat{t}_2$, due to the constraint dictating that the power system can include at most two different engine types. The possible values of $\hat{t}_4$ are similarly dependent on the values of the first three elements, with the additional constraint of non-zero values only being allowed if $\hat{t}_3$ is also non-zero, since the engine slots must be filled in increasing order. The number of engine configurations is then as follows:

$$
\begin{aligned}
&2 \text{ engines, } \hat{t}_1 = \hat{t}_2 : 4 \\
&2 \text{ engines, } \hat{t}_1 \neq \hat{t}_2 : 4^2 - 4 = 12 \\
&3 \text{ engines, } \hat{t}_1 = \hat{t}_2 : 4 \cdot 4 = 16 \\
&3 \text{ engines, } \hat{t}_1 \neq \hat{t}_2 : 12 \cdot 2 = 24 \\
&4 \text{ engines, } \hat{t}_1 = \hat{t}_2 = \hat{t}_3 : 4 \cdot 4 = 16 \\
&4 \text{ engines, } \hat{t}_1 \neq \hat{t}_2 \text{ or } \hat{t}_1 \neq \hat{t}_3 : (24 + 12) \cdot 2 = 72.
\end{aligned}
$$

In total, there are 144 possible configurations for the ship in the case study, although some of them might not be feasible due to other constraints. In general, the number

of possible configurations for a with n engines and m distinct engine types is given by

$$y_n = \binom{m}{2} \cdot (2^n - 2) + m = 2m - 2^{n-1}m - m^2 + 2^{n-1}m^2, \tag{7}$$

assuming a maximum of two distinct engine types, a convention which ship designs typically follow. The binomial coefficient in Equation (7) represents the number of distinct engine pairs, and the term $(2^n - 2)$ is the number of ways each pair can be used to fill all n engine slots, excluding configurations in which all slots are filled by the same engine type. The last term, m, represents all configurations which only include one engine type.

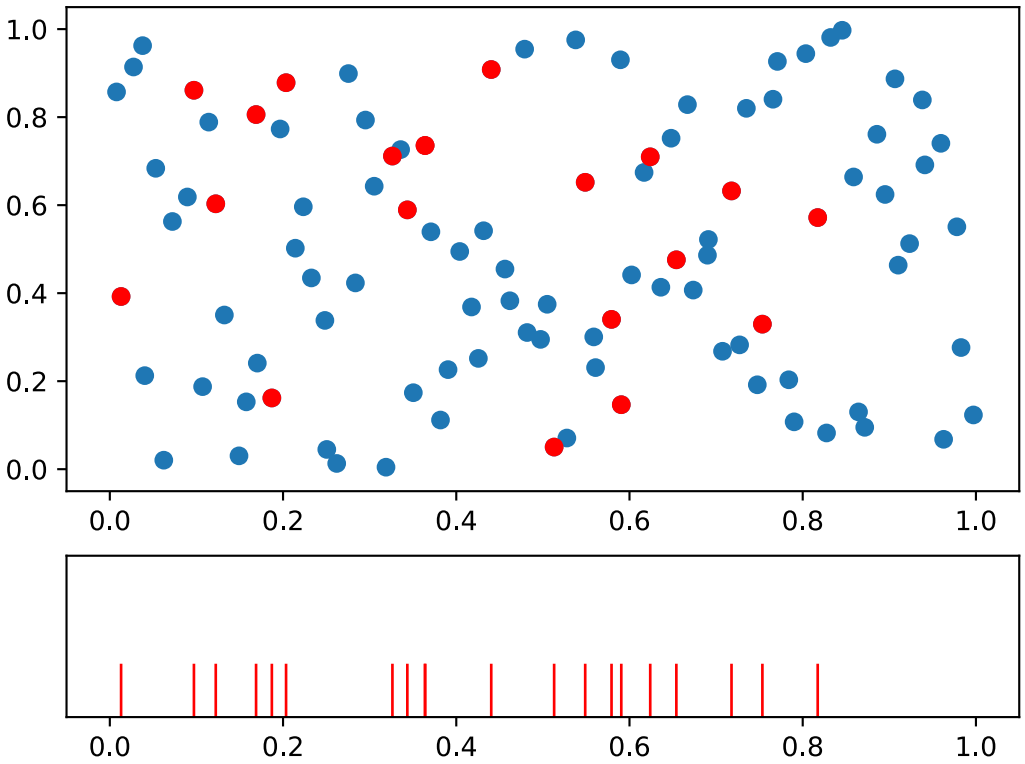

Figure 10: LHS sample and a subset projection

Due to the large number of configurations, it is infeasible to have a sample size large enough where separately sampling $\mathbf{p}$ for each configuration would make sense. Instead, three LHS samplers are defined, with dimensionalities 2, 3, and 4, corresponding to the number of engines in a configuration. A sample is drawn from each sampler, with the sample sizes depending on the number of sampled configurations with the corresponding number of engines. While a four-dimensional

LHS sampler could be used, with projections of $\mathbf{p}$ into two or three dimensions used for configurations with less than four engines, this approach would not guarantee that samples corresponding to two- and three-dimensional configurations fill the parameter space well. Figure 10 illustrates this with a subset of a two-dimensional LHS sample projected into one dimension. Clearly, the projection does not fill the space well, although the two-dimensional sample is quite space-filling. Therefore, unless the subset is selected with a strategy which ensures that the projections to lower-dimensional spaces are space-filling, it is best to use separate LHS samples for each configuration group. Separate four-dimensional samples could be drawn for each group with the space-filling qualities retained, but doing so would offer no clear benefit over the use of separate samplers.

Problems

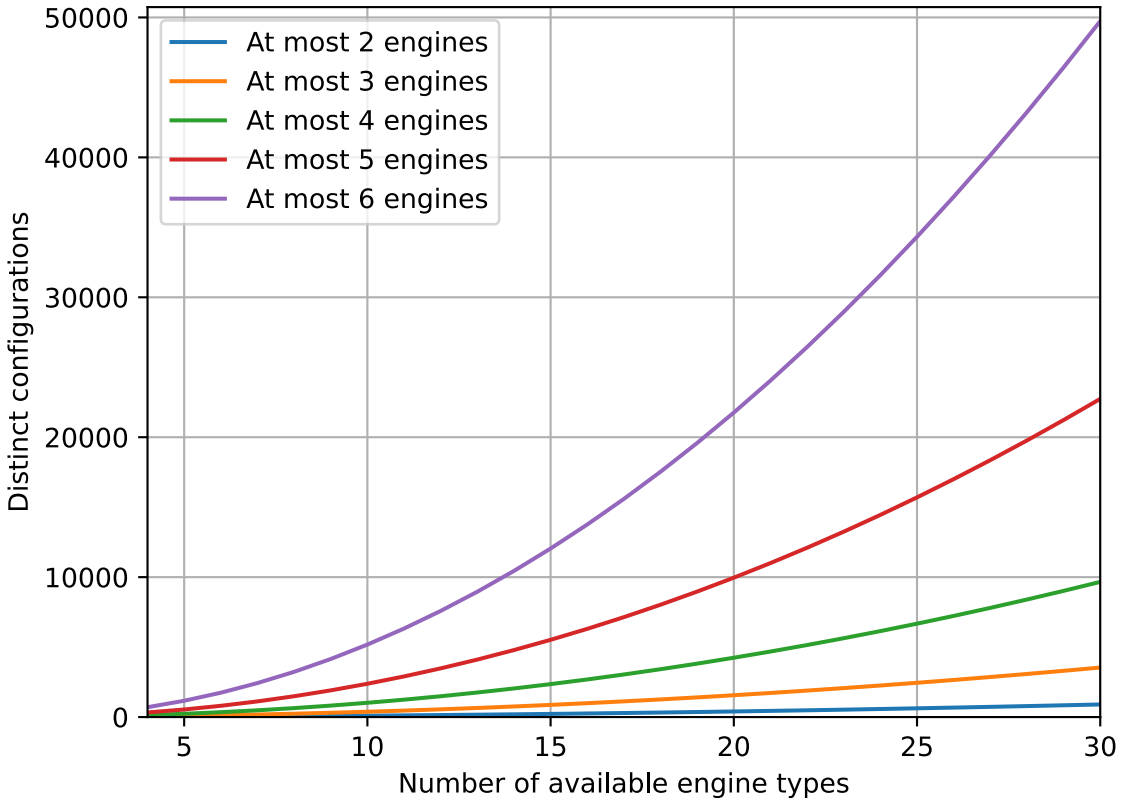

Figure 11: Number of configurations as a function of m and n

In the enumerative sampling approach, adding a new qualitative parameter with m_i levels multiplies the size of the dataset required for an even sampling by m_i. For a large model, this approach falls victim to the curse of dimensionality. The problem

with exponentially growing parameter spaces is that the number of samples required for training a globally representative surrogate model tends to become infeasibly large.

The number of configurations for different settings is visualized in Figure 11, which shows that the total number of configurations grows quickly as a function of available engine types when the number of engine slots is large, as is typical for larger ships. The enumerative sampling approach might therefore not scale well for larger problems.

5.4.2 Parametric sampling

Basic idea

In the parametric sampling approach, instead of treating each qualitative parameter as individual and non-numeric, some or all of the parameters are described using numeric hyperparameters. The approach is only applied to sampling, so the structure of the simulation model and the surrogate model are the same as when using enumerative sampling. While the values of the qualitative parameters are treated as non-numeric, the parametric sampling approach can provide the ability to sample a large, partially qualitative parameter space, in a space-filling manner without the need to include each configuration in the initial training set. While it is not necessarily sensible to describe sampling approaches in a partially qualitative parameter space as space-filling, there are some ways to approximately describe similarity between different designs. For example, a ship power plant with two engines of type *1* and one engine of type *2* can be assumed to perform more similarly as a design with three engines of type *1*, than one with four engines of type *4*. Clearly the parameterization of qualitative parameters is very problem-specific, and for some problems it might not be viable. The approach requires a hyperparameterization which translates the domain knowledge about the problem well.

Case study implementation

Number of engines	Probability
2	$\frac{1}{6}$
3	$\frac{2}{6}$
4	$\frac{3}{6}$

Table 5: Probability distribution for the total number of engines using sampling parameterization 1

As some of the qualitative parameters of the case study are dependent on the values of other parameters, a parametric approach might be suitable to be able to sample the parameter space. The objective of the sampling approach is to form a diverse dataset for the training of the surrogate model without having to have samples for each engine configuration. For the case study, the matrix T consists of one-hot column

vectors describing the selected engine types for each engine slot. The number of engines can change between two and four, and at most two different engine types can be selected. The first parameterization uses the types of engines present in a design, along with the number of both engine types. The order of the engines is the only information missing from this parameterization of T. The suggested hyperparameters are a, b, n_a, and n_b, representing the selected engine types and the number of those engines present, respectively. The parameterization forms a four-dimensional discrete hyperparameter space, where $a \in \{1, 2, 3, 4\}, b \in \{1, 2, 3, 4\}, n_a \in \{1, 2, 3\}, n_b \in \{1, 2, 3\}$, and $2 \leq n_a + n_b \leq 4$.

An alternative parameterization could consist of the engine types, (a, b), the total number of engines k, and the number of engines of type a, with $n_a \leq k$. The two parameterizations lead to different distributions in terms of the number of engines, as the first formulation assigns the number of engines a probability distribution that is skewed toward designs with more engines, as shown in Table 5, whereas the second parameterization leads to a uniform probability distribution in terms of number of engines. It is unclear which parameterization is preferable, as even though the number of feasible configurations increases when the number of engines in the configuration increases, the effect of additional engines is presumably smaller than that of the first two engines, as the usage rate of the engines is dependent on their position, as discussed in Section 5.1. It might therefore be reasonable to increase the amount of samples with fewer engines to try and increase the performance of the surrogate for such designs.

The first parameterization produces configurations with too many engines with a probability of $\frac{1}{3}$, meaning that more points needs to be sampled to get a sample size that matches that of the second parameterization. Both parameterizations share the sampling strategy for the engine types, and have a probability of $\frac{1}{4}$ for producing a configuration with only one engine type present. In reality, such configurations are common due to maintenance benefits, and it could be worthwhile to increase the share of such designs. Due to the fact that the first parameterization should lead to a more even distribution between all possible configurations, it is selected as the implementation of the parametric sampling approach.

If the engine types could be sorted in a manner which means that engine i is "more similar" to engine j than to engine k when $|j - i| < |k - i|$, the parameter space could be justifiably sampled using a space-filling algorithm. If no such ordering can be achieved, the parameters (a, b) are essentially sampled randomly, and no space-filling qualities are guaranteed. In the case study, each engine is defined by its fuel, fuel consumption curve, maximum continuous load, and MCR range. As such, there is no clear way to sort the engines. In this case study, the engines are sorted primarily by their fuel type, and secondarily by their maximum MCR. Their sorting can then be seen as a grouping, but there is no guarantee that engines i and $i+1$ can be seen as "similar". Still, treating the parameters as numeric does not seem to cause any harm, so a space-filling algorithm is used. An eight-dimensional LHS sampler is used for sampling points defining both the hyperparameters and $\mathbf{p}$. Since LHS produces continuous values, the first four elements of each sample are rounded down, providing an even distribution for the integer-valued hyperparameters. The order of

the columns of each engine type matrix T constructed using the parameterization is random. This means that some configurations might be unrepresented, especially if a small sample size is used. Since some of the produced samples are infeasible due to an excessive number of engines, the projection of the parameter space onto $\mathbf{p}$ might not be filled optimally. As only one sampler is used, the risks related to $\mathbf{p}$ being unevenly sampled for two- and three-engine configurations, as described in the previous section, are present.

5.4.3 Numeric parameter scaling

The numeric parameters in the model are represented by the MCR vector $\mathbf{p}$, whose size varies between two and four, depending on the number of selected engines. While the values of $\mathbf{p}$ present in the simulation model depend on the engine types represented by T, for sampling purposes, they can be considered as taking values from the closed interval $[0, 1]$. The sampled values can then be shifted and scaled according to the following equation:

$$p_i = m_{j_i}^{min} + \hat{p}_i \cdot (m_{j_i}^{max} - m_{j_i}^{min}), \tag{8}$$

where $\hat{\mathbf{p}}$ is the sampled vector, and $m_{j_i}^{min}$ and $m_{j_i}^{max}$ represent the minimum and maximum available MCR of the engine type in engine slot i, as listed in Table 2.

5.4.4 Comparison of sampling approaches

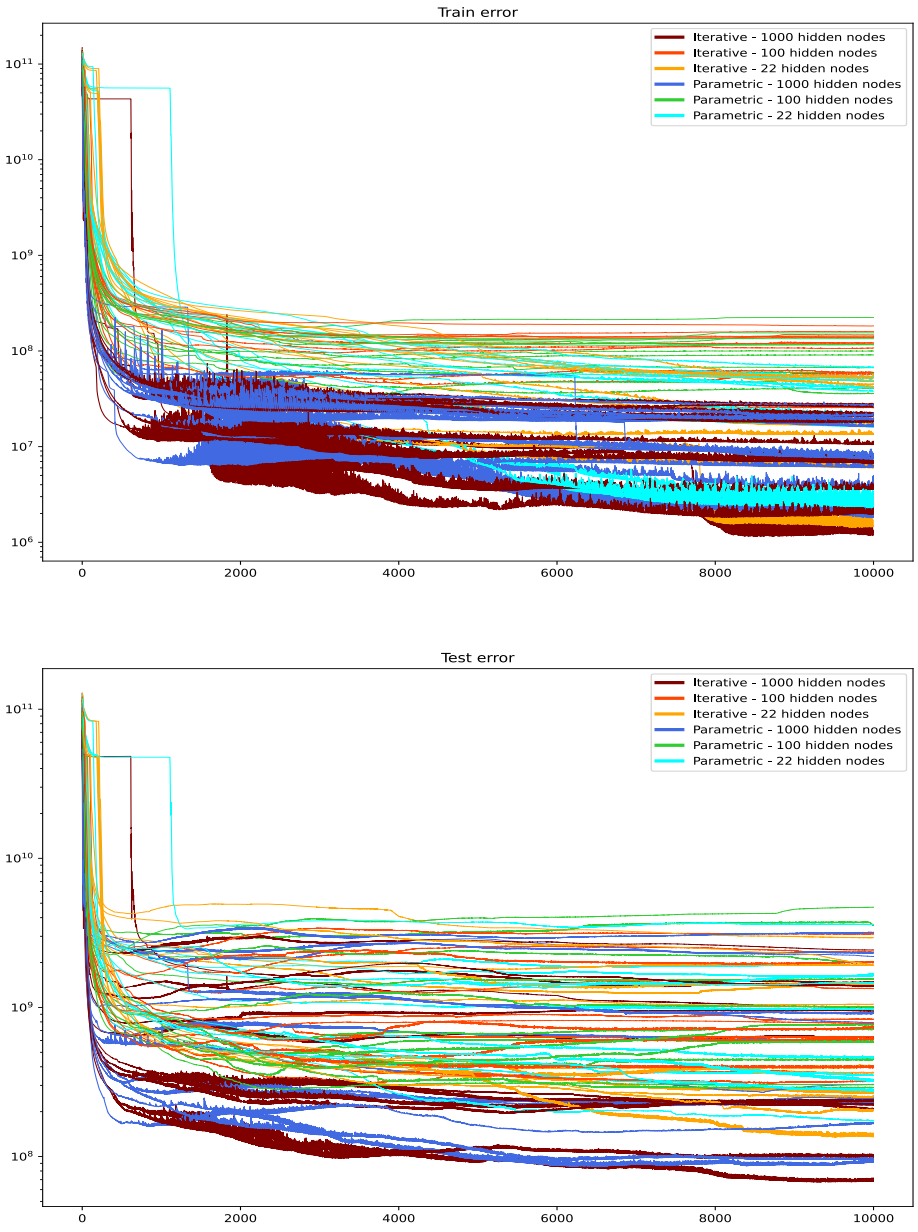

Figure 12: Convergence of sampling test models

To test how the two sampling approaches perform in the setting of the case study, a test is performed. In the test, the approaches are used to sample three sets of 100, 200, 600 and 1200 points. The validation set from Section 5.3 is reused. The training sets are all separately used to train a surrogate model with the three architectures selected in Section 5.3. The models are evaluated on the validation set, and the best-performing model is selected. Both the convergence of the training and the predictive ability are then compared for each sampling technique and sample size. The time spent on sample construction is not measured, because it is negligible in comparison to the time spent on simulation.

Sample size	Enumerative sample	Parametric sample
100	$1.5 \cdot 10^9$	$2.4 \cdot 10^9$
200	$6.8 \cdot 10^8$	$1.0 \cdot 10^9$
600	$2.2 \cdot 10^8$	$2.5 \cdot 10^8$
1200	$8.7 \cdot 10^7$	$1.6 \cdot 10^8$

Table 6: Sampling approach mean test errors

The results of the test are shown in Table 6 and Figure 12. The reported errors are the average of all three corresponding samples. The enumerative sampling strategy seems to outperform the parametric sampling strategy in all four sample sizes, although with larger sizes the difference gets smaller. Therefore, enumerative sampling is selected as the used sampling approach.

5.5 Active learning

In machine learning tasks, large datasets are often necessary for good model performance. As discussed in Section 2, ship design projects are iterative, so a model may need to be changed and analyzed several times. Therefore, minimizing the time spent on simulations is of vital importance. Usually, machine learning models are trained on a fixed set of n labeled samples $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i) | 1 \leq i \leq n\}$, attempting to learn an accurate approximation $\hat{f}$ of the mapping $f : \mathbf{X} \to \mathbf{Y}$, so that $\hat{f}(\mathbf{x}_i) \approx \mathbf{y}_i$. In *active learning*, instead of a single completely labeled set, there are two sets: one labeled set $\mathcal{L} = \{(\mathbf{x}_i, y_i) | 1 \leq i \leq n\}$, and one unlabeled set $\mathcal{U} = \{\mathbf{x}_i | 1 \leq i \leq N\}$, with $N \gg n$. The explanatory variables $\mathbf{x}$ of both sets are assumed to be subsets of the same set $\mathcal{X}$. The general idea of active learning is to first teach a model $\hat{f}$ using $\mathcal{L}$ as the training set, and then use information about $\hat{f}$ and $\mathcal{U}$ to select points in $\mathcal{U}$ to label. That is, to use the process f to determine $f(\mathbf{x})$ for the selected values of $\mathbf{x}$. A review of active learning is provided in Settles [2009], and a practical guide is given by Settles [2011]. In this system, the active learning routine is implemented using an initial labeled set $\mathcal{L}$ of size 400, sampled using the enumerative sampling approach described in Section 5.4. The set of 445 points described in Section 5.3 is used as the validation set $\mathcal{V}$. The unlabeled set $\mathcal{U}$ consists of all feasible parameter values.

There are various ways of selecting the queried points in an active learning setting, but generally some metric of expected potential improvement is used, or at the very

Algorithm 1 Active learning algorithm

1: Initialize $\hat{f}$ and train it using $\mathcal{L}$
2: $L(f) \coloneqq \max(||\mathbf{y} - f(\mathbf{x})||_1, (\mathbf{x}, \mathbf{y}) \in \mathcal{V}$
3: **while** $L(\hat{f}) > \sigma$ **do**
4: $\quad T^* \leftarrow \max_T(\text{median}(\{||\hat{f}(\mathbf{x}) - \mathbf{y}||_1 \mid \mathbf{x}[1:16] = T, (\mathbf{x}, \mathbf{y}) \in \mathcal{V}\}))$
5: $\quad$ Sample a set $\mathcal{U}^*$ of size $\lceil 0.05 \cdot |\mathcal{L}| \rceil$, where $\mathbf{x}[1:16] = T^* \; \forall \; \mathbf{x} \in \mathcal{U}^*$
6: $\quad$ Evaluate the simulation model for all points in $\mathcal{U}^*$ to obtain a labeled sample L^*
7: $\quad \mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{L}^*$
8: $\quad$ Train $\hat{f}$ using $\mathcal{L}$
9: **end while**
10: **return** $\hat{f}$

least hypothesized. In the setting of the case study, $\mathcal{U}$ is continuous in terms of $\mathbf{p}$ and discrete in T, as any feasible point can be simulated. It might make sense to develop a distance measure and look at areas where the model performs poorly in order to achieve greater improvement. However, due to the mixture of qualitative and numeric parameters, no such metric is used. Instead, a guess is made that the values of T have a correlation with the model performance. Although this assumption is not verified, there seems to be configurations for which the prediction error is significantly larger that the mean error. The entire active learning algorithm used in the case study is described in Algorithm 1.

Some parts of the Algorithm 1 warrant an explanation. Firstly, in order to comply with the objective to develop a robust surrogate model, maximum absolute loss is used to evaluate the performance of the model. The value of σ on line 3 of the algorithm is set to 10% of the mean value of $|\mathbf{y}|$, meaning that the algorithm is ran until the surrogate model prediction errors are at most 10% of the average fuel consumption.

Line 4 consists of calculating the absolute error for each point in the validation set, and selecting the engine type configuration with the largest median error among those points. The idea of the approach is to emphasize poorly performing configurations in the training set $\mathcal{L}$ in order to encourage the model to learn to represent those configurations.

As seen in line 5 of the algorithm, the size of new samples increases during training. This is due to the fact that as the size of $\mathbf{L}$ increases, an addition of constant size would have less effect on the training.

Lastly, in line 8, the existing model $\hat{f}$ is trained using the expanded training set $\mathcal{L}$. During testing, this approach was deemed to produce better performing models than re-initializing the model each time.

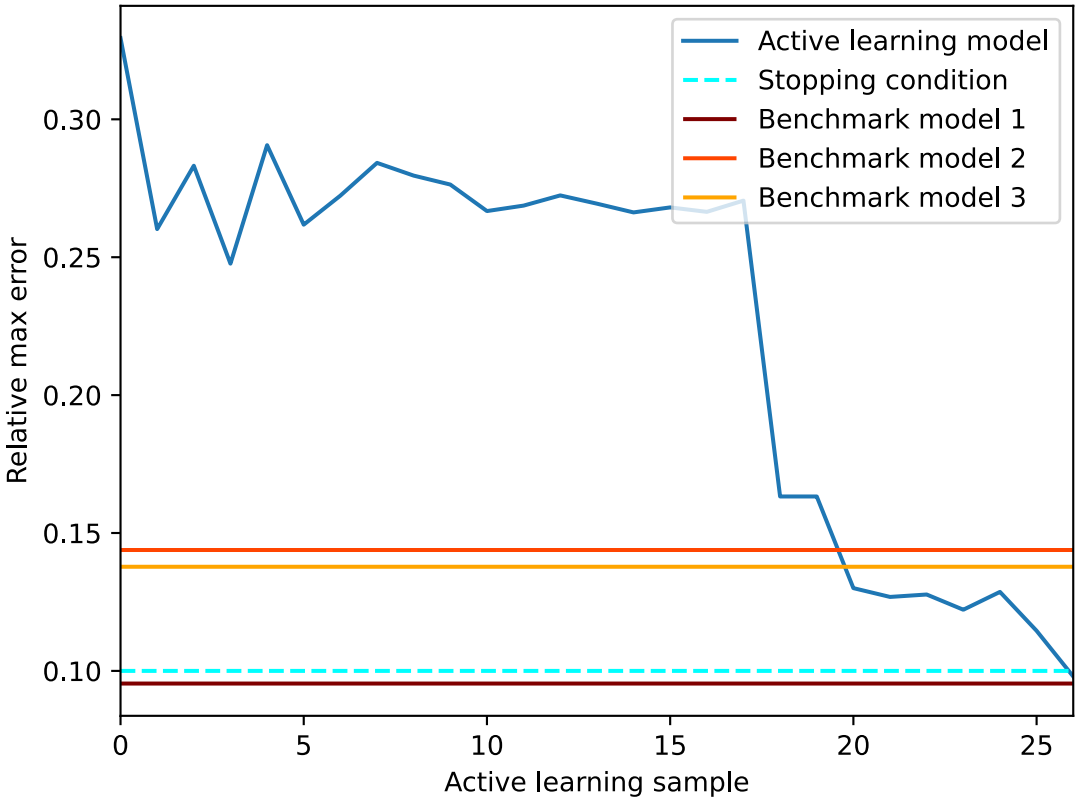

Figure 13: Active learning max error

The convergence of the model is shown in Figure 13, and shows the progression of the maximum error in the active learning routine, compared to three benchmark models trained on samples of size 1200, sampled using the enumerative sampling method. The final active learning model, which achieves a maximum relative error of 9.8% on the validation set, uses a training set consisting of 650 points. The set started with a 200-point set sampled using the enumerative sampling method, and a total of 27 new sets were added to it during the training process. When tested on another 445-point set, the maximum relative error is 13.6%. The benchmark models have a maximum relative test error of 10.1%, 14.4%, and 14.1% on this set, suggesting that the active learning approach manages to train a relatively robust model requiring barely half the number of simulations in comparison to the single-batch samples.

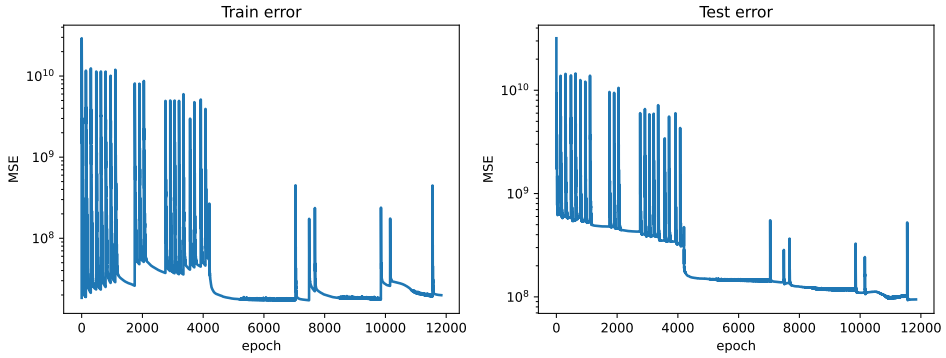

Figure 14: Active learning model convergence

Figure 14 shows the model training and test errors. As expected, the error spikes after adding samples to the training set, but then decreases rather quickly. Early stopping was used to avoid overfitting to the new data, which can be seen as a varying distances between the spikes.

6 Discussion

Although the approximated simulation model is quite simple, this project has demonstrated that an active learning approach seems to work well in building a robust neural network surrogate model for a ship energy systems model. While the surrogate is not used for analysis, there is no reason to believe that it is not suitable for such tasks. No sensitivity analysis was performed, but neural networks are smooth functions, and the performance of the surrogate on the two test sets are comparable, suggesting a decent robustness.

The usefulness of a routine such as the one developed in this thesis is ultimately limited by the accuracy of the approximated simulation model. It is therefore necessary to spend an adequate amount of resources on developing a simulation modeling framework to ensure the applicability of a surrogate modeling routine in ship design projects. While the routine was developed with ship power systems in mind, there is no reason the same approach should not be applicable for other problems, where there is a process with an input/output structure that can be queried at will.

The routine is also developed with scalability in mind, meaning that the addition of new parameters and components should be viable. In terms of energy system design, these new components could be new power producers or consumers, such as a battery, a shaft generator, or mechanical engines. Simulating other domains is also a possibility, for example by using computational fluid dynamics to represent the effect of the hull design on the power consumption. Models significantly larger than the one used in the case study could also be used, especially if the simulation model evaluations can be parallelized. The active learning scheme lends itself well to parallelization, especially as the time spent on neural network training tends to be significantly smaller than the time spent on simulating. With proper parallelization, the time spent on simulations in Section 5.5 after the evaluation of the initial 200 point set could be reduced by a factor of $\frac{650-200}{27} \approx 17$.

While all parts of the routine could certainly be improved, the major area of development is the active learning algorithm. Identifying a generalizable method for finding poorly performing areas, or areas with the largest expected improvement, would improve the entire routine considerably. The main challenge in selecting a general method is the combination of qualitative and quantitative variables.

Alternative formulations of the surrogate input data should be investigated. In the case study, the input vector size could have been reduced from 20 to 16 by inserting the MCR of each engine into the one-hot representation. The input could be formulated as

$$\hat{\mathbf{X}} \coloneqq \mathbf{T} diag(\mathbf{p}) \tag{9}$$

The input defined in Equation (9) was not used, as it is not as generalizable as the approach of flattening and concatenating all inputs. It was also hypothesized that the effect of $\mathbf{p}$ would have been harder to learn for this formulation than for the one used.

By changing the neural network class, different input structures could be used. For example, if a convolutional neural network was used, the input matrix would not

need to be flattened. While convolutional neural networks are most famously used for image inputs, their ability to take into account the distance between elements could also be useful when working with one-hot inputs.

7 Conclusion

In this thesis, the use of surrogate modeling for simulation-based analysis of ship power system design was explored. The basics of ship power system design were discussed, and the use of mathematical methods in design projects was reviewed. A potential for improvement in simulation-based analysis via the use of surrogate modeling was identified.

The objective of the thesis was to develop a proof-of-concept level implementation of a surrogate modeling routine to assist simulation-based design analysis. Generalizability and scalability of the routine were emphasized. A case study using mock data for an imaginary medium-sized cargo ship was conducted to support the development of the routine. In order to minimize the number of simulation model evaluations, space-filling sampling algorithms were examined, and an active learning approach was implemented. The surrogate modeling was conducted using neural network surrogates, as they were deemed to provide an accurate and scalable approximator.

Using active learning, a surrogate model trained with a set of 650 points achieved a similar maximum absolute prediction error as models using training sets of 1200 points. The model achieved a relative maximum error of around 14 %, with a mean squared error of $\approx 10^8$, with the simulation model mean output being between $2 \cdot 10^6$ and $3 \cdot 10^6$.

The proposed methodology facilitates robust analysis of simulation models, and should scale well for larger systems, provided adequate computational resources. Parallelization of the simulations can accelerate the entire routine considerably. The routine can provide a helpful tool to be used as part of a ship design process, although ultimately the usefulness of the system is dependent on the accuracy of the used simulation model.

References

United Nations Conference on Trade and Development. Review of maritime transport 2022, Nov 2022. URL https://unctad.org/system/files/official-document/rmt2022_en.pdf.

Anthony King. Emissions-free sailing is full steam ahead for ocean-going shipping, Sep 2022. URL https://ec.europa.eu/research-and-innovation/en/horizon-magazine/emissions-free-sailing-full-steam-ahead-ocean-going-shipping.

Francesco Baldi, Fredrik Ahlgren, Tuong-Van Nguyen, Marcus Thern, and Karin Andersson. Energy and exergy analysis of a cruise ship. *Energies*, 11(10):2508, 2018.

Francesco Baldi, Hannes Johnson, Cecilia Gabrielii, and Karin Andersson. Energy and exergy analysis of ship energy systems–the case study of a chemical tanker. *International Journal of Thermodynamics*, 18(2):82–93, 2015.

Gequn Shu, Youcai Liang, Haiqiao Wei, Hua Tian, Jian Zhao, and Lina Liu. A review of waste heat recovery on two-stroke ic engine aboard ships. *Renewable and Sustainable Energy Reviews*, 19:385–401, 2013.

Hai Lan, Shuli Wen, Ying-Yi Hong, C Yu David, and Lijun Zhang. Optimal sizing of hybrid pv/diesel/battery in ship power system. *Applied energy*, 158:26–34, 2015.

Ahad Al-Enazi, Eric C Okonkwo, Yusuf Bicer, and Tareq Al-Ansari. A review of cleaner alternative fuels for maritime transportation. *Energy Reports*, 7:1962–1985, 2021.

Christos A Frangopoulos. Developments, trends, and challenges in optimization of ship energy systems. *Applied Sciences*, 10(13):4639, 2020.

Francesco Baldi. *Modelling, analysis and optimisation of ship energy systems*. Chalmers University of Technology Gothenburg, Sweden, 2016.

Frances E Yates. Complexity and the limits to knowledge. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 235(5):R201–R204, 1978.

James Ladyman, James Lambert, and Karoline Wiesner. What is a complex system? *European Journal for Philosophy of Science*, 3:33–67, 2013.

Apostolos Papanikolaou. *A holistic approach to ship design*, volume 1. Springer, 2019.

Jari TJ Lappalainen, Timo Korvola, Jukka K Nurminen, Vesa Lepistö, and Tero Mäki-Jouppila. Cloud-based framework for simulation-based optimisation of ship energy systems. In *Proceedings of the 2nd International Conference on Modelling*

and Optimisation of Ship Energy Systems (MOSES2019). University of Strathclyde Publishing, 2019.

Nikoletta L Trivyza, Athanasios Rentizelas, and Gerasimos Theotokatos. A novel multi-objective decision support method for ship energy systems synthesis to enhance sustainability. *Energy Conversion and Management*, 168:128–149, 2018.

Nikoletta L Trivyza, Athanasios Rentizelas, and Gerasimos Theotokatos. Impact of carbon pricing on the cruise ship energy systems optimal configuration. *Energy*, 175:952–966, 2019.

Enrico Baldasso, Mia Elg, Fredrik Haglind, and Francesco Baldi. Comparative analysis of linear and non-linear programming techniques for the optimization of ship machinery systems. *Journal of Marine Science and Engineering*, 7(11):403, 2019.

J Harvey Evans. Basic design concepts. *Journal of the American Society for Naval Engineers*, 71(4):671–678, 1959.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

Patrick J Driscoll, Gregory S Parnell, and Dale L Henderson. *Decision Making in Systems Engineering and Management, 3rd Edition*, chapter 2.7.1. Wiley, 2022. ISBN 1-119-90140-5.

Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 0210–0215. IEEE, 2018.

Octavio Loyola-Gonzalez. Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view. *IEEE access*, 7:154096–154113, 2019.

Jeffrey Goldstein. Emergence in complex systems. *The sage handbook of complexity and management*, pages 65–78, 2011.

R.A. Fisher. *The Design of Experiments*. Oliver and Boyd, 1935.

Averill M Law. A tutorial on design of experiments for simulation modeling. In *2017 Winter Simulation Conference (WSC)*, pages 550–564. IEEE, 2017.

W David Kelton. Experimental design for simulation. In *2000 Winter Simulation Conference Proceedings (Cat. No. 00CH37165)*, volume 1, pages 32–38. IEEE, 2000.

Thomas J Santner, Brian J Williams, William I Notz, and Brain J Williams. *The design and analysis of computer experiments*, volume 1. Springer, 2003.

Jerry Banks. *Handbook of simulation: principles, methodology, advances, applications, and practice*. John Wiley & Sons, 1998.

Atharv Bhosekar and Marianthi Ierapetritou. Advances in surrogate based modeling, feasibility analysis, and optimization: A review. *Computers & Chemical Engineering*, 108:250–267, 2018.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Claudio Angione, Eric Silverman, and Elisabeth Yaneske. Using machine learning as a surrogate model for agent-based simulations. *Plos one*, 17(2):e0263150, 2022.

Francesco Lamperti, Andrea Roventini, and Amir Sani. Agent-based model calibration using machine learning surrogates. *Journal of Economic Dynamics and Control*, 90:366–389, 2018.

Ross Anderson, Joey Huchette, Will Ma, Christian Tjandraatmadja, and Juan Pablo Vielma. Strong mixed-integer programming formulations for trained neural networks. *Mathematical Programming*, 183(1-2):3–39, 2020.

Peter Fritzson and Vadim Engelson. Modelica—a unified object-oriented language for system modeling and simulation. In *ECOOP'98—Object-Oriented Programming: 12th European Conference Brussels, Belgium, July 20–24, 1998 Proceedings 12*, pages 67–90. Springer, 1998.

Gerald Schweiger, Henrik Nilsson, Josef Schoeggl, Wolfgang Birk, and Alfred Posch. Modeling and simulation of large-scale systems: A systematic comparison of modeling paradigms. *Applied Mathematics and Computation*, 365:124713, 2020.

Shuxiang Xu and Ling Chen. A novel approach for determining the optimal number of hidden layer neurons for fnn's and its application in data mining. 2008.

A. Blum. *Neural Networks in C++: An Object-oriented Framework for Building Connectionist Systems*. Number v. 1 in Neural Networks in C++: An Object-oriented Framework for Building Connectionist Systems. Wiley, 1992. ISBN 9780471552017. URL https://books.google.fi/books?id=9H0pAQAAMAAJ.

Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760*, 2018.

Francesco Ceccon, Jordan Jalving, Joshua Haddad, Alexander Thebelt, Calvin Tsay, Carl D Laird, and Ruth Misener. Omlt: Optimization & machine learning toolkit. *The Journal of Machine Learning Research*, 23(1):15829–15836, 2022.

Pande Lab at Stanford University. Some dangers of label normalization in ml, May 2018. URL https://medium.com/@pandelab/some-dangers-of-label-normalization-in-ml-6482d955882e.

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

Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Sebastian Burhenne, Dirk Jacob, Gregor P Henze, et al. Sampling based on sobol'sequences for monte carlo techniques applied to building simulations. In *Proc. Int. Conf. Build. Simulat*, pages 1816–1823, 2011.

Michael D McKay, Richard J Beckman, and William J Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000.

Il'ya Meerovich Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967.

Burr Settles. Active learning literature survey. 2009.

Burr Settles. From theories to queries: Active learning in practice. In *Active learning and experimental design workshop in conjunction with AISTATS 2010*, pages 1–18. JMLR Workshop and Conference Proceedings, 2011.