Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

Herman Skogseth

Temporary zero emission operation on cruise ships

Master's thesis in Applied Physics and Mathematics
Supervisor: Jon Andreas Støvneng
Co-supervisor: Armin Hafner, Cecilia Gabrielii
June 2022

NTNU
Norwegian University of
Science and Technology

Herman Skogseth

Temporary zero emission operation on cruise ships

Master's thesis in Applied Physics and Mathematics
Supervisor: Jon Andreas Støvneng
Co-supervisor: Armin Hafner, Cecilia Gabrielii
June 2022

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

NTNU
Norwegian University of
Science and Technology

Preface

This thesis has been done in close collaboration with SINTEF Energi AS, and in particular with the CruiZE project led by Cecilia Gabrielii. The master thesis has been done as an extension of the specialization project in the fall of 2021 as well as the work I did as part of CruiZE the summer of 2021. I want to thank SINTEF Energi AS for all the help and resources that I have received, you have been very kind.

I want to specially thank my external supervisor Cecilia Gabrielii for all the help and for all the great and insightful conversations we have had about cruise ships and thermal systems. I also want to thank all the other wonderful people who have helped me out at SINTEF, especially Sigurd Sannan and the help I received from him during my time at SINTEF in the summer of 2021.

From NTNU I would like to specially thank Armin Hafner, from the Department of Energy and Process Engineering (EPT), for his great engagement in the project, and for the pointers and inspiration he has given me. Muhammad Zahid Saeed has also been of great help. I also want to thank my internal supervisor at the Department of Physics: Jon Andreas Støvneng. Thank you for giving me the freedom to study this.

My final thank you goes out to everyone who has supported me during the last five years, thank you for being there. In particular I want to thank: My three partners in physics Torstein, Sondre & Amund, my friends Martin & Sindre, my parents Ingrid & Toro, my dog Mabel, and my wonderful girlfriend Anne Joo.

Abstract

Cruise ships are currently facing increasing demands with respect to their emissions, particularly on the local scale. One example of this is that the Norwegian government has decided that the world heritage fjords, such as Geirangerfjorden, will be zero emission zones by 2026. It is therefore of interest to study the possibility for traditional cruise ships, read diesel-powered, to temporarily operate with zero emissions. This thesis will mainly focus on achieving this in port, but the solutions presented will be just as valid for short cruising, if one were to add a battery.

A big problem standing in the way of temporary zero emission operation for cruise ships is how to cover the heat demands of the ship since this is, under normal operation, usually covered by utilizing the waste heat from the diesel generators. This thesis will study various solutions utilizing thermal energy storage, as well as heat pumps/electric boilers, to cover these heat demands. A dynamical model of the thermal energy system on traditional cruise ships was created, using Dymola/Modelica, to properly study these solutions.

The simulations show very positive results for meeting the demands for high temperature water, particularly when using thermal energy storage based on phase change materials. For the pillow-plate based design a volume of merely 265 m^3 could cover all heat demands in port! Doing this with a water tank required a volume of 850 m^3. The simulations run does not, however, show great results for the steam. For the case ship used in the simulations there was too little waste heat available from steam to solve the heat demand using a thermal energy storage. Instead, an electric boiler was used, ant it was found that it needed to deliver an average power of 3.4 MW. However, the challenges of simulating the steam has given significant uncertainty to these results.

Sammendrag

Cruiseskip-industrien møter for tiden økende krav når det kommer til utslipp, spesielt på lokal bane. Et eksempel på dette er at de norske myndighetene har bestemt at alle verdensarvfjorder, inkludert Geirangerfjorden, skal være nullutslippssoner innen 2026. Det er derfor interessant å studere muligheten for tradisjonelle cruise skip (les diesel-drevet) å tidvis operere med nullutslipp. Denne oppgaven vil i hovedsak fokusere på å oppnå dette når skipet ligger til havn, men løsningene som vil bli diskutert er like gyldige for å oppnå dette også for korte tider på havet. Det er da naturligvis nødvendig å kombinere løsningene med et elektrisk batteri.

Et stort hinder som står i veien for at tradisjonelle cruiseskip kan operere med nullutslipp er å møte varmebehovet til skipet, ettersom dette under vanlig drift dekkes av restvarme fra dieselmotorene. Denne oppgaven vil ta for seg løsninger hvor man bruker termiske energilagre, i tillegg til varmepumper/elektriske varmtvannsberedere, til å møte skipets varmebehov. For å studere disse løsningene ble det laget en dynamisk modell av det termiske energisystemet på tradisjonelle cruiseskip i Dymola/Modelica.

Simuleringene har vist svært gode resultater for å møte varmtvannsbehovene til skipet, spesielt når det termiske energilageret var basert på faseendringsmaterialer. Det var da mulig å møte alle varmtvannsbehovene i havn med et volum på kun 265 m^3, gitt at varmeveklingsdesignet var basert på plater av typen ”pillow-plate”. En vanntank trengte et volum på 850 m^3 for å møte disse behovene. Simuleringene ga derimot ikke spesielt bra resultater for vanndamp. Skipet som ble studert i simuleringene hadde for lite restvarme, i form av vanndamp, under cruising til å dekke behovene i havn med et energilager. Det ble derfor kjørt en simulering for å studere potensialet bak å bruke elektrisk varmtvannsbereder, som viste at varmtvannsberederen måtte levere en gjennomsnittlig effekt på omtrent 3.4 MW. Men, utfordringene knyttet til det å simulere vanndampen har skapt stor usikkerhet rundt disse resultatene.

Contents

Acronyms

CoP Coefficient of Performance. 17, 46–48, 61, 89, 97

DAE Differential Algebraic Equation. 18–21, 69

DG Diesel Generator. 1, 2, 7, 21, 23, 24, 84, 85, 87, 88, 93, 95, 108, 111

EGB Exhaust Gas Boiler. 6, 7, 49–51, 55, 59, 76–80, 83–85, 87, 88, 94, 95

FWG Fresh Water Generator. 5–7, 87, 88

GUI Graphical User Interface. 23, 90, 91, 93, 95

HP Heat Pump. 27, 81, 82

HT High Temperature. 1, 2, 5–7, 23, 24, 27–29, 32–35, 44–47, 49, 51, 55–59, 61, 83, 84, 87, 88, 90, 93, 98, 108, 111

IDE Integrated Development Environment. 21

LT Low Temperature. 5, 6, 84, 87, 93, 108, 111

ODE Ordinary Differential Equation. 20

OFB Oil Fired Boiler. 1, 2

PCM Phase Change Material. 5, 9–11, 27, 33, 35–45, 53–58, 60, 61

PHE Plate Heat Exchanger. 10, 53, 54

PPHE Pillow-Plate Heat Exchanger. 11, 53, 54

TES Thermal Energy Storage. 2, 5, 7–9, 28, 30–32, 34, 35, 41, 51, 53–55, 57, 58, 60, 61, 75, 76, 90–92, 98

VLE Vapour Liquid Equilibrium. 22, 23, 49, 55, 56, 59, 88, 103

Chapter 1

Introduction

In this chapter three things will be discussed: The general idea behind the thesis, the motivation for the thesis and the choice of format. The section covering the general idea behind the thesis will be a general introduction into the problems that will be tackled in the thesis, as well as the general notion of the solutions presented to solve these. When it comes to motivation, the project in itself will obviously be discussed, but it will also be put into perspective with its larger role within the CruiZE project. The format section will mostly focus on the ways in which this thesis deviates from common formats and why large segments of the content reside in the appendices. This chapter is largely based on the introduction written for the specialization project done prior to this master thesis. The work done here is done in continuation of that work, so there will be significant overlap between the two theses. When segments from the prior thesis is reused for this thesis it will, similarly to here, be pointed out in the introductory segment of that chapter.

1.1 Idea

Traditional cruise ships, and their thermal energy systems, will be discussed in detail in section 2.1, but a brief summary is given here: A set of Diesel Generators (DGs) produce electricity for propulsion as well as the hotel facilities of the ship. They also produce waste heat, in form of exhaust gas and High Temperature (HT) water. The exhaust gas is used to produce steam, which is used to cover many of the heat demands on the cruise ship. Some of the heat demand is also covered by the HT water.

The problem that this project is tackling is how cruise ships can meet the heat demands in port with zero emissions. The electricity demand can often be covered by shore power, but the shore power is limited; it cannot necessarily cover the heat demands as well. Currently, the most common solution for covering heat demands in port on cruise ships is to run Diesel Generators and/or Oil Fired Boilers, both of which have a large amount of emissions. This project will study the following, alternative solution:

Use Thermal Energy Storage (TES) to store excess heat from HT water and steam when cruising, and use the stored energy to cover heat demands in port. Solutions using heat pumps and/or electric boilers, either alone or in combination with TES, will also be studied and/or discussed.

1.2 Motivation

As mentioned earlier this project has been done in close collaboration with SINTEF Energi AS, and in particular the CruiZE project led by Cecilia Gabrielii. The CruiZE project is described, in the project description written by C. Gabrielii, in the following way: ”The goal is to develop energy-efficient technologies towards a more environment-friendly cruise industry. The main focus of the project is to suggest innovative heating and cooling concepts, optimised for the ships’ propulsion system and varying operating conditions, that can enable zero emissions in ports, minimised emissions at sea, and an average reduction of the ships’ total energy usage by 10-20%.”[1]

This project focuses on a few of the areas described here, first and foremost innovative heating concepts for enabling zero emissions in port. Whilst the solutions presented in this thesis is focusing on this they will also as a consequence lower the total emissions of the cruise ships: Not running Diesel Generators and Oil Fired Boilers will naturally help lower the total emissions from the cruise ships. In addition, the solutions using TES *could* be used as a buffer for heat use when cruising, which could help lower peak demands if implemented correctly, although this use in not studied here. Hopefully this thesis can shed some light on the potential for solutions using TES on cruise ships, whilst also building a model that can be utilized to study thermal energy systems on other passenger ships.

1.3 Format

The format of the thesis follows, for the most part, that of a traditional one. The literary review, a segment sometimes excluded in this type of thesis, is included to go through background information needed on the thermal energy systems of cruise ships as well as information on thermal energy storage. It will also, very briefly, mention some of the relevant studies done prior to this project.

However, a bigger deviation from common formats is Chapter 4: Analysis and Results. The analyses used to find the results in many theses is something that often can be classified as separate from the results, and would be included as part of the method. This was not considered to be the case for the work done here. There is no coherent strategy for the majority of the analysis, it is shaped by the results in a very fluid way. Separating the two became impractical and, most importantly, more confusing format-wise. The discussion was, however, separated from both of these as it did not

rely on an understanding of the analysis. Key results are repeated in the discussion chapter as their relevance shows up.

Another point to mention here is that a relatively large portion of this thesis resides in the appendices. The thesis is heavily based on simulation models made in Dymola/Modelica, and although this has been central to the work done, most of it is not fundamentally important to the final results of the thesis. For this reason the numerical models are only mentioned briefly throughout the thesis when relevant. The numerical model is instead explained in detail in the appendices.

Chapter 2

Literary review

This chapter will largely focus on giving background information on important systems used in this thesis. The first section fill focus on thermal energy systems on cruise ships, including some relevant work done in this field. This section has been taken from the specialization project. The sections following will discuss thermal energy storage, with a large focus on PCMs.

2.1 Thermal energy systems on cruise ships

Cruise ships are quite complicated energy wise. They need large amounts of energy in various forms, for various purposes, all dynamically changing depending on the behaviour of the passengers and crew. That description, and much of this following section, is based on an internal project memo from the CruiZE project at SINTEF Energi AS [2]. Figure 2.1 gives a good illustration of the thermal energy system of traditional cruise ships. The main heat flows of interest for this report are those of High Temperature (HT) water and steam. Low Temperature (LT) water is also needed in the engine model, but as one can easily see from the figure the LT water is not used for heat recovery, so its importance is minimal.

The HT water is used for three purposes in the cruise ship studied in this report:

- Fresh water generation
- Air conditioning
- Potable water preheating

Potable water is clean, drinkable water at high temperatures, typically between 60°C and 70°C. When cruise ships produce potable water they typically take in clean water stored in tanks, usually generated by the Fresh Water Generators (FWGs), and preheat it using HT water before heating it to the desired temperature using steam. Alternative solutions using a Thermal Energy Storage have been studied [4].

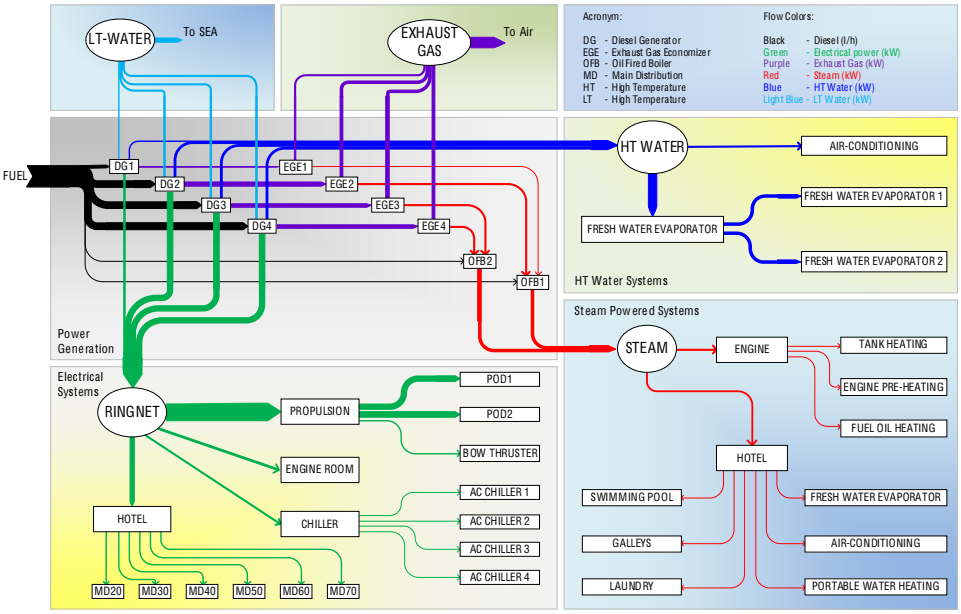


Figure 2.1: Thermal energy system of a traditional cruise ship. A set of diesel generators produce electricity, which is used for propulsion, hotel facilities, etc. Heat is recovered from the diesel generators in form of High Temperature water, and using Exhaust Gas Boilers the exhaust gas is utilized to generate steam. [3]

As can also be seen from figure 2.1 there several steam users on cruise ships, many of which consist of delivering some service as part of the hotel facilities on the ship. This includes, among others, fresh water generation, potable water heating, swimming pools and laundry. For the data used in this case this is simply split into three categories: Potable water heating, fresh water generation and other.

In order to accurately model the engine and heat recovery system of the cruise ship, SINTEF was given access to confidential schematics of the systems. These schematics can, obviously, not be shared here. However, a schematic was created by Cecilia Gabrielii to retain the core functionality of the system without compromising the integrity of the confidentiality agreement. That schematic can be seen in figure 2.2. The figure only shows engine 4 to 6 (top left) and heat recovery for HT water (middle and lower segment). Engine 1 to 3 are set up similar to engine 4 to 6. Note that the engine system interacts with heat recovery using two heat exchangers: One for engines 1 through 3 (middle, slightly to the right) and one for engines 4 through 6 (middle, slightly to the left). This water is then combined and subsequently split before its used in air conditioning (bottom left) and potable water preheating (bottom right). Another important note is the FWGs: There is, similarly to the heat exchangers for HT recovery, one of these for each ”triplet” of engines (1-3 & 4-6). Each triplet also has its own cooler, the one for engines 4-6 can be seen in the top middle of the figure (2.2). The LT cooling system is not shown in this figure, nor are the EGBs. In this report it is assumed that the LT cooling system function similarly to the HT

cooling system: One cooler for each engine triplet. The six DGs are, in addition, all hooked up to their own EGB for steam production[5].

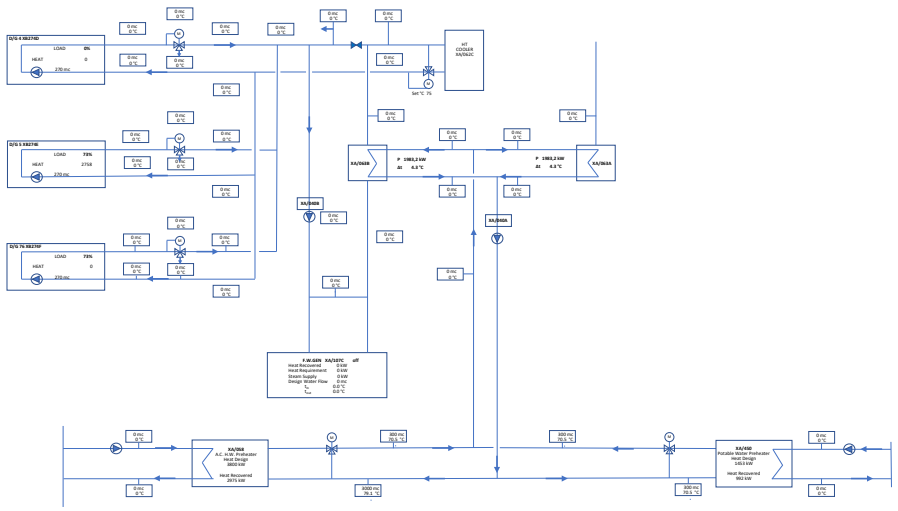


Figure 2.2: Model showing the flow of High Temperature (HT) water for the engine system of the case ship, as well as the HT recovery system. The model shows only the engine triplet of DGs 4, 5 & 6 (on the upper left of the figure), the triplet of DGs 1, 2 & 3 and its corresponding HT flow system is a mirror image of this triplet. The model also shows the location of Fresh Water Generator and cooler for triplet 4, 5 & 6, the separation of heat exchange with air conditioning (lower left) and potable water preheating (lower right) in the HT recovery system, as well as some basic information for *some* of the water flows. This is a recreation of a set of blueprints for the case ship given by Carnival Corporation & plc, modified by Cecilia Gabrielii to uphold confidentiality agreement. This model retains the fundamental functionality of the engine system without remaining true to the accurate details of it.

2.2 Thermal Energy Storage

A fundamental concept used throughout this thesis is that of the Thermal Energy Storage (TES). These types of energy storage are currently seeing increased use in industrial applications as society is moving towards more effective energy use as well as more fluctuating energy sources (e.g. moving from coal to wind), as this naturally brings with it the need for an energy buffer to cover heat demands when energy availability is low or expensive. However, the use of TES on passenger ships is still limited.

There are many reasons for utilising a TES over other types of energy storage, such as electrical storage (batteries). The most important reason, in many cases, is that the energy in question is in the form of heat. It's easier to store heat directly in a Thermal Energy Storage than converting it to electric energy, then store it in a battery, then

convert it back to heat when it is needed again. Converting via electric energy also carries with it significant losses[6].

There are mainly two types of Thermal Energy Storage: Sensible heat storage and latent heat storage. These will be discussed in the following two sections.

2.3 Sensible Heat Storage

A sensible heat storage uses *sensible heat*, meaning the energy change is connected to the changing temperature of the material. This relationship is defined by the heat capacity C of the object:

$$C = \frac{\mathrm{d}Q}{\mathrm{d}T}, \tag{2.1}$$

where $\mathrm{d}Q$ is the heat flow into the system and $\mathrm{d}T$ is the corresponding change in temperature.

The most common type of sensible heat storage is the water tank, which is also the only type that will be used in this thesis. A water tank can function in many different ways, but will in this thesis function in the following way: The water tank will have a volume V_{wt} which is constantly filled with water at a given temperature T_{wt}. Flowing water can then be ”exchanged” with the water in the tank, for example if there is excess water available with a temperature $T_{flow} > T_{wt}$ then this water can flow into the water tank and an equivalent amount of water with temperature T_{wt} will flow out. The water flowing in will then have replaced colder water, meaning the temperature of the water in the tank will rise. The water in the tank will be perfectly mixed at all times in the models used.

2.4 Latent Heat Storage

A latent heat storage is, naturally, based on the principle of *latent heat*, meaning the energy required for a material to change phase. The latent heat L of a phase transition is defined in the following way:

$$L = \frac{Q}{m}, \tag{2.2}$$

where Q is the amount of heat needed for all of the material to phase transition and m is the total mass. This means the latent heat L of a material gives us the energy stored in a phase transition for that material. The most common phase change used in latent heat storage is melting (solid to liquid) and its opposite solidification (liquid to solid).

2.4.1 Materials

All latent heat storage require some sort of Phase Change Material (PCM) for storing energy, it is therefore common to simply refer to this type of TES as a PCM. This is technically a bit inaccurate, and will inevitably lead to some confusion. It is nevertheless done in this thesis as well. Instead, when referring to the actual phase change *material*, this will be done by simply referring to it as the "material of the PCM". Please forgive this somewhat awkward convention. So: The material used in the PCMs throughout this thesis can be seen in table 2.1. These materials, along with the key parameters which can be seen in table 2.1, were taken from a long list of known materials put together by SINTEF. The materials were chosen based on their temperature as well as how many of these key parameters were known and included in the list. The numbers in parenthesis were unknown, and were estimated based on other materials with a similar melting temperature within the same category of material (e.g. organic).

Table 2.1: Table showing key values for all materials used, or discussed, as Phase Change Materials (PCMs) in this thesis. Values in parenthesis were not available and are estimates based on typical values for PCMs in this temperature range for other materials of similar type (e.g. organic). T_m is the melting temperature of the material, whilst L is the latent heat. For the remaining values: Subscript "s" denotes values for solid phase whilst "l" denotes values for liquid phase. ρ is the density of the material, c_p is the specific heat capacity (heat capacity per mass unit) under constant pressure and λ is the thermal conductivity. The physics behind many of these quantities, specifically those related to thermodynamics, is studied further in section 3.1.

	PureTemp 68	CrodaTherm™ 74
T_m[°C]	68	74.8
L[kJ/kg]	213	224
ρ_s[kg/m^3]	960	939
ρ_l[kg/m^3]	870	858
$c_{p,s}$[kJ/(kgK)]	1.85	(2.00)
$c_{p,l}$[kJ/(kgK)]	1.91	(2.00)
λ_s[W/(mK)]	0.25	(0.20)
λ_l[W/(mK)]	0.15	(0.20)

In some sense its an important point to make that a PCM will act as a sensible heat storage outside its latent area, meaning when the temperature is either higher or lower than the melting temperature energy change in the PCM will lead to a change in temperature. This is why two of the key parameters for the materials are the specific heat capacities for solid and liquid phase; as was explained in section 2.3 the (specific) heat capacity of a material determines how much energy must be added to/removed from a material in order to change its temperature.

2.4.2 Heat exchange

A PCM requires not only a material to store the energy in, but also a way of exchanging heat between the working fluid (in our case water) and the material. The two strategies for heat exchange that will be used in this thesis are plate in block and pillow-plate in block. These have been chosen due to their high rate of heat exchange. Why this is necessary is discussed in chapter 4.

The plate in block based PCM utilizes a set of plates, stacked in one direction (in this thesis the width of the PCM, with fluid flowing through the ducts of the plates. This design structure gives a large amount of heat exchange area compared to, for example, a tube in block based design. The material in the PCM lies in blocks between the plates. An image showing a visualization of this design can be seen in figure 2.3. The figure is taken from the corresponding PCM model from the TIL library for Dymola/Modelica, more on that in section 3.2. This type of PCM is also referred to as a Plate Heat Exchanger (PHE) PCM.

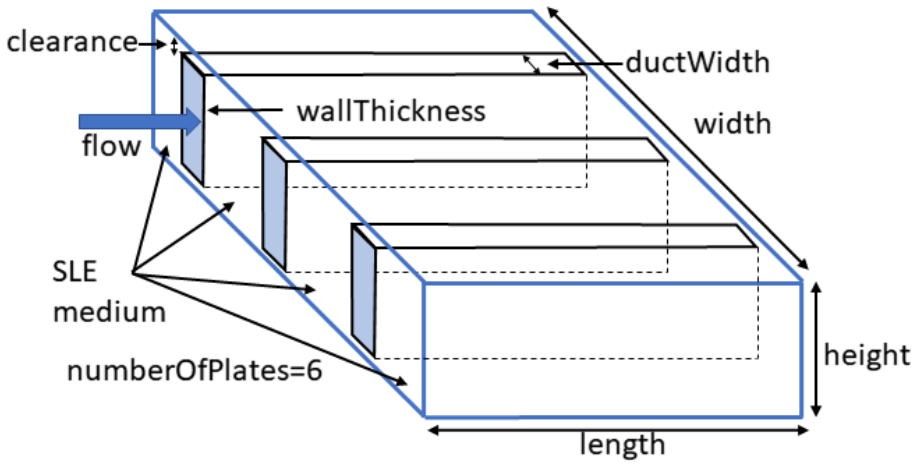


Figure 2.3: Visualization of Plate Heat Exchanger PCM, taken from the corresponding model in the TIL library for Dymola/Modelica (see section 3.2). Many of the quantities shown in this figure will become important later, such as the duct width d, wall thickness w and clearance Δ.. Note also in which direction the plates are stacked, namely in the width of the PCM.[1]

[1]Please ignore the fact that this figure shows the number of plates to be equal to 6 (one for each side of each flow). Simulations run using this model indicates that this is an error, and that the correct number of plates in this image should be 3 (one for each flow).

The pillow-plate in block based PCM, sometimes referred to as Pillow-Plate Heat Exchanger (PPHE) PCM, is very similar to the plate in block PCM. The main difference is, unsurprisingly, related to the plate design. A visualization of the pillow-plate design can be seen in figure 2.4. Note, in particular, that the terminology for the plate design remains the same, even if the meaning of the terms changes slightly. The numerical models for the pillow-plate PCM are based on a review done on PPHE PCMs [7].

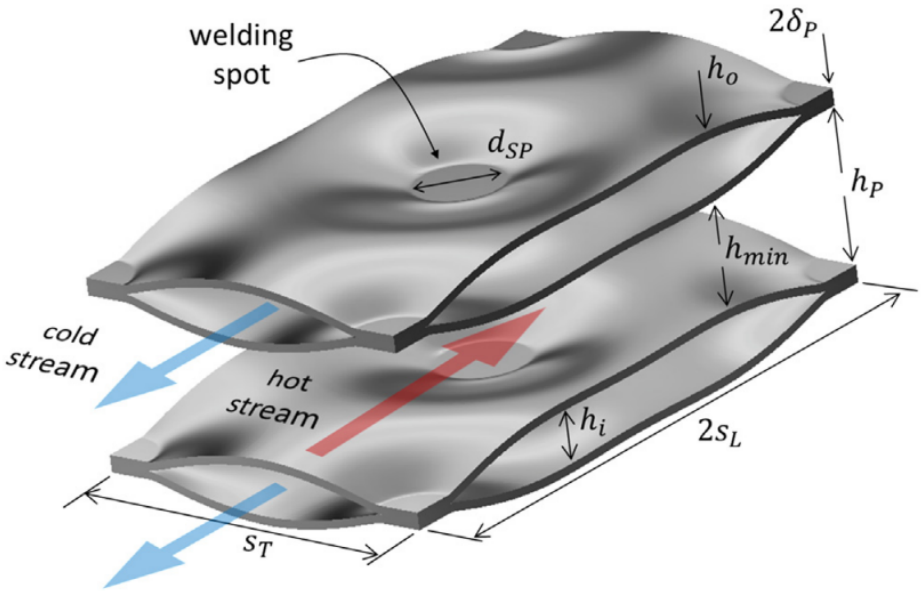


Figure 2.4: Figure showing pillow-plate design in a Pillow-Plate Heat Exchanger (PPHE)[8]. This is the design used by Pillow-Plate Heat Exchangers, not PPHE PCMs, which is why there are two plates and not one. The design of the plates, however, remain the same. The physical quantities shown here, which are significant to this thesis, is the duct width h_i and the wall thickness δ_P. To avoid confusion with other quantities these are renamed to d and w, respectively.

Chapter 3

Theory and method

This section will discuss the theory needed in this report and the methods/numerical tools used in the models. The theory focuses on mainly one area: Thermodynamics. The methods cover mainly the numerical tools chosen to build the model and the case used as basis for the tests. This chapter is based on the corresponding chapter from the thesis of the specialization project, however, it contains significant changes and additions. The section on numerical tools remain unchanged.

3.1 Thermodynamics

This section will briefly go over some of the basic thermodynamics needed in this project. The section is largely based on various segments from ”Termisk fysikk”, by P. C. Hemmer[6], with some segments being based on the Mechanical Engineeer's Data Handbook (most importantly the heat transfer section)[9].

3.1.1 General

A good place to start with any discussion on thermal systems is with the first law of thermodynamics, which is usually defined via its infinitesimal form:

$$\mathrm{d}Q = \mathrm{d}U + \mathrm{d}W\,, \tag{3.1}$$

where $\mathrm{d}Q$ is heat flow into the system, $\mathrm{d}U$ is the change in internal energy and $\mathrm{d}W$ is work done by the system on the environment. Really, this is just defining heat as a type of energy, which must be included in calculations using conservation of energy. The second law of thermodynamics will not be used for any calculations, but forms the basis for the entire thesis. Although this law has many formulations, the following was chosen for use here: The energy of a closed system remains constant, and the entropy tends towards a maximum ([6] p. 60).

The entropy of a system S, although commonly referred to as the "chaos of a system", is best understood in this thesis using the context of its origins in classical thermodynamics. Entropy was introduced to understand when engines can do work, and is commonly attributed to two main formulations: Heat can only flow from a reservoir "A" to a reservoir "B" if $T_A > T_B$ (Clausius' formulation) and no process can occur where the only result is heat energy being fully converted to work (Kelvin's formulation). The entropy is defined, through classical thermodynamics, in the following way:

$$\mathrm{d}S = \frac{\mathrm{d}Q_{rev}}{T}. \tag{3.2}$$

The first and second law of thermodynamics can be formulated into one central equation, commonly known as *the thermodynamic identity*. This equation is based on two main assumptions: The given process is reversible, meaning $\mathrm{d}Q = T\,\mathrm{d}S$, and the work done by the system is of the form $P\,\mathrm{d}V$. Equation (3.1) can then be rewritten as

$$T\,\mathrm{d}S = \mathrm{d}U + P\,\mathrm{d}V\,, \tag{3.3}$$

This can be rewritten as

$$\mathrm{d}U(S,V) = T\,\mathrm{d}S - P\,\mathrm{d}V \tag{3.4}$$

where the internal energy $U = U(S,V)$ is clearly written out as a function of its *natural variables* S and V. One can now do a Legendre transformation on internal energy to receive enthalpy[1]:

$$H(S,P) = PV + U(S,V). \tag{3.5}$$

Enthalpy can be thought of as the energy in a system that can be used to produce heat under constant pressure, the reason for this will be evident very soon. Equation (3.5) can be rewritten to a differential form similar to equation (3.4):

$$\mathrm{d}H(S,P) = T\,\mathrm{d}S + V\,\mathrm{d}P\,. \tag{3.6}$$

Stepping away from various forms of energy, and over to another fundamental concept in thermodynamics: Temperature (which is also, in some sense, a form of energy). The relationship between heat and temperature is defined by the heat capacity

$$C = \frac{\mathrm{d}Q}{\mathrm{d}T}, \tag{3.7}$$

[1]A Legendre transformation takes a function $f(q)$ and transforms it to a function $g(p)$ by the simple formulae $g(p) = pq - f(q)$. So, technically, the definition for enthalpy is only very *close* to a Legendre transformation, since the sign is opposite.

where $\mathrm{d}Q$ is the heat flow into the system and $\mathrm{d}T$ is the corresponding change in temperature. Very often the more relevant physical quantity is specific heat capacity, meaning (here) heat capacity per mass unit: $c = C/m$.

3.1.2 Heat in flowing fluids

When studying fluids it is often convenient to use physical quantities per mass unit, such as the specific heat capacity c mentioned in the previous section. Another important quantity that will be needed in this form is enthalpy. The specific enthalpy is, similar to specific heat capacity, simply defined as the enthalpy per mass unit: $h = H/m$. One can then find the enthalpy flowing with a fluid:

$$\dot{H} = h\dot{m}. \tag{3.8}$$

An important point of study when discussing fluids is when heat is either put in to or taken out of the flowing fluid. In many cases this happens under constant pressure. If one assumes a constant pressure, then the change in enthalpy of a system is equivalent to the heat flow into it. This can be seen by inserting $\mathrm{d}P = 0$ and equation (3.2) into equation (3.6), :

$$\mathrm{d}H = T\,\mathrm{d}S + V\,\mathrm{d}P = \mathrm{d}Q + 0 = \mathrm{d}Q\,. \tag{3.9}$$

The heat flow out from a flowing fluid can then be found be combining this with equation (3.8):

$$\dot{Q} = \Delta\dot{H} = \Delta h \cdot \dot{m}. \tag{3.10}$$

In this equation Δh is the change in specific enthalpy of the fluid, and $\dot{m}$ is the mass flow of the fluid. An example of a system where this is relevant is one that will show up many times in this report: A fluid is flowing through a pipe with mass flow $\dot{m}$. The heat flowing into the pipe is $\dot{Q}$. The change in specific enthalpy of the water, from entering the pipe to exiting it, is then implicitly given by equation (3.10).

If the fluid is purely in one phase, either liquid or gas, it is often more interesting to study the change in temperature than the change in enthalpy. If one also here assumes constant pressure then equation (3.9) can be used to find the heat capacity under constant pressure (C_p) via the enthalpy:

$$C_P = \left(\frac{\partial Q}{\partial T}\right)_P = \frac{\mathrm{d}H}{\mathrm{d}T}, \tag{3.11}$$

which further leads to the specific heat capacity under constant pressure:

$$c_P = \frac{C_P}{m} = \frac{\mathrm{d}(H/m)}{\mathrm{d}T} = \frac{\mathrm{d}h}{\mathrm{d}T}. \tag{3.12}$$

One can then rewrite eq. (3.10) into

$$\dot{Q} = c_p \cdot \Delta T \cdot \dot{m}, \tag{3.13}$$

Here we have assumed that c_p is a constant independent of the temperature, such that the differentials can be replaced by finite differences ($c_P = \Delta h/\Delta T$). Similarly one can find that

$$\dot{H} = c_p \cdot T \cdot \dot{m}. \tag{3.14}$$

3.1.3 Thermal resistance

The heat transfer of complex situations can more easily be analysed using the concept of *thermal resistance* R, an analogous formalism to that of electric circuits. The heat transfer rate $\dot{Q}$ is then analogous to electric current I, whilst ΔT, the difference in temperature, takes on the role of voltage V. This gives us an equivalent formulae to Ohm's law for circuits:

$$\dot{Q} = \frac{\Delta T}{R}. \tag{3.15}$$

The thermal resistance R follows the inverse law for addition when multiple heat pools are connected in serial, meaning

$$\frac{1}{R} = \sum_i \frac{1}{R_i}. \tag{3.16}$$

An important parameter for many heat exchangers is the conductive heat transfer coefficient α, which is defined as

$$\alpha = \frac{q}{\Delta T}. \tag{3.17}$$

where q is the heat flow per area A, meaning $q = \dot{Q}/A$. This gives a thermal resistance

$$R = \frac{1}{\alpha A}. \tag{3.18}$$

In much a similar fashion one finds that the thermal conductivity of a material, λ, gives a thermal resistance

$$R = \frac{L}{\lambda A}, \tag{3.19}$$

where L is the length of the component. This section was based on pages 128 & 129 in Mechanical Engineers Data Handbook[9].

3.1.4 Heat pumps

An ideal heat pump can be modelled using a single parameter: Its Coefficient of Performance (CoP), which for a general engine is known as the energy conversion efficiency η. These two terms will be used interchangeably. The CoP of any heat pump is given by

$$\eta = \frac{|Q_H|}{|W|} \tag{3.20}$$

where $|Q_H|$ is the heat energy on the hot side, defined s.t. $Q_H < 0$. $|W|$ is the power applied to the heat pump, which is defined s.t. $W > 0$ positive for. In addition to these there is also the heat extracted from the cold side Q_C, which is defined s.t. $Q_C > 0$. A complete cycle for the heat pump will have a heat change $dQ \rightarrow Q_C + Q_H$, no change in internal energy ($dU = 0$) and negative work done by the system equal to the power applied, meaning $dW \rightarrow -W$. Inserting this into (3.1) one will find that

$$Q_C + Q_H = -W. \tag{3.21}$$

Combining this with (3.20) one can find that:

$$|Q_C| = (\eta - 1)W \tag{3.22a}$$
$$|Q_H| = \eta W \tag{3.22b}$$

3.1.5 Phase diagrams

A very important tool for studying thermodynamic systems is phase diagrams: Diagrams showing the various phases of a certain material. In our case they will be used only when studying steam. In figure 3.1 a steam cycle is plotted in a phase diagram. The diagram is of the type log(p)-h, plotting logarithmic pressure against specific enthalpy for water (R718).

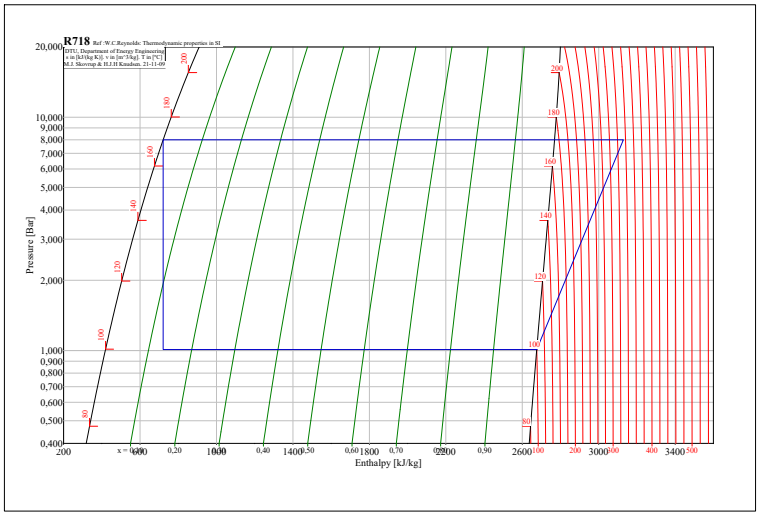


Figure 3.1: This figure shows a log(p)-h diagram of a steam cycle using Coolpack [10]. The blue lines show the steam cycle, which is counter-clockwise in this diagram. Hard black lines show phase transitions, green lines show two-phase quality and red lines show isotherms.

3.2 Numerical tools

This project has revolved around building simulations in the Dymola environment, using the Modelica language. In order to understand the results and limitations of the simulations it is vital to first understand these tools.

3.2.1 Modelica

The Modelica language is a free, object-oriented simulation language developed by the Modelica Association [11]. In this section the basics of the programming language will be laid out, and connected to a real example: A damped pendulum.

Modelica is based on creating models defined by a set of equations rather than the classical assignments which define the programming in most languages. These equations are Differential Algebraic Equations (DAEs), and the solver, which will be discussed in more detail later, rearranges these equations algebraically in order to solve the system. Boundary conditions are then applied, and the simulation is run. To better understand the somewhat weird nature of Modelica code, let's look at an example. Listing 3.1 shows an example of a Modelica model: A damped pendulum.

Code listing 3.1: A model describing a damped pendulum in Modelica, created in the Dymola environment. The first section of the code, everything up to ”equation”, defines the parameters and variables of the system. These may be parameters for the user of the model to tweak, such as the mass m, variables defining the dynamics of the system, such as the angle θ, or variables introduced for convenience, such as the gravitational constant g. The variables are combined into Differential Algebraic Equations in the section labelled ”equation”. These follow mathematical standards of equality rather than the standard assignments of conventional programming languages.

```
model DampedPendulum
  import SI = Modelica.SIunits;

  parameter SI.Mass m = 1 "Mass";
  parameter SI.Length L = 1 "Length";
  parameter SI.TranslationalDampingConstant b = 0.1 "
     Damping constant";
  parameter SI.Angle theta_0 = 0.78539816339745 "
     Initial angle";
  parameter SI.AngularVelocity omega_0 = 0 "Initial
     angular velocity";

  SI.Angle theta(start=theta_0, fixed=true) "Angle";
  SI.AngularVelocity omega(start=omega_0, fixed=true)
     "Angular velocity";
protected
  constant SI.Acceleration g = 9.81 "Gravitational
     acceleration";
  SI.Velocity v "Velocity";
equation
  omega = der(theta);
  v = omega*L;
  m*L*der(omega) + b*v + m*g*sin(theta) = 0;
end DampedPendulum;
```

Ignoring for now the first two sections, one can see at the bottom under the section ”equation” the three DAEs defining this system:

$$\omega = \dot{\theta} \tag{3.23a}$$

$$v = \omega L \tag{3.23b}$$

$$mL\dot{\omega} + bv + mg\sin(\theta) = 0 \tag{3.23c}$$

Modelica’s use of equations instead of assignments makes the language *non-causal*. In programming it is usually important which variable is on the left-hand side and

which is on the right. The program is read by the computer causally, from start to finish, one statement at a time. So to calculate $I = V/R$ both V and R must already be stored in memory. Modelica, on the other hand, does not need this. This means that $V = RI$ is equivalent to $I = V/R$, similar to what is normal in mathematics. Continuing on with the example in listing 3.1, Modelica rearranges equations 3.23 to the following:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\theta\\ \omega\end{bmatrix} = \begin{bmatrix}\omega\\ -\frac{b}{m}\omega - \frac{g}{L}\sin(\theta)\end{bmatrix} \tag{3.24}$$

These equations can now be solved numerically by the solver.

It is important to note that the explanation given here is an abstraction of how Modelica handles equations, not an accurate portrayal of the inner workings of Modelica-solvers. These solvers also solve DAE systems that cannot be expressed as a system of ODEs, like the one above[2].

All Modelica variables, unless otherwise specified, are assumed to be continuous differentiable variables; they can change values throughout the simulation. There are, however, a couple of ”qualifiers” that govern their behaviour towards the user of the model:

- The qualifier ”parameter” denotes variables that the user can change after compilation, so you can test the simulation for various values. An example is to run the simulation for multiple values of the damping constant b.
- The qualifier ”constant” denotes variables that can't be changed after compilation.
- Variables not denoted with either ”parameter” or ”constant” are regular variables, fully defined by the dynamics of the model.

Variables under the ”protected” section are local variables hidden to the user. They can be used in equations to simplify coding or increase readability. Another important aspect of Modelica is it's use of units. All variables can be assigned a unit, in this example they are assigned units through the use of the Modelica standard library SIunits. If all variables in an equation has a specified unit the solver can check that the units of the equation match. If they don't match, a warning is issued. There are, of course, many other aspects of Modelica that are important to know, but there is not enough space in even the margins for that. For more information, see [11].

[2]To solve these the solver needs to flatten the DAE and manipulate it to resemble the ODE form ([11], Appendix B). This is well beyond the scope of what will be discussed here.

3.2.2 Dymola

Dymola, as mentioned earlier, is a simulation environment. This includes much more of a complete environment than traditional IDEs (Integrated Development Environments). Dymola includes, most notably, a graphical interface for model building, a text-editor for classical programming, a plethora of numerical solvers, and a plotting environment.

There will be many examples of the graphical interface throughout this report, so it will not be discussed here. The text editor is not very exciting: It's a box you can write code in, similar to what you expect to find in most IDEs. The plotting environment, although practical when testing the models, has limited use; it will not be discussed further. However, the numerical solvers are of great interest. They will be discussed in some detail now.

The numerical solvers that come as part of Dymola offer a variety of options for achieving the wanted accuracy with a satisfactory runtime. A couple of the options include:

- Euler's method
- Runge Kutta 4
- DASSL
- etc.

These methods can be based on one of two principles: Fixed time step or variable time step. For our simulations this is the most important aspect of the solver to consider. As discussed further below, in section 3.3, the simulation will go over a considerable time frame (8 days). In large segments of this simulation there will be minor changes over time. This warrants taking large time steps. On the other hand there will be periods of extreme change, for example when DGs are turned on/off; this warrants a very small time step to achieve accurate solutions. For these reasons it is of the utmost importance to use a numerical solver with a variable time step. The numerical solver that was chosen, for this very reason, was DASSL. DASSL is a multi-step solver for DAEs created by Linda R. Petzold in 1982 [12], the version used in Dymola is a modification of this method by Dassault Systèmes, the company behind Dymola. And as will be seen later, it gives more than sufficient results for our simulations. Simulation time for the chosen precision, a tolerance of 10^{-3}, has also proven sufficient.

3.2.3 TIL

The last point to go over when it comes to numerical tools is TIL, a library that will be *heavily* utilized in the simulations. TIL offers a suite of models covering all aspects of thermal system simulations. It offers, among others, heat exchangers, pumps, compressors, valves and volumes, making it excellent for use in our simulations.

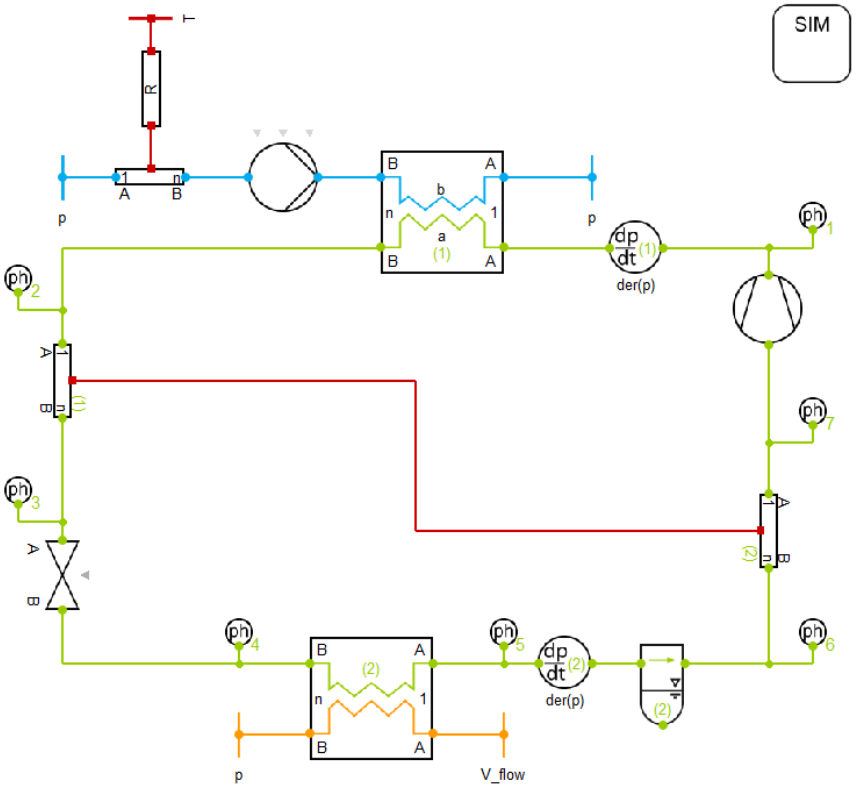


Figure 3.2: Example of a model created using the TIL library; the model shows a simple CO2 heat pump. Blue lines indicate some form of liquid, orange lines indicate a type of gas. The green lines indicate a Vapour Liquid Equilibrium (VLE) fluid. The red line indicates heat flow.

A notable feature of TIL is that it separates between three different types of fluids: Pure liquid, pure gas, and VLE fluid, where VLE is short for Vapour Liquid Equilibrium. The pure, one-phase fluids are simpler to work with, but the user needs to make sure the model makes sense as the fluid will artificially stay in it's phase even outside of where this makes physical sense. E.g. one can have liquid water at 1 atm heated to 150 °C, which is obviously nonsensical. These fluids are, nevertheless, very advantageous both when it comes to their ease of use and speed in simulations. Figure 3.2 shows the three fluid types all in action in a model for a heat pump. Liquid is blue, gas is orange and VLE fluid is green.

The block labelled "SIM" determines the type of fluids used in the simulation. The block supports up to three fluid-types per fluid category (liquid, gas, VLE fluid), and the correct fluid type is then chosen by the user in the GUI of all relevant blocks.

3.3 Case

The case chosen for testing of the complete model was based on an eight-day cruise from august of 2018, the ship in question had a capacity for approximately 3500 passengers. The cruise includes numerous port stays, of various lengths, and with various lengths of cruising inbetween. Data for engine loads, amongst others, was provided by Carnival corporation & plc for the entire duration of the cruise, with a measurement interval of three minutes. This was combined with operation data for the ship under various conditions, also provided by Carnival, to put together the data necessary for the tests.

Figure 3.3 shows the engine loads for all six DGs for the entirety of the cruise. The graph is split into DGs 1,2,3 and DGs 4,5,6. This is, here, mostly done for readability, but this specific split is important for the internal functionality of the heat recovery system for HT water.

Figure 3.4 shows the state of the ship for the full duration of the eight-day cruise. The ship-state is represented by an enumerator, which will all be explained shortly: 0 for 'Dock', 1 for 'Ready', 2 for 'Maneuvering' and 3 for 'Cruising'. 'Cruising' and 'Dock' are mostly self-explanatory; 'Cruising' is when the ship is travelling out at sea whilst 'Dock' is when the ship is properly docked in a port. 'Maneuvering' is when the ship is going to the port, but is not yet docked yet. Usually the speed is much slower, which also means lower engine loads. 'Ready' is the state the ship is in when the ship is still docked, but the engines are either being turned off after maneuvering into docking or being turned on before maneuvering out of docking.

The goal of this thesis is to study possible solutions for not running the DGs whilst in port. The choice was made to allow the ship to run the DGs during a short time interval both when docking and when leaving. On a technical level this was implemented by allowing diesel generators to run when the ship would normally be in the 'Ready' state, but turning them all off when in the 'Dock' state. The results of this on the

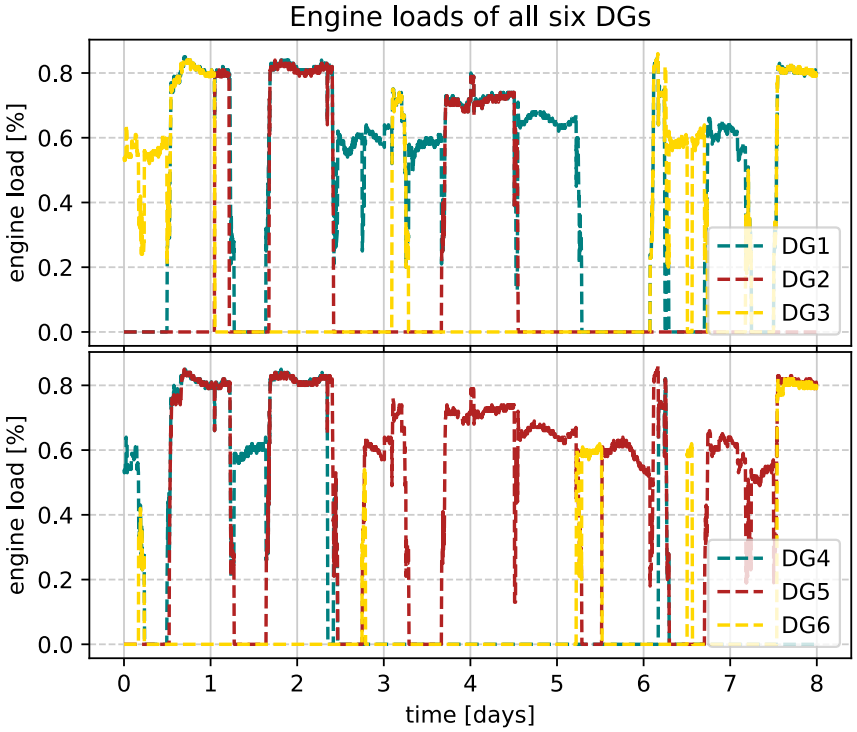


Figure 3.3: Engine loads from case data, for all six DGs. DGs 1,2,3 can be seen in the upper plot, whilst DGs 4,5,6 can be seen in the lower. The split of the DGs is done here for readability, but this particular split is closely connected to the functionality of the HT recovery system, and will therefore be of importance in other figures.

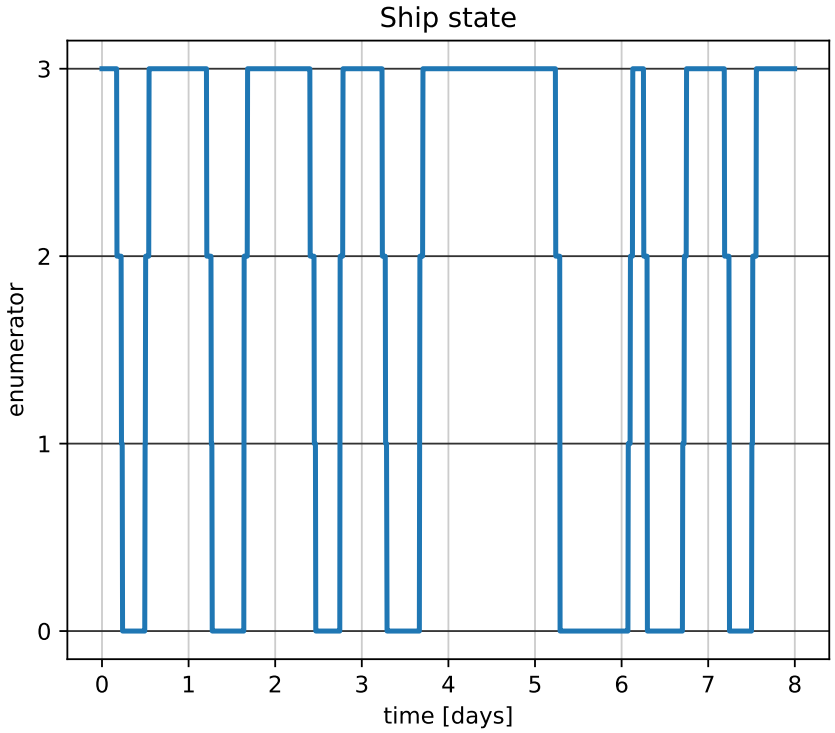


Figure 3.4: Figure showing the ship state over time for the full duration of the eight-day cruise. The ship state is enumerated: 0 for 'Dock', 1 for 'Ready', 2 for 'Maneuvering' and 3 for 'Cruising'.

engine loads can be seen in figure 3.5.

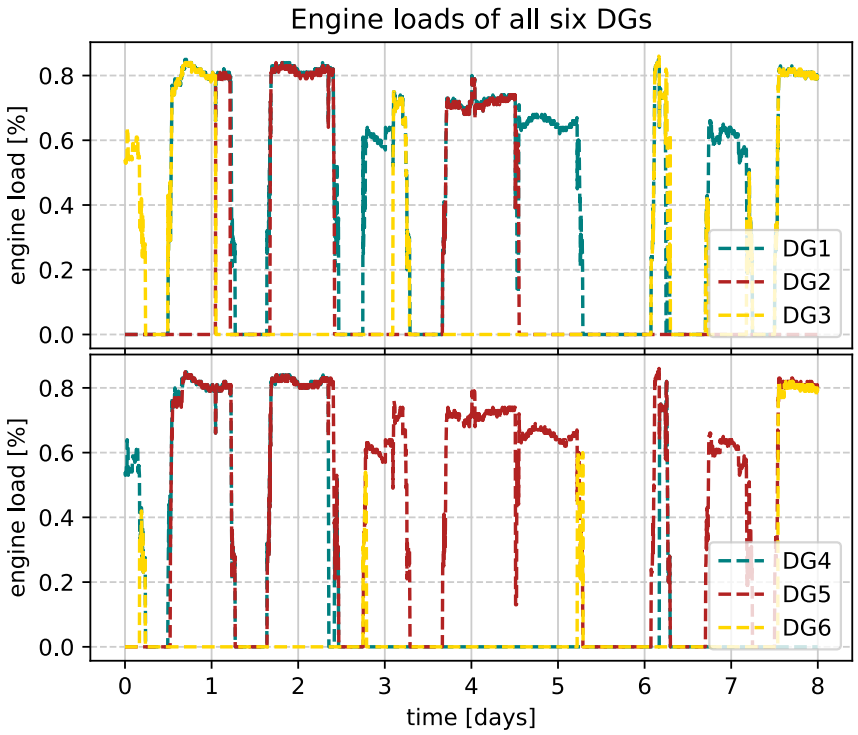


Figure 3.5: Engine loads used in tests; all engine loads were set to zero when the ship state is 'Dock'. See figure 3.3 for more information on the plot.

Chapter 4

Analysis and results

This chapter will go through the results produced and the analyses that led to them. The first section will cover some basic groundwork done, and explain the various solutions proposed to solve the need for HT water in port with zero emissions. Following this will be each proposed solution, the analysis used to find the correct parameters for it as well as the results from simulations. In the final section of the chapter various steam solutions will be proposed, and results from these will be shown.

4.1 Base analysis HT water

4.1.1 Solutions - overview

The type of solutions that have been studied to solve the need for HT water can be divided in the following way:

- Solution 1: A water tank connected to the HT recovery system.
- Solution 2: A PCM connected to the HT recovery system.
- Solution 3: A Heat Pump connected to the HT recovery system.

The requirements for the performance of the solutions are the following:

- The solution must deliver HT water to the engine system at a temperature equal to or above 70.5 °C, whenever the engines are running. This can be broken for very short time intervals.
- The solution must meet the heat demands of the HT users at all times, delivering HT water with a temperature of atleast 50 °C.
- The initial temperature for PCMs will be set to their melting temperature. For the water tank the initial temperature is equal to the lower threshold value: 70.5 °C.

4.1.2 Analysis

In order to make things easier when multiple types of TES are to be implemented a base analysis on the requirements for energy stored in the TES, as well as necessary heat exchange rate, will be done. It is not possible to use the case data for waste heat from the engine system for HT water directly, as the HT recovery system necessarily carries with it a significant loss. Instead, a simulation was run where all waste heat was extracted from the water, given the HT recovery system solution on the ship. Figure 4.1 shows the results from this simulation for accumulated energy and heat flow.

Studying figure 4.1 closely, one will quickly find that the most challenging period happens between day 5 and day 7, where two back-to-back port stays occur with only minimal time in between. Although this is the time period which requires the largest energy storage throughout the cruise, it is the initial 12 hours which impose the largest constraints on heat exchange rate. Whilst day 5 to 7 have approximately 36 hours of charging time ahead of the back-to-back port stays, the first port stay comes after only a few hours. The challenges both of these impose on the various solutions will be returned to.

One can use these results to estimate the energy capacity needed by the TES by looking at the period from day 5 to day 7, as this is the most challenging period capacity-wise. Looking at figure 4.2, a zoomed-in version of net, accumulated energy in figure 4.1, one can see that the difference between the top point, at around 5.25 days, and the bottom point, at a little past 6 days, is about 7 MWh. This gives us a rough estimate of the capacity needed, given that the heat exchange rate is sufficient so that enough energy is stored in the short cruise during day 6 to hold through the second of these port stays.

One additional point to analyse is the required heat exchange rate. Looking first at figure 4.2 the period cruising between the two port stays need to meet the demand in the second port stay, which seems to be a around 3 MWh. This energy must be charged to the TES in the span of about 6 hours, which gives a required heat exchange rate of approximately 0.5 MW. However, this heat exchange rate can be neglected if the storage capacity of the TES is sufficiently large, so that enough energy has been stored in the previous cruise to also last through this port stay.

Turning to the first 12 hours of the cruise: The net energy accumulated in this period can be seen in figure 4.3. As can be seen in the figure the requirements for the port stay, which is the difference between the peaks at about 5 hours and 11 hours, is around 2 MWh. This must be charged to the TES beforehand, meaning in the first 5 hours of the cruise. This gives an average charging heat flow requirement of about 0.4 MW. This is lower than what was needed in the section study further up, but as discussed above that number could be compensated for with a larger energy capacity on the TES.

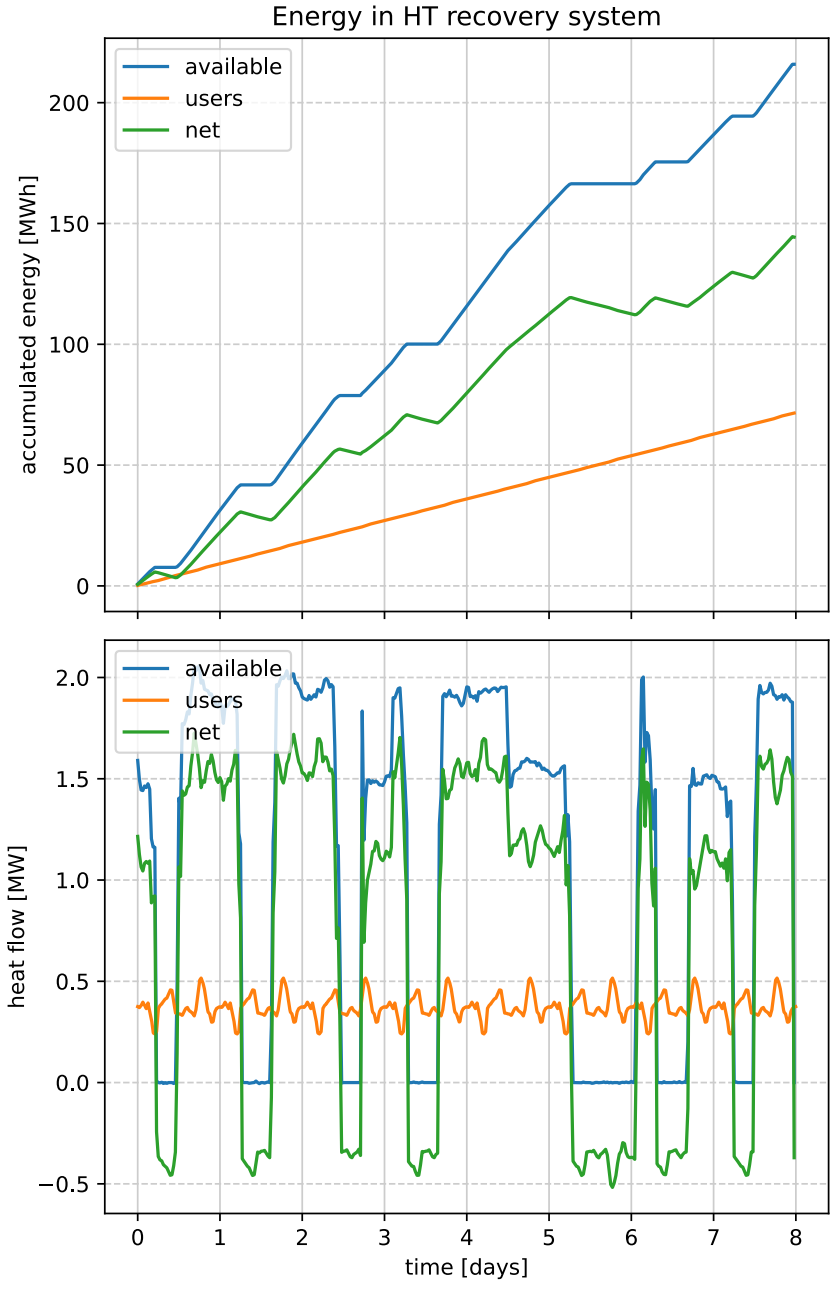


Figure 4.1: Figure showing accumulated energy and heat flow rate in the HT recovery system in the model. Values for the users is given by the case data, whilst available energy/heat flow was calculated by the enthalpy difference between the water after users and after cooling it to the lower threshold value (70.5 °C). Net energy/heat flow was calculated by subtracting the energy/heat flow to users from the available energy/heat flow.

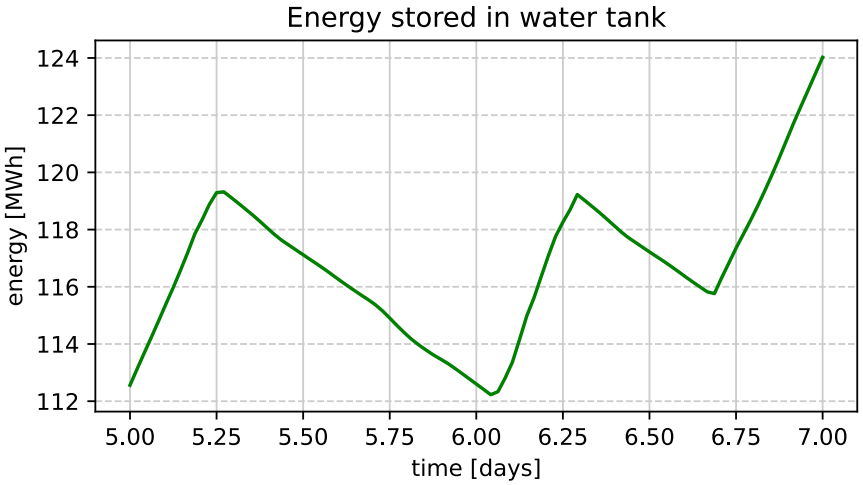


Figure 4.2: Figure showing net accumulated energy in a critical period during the cruise: Day 5 and day 6. See figure 4.1 for the full duration of the cruise. The net accumulated energy rises when the ship is cruising (showing how much energy can be stored), and falls when the ship is in port and the engines are turned off (showing how much energy that must be used). Prior to the period shown here the cruise ship has spent approximately 36 hours cruising, meaning there is good opportunity to store large amounts of heat in a TES beforehand. Nevertheless, meeting the heat demands in this period will be challenging, as will be seen later.

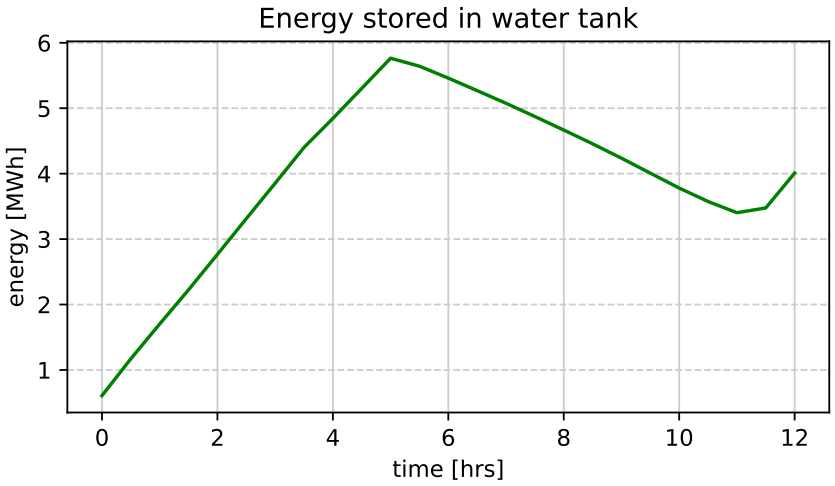


Figure 4.3: Figure showing net accumulated energy in a critical period at the start of the cruise: The first 12 hours. See figure 4.1 for the full duration of the cruise. The net accumulated energy rises when the ship is cruising (showing how much energy can be stored), and falls when the ship is in port and the engines are turned off (showing how much energy that must be used). The period shown here is extremely challenging when it comes to heat transfer rate for any solution using TES, as enough energy must be stored in the first 5 hours to last about 6 hours in port.

4.2 Solution 1

In the previous section, section 4.1.2, it was concluded that a thermal energy storage with sufficient heat transfer rate will need about 7 MWh of heat capacity. Assuming a maximum temperature of 77 °C, and a minimum temperature of 70.5 °C, one can find the energy effectively stored in the water tank by finding the difference between the energy stored at these temperatures. That energy, per volume, will be given by the formula $u(T) = \rho(T) \cdot h(T)$ where $u(T)$ is the energy density at a temperature T, whilst $\rho(T)$ and $h(T)$ are the mass density and specific enthalpy of water, respectively, at that same temperature. The total energy density stored is then

$$u_{tot} = \rho(77.0\,°\text{C}) \cdot h(77.0\,°\text{C}) - \rho(70.5\,°\text{C}) \cdot h(70.5\,°\text{C}) \tag{4.1}$$

Using CoolProp ([13]) one can find the values for the mass density and specific enthalpy of water at any given temperature. The results from this be seen in table 4.1.

Table 4.1: Table showing some physical quantities for water needed in the analysis. Values were found using CoolProp[13].

T[°C]	ρ[kg/m^3]	h[kJ/kg]
77.0	973.0	322.5
70.5	977.5	295.2

Inserting this into equation (4.1) yields $u = 25.40\,\text{MJ/m}^3 = 7.055\,\text{kWh/m}^3$. It is then easy to calculate the necessary volume needed by the water tank:

$$V = \frac{E}{u} = \frac{7\,\text{MWh}}{7.055\,\text{kWh/m}^3} = 992\,\text{m}^3. \tag{4.2}$$

An initial testing volume of 1000 m^3 was chosen based on this result. The results from a simulation using this volume for the water tank can be seen in figure 4.4. The model used for testing can be seen in figure 4.5 As one can see the temperature of the output to the engine system T_{out} remains equal to or higher than the lower threshold T_0 at almost all times, only falling below for a few very short instances. The temperature of the water tank T_{wt} rises quickly during charging, and one can see that the TES easily meets the heat demands at all times. Notice in particular the two most problematic periods (discussed in section 4.1.2): The first port stay, which happens in the first 12 hours of the cruise, and the back-to-back port stays during day 5 and 6. The tank meets the demands in this periods, but as one can see this is the two periods where the water tank is close to being fully discharged. The clearing is, however, quite significant, which indicates that the TES is oversized for the problem.

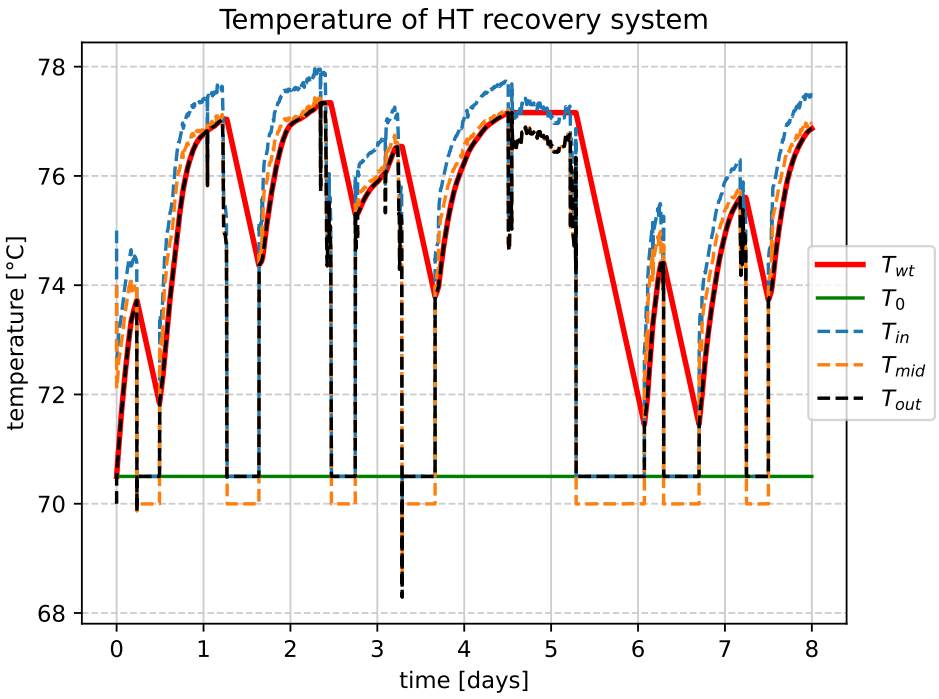


Figure 4.4: Temperature at various points in the HT recovery system. T_{in} is the temperature coming from the heat exchange with the engine system, T_{mid} is the temperature after heat exchange with the HT users. T_0 is the lower threshold for the output temperature for the HT recovery system, which is shown in the figure as T_{out}. Finally, the temperature of the water tank is T_{wt}. This result is from a simulation using a water tank with volume 1000 m^3 as the Thermal Energy Storage (TES).

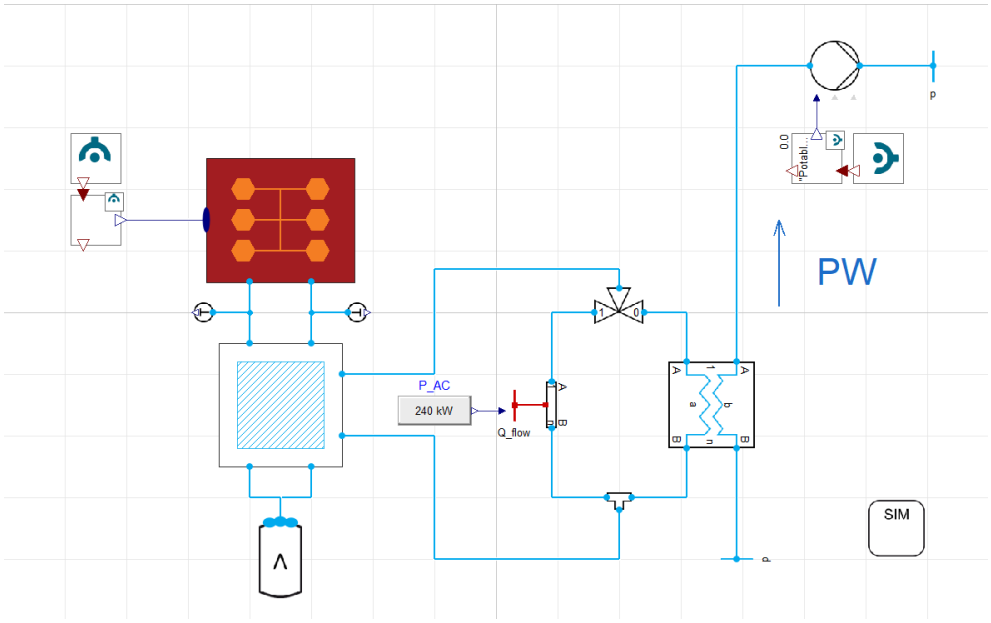


Figure 4.5: This figure shows the model used for simulations of solution 1. The red box is the power supply system, all info regarding this can be found in the section B of the appendix. Below the power supply system is a helper model for the HT recovery system, including controller, see B.1.5. To the right of the figure one can find the users: The air conditioning is a constant heat flow of 240 kW, which is to the left in the parallel flow. The potable water preheating is modelled using a heat exchanger model connected to a water flow where the mass flow is given by the case data. In the bottom of the figure, the box with the big ”V”, one can find the water tank. This will in solution 2 be swapped for a PCM.

Running tests it was found that a water tank with size 850 m^3 is right on the edge of being enough, falling less than 0.02K below the lower threshold value of 70.5 °C. The results from a simulation using this can be seen in figure 4.6.

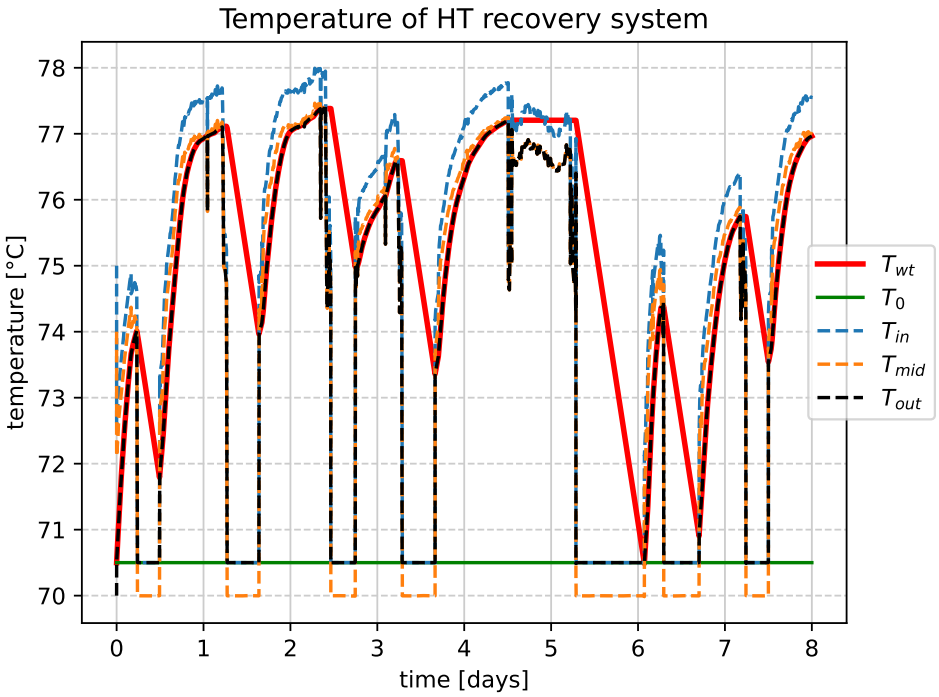


Figure 4.6: Temperature at various points in the HT recovery system, see figure text of figure 4.4 for more details. This is from a simulation using a water tank with volume 850 m^3 as a TES.

The energy capacity needed by this water tank is about 6.35 MWh, which is about 0.65 MWh less than first estimated. Exactly why the initial estimate is off is unclear, but not really important, it was always meant as a rough estimate. Another point to mention, as it will be discussed more on a later point, is the weight of the water tank. With a volume of 850 m^3 the average weight of the water in the tank is about 829 tonnes, deviating from this average by less than 2 tonnes at the most.

4.3 Solution 2

The first approach that was done to implementing a PCM as TES instead of a water tank was trying to simply replace the water tank with some form of PCM and tune parameters accordingly. Trying to implement a PCM around the working temperature of the HT recovery system is challenging as it carries with it one fundamental problem: ΔT, the difference between the temperature of the fluid and the melting temperature of the PCM, becomes very small. This happens both when charging and discharging the PCM. This makes it challenging to achieve sufficient rates of heat transfer, as will be seen. But, as will also be seen, not impossible. First of all: A discussion on the choice of material.

4.3.1 Choice of material

The temperature available to the heat exchange with the PCM is typically between 72 °C and 78 °C, but this is highly dependent on the output temperature of the HT recovery system, it will typically be 1-2 Kelvin above that temperature. It also, very much, depends on the engine loads and the number of engines which are running. The choice for material in this solution therefore heavily depends on the temperature, as the PCM must get sufficient heat exchange for both charging and discharging, all in a temperature interval of less than 10 K. The PCM must in charging mode heat exchange with water at about 76 °C – 78 °C, and in discharging it must heat the water to 70.5 °C. The chosen temperature range for the material was then chosen to be 72 °C – 75 °C, which led to the material CrodaTherm™ 74 (important physical quantities for this material can be seen in table 2.1). The material was chosen solely on its melting temperature of $T_m = 74.8\,°\text{C}$, as it was the only material with known (or actually semi-known) parameters in the wanted temperature range.

4.3.2 Initial analysis and testing

With the choice of material out of the way it is possible to estimate the size necessary for the PCM. In section 4.1.2 it was found that the Thermal Energy Storage needed to store about 7 MWh, but section 4.2 showed that a water tank only needed to store a little over 6.35 MWh in order to meet the demands. The energy capacity used in the initial analysis for the PCM will be set to 6.5 MWh, adding a little buffer to the water tank results, but not going quite as high as the initial analysis. The heat stored in the phase transition of the material is the latent heat, see section 2.4. For CrodaTherm™ 74 this is given as $L_{pcm} = 224\,\text{kJ/kg}$. An estimate of the necessary volume is then given by

$$V_{pcm} = \frac{m}{\rho} = \frac{Q_{tot}}{\rho L_{pcm}}. \tag{4.3}$$

This will only be an approximation as the density ρ changes during phase transition, but as this is only an estimate this is not important. The lowest of the two densities was chosen for this estimate: $\rho_l = 858\,\mathrm{kg/m^3}$ (see table 2.1). Inserting this, as well as $L_{pcm} = 224\,\mathrm{kJ/kg}$ & $Q_{tot} = 6.5\,\mathrm{MWh}$, gives a volume of $V_{pcm} = 122\,\mathrm{m^3}$. This is very low, especially considering that this only accounts for the latent heat of the PCM, there will be a little additional capacity coming from the sensible heat. However, this volume does not contain the volume needed by tubes/plates for the heat exchange, only the volume of the material. To find this one must study the heat flow rate.

Assume, as was found in section 4.1.2, that in order to meet the heat requirements in the crucial day 5 to day 7 period one needs a heat transfer rate of 0.5 MW (it is for now not assumed that there is excess energy capacity to cover the second port stay in this period, again, see section 4.1.2 for more in depth explanation). Using equation (3.15), assuming a bad case of temperature difference $\Delta T \sim 1\,\mathrm{K}$, one gets a required thermal resistance:

$$R = \frac{\Delta T}{\dot{Q}} \sim \frac{1\,\mathrm{K}}{0.5\,\mathrm{MW}} = 2 \times 10^{-6}\,\mathrm{K/W}. \tag{4.4}$$

Initial testing will be done with a plate in block PCM. One can imagine the heat exchange will look something like figure 4.7. The water is shown in blue, the wall of the plate is shown in grey and the PCM material is shown in orange. The water will be assumed to be a source/sink such that the convective heat transfer coefficient α can be used to determine the thermal resistance related to convection to the wall. The resistance in the wall and the PCM material will be determined via their thermal conductance parameters λ_w & λ_{pcm}. The heat exchange area A will be the same for all three resistances. By equation (3.16), combined with (3.18) and (3.19), the total resistance of the system is given by

$$R = \frac{1}{A}\left(\frac{1}{\alpha} + \frac{w}{\lambda_w} + \frac{L_{pcm}}{\lambda_{pcm}}\right), \tag{4.5}$$

where w is the thickness of the wall separating the fluid from the material and L_{pcm} is half the width of the PCM material section.

It has been found that $\alpha \approx 12\,\mathrm{kW/(m^2K)}$ for a plate heat exchanger with water at a temperature of about 70 °C[14]. This will be used in the following calculations. Assuming the material in the wall is steel, which it will be in the models run, then λ_w is of order $10\,\mathrm{W/(mK)}$([9], p. 131). The wall thickness w will typically be of order 1 mm. From table 2.1 one can see that $\lambda_{pcm} = 0.20\,\mathrm{W/(mK)}$ for the material CrodaTherm™ 74, but really, no matter the material $\lambda_{pcm} \sim 10^{-1}\mathrm{W/(mK)}$. This means that if L_{pcm} is of the order 0.1 mm or higher, it will be the dominating term. This can be assumed to hold. So:

$$A \approx \frac{1}{R} \cdot \frac{L_{pcm}}{\lambda_{pcm}} \tag{4.6}$$

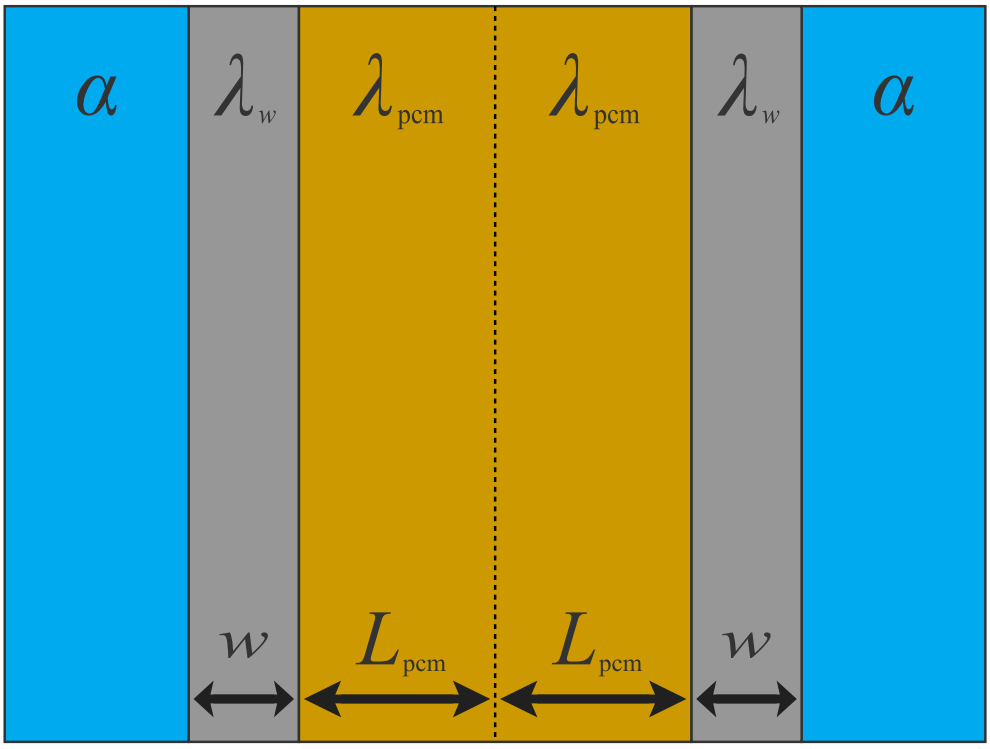


Figure 4.7: This figure shows the cross section of a plate-in-block PCM, along with important parameters for calculations on heat transfer: α is the conductive heat transfer coefficient from the water to the wall, w is the thickness of the wall separating the fluid from the material, L_{pcm} is half the width of the PCM material section and finally λ_w & λ_{pcm} is the thermal conductivity of the wall and PCM material, respectively.

It is now assumed that the total heat exchange area A is split into n plates with side lengths l, meaning $A = 2nl^2$ (since each plate has a heat exchange area on each side). These plates have a duct width d and, as mentioned previously, a wall thickness w. Each plate also have PCM material on each side with length L_{pcm}. This means the total volume for the material is given by

$$V_{pcm} = n \cdot (l^2 \cdot 2L_{pcm}) = 2nl^2 \cdot L_{pcm} = AL_{pcm} \approx \frac{L_{pcm}^2}{R\lambda_{pcm}} \tag{4.7}$$

which gives

$$L_{pcm} \approx \sqrt{V_{pcm} R\lambda_{pcm}}. \tag{4.8}$$

This gives, with the values found earlier, a length $L_{pcm} = 0.7\,\text{cm}$.

The total volume of the PCM, which is to be minimized, is a box with length l, height $l + \Delta$, and width $n(d + 2w + 2L_{pcm})$. Δ is the clearance in the PCM, meaning the area between the plates/material of the PCM and the roof of the storage. This gives us a volume

$$V(n, l) = n(d + 2w + 2L_{pcm})(l + \Delta)l, \tag{4.9}$$

Inserting $n = A/2l^2$ into equation (4.9) gives

$$V(l) = \frac{A}{2}(d + 2w + 2L_{pcm})\left(1 + \frac{\Delta}{l}\right), \tag{4.10}$$

Inserting for all known variables and plotting the volume as a function of the side length l gives the results shown in figure 4.8. As can be seen the volume falls as a function of l, but the difference between the various values is minimal, particularly for the higher values. A value of $l = 5\,\text{m}$ was chosen as appropriate as the volume falls less than $0.1\,\text{m}^3$ for values higher than this. The remaining parameters can now be determined based on our choices. The final parameters of the PCM can be seen in table 4.2 under ”PHE initial test”.

The model used for testing of solution 2 is identical to the one shown in figure 4.5, except the water tank has been swapped for a PCM model. Using the parameters shown in table 4.2 yields the results shown in figure 4.9. As one can see, the PCM isn't quite sufficient to meet the demands. At around 0.5 days and 6.7 days the energy falls below the initial value. In both periods this is caused by too low heat exchange rate in the cruising beforehand. In the first case the PCM has, pretty consistently, a heat flow rate $\dot{Q}_{pcm} \sim 0.2\,\text{MW}$. As discussed earlier the required heat exchange rate in this period was 0.4 MW. In the second ”troublesome” area the heat exchange rises to a maximum of 0.4 MW, but in this period it was required to have an average of 0.5 MW. Exactly why the heat exchange doesn't meet the expectations from the

analysis is unclear, but a good guess is the assumption that the water is a source/sink with temperature difference $\Delta T = 1$ K. Studying the temperature from this simulation it was found that the water has at temperature difference of around 0.5 K at its highest, falling to about 0.2 K at exiting the PCM.

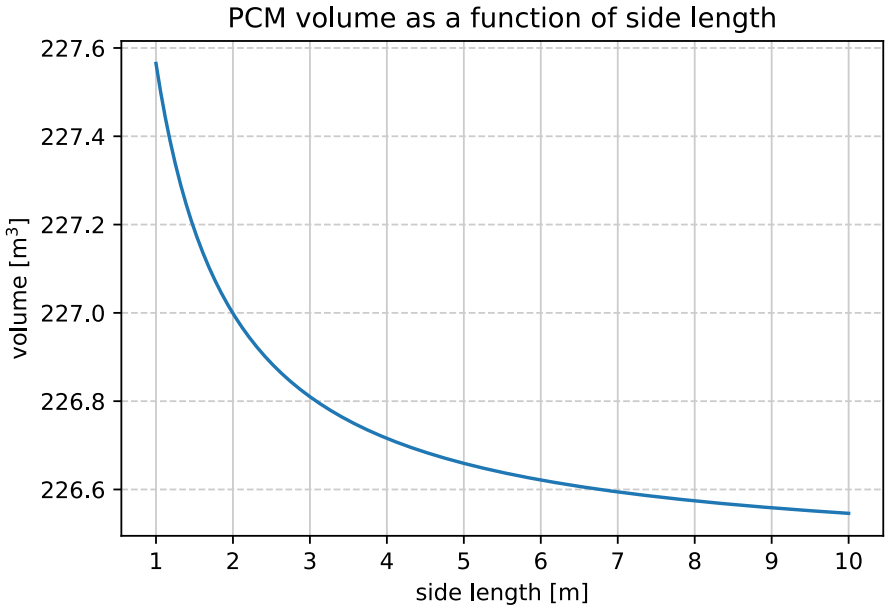


Figure 4.8: Figure showing the volume of the plate in block pcm as a function of its side length l, given by equation (4.9). Notice that the change in volume is relatively small, particularly for values higher than 5 m. The dependence of the volume on the side length is caused by the clearance of the PCM: Δ. For $\Delta \to 0$ the volume becomes fixed.

Table 4.2: Table showing parameters for all PCMs used in testing. ”PHE”is short for ”plate heat exchanger” whilst ”PPHE” is short for ”pillow-plate heat exchanger”. n is the number of plates, d is the duct width, w is the wall thickness and Δ is the clearance (height between the contents of the PCM and the roof of it).

	PHE initial test	PHE final result	PPHE
width $[m]$	6.275	15.04	10.58
length $[m]$	5	5	5
height $[m]$	5.005	5.005	5.005
$n[1]$	349	847	820
d[mm]	10	10	10
w[mm]	1	1	1
Δ[mm]	5	5	5

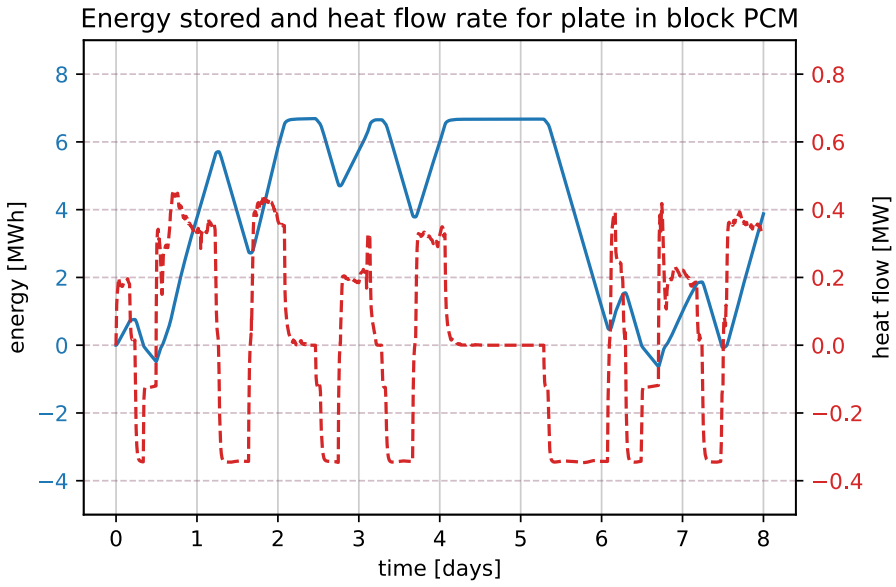


Figure 4.9: This figure shows the energy stored and the heat flow rate of the plate in block Phase Change Material (PCM) used in initial testing. The energy set to stored is set to zero for complete solid phase at the melting temperature. The heat flow rate is set to positive for heat flowing into the PCM.

4.3.3 Improving upon initial results

One way to move forward and improve upon the initial results found, is to study the effect the thermal resistance has on the final results from the dynamic simulation. The same calculations that were done previously was done for each value of R, and simulations were then run with the parameters following from this. The side length l was, again, chosen to be 5 m. In all simulations run the TES had, at some point, an energy deficit. The maximum energy deficit, along with the maximum energy deficit in the first 12 hours, are plotted in figure 4.10 as the hard and dashed blue lines, respectively. The volume of the PCM is also plotted, this can be seen in green. Note that the figure is logarithmic in R.

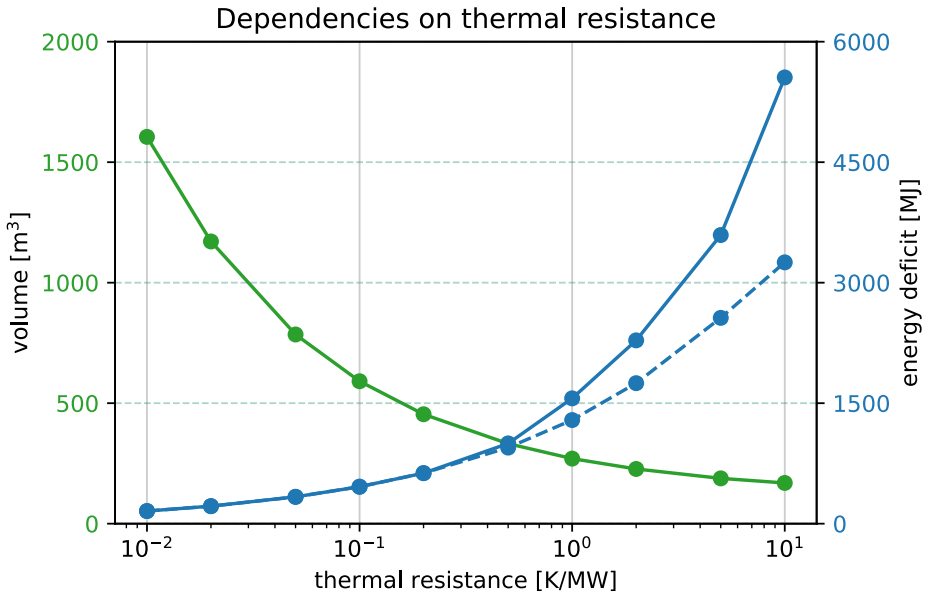


Figure 4.10: The effects of changing thermal resistance R on the dynamic simulation. The maximum energy deficit in the simulation is shown by the hard blue line. The dashed blue line is the maximum energy deficit in the first 12 hours of the cruise, as this period is of special interest this is also plotted. For small values of R these are the same. In addition, the volume of the PCM is shown in green.

As one can easily see the energy deficit(s) decrease for lower values of R, whilst the volume increases. The first thing to note is that it seems like the energy deficit is only lowered by decreasing R, it never disappears. This maximum energy deficit is, for all measured values of $R \leq 0.2$ coming from the first 12 hours of the cruise. For the lowest values of R this was the only moment in the simulation with energy deficit.

The second thing to note is that since the advantage of lowering the thermal resistance falls for low values another strategy should be found. The following solution is proposed: Allow some energy deficit, but keep the temperature of the PCM above the

crystallization temperature 71 °C. This means some of the demand will be met with the sensible heat of the solid-state of the PCM. This heat is given by

$$Q_{SH} = c_{p,s} \cdot \Delta T \cdot m. \tag{4.11}$$

Here m is the mass of the additional PCM material needed, which will be equal to $\rho_s V_{SH}$; V_{SH} is then the additional volume needed to meet this heat. The temperature difference ΔT is here the difference between the melting temperature and the crystallization temperature, meaning $\Delta T = 74.8\,°\text{C} - 71\,°\text{C} = 3.8\,\text{K}$. All in all this gives

$$V_{SH} = \frac{Q_{SH}}{\rho_s \cdot c_{p,s} \cdot \Delta T}. \tag{4.12}$$

The required heat Q_{SH} must the difference between the energy deficit at the crystallization temperature and the maximum energy deficits in the simulations. However, it can be assumed that increasing the volume of the material will lead to the energy deficit at day 6 disappearing before the energy deficit in the first 12 hours. This is because the port stay at day 6 can take advantage of the increase in sensible heat for *both* phases as well as the latent heat, will the port stay at the beginning of the cruise can only take advantage of the increased sensible heat in the solid phase. All of this to say: The energy deficit in the first 12 hours will be the one to adjust for. Plotting this gives the results shown in figure 4.11.

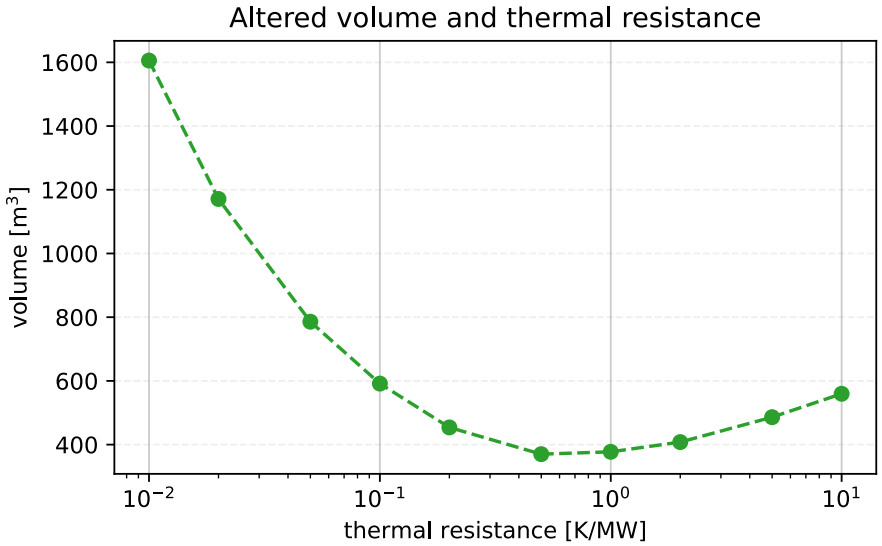


Figure 4.11: Figure showing total estimated volume necessary to keep temperature of PCM higher than its crystallization temperature of 71 °C. Estimates are found be taking the energy deficit found for the given resistance, subtracting the enthalpy at crystallization and using this as Q_{SH} in equation (4.12).

As one can see the minimum here is for $R = 0.5\,\mathrm{K/MW}$, which has an additional volume of $V_{SH} = 21\,\mathrm{m}^3$. This results in a total volume of $370\,\mathrm{m}^3$. Running a simulation with this shows that the additional volume calculated isn't quite enough, the simulation required a volume $V_{SH} = 39\,\mathrm{m}^3$ to meet the demands. This results in a total volume of $402\,\mathrm{m}^3$. Running some manual tests for the resistances close to this, $R = 0.2\,\mathrm{K/MW}$ & $R = 1\,\mathrm{K/MW}$, showed an increase in total volume in both directions. This means the volume found can be assumed to be very close to the ideal volume. Some further testing showed that $R = 0.34\,\mathrm{K/MW}$ with zero additional volume gave the best results: $V = 376\,\mathrm{m}^3$. Adding extra storage capacity was not the solution after all. The parameters for this PCM is shown in table 4.2 under the name "PHE final result".

4.3.4 Pillow-plate in block

Using a pillow-plate in block PCM will give even better results than what was found with the plate in block PCM. The dimensions of the PCM were initially chosen to be identical to the final plate in block PCM found (see table 4.2). After this, two variables were tuned: The number of plates n and the typical length between plates L_{pcm}[1]. One advantage that showed up during this tuning process was the independence of variables for the pillow-plate PCM: The energy deficit in the first 12 hours of the cruise was almost entirely independent of L_{pcm}. The PCM could therefore be tuned by number of plates first, until the temperature at the initial port stay was just on the border to 71 °C. After this, L_{pcm} could be tuned so that the total energy capacity was just enough to meet demands in the day 5 to day 7 period. The parameters from this tuning can be seen in table 4.2 under the name "PPHE", whilst the results from a simulation using this is shown in figure 4.13. The total volume of this PCM is only $265\,\mathrm{m}^3$.

[1]For the plate in block PCM this was a neatly defined length, but due to the geometry of the pillow-plates this is no longer the case. In this case the length L_{pcm} is defined such that the total width of the PCM is still equal to $n \cdot (d + 2w + 2L_{pcm})$.

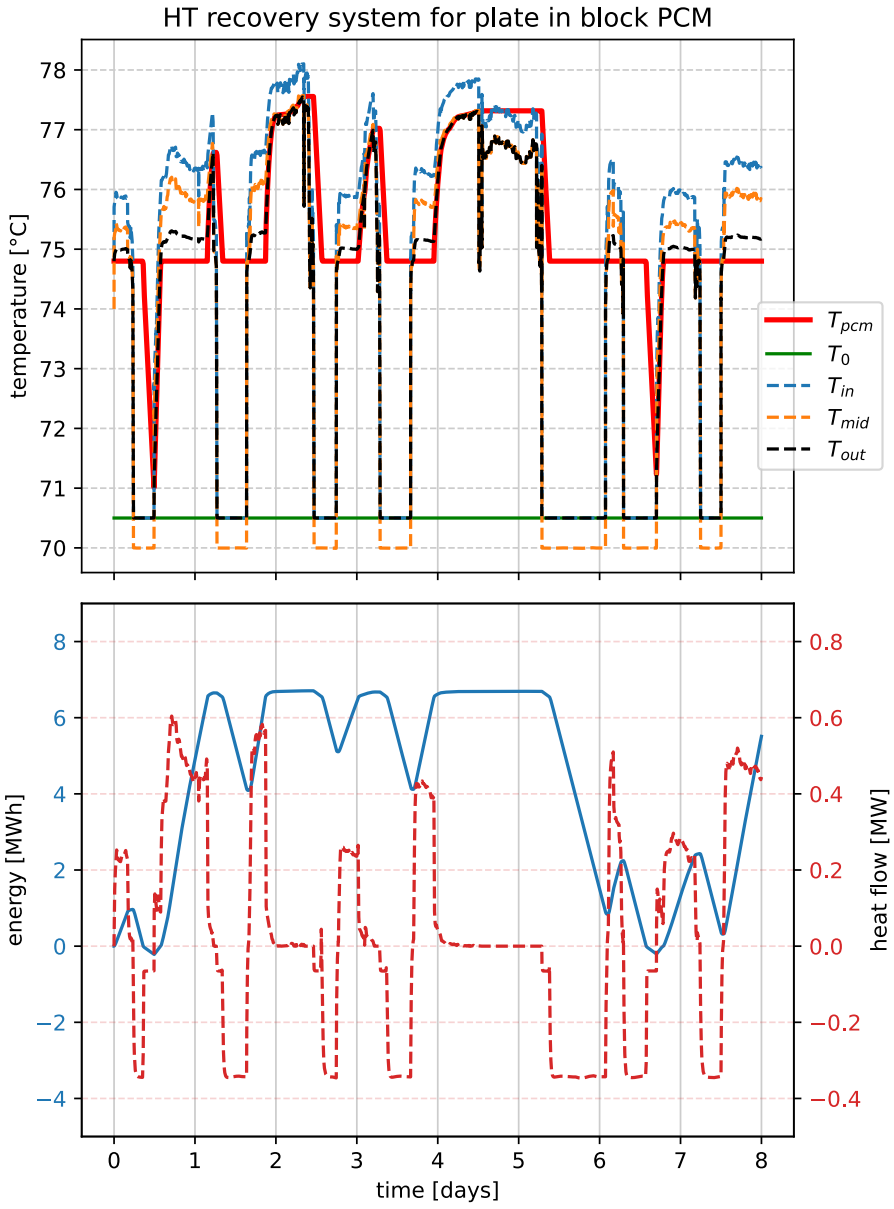


Figure 4.12: This figure shows the temperature and energy stored/heat flow rate of the final plate in block Phase Change Material (PCM), which was found after extensive analysis. T_{in} is the temperature coming from the heat exchange with the engine system, T_{mid} is the temperature after heat exchange with the HT users. T_0 is the lower threshold for the output temperature for the HT recovery system, which is shown in the figure as T_{out}. Finally, the temperature of the water tank is T_{wt}. The energy set to stored is set to zero for complete solid phase at the melting temperature. The heat flow rate is set to positive for heat flowing into the PCM. This PCM has a total volume of 376 m^3.

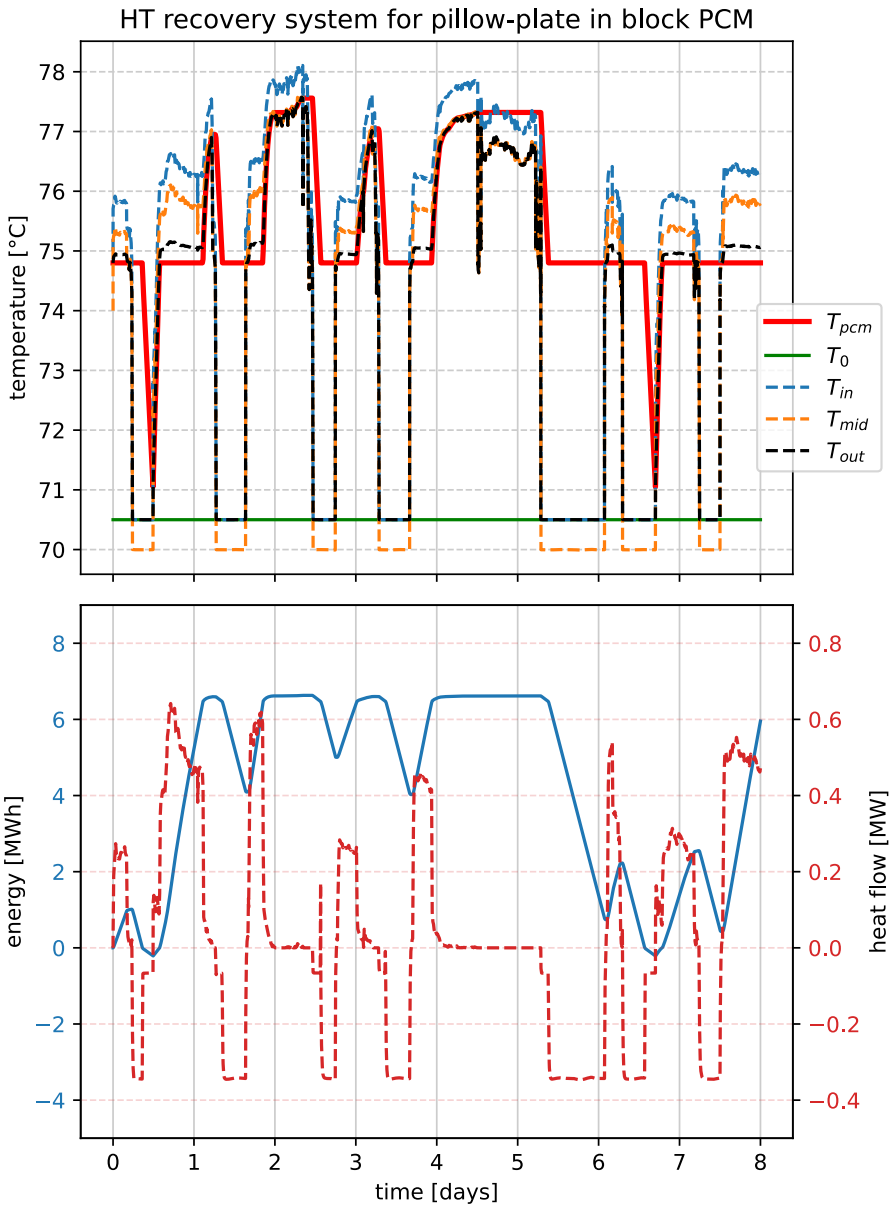


Figure 4.13: This figure shows the temperature and energy/heat flow rate in the HT recovery system from a simulation using a pillow-plate in block PCM. See figure 4.12 for explanation on the graphs. The PCM used had a volume of 265 m^3.

4.4 Solution 3

The last approach to meeting the heat demand for HT water was implementing a simple heat pump to heat up the water between the users and the engine system. The heat pump was modelled as being ideal, and the implementation of this can be found in the appendix (section B.1.4). Details on the controller used for the heat pump can be seen in A.2.4. One main point to note here is that a small water tank, volume of 1 m^3, was used to create a buffer for when the engines are turned off. This gives enough time for the controller to stabilize, again, see A.2.4. The model used for testing this solution can be seen in figure 4.14.

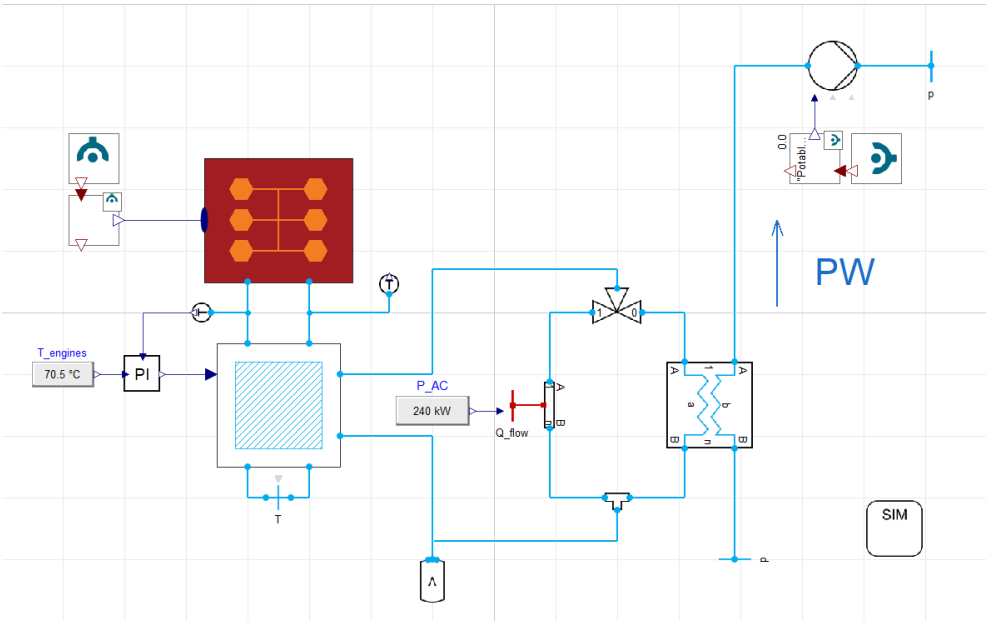


Figure 4.14: This figure shows the model used for simulations of solution 3. An in depth explanation of a very similar model, that of solution 1, can be found in the figure text of figure 4.5. The difference in this model from that model is that the water tank has been removed, and a heat pump inside the HT recovery system helper model, blue box, is activated. The boundary condition, which is located where the water tank was for solution 1, provides water to the cold side of the heat pump. This was set to 25 °C for the simulations run here. A small water tank, with volume 1 m^3, has been added after the users; this acts as a buffer when heat demand or availability changes.

A Coefficient of Performance (CoP) of 3.5 was chosen, quite arbitrarily, as an initial test. The results for the HT recovery system from this test can be seen in figure 4.15. The average power of the heat pump, when it was active, was about 98 kW. The maximum power required was just over 99 kW.

It is also of interest to see how the average power required by the heat pump is affected by the heat pump coefficient, and how close this will lie to the theoretical results. To

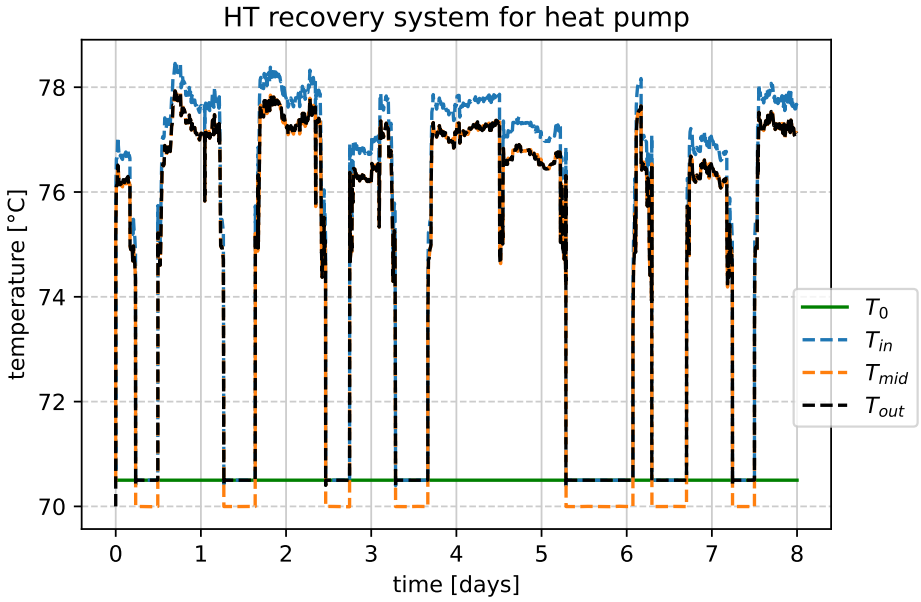


Figure 4.15: This figure shows the temperature in the HT recovery system from a simulation using a heat pump to meet HT water demand during port stays. T_{in} is the temperature coming from the heat exchange with the engine system, T_{mid} is the temperature after heat exchange with the HT users. T_0 is the lower threshold for the output temperature for the HT recovery system, which is shown in the figure as T_{out}. The heat pump had a Coefficient of Performance (CoP) of 3.5.

test this many simulations were run with various CoP, the results from this can be seen in figure 4.16. As one can easily see the simulation results lie very close to the theoretical values, as is expected.

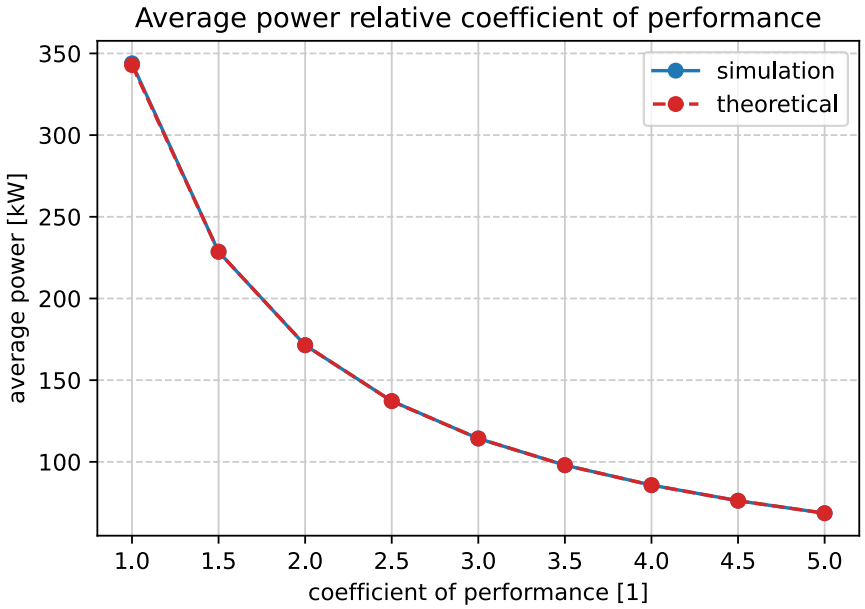


Figure 4.16: This figure shows the relationship between Coefficient of Performance (CoP) and average power. Values for average from the simulation were calculated as the time average of all strictly positive values, this equals the average power when the heat pump is active. The theoretical values have been calculated using equation (3.20), based on the average $|Q_H|$ from the simulation using the CoP equal to 3.5.

4.5 Steam

Up to this point all analysis, and all results, have been regarding HT water and its heat demand. There are mainly three reasons for why the thesis gives the majority of its focus to this, rather than steam. The first is that much more information has been available about the functionality of the systems. The second is that in order to properly model the steam demands, VLE fluid must be used. This is much more challenging, both when creating the models and when it comes to performance. Many of the systems simply could not be modelled in any reasonable time for the length of the simulations required. The third is that the usage of steam on the case ship is already a problem: Data indicates that there is more energy need in port than there is waste heat available at sea, given an equal amount of time in each state. Anyway, studying steam *in some form* is of interest.

Similar to earlier, a simulation was run for the eight-day cruise. Plotting the results for the steam takes on quite a different form than for the HT water. For the HT water the parameter defining the energy flow at various points was the temperature, but all points in the steam cycle are constant in both pressure and specific enthalpy[2], when it stabilizes. These points can be seen in figure 4.17.

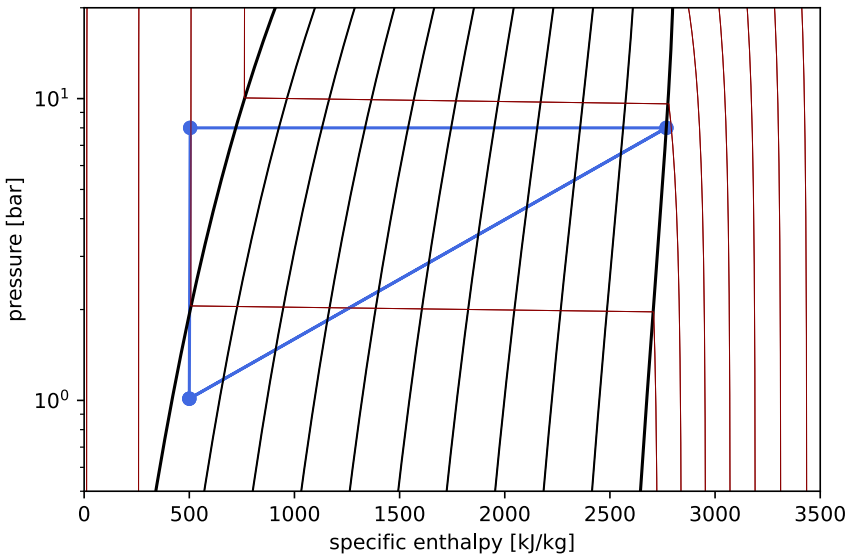


Figure 4.17: Figure showing a steam cycle taken from a simulation of the model. The datapoints were taken early in day 1, after the cycle had stabilized. The steam cycle is clockwise in the figure shown: The compression of the gas, left line, increases the pressure of the gas from 1 atm to 8 bar. At 8 bar the steam is evaporated by the heat from the Exhaust Gas Boilers (EGBs), this is the top line in the figure. After this the steam is condensed by the users, which also lowers the pressure back down to 1 atm. This condensation is the diagonal line in the figure.

[2]For VLE fluids specific enthalpy must be used instead of temperature due to the latent heat.

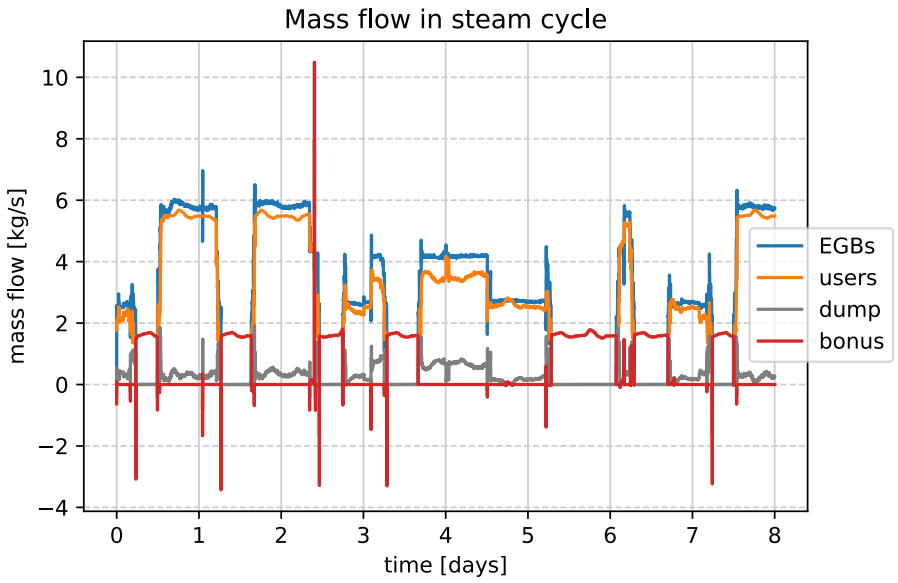


Figure 4.18: Figure showing the mass flow at key point in the steam cycle model. The mass flow through the heating by the EGBs is shown in blue, the mass flow to the users is shown in red. The mass flow labelled ”dump”, shown here in gray, is the mass flow not needed by the users. This mass flow is, simply, cooled down and recycled along with the mass flow from the users. Finally, the mass flow labelled ”bonus” shows the mass flow bypassing the EGBs, which then passes through an electric boiler and flows to the users. This mass flow is equal to the mass flow needed by the users not met by the mass flow from the boilers.

Instead, the parameter defining the energy flow at the various points is the mass flow; figure 4.18 shows the mass flow at important points in the steam cycle. The mass flow through the heating by the EGBs, shown in blue, is indirectly controlled by a PI-controller. See section A.1.3 for general info in PI-controllers, or just section A.1 for general control theory. The controller regulates the mass flow to the boilers, whilst a separator ensures only pure vapour flows from the boilers. The mass flow out of the separator is then the fraction of the mass flow in which is vapour (see ”quality” in section 3.1.5). All this means is that the controller seeks to have the same mass flow into the boilers as what comes out of the boilers, but the controller stabilizes so quickly one would not be able to see this in the graph. For that reason only the flow out of the boilers is shown.

The mass flow rate to the users, shown in orange, is calculated using the case data and equation 3.10 in section 3.1.2. The remaining steam is ”dumped”, shown in gray, which in our case means it is cooled (to no use) and flows back into the cycle along with the water from the users. Finally, the mass flow marked as ”bonus”, shown in red, is mass flow which bypasses the EGBs. This mass flow is determined as the mass flow available from the EGBs subtracted from the mass flow needed by the users.

Both the mass flow for ”dump” and the mass flow labelled ”bonus” pass the electric boiler, as a similar solution was planned using TES. However, as will be discussed further down, this has not been implemented.

First of, let’s discuss the parts of figure 4.18 which are, frankly, an eye-sore. At many points in the simulation the various mass flows have ”noisy” datapoints. What is meant by this is that for very short periods of time (usually only one time-point in the simulation) they have an uncharacteristic value, a value which differ vastly from neighbouring datapoints. This happens during specific events in the simulation, e.g. large, instantaneous changes to thermal needs. The bad datapoints occur due to weak points in the model, which is why they are frequent here in the steam simulation and not in the model for the HT recovery system. As will be discussed more later the steam model is quite a bit more complex, which has caused more weak points. To remedy some of this it was chosen to try to clean up the data a bit. Values under zero were removed, and the outlier point for ”bonus” mass flow at about 2.5 days were also removed. This gave the results shown in figure 4.19. Note that the high points in the mass flow for the EGBs are not like this, as they last for many datapoints, usually several minutes in simulated time.

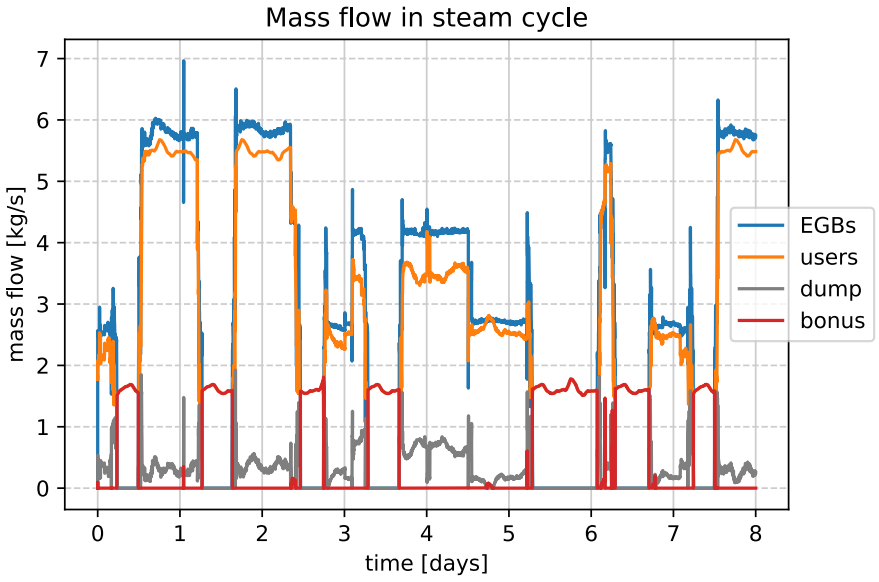


Figure 4.19: This figure shows the mass flow at key points in the steam cycle after some ”cleanup” has been done. See figure 4.18 for the raw data, or for more information on the different mass flows.

Finally, some average and total values were calculated, using the cleaned up data. These can be seen in table 4.3. The rest heat was found using the mass flow dumped and multiplying this with the enthalpy difference over the condensation. The heat (flow) needed to be covered by a boiler/TES was found using the ”bonus” mass flow, in a similar fashion to the rest heat.

Table 4.3: Table showing some average and total values calculated using the cleaned up results from the steam simulation. Average values are for non-zero periods, whilst total values are for the complete eight-day cruise.

	Rest heat	Heat need
Average [kW]	663	3373
Total [MWh]	114	241

Chapter 5

Discussion

In this chapter the results from the previous chapter will be discussed and compared, were necessary critical information will be repeated/summarized. Following this there will be a section discussing other solutions that was proposed, why they are not included here and some advantages/disadvantages of the solution. Finally there will be some further discussion on things such as the model, the analysis done to find parameters for TES and case choice.

5.1 Discussion on solutions

In this section the various solutions will be discussed and compared.

5.1.1 Comparing TES

First and foremost the two solutions using TES, solution 1 & 2, will be discussed. In section 4.2 a water tank was studied, and the smallest possible volume for this tank was found (given the conditions necessary for the solution to function properly, see section 4.1.1). Section 4.3 studied the possibility to swap this water tank out with a PCM using the material CrodaTherm™ 74. For the PCM achieving a satisfactory heat exchange rate proved difficult. An extensive analysis was, however, done, and an optimal solution with minimal volume was found. This was then improved upon by switching heat exchange strategy for the PCM, going from a Plate Heat Exchanger (PHE) to a Pillow-Plate Heat Exchanger (PPHE). The final results from these analyses can be seen in table 5.1, showing both the minimal volume possible as well as the weight of this solution[1]

As one can see from table 5.1 the PCMs can achieve significantly smaller volumes

[1]This may not be the minimal weight for the PCMs. A bigger volume but smaller weight could be possible by shifting the relative portion of material vs. plates & water (for heat exchange).

than the water tank. However, the PCMs were much harder to tune in order to meet the demands of the cruise ship. Any real implementation of TES will need to be tuned to have significant buffer. Doing this for the water tank is much simpler than for the PCMs, as the water tank only has one parameter (volume) whilst the PCMs have many (side lengths of the plates, number of plates, width of the PCM, etc.). The solutions for PCMs in this thesis even fixed many of the parameters for the plates (e.g. duct width) in order to limit the amount of parameters that could be tuned.

Table 5.1: Final results from simulations and analysis on solutions using Thermal Energy Storage (TES): V is volume and m is mass[2]. All three TES were optimized for minimal volume. WT is short for water tank, whilst PHE and Pillow-Plate Heat Exchanger is short for Plate Heat Exchanger and PPHE, respectively. These are also known as plate in block and pillow-plate in block PCMs.

	WT	PHE PCM	PPHE PCM
$V[\mathrm{m}^3]$	850	376	265
m[tonnes]	829	378	368

Another problematic point when it comes to the PCMs is the trouble with achieving a successful heat exchange rate. The water tank had no trouble with this, whilst the PCMs needed to be finely tuned in order to get good heat exchange rates within a reasonable volume. This was, however, some of the intent behind the choice of requirements for the TES: To test the limits of what was possible to achieve with a PCM. It was probably unreasonable to expect the TES to charge so quickly from zero energy state in so short time span before the first port stay. Most likely it would be possible to have the TES charged to some degree beforehand.

Finally, on to weight, as this is also of great importance to ships. As one can see in table 5.1 the difference in weight is close to proportional to the difference in volume for the water tank and the Plate Heat Exchanger PCM. The water tank, weighing in on 829 tonnes, is over twice as heavy as both PCM variants. And it is quite interesting to note that the Pillow-Plate Heat Exchanger PCM has a volume 30% smaller than that of the Plate Heat Exchanger PCM, but weighs under 3% less.

5.1.2 TES vs. HP

There are significant differences between the solutions utilizing Thermal Energy Storage, and the one utilizing a heat pump. First of all is the availability of energy: The TES needs to store the available energy at sea in order to use it in port, that's just how they function. This means that if all the energy stored is used, there is no way to get more before the ship is back to cruising. For that reason any solution using a TES would need to dimensioned to meet significantly more demand than what is

[2]Important to note here that "tonnes" is the metric unit equivalent to 1000 kg. This is not to be confused with imperial "tons", which come in both British and American variants.

normal, to be safe. A heat pump does not need this, as the heat pump need only be dimensioned to meet a maximum heat flow. The water tank in the model did not need to be dimensioned with heat flow in mind, but in reality this would need to be accounted for. With the type of tank shown here, a constantly well mixed tank, the water flow in/out would need to be able to meet the demands of the ship. But it is easier to dimension for mass flow than for heat flow, which is an advantage of the water tank. The PCM would need to be dimensioned for both mass flow and heat flow, similarly to the heat pump.

There are mainly two disadvantages to the heat pump solution. The first is that the heat pump requires power in order to deliver heat, something that may not be available in port to the capacity needed. The power available to the ship *needs* to meet the direct power needs of the crew and passengers on the ship, so adding the heat need may be too much. This will, naturally, depend on the ship and port in question[3]. The second reason for trouble with heat pumps relates to the working fluid. Many working fluids in heat pumps are flammable, and since cruise ships are very strict regarding safety this is problematic. For example, a common working fluid for heat pumps is CO2, which many in the industry has given clear indication is a no-go.

5.1.3 Steam

First off, the most important results from the steam simulation for the purposes of this discussion can be found in table 4.3, but will be repeated here. The total amount of energy needed to cover heat demands for steam, when this cannot be met by waste heat from the EGBs, was found to be 241 MWh. On the other hand, the total amount of energy dumped was only 114 MWh. This means that before even discussing any sort of heat loss related to storing excess energy, there is already too little energy available for a solution using TES.

There are mainly two ways to do this. The first is to meet some of the demand with a TES, and combine this with either a heat pump or an electric boiler. The Thermal Energy Storage would then not be there to meet heat demands, but rather serve as an extra heat addition to the solution lowering the power required by the heat pump/electric boiler. The second solution is to change the amount of steam that is used by the ship. It is possible to optimize the steam usage, and/or change the heat source used for many of the areas. One can, for example, use more of the heat in the HT water to meet user demands, e.g. by further preheating the potable water. The heat exchange between HT water and potable water only heats the potable water to approximately 35 °C. It should be possible to increase this to at least 50 °C.

One big problem with testing steam solutions is the complicated nature of simulating using VLE fluid. A good measure of this shows up with respect to the file size of

[3]Another, additional power user that could rear its ugly head would be the need to store energy to a battery if the ship is not only to stay zero emission in port but also when cruising in to/out of the port (e.g. Geirangerfjorden). More discussion surrounding this can be found further down in section 5.3.4.

the results: A simulation of the types shown previously in this chapter has all been between 1 and 3 GB, the simulation run for the steam had an original result file of 35 GB. This was caused by the fact the the VLE fluid would crash with the numerical precision used in all other simulations. The tolerance had to be cut in half, from 10^{-3} to 5×10^{-4}, and the number of points increased by 300%.

5.2 Other solutions

Many solutions have been considered, and tested, but not all of them can be given a significant spot in this thesis. For many of the solutions they have been left out because they, for some reason or another, function very poorly or lead to huge problems that need to be addressed. They will all, however, be briefly discussed in this section.

5.2.1 PCM below working temperature

One of the core difficulties with solution 2 is that both charging and discharging happen around the working temperature of the HT recovery system, so the temperature difference in the heat exchange becomes quite low. This must then be compensated by having a large surface area for heat transfer in the PCM, which can increase both its weight and price.

A proposed solution for this was to lower the temperature of the PCM a little bit, around doubling the temperature difference in both charging and discharging. This has the consequence that the water will not be heated to the traditional working temperature of the HT recovery system. Using, for example, PureTemp 68 (see table 2.1 for info on the physical quantities of this PCM) a temperature difference of about 10 K can be achieved.

The water will then only be heated to about 58 °C during discharging, which is far below the required temperature of 70.5 °C. However, the requirement is there to ensure that the HT cooling water for the engines does not fall below 74 °C. This can happen if too much heat is exchanged to the HT recovery system, which in turn can happen if the water in the HT recovery system falls below 70.5 °C. The requirement should therefore only be *strictly* necessary when the engines are running. One could, for example, use additional piping and/or small water tanks to ensure that the water heat exchanging with the engine cooling water is at 70.5 °C when the engines are turned on. The solution should then be satisfactory.

There is another problem, however. The users of the HT water also have temperature requirements. These users, air conditioning and potable water preheating, should in theory be work with this solution; the temperature requirement for air conditioning should not be above 50 °C in general, and the potable water does in this case never go above 40 °C. The biggest problem with the lower temperature of the HT water will likely be that the heat exchange rate is lowered, due to the smaller temperature

difference between the HT water and its users. It was concluded that these problems were ill-suited for being studied by the numerical model as-is, so they were not studied any further.

5.2.2 One PCM per user

Another solution inspired by the problem of achieving sufficient heat exchange was the proposed solution to have one PCM for each user of HT water, meaning one PCM for air conditioning and one for potable water preheating. The idea was to charge these, as normal, using waste heat in the HT recovery system. The two PCM were then discharged directly to the users, to the air conditioning system or potable water flow. This removed the heat exchange between the HT recovery system and TES when discharging.

There was, however, a problem regarding any testing of this solution: Very little information was known about the air conditioning system of the ship. It has been known that the air conditioning heat exchanges with the HT recovery system parallel to the potable water preheating, but beyond this the water is murky. A constant heat flow of 240 kW has been used, as this was an average given, but without dynamic heat flow and temperature information on the other side of the heat exchange it was viewed as impossible to create meaningful results regarding this solution. Another point to make is that this solution doesn't solve the problem of heat exchange rate on the troublesome side, charging, only on the unproblematic side, discharging.

5.2.3 WT and PCM

It could also be of interest to study a mixed solution where both types of TES is utilized: A water tank for quick charging/discharging and a PCM for long charging/discharging. A very interesting problem that would show up with this solution is how to then manage the waste heat. One can no longer store as much waste heat in the TES as possible, it must be distributed among the two. Does a hypothetical controller need to know an approximate time until next port stay to do smart distribution, or is this not necessary? Can heat be distributed to the PCM first, where not all waste heat will be extracted, and the remainder can be put in the water tank? How will the change in temperature for the water tank change the heat exchange to the PCM? These are all good questions, but they will not be answered by this thesis.

5.2.4 One PCM in total

The final solution that was up for discussion, that shows some merit of being a good one, is to have only one PCM for both HT water and steam. Many choices can be made for how charging/discharging should be done, all very dependent on the melting temperature of the material used. For example, one could put the melting temperature

above the working temperature of the HT water and below that of the steam. The PCM can then be naturally charged by the steam and discharged by the HT water. However, doing the opposite, discharging to steam and charging from HT water would not be directly possible. This could possibly be done using heat pumps, but the exact implementation of this could prove tricky.

5.3 Further discussion

This section will briefly discuss some areas that have not been covered in the previous sections, as they are more related to the methodology used than the results of the thesis.

5.3.1 Solutions

One important point to take from all of this, that has not yet been addressed, is the challenge behind meeting the requirements for charging/discharging TES when heat exchange is involved. This is also a damper for many solutions where the TES operates at a different level, or by a different fluid, than the working fluid(s). Every heat exchange leads to loss of energy, that cannot be extracted in the time required, as well as a lowering of temperature. This means that any well thought out solution seems to depend on a minimal number of heat exchanges.

5.3.2 Analysis

An area that has been viewed as somewhat unsatisfactory in respect to the work done has been the strategies used for finding parameters for the PCMs. The requirements for the PCMs were not to meet some given heat exchange rate, but rather meet the requirements of a dynamical simulation. It was therefore noted from pretty early on that it would be advantageous to use numerical optimization to find the minimal volume. This was, however, regarded as being outside the scope of the master thesis. It was also noted that the student, me, lacked the knowledge needed to accomplish this.

5.3.3 Model

The results presented in this thesis are only as good as the model used to find them, and with any dynamical model there is always the issue of accuracy. All the details on the models can be found in appendix B, but a few point of interest will be discussed here.

The core of the entire model lies in the model for the engines used (see B.1.1), which was based on data found in the datasheets of the engine type used by the case ship

(see C.1)[15][16]. This model has been tested extensively, in both the specialization project in the fall of 2021 and summer internship at SINTEF in the summer of 2021. The model produced the wanted, and expected results, in the known engine load range of 50-100%. However, the simulations frequently use engine loads lower than 50%, and in this range the model is based on assumptions. For example, it was assumed that there is a linear relation for mass flow of exhaust gas between 0% and the last known point (50%). It is fairly safe to assume that the mass flow is zero for no load, but the linear relation is likely not true. Another solution to this has not been found.

The model for the exhaust gas boiler is based on a controller finding the heat necessary to cool the exhaust gas to a given temperature (190 °C). This energy is then extracted and put directly into the steam in order to heat it. A more realistic model, that would dynamically change given the conditions of the exhaust gas and steam, would use a heat exchanger between the two. This model was never completed, as troubles with VLE fluid kept showing up. More information on this model can be found in appendix B.1.2.

The model for the power supply system, sometimes referred to as the engine system, was based on figure 2.2 in section 2.1. The model can be found in B.2.3 It utilizes the two models previously discussed here, the engine model and the EGB model. Not a lot of error sources are added in this layer, the biggest being heat exchange between the engine cooling water and the HT recovery system. These were tuned to give results on line with what was known, but they cannot be assured to give the same dynamics as their real equivalents.

The steam model, used only for the results in section 4.5, is likely the most lackluster of the models used. As discussed in section 4.5 the simulation run resulted in many extreme datapoints, datapoints vastly different from their neighbours. These points are known to be caused by quick swaps in values such as the thermal need for steam. In an ideal world these problems would be solved by improving the model, but this was simply not possible to do in the time given to this master thesis. The current model for the steam took the majority of the time used on building the models, and is still the most lackluster. The main reason for this is that the VLE fluid is much harder to work with than the pure liquid fluid. See section 3.2.3 for explanation on the various fluid types used by the TIL library in Dymola/Modelica. The main problem that remains in the model is that the mass flow labelled as ”bonus” in figure 4.18 and 4.19 is not given explicitly by the other mass flows, but found through mathematical manipulation of sensor values. This is levelled by an artificial boundary condition at the high pressure. This causes weird mass flows when the values change quickly, or passes certain values, e.g. when the mass flow to ”dump” hits zero. See section B.1.6 of the appendix for the model of the steam cycle.

5.3.4 Case

One of the areas which has affected the results in a significant way is the case choice. The eight-day cruise was a natural choice when that data became available, as it posed a significantly more realistic case than what was done previously. In the specialization project done prior to this a simple two-day case was used, which consisted of 24 hours cruising followed by 24 hours in port. Although it worked very well at providing an initial analysis of the models and the solution using a water tank as TES it lacked in many areas, most notably when it comes to typical times for cruising port stays. The eight-day case provided better dynamics with various length on both port stays and cruising periods, but also gave rise to the more challenging periods such as the back-to-back port stays in day 5 and 6 of the case.

However, the eight-day case lacked, similar to the two-day case, much live data. This data was found using general data on the ship provided by Carnival earlier, where the relationship between quantities such as load/speed and heat need/availability could be found. Since load and speed was available for the eight -day cruise it was easy to find values for most of the data needed. However, some assumptions had to be made and these can have affected the results. One key assumption that was made in the development of the case-files is that when cruising under 5 kn the power need was assumed to be that of 5 kn. There was no way around doing an assumption here, as no values were known for speeds lower than 5 kn, but one could for example have chosen a linear interpolation between the known point at 5 kn and the point zero, zero[4]. Instead the power need is equal to that of 5 kn right up until the speed is exactly zero, then the power need is zero. The reason this was chosen over linear interpolation is that it will overestimate the power need, whilst linear interpolation would likely underestimate it.

5.3.5 Material

The last source of error that will be discussed here is that of the material used in solution 2, CrodaTherm™ 74. This material has, as mentioned both in section 2.4 and 4.3, multiple unknown physical values. These quantities are the heat capacities and the thermal conductivities of the material, for both liquid and solid phase. The estimated values, along with all known values can be seen, as mentioned many times earlier, in table 2.1. The estimates were made based on similar types of material, organic, with melting temperatures close to that of CrodaTherm™ 74. It is naturally very hard to say how accurate these estimates are, as the material CrodaTherm™ 74 could have very different properties from the materials the estimate is based on. One thing is, however, clear: The estimates only affect the amount of heat stored in the sensible heat of the PCM and the heat transfer rate, but both of these have significant impact on the final results for volumes for the PCM solutions. The latent heat, which accounts for the majority of the storage capacity of the PCM, remains the same.

[4]It is fairly safe to assume that the power need for propulsion is zero for zero propulsion.

Chapter 6

Summary

Three solutions were proposed for meeting heat demands for HT water in port: Introduce a water tank, PCM or a heat pump to the HT recovery system. Models were created to test these solutions, and the simulations run showed that all three could meet demands.

The total volume needed for the water tank was 850 m^3, whilst the solution using a pillow-plate in block PCM could get as low as 265 m^3. The more standard plate in block PCM needed a volume of 376 m^3. The heat pump solution showed that a heat pump with Coefficient of Performance (CoP) of 3.5 needed an average power of 98 kW, with a maximum power of 99 kW. The heat pump has the disadvantage that it needs power to operate, altough less than a direct heater. However, in contrast to the solutions using TES it does not have a maximum energy it can deliver, only heat flow. The PCM needs to be designed with both total energy and maximum heat flow in mind, whilst the water tank need only be tuned for total energy.

When it comes to steam the numerical complexity made it difficult to study solutions, however, a simple solution using a heater (representing an electric boiler) gave an initial test. This showed that there was less excess energy when cruising, than energy needed for the port stays. This means that no solution using only a TES, no matter how perfect, could be implemented to meet the heat demands. Two solutions to this was proposed: Either combine the TES with a heater (heat pump or electric boiler) or change the heat use to better utilize waste energy from HT water, in order to lower the demand for steam both in and out of port.

Bibliography

[1] C. Gabrielii, *Cruize - cruising towards zero emissions*, SINTEF Energi AS, Nov. 2020. [Online]. Available: https://www.sintef.no/en/projects/2020/cruize-cruising-towards-zero-emissions/.

[2] Á. Á. Pardiñas and A. Sevault, "Integration of compact thermal energy storage on cruise ships: Opportunities and challenges," SINTEF Energi AS, Trondheim, Norway, Project memo, Dec. 2020.

[3] Z. Guangrong, "Ship energy efficiency technologies - now and the future," VTT Technical Research Centre of Finland Ltd, Espoo, Finland, Research highlight, 2017.

[4] *Potential of thermal storage for hot potable water distribution in cruise ships*, Italian Thermal Machines Engineering Association, Pisa, Italia: Elvsevier Ltd., Sep. 2018.

[5] *"DIESECON" - Exhaust Gas Boiler - operation, maintenance and instruction manual*, Casinghini Heatex s.r.l., Brescia, Italy, Oct. 2011.

[6] P. C. Hemmer, *Termisk fysikk*, 2nd ed. Tapir akademisk forlag, 2017.

[7] M. Mastani Joybari, H. Selvnes, A. Sevault, and A. Hafner, "Potentials and challenges for pillow-plate heat exchangers: State-of-the-art review," *Applied Thermal Engineering*, p. 118 739, 2022, issn: 1359-4311. doi: https://doi.org/10.1016/j.applthermaleng.2022.118739. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1359431122006834.

[8] M. Piper, A. Olenberg, J. Tran, and E. Kenig, "Determination of the geometric design parameters of pillow-plate heat exchangers," *Applied Thermal Engineering*, vol. 91, pp. 1168–1175, 2015, issn: 1359-4311. doi: https://doi.org/10.1016/j.applthermaleng.2015.08.097. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1359431115008947.

[9] J. Carvill, *Mechanical Engineer's Data Handbook*, 1st ed. Elsevier Science & Technology, 1994.

[10] [Online]. Available: https://www.ipu.dk/products/coolpack/.

[11] *Modelica language specification*, 3.5, Modelica Association, Linköping, Sweden, Feb. 2021.

[12] L. R. Petzold, *A description of DASSL: A differential/algebraic system solver*, Sep. 1982.

[13] I. H. Bell, J. Wronski, S. Quoilin, and V. Lemort, "Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical property library coolprop," *Industrial & Engineering Chemistry Research*, vol. 53, no. 6, pp. 2498–2508, 2014. doi: 10.1021/ie4033999. eprint: http://pubs.acs.org/doi/pdf/10.1021/ie4033999. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/ie4033999.

[14] O. Arsenyeva, J. Tran, M. Piper, and E. Kenig, "An approach for pillow plate heat exchangers design for single-phase applications," *Applied Thermal Engineering*, vol. 147, pp. 579–591, 2019, issn: 1359-4311. doi: https://doi.org/10.1016/j.applthermaleng.2018.08.083. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S135943111832492X.

[15] *Wärtsilä 46 - project guide for marine applications*, Wärtsilä Finland Oy, Vaasa, Finland, Jan. 2001.

[16] *Wärtsilä 46F - product guide*, Wärtsilä Finland Oy, Vaasa, Finland, Feb. 2020.

[17] J. G. Balchen, T. Andresen, and B. A. Foss, *Reguleringsteknikk*, 6th ed. Institutt for teknisk kybernetikk, NTNU, 2016.

Appendix A

Controllers

In this appendix everything related to controllers will be discussed. First up is some control theory, both covering the basics and some application of the theory on areas related to the controllers used for this thesis. Following this will be a a walktrough of the various controllers implemented in the models, and their respective tuning.

A.1 Control theory

This section will begin with a bit of a general introduction to control theory, as it is assumed the reader does not necessarily have any knowledge about this subject. After this there will be a discussion around the details behind the specific controllers.

A.1.1 General

This section will, as mentioned above, speak a bit about the core concepts of control theory. It will only focus on the simplest form of control theory: Linear control theory. One can, in linear control theory, describe any system with the form

$$\dot{\boldsymbol{x}} = \mathbf{A}\boldsymbol{x} + \mathbf{B}\boldsymbol{u} \tag{A.1a}$$

$$\boldsymbol{y} = \mathbf{C}\boldsymbol{x} \tag{A.1b}$$

where $\boldsymbol{x}$ is a vector of all state variables, $\boldsymbol{y}$ is a vector of all measurement variables and $\boldsymbol{u}$ is a vector of control inputs. $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are matrices describing, respectively, the dynamics of the system, the dynamics of the control input and the correspondence between state variables and measurement variables [17]. Let's simplify the case, and study only scalar systems. This yields

$$\dot{x} = ax + bu \tag{A.2a}$$
$$y = cx \tag{A.2b}$$

where now x, y and u are scalar variables instead of vectors, and a, b and c are scalar quantities instead of matrices. All variables still describe the same as before. One may now substitute in y for x in equation (A.2a) using equation (A.2b):

$$\dot{y}/c = ay/c + bu \implies \dot{y} = ay + bcu \tag{A.3}$$

Letting $b \to b/c$, to simplify the notation, yields

$$\dot{y}(t) = a \cdot y(t) + b \cdot u(t), \tag{A.4}$$

where the notation now specifies that y and u are variables in the time-domain by writing them as $y(t)$ and $u(t)$ respectively. This is done to separate them from their corresponding variables in the s-domain, the domain of the Laplace-transformed variables (note that s is a complex variable, so by taking the Laplace transform one moves from a real variable t to a complex variable s). Taking the Laplace transform of equation (A.4), and assuming initial conditions of zero, yields

$$s \cdot y(s) = a \cdot y(s) + b \cdot u(s), \tag{A.5}$$

where the variables are now denoted as $y(s)$ and $u(s)$ to emphasize that they are in the s-domain. This can be rearranged to

$$y(s) = \frac{b}{s - a}\, u(s), \tag{A.6}$$

which in turn gives the *transfer function*

$$h_u(s) \equiv \frac{y(s)}{u(s)} = \frac{b}{s - a}. \tag{A.7}$$

The transfer function, in effect, describes the way in which an input (in this case u) will affect an output (in this case y). The transfer function for a controller is defined as

$$h_c(s) \equiv \frac{u(s)}{e(s)} \quad ; \quad e(s) = r(s) - y(s) \tag{A.8}$$

where r is the reference value and e is the error: The deviation from the reference value. This can now be put together to make a complete feedback controller system, albeit a simple one; this is shown in figure A.1.

More complicated systems can be described by studying more complicated equations of motions, which may require a more complex control system. For more information on control theory see [17].

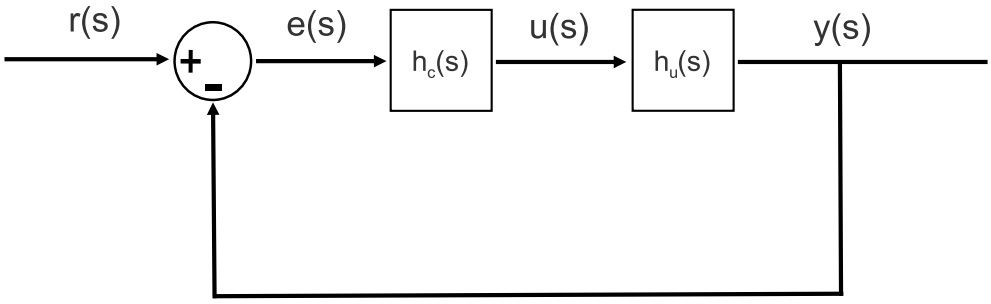


Figure A.1: Block diagram of a basic feedback controller system. The two transfer functions defining the system and controller are, respectively, $h_u(s)$ and $h_c(s)$. $y(s)$ denotes the parameter one wishes to control, $r(s)$ denotes the reference value, $e(s) = r(s) - y(s)$ denotes the error from the reference and $u(s)$ denotes the control output.

A.1.2 Valve controller

Throughout the work a system started to show up, and there came a need for an appropriate controller. An example of the type of system is shown in figure A.2. To properly control this system one must first understand the dynamics of it.

An input mass flow is split into two pipes, one which changes the temperature and one which doesn't. The goal is to control the output temperature of the system by adjusting a valve which determines the amount of mass which flows through each tube. The input to this valve, which is placed where the mass flows are joined, is the relative amount which is sent through the heating/cooling pipe. From the law of conservation of energy the following must hold: The energy going into the valve must be the same as the energy coming out of it, meaning

$$\dot{H}_{out} = \dot{H}_0 + \dot{H}_{BP}, \tag{A.9}$$

where 0 denotes the water passing through the heating/cooling, and *BP* indicates the pipe bypassing this. As was shown in subsection 3.1.2: $\dot{H} = c_p T \dot{m}$. Inserting this into equation (A.9) yields

$$T_{out} \cdot \dot{m}_{out} = T_0 \cdot \dot{m}_0 + T_{BP} \cdot \dot{m}_{BP}. \tag{A.10}$$

Dividing by $\dot{m}_{out}$, and replacing T_{BP} with T_{in} and $\dot{m}_{BP}$ with $\dot{m}_{in} - \dot{m}_0$ yields

$$\begin{aligned} T_{out} &= T_0 \frac{\dot{m}_0}{\dot{m}_{out}} + T_{in} \frac{\dot{m}_{in} - \dot{m}_0}{\dot{m}_{out}} \\ &= T_0 \frac{\dot{m}_0}{\dot{m}_{out}} + T_{in} \frac{\dot{m}_{in}}{\dot{m}_{out}} - T_{in} \frac{\dot{m}_0}{\dot{m}_{out}}. \end{aligned} \tag{A.11}$$

The following is now assumed: $\dot{m}_{in} = \dot{m}_{out}$. In section A.2.1 the validity of this assumption will be checked. One can now insert the parameter to be controlled, $y(t) = T_{out}$, and the control parameter, $u(t) = \dot{m}_0/\dot{m}_{out}$. This gives

$$y(t) = (T_0 - T_{in}) \cdot u(t) + T_{in}. \tag{A.12}$$

This control system is now so simplified that it is possible to create an ideal feed-forward controller, meaning no need for a feedback-loop. This can be seen by the fact that there are no derivatives in the governing equation of the system (eq. (A.12)). One then simply sets the output value $y(t)$ equal to the reference temperature T_{ref}, and rearrange the equation to get

$$u(t) = \frac{T_{ref} - T_{in}}{T_0 - T_{in}}. \tag{A.13}$$

So long as T_{in} and T_0 are measurable, and T_{ref} is given, this will output the ideal controller output. One important thing to note is the fact that there is a fraction in the expression in equation A.13. This means when $T_0 = T_{in}$ the control output is undefined. The reason for this can be seen by looking at equation (A.12), by inserting $T_0 = T_{in}$ one can see that $y(t) = T_{in}$: It is independent of $u(t)$. Since $u(t)$ can be chosen to be anything in these cases, it is chosen to be equal to zero. This is chosen because $u(t) = 0$ can in some sense be considered the "natural state" of the system: Passing the water through T_0 is the addition to the overall system, the action that changes the ”status quo”.

A.1.3 PI controllers

The PI-controller is a type of controller that outputs a value proportional to the error, $e = r - y$, and its integral. Meaning the control output is given by

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau)\, \mathrm{d}\tau \right), \tag{A.14}$$

which has the transfer function

$$h_c(s) = K_p \left(1 + \frac{1}{T_i \cdot s} \right). \tag{A.15}$$

Here K_p is the proportionality constant for the controller and T_i is the time constant of the integral portion. This time constant can be thought of as determining the time scale at which integration should affect the output of the controller. An important aspect of feedback controllers such as the PI-controller is tuning: Finding the correct values for the control parameters, K_p and T_i in this case. No specific method was used to find these values, but a common technique of applying a step function for testing was utilized heavily. More info on the tuning, and the corresponding tests, can be found in the section following this.

A.2 Controllers

In this section the implementation of various controllers will be discussed. In addition, results from simulations meant to assure the controllers are working as expected will be shown and discussed. This includes, in the case of the feedback-controller, the choice of both controller type and tuning parameters.

A.2.1 Ideal valve controller

This section will deal with the ideal valve controller, which have been implemented according to subsection A.1.2 using mainly the text editor. The code can be found in section B.2.7. The output from the controller is bounded to be between zero and one, as that is the minimum and maximum input values for the control valve.

To assure that the controller for the valves behaves as intended a few tests must be run. The most important points to check are:

- That he assumption $\dot{m}_{in} = \dot{m}_{out}$ holds.
- Behaviour for various relations between the different temperatures.
- Behaviour when the denominator is equal to, or close to, zero.

In order to check the first point, that our assumption $\dot{m}_{in} = \dot{m}_{out}$ holds, the model shown in figure A.2 was created. To the top right are the input values, a constant temperature $T_{in} = 80\,°\text{C}$ and a step function for mass flow, to the *overdetermined* boundary condition, labeled by the plus sign. This boundary condition takes in more values than what is necessary for the number of DAEs it corresponds to, and must therefore have a corresponding *underdetermined* boundary condition, which has too few. The step function follows the following behaviour:

$$\dot{m}_{in}(t) = 100\,\text{kg/s} - \Theta(t - 50\text{s}) \cdot 50\,\text{kg/s} \tag{A.16}$$

where $\Theta(t - \tau)$ is the unit-step function, also known as the Heaviside-function. It is defined to be zero for $t < \tau$ and one for $t > \tau$. This means that $\dot{m}_{in} = 100\,\text{kg/s}$ for $t < 50\,\text{s}$ and $\dot{m}_{in} = 50\,\text{kg/s}$ for $t > 50\,\text{s}$.

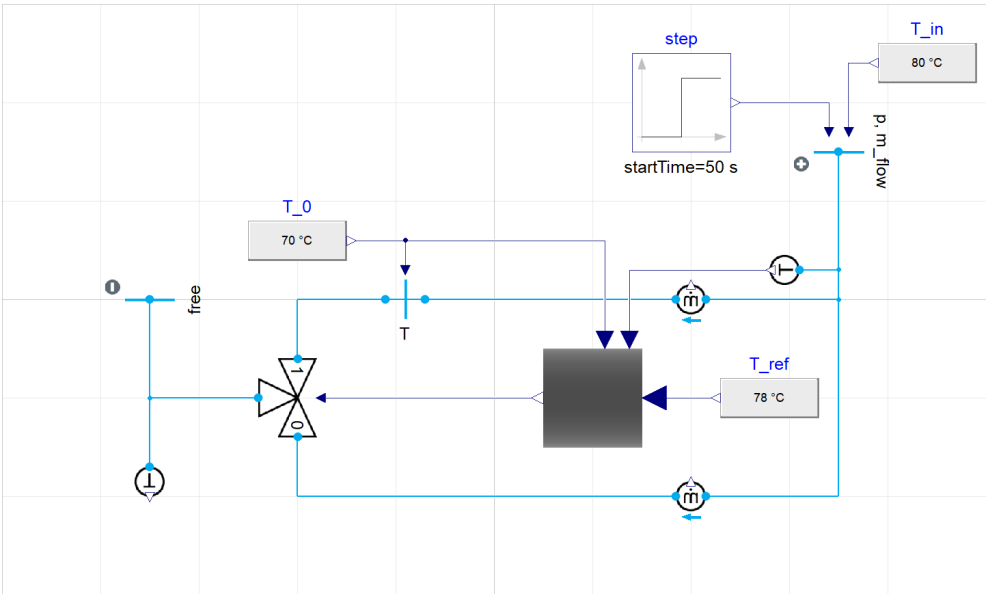


Figure A.2: Model used for testing ideal valve controller. The water flows from the *overdetermined* boundary condition in the top right (recognizable by its plus sign), is split and subsequently reunited in the controlled valve before flowing to the *underdetermined* boundary condition (recognizable by its minus sign). The valve controller is the black box in the middle, it takes in three temperatures: T_{in}, T_{ref} and T_0, and gives out the ideal control output for the valve given those temperatures (see section A.1.2).

In the model shown in figure A.2, water flows down from the top right, splits in two, and the two flows flow to the left before combining in the valve. The dark square block that can be seen in the middle is the ideal valve controller. As one can see from the figure, the controller takes in a reference temperature T_{ref} and two measurement temperatures: T_{in} and T_0. In this simple model T_0 is defined by a boundary condition, and it is therefore constant. From the valve the water flows out to the underdetermined boundary condition, labeled by a minus sign and the text ”free”. Simulating this model for 100s gives the results shown in figure A.3.

As one can easily see from the graph, the change in mass flow at time $t = 50\,\text{s}$ has no effect on the output temperature of the system. Considering the derivative of the Heaviside function is the Dirac delta-function, meaning the derivative of the step function approaches infinity in the instant of the step, one would assume that if a problem would arise with the assumption $\dot{m}_{in} = \dot{m}_{out}$ it would arise at this point. This does not happen, which indicates that the assumption holds.

Turning now towards verifying that the behaviour of the controller is as expected, and wanted. In order to test this, a very similar system to A.2 was built. The main difference from the model shown in figure A.2 is the input values to the overdetermined boundary condition: The mass flow is now set to a constant value of $\dot{m}_{in} = 100\,\text{kg/s}$, and the input temperature is set to a ramp-function. This ramp function, which is the

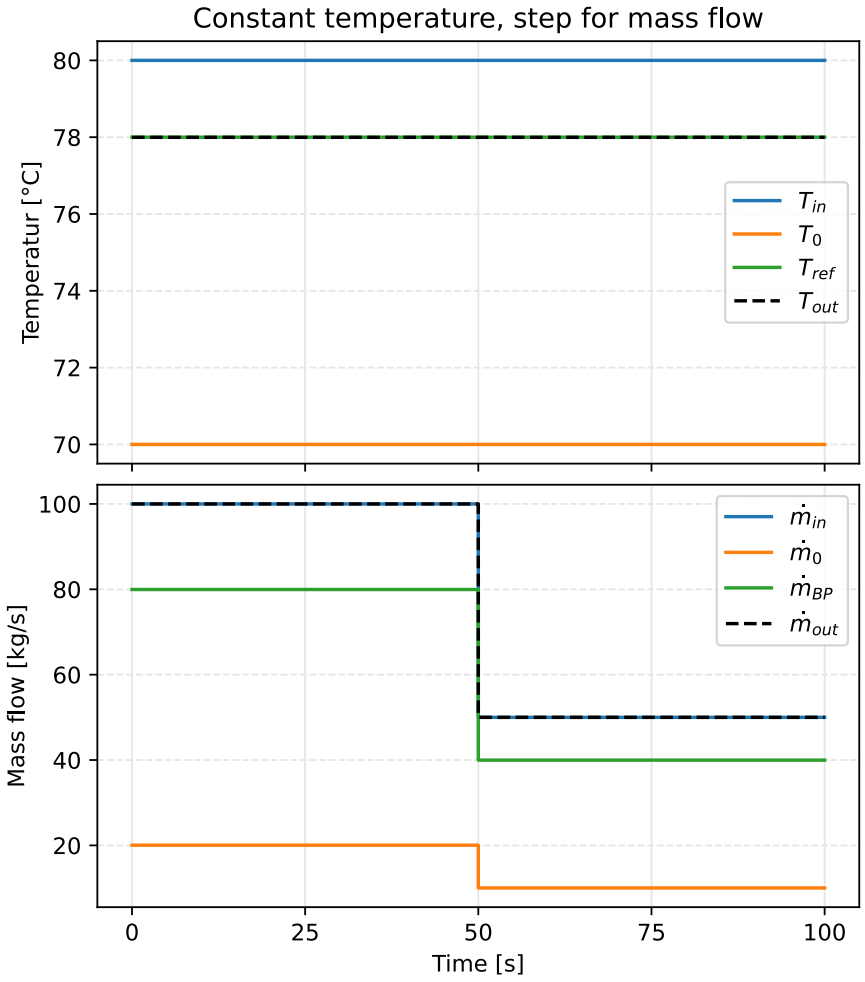


Figure A.3: Results from testing the assumption $\dot{m}_{in} = \dot{m}_{out}$ for the ideal valve controller. All temperatures where held constant, and a step function was applied to the input mass flow which activated at $t = 50\,\text{s}$. If the assumption did not hold there should have been an error in the output temperature visible around this point in time. This cannot be seen, which indicates that the assumption holds.

one implemented in the Modelica Standard Library, is a limited time ramp. They have the following behaviour:

$$B(t-\tau) = \begin{cases} B_0 & ; \quad t < \tau \\ B_0 + \Delta B \cdot (t-\tau)/T & ; \quad \tau < t < \tau + T \\ B_0 + \Delta B & ; \quad \tau + T < t \end{cases} \tag{A.17}$$

where τ is the start time of the ramp, T is the duration of the ramp, B_0 is the initial value and ΔB is the change. The ramp function used in this test has values $\tau = 20\,\text{s}$, $T = 60\,\text{s}$, $B_0 = 80\,°\text{C}$ and $\Delta B = -10\,\text{K}$. In addition to the change in inputs to the overdetermined boundary condition the value for T_0 was changed from 70 °C to 72 °C. Running the simulation for 100 s gives the results shown in figure A.4. As can be seen in the graph the output temperature follows the reference temperature for, approximately, the first 30 s. After the input temperature falls below the reference temperature one can see that the output temperature follows the value for the maximum of the two temperatures T_{in} and T_0 at all times. This is very satisfactory behaviour.

This test also shows another important aspect of the controller that was mentioned further up: Behaviour when the denominator is equal to or close to zero. As can be seen in the graph, the value for T_{in} passes the value of T_0; when they are equal, the denominator is zero. This may lead one to think that the denominator at one point is zero, but this is a numerical simulation: It is very unlikely that they are ever truly equal as the time-steps are discrete. They should however at some point, assuming the time steps are quite small, be very close to equal, meaning the fraction should blow up. This behaviour cannot be seen for the controller output, since the output is bounded in the interval $[0, 1]$.

It is possible to study this by looking at the unbounded output value stored in the controller, the temporary value calculated before it is bounded within the allowed interval. This can be seen in figure A.5. As one can see at the instance of a glance, the value blows up where it is expected to do so, when $T_{in} \approx T_0$.

The magnitude of the value reaches an order of 10^{12}. The worst case that was received in all tests done was an order of 10^{60}! This is obviously not great. However, Modelica supports floating point values up to about 10^{300} on most computers. It was therefore concluded that the overhead brought in by a ”fix” for this, e.g. by also avoiding values close to when the denominator is equal to zero, not just when it is exactly zero, was not worth it. Maybe in a revision of the block this will be added, but for now this strange behaviour is allowed to continue; it does not affect any of the results, the worst that can happen is that the simulation crashes due to value overflow.

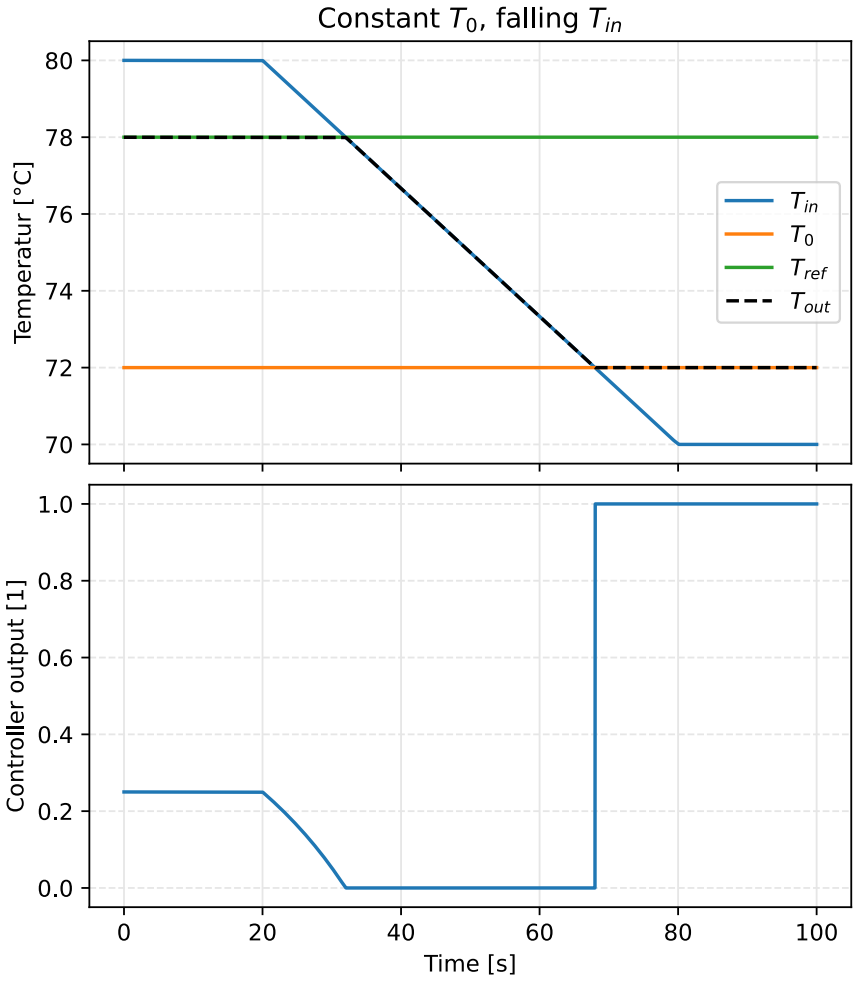


Figure A.4: Results from testing behaviour for various relations between measurement temperatures. T_0 and T_{ref} are both held constant, whilst T_{in} is defined by a ramp function; falling from 80 °C to 70 °C in the time interval $[20\,\text{s}, 80\,\text{s}]$. The output temperature remains at the reference temperature so long as there is a mix between T_{in} and T_0 which allows this. After this it is equal to the highest of the two.

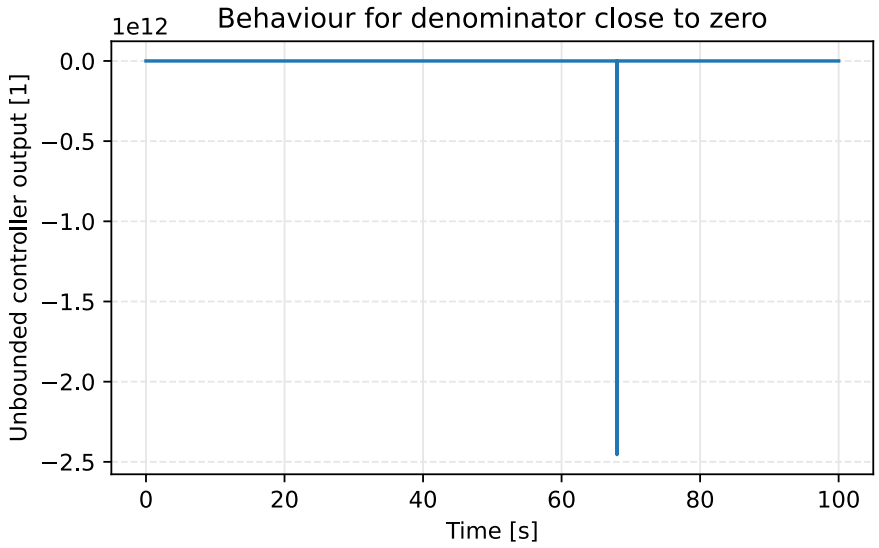


Figure A.5: Temporary, unbounded value for controller output. The value blows up a little bit before $t = 70\,\text{s}$, reaching a magnitude of order 10^{12}. This is not great, but it will not be dealt with any further since Modelica supports floating point values up to an order of about 10^{300}. The problem will not affect the results produced by the model, worst case scenario is that the model crashes.

A.2.2 Valve controller and water tank

Looking now at the dynamical behaviour of the controller, inserting a water tank for T_0, gives the results shown in figure A.6 for charging and figure A.7 for discharging. In both settings the results are as expected, and wanted:

- When there is an excess of heat ($T_{in} > T_{ref}$) this is stored in the water tank in such a way that the output temperature is kept equal to the reference temperature, when possible. When both temperatures are higher than the reference temperature all water flows through the coldest of the two, in this case the water tank.
- When there is a lack of heat ($T_{in} < T_{ref}$) this is extracted from the water tank in such a way that the output temperature is kept equal to the reference temperature, when possible. When both temperatures are lower than the reference temperature all water flows through the warmest of the two, in this case the water tank.

One final thing related to this controller which is interesting to touch upon is the time delay of measurements; in reality all measurements will not be instantly accurate and available. The measurements are both subject to delays within the instruments, delays due to communication between sensors and controller and delay due to the inertia of the physical system. Real water, unlike ”TIL-water”, does not instantly mix.

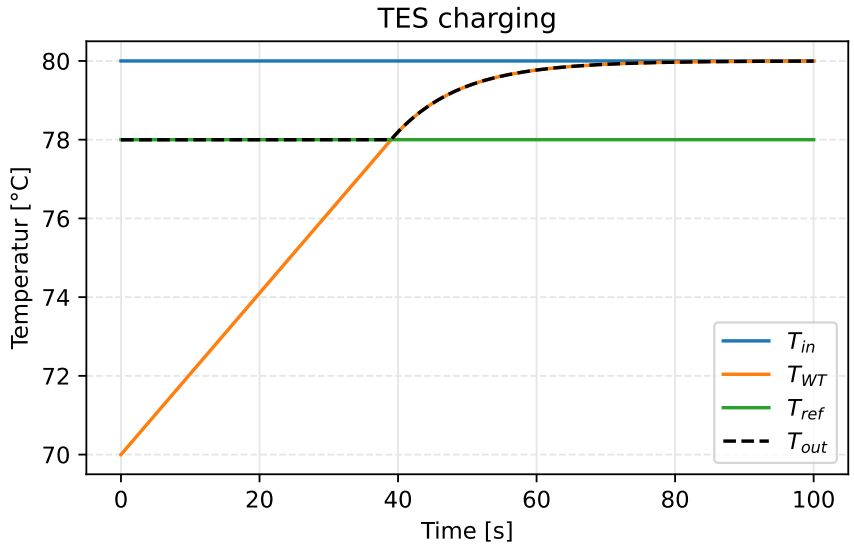


Figure A.6: Results from testing valve controller with a charging TES (water tank). When the temperature of the tank is lower than the reference temperature the output temperature is kept at the reference (water flow is a balance between the two pipes). After this, the output temperature follows that of the water tank (all water flows through tank).

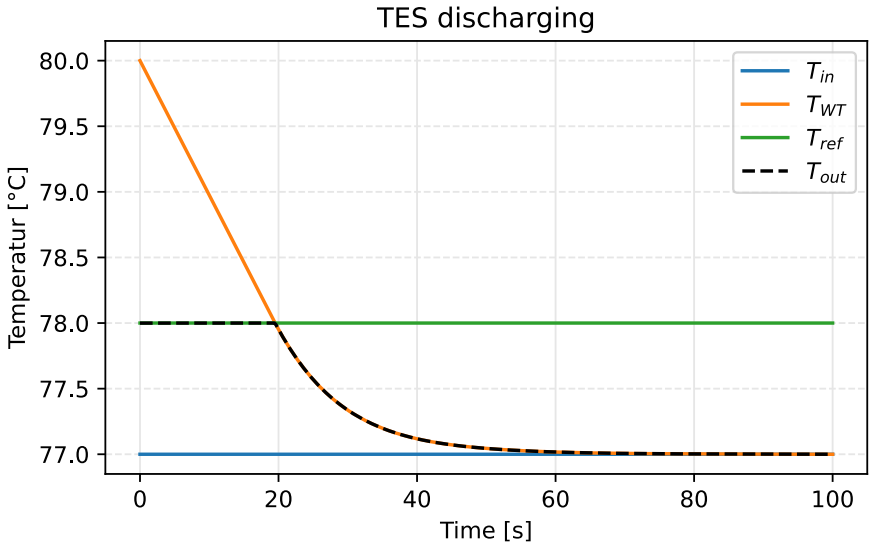


Figure A.7: Results from testing valve controller with a discharging TES (water tank). When the temperature of the tank is higher than the reference temperature the output temperature is kept at the reference (water flow is a balance between the two pipes). After this, the output temperature follows that of the water tank (all water flows through tank).

Measurement delay will not be touched upon outside of this section, it is however important to note the discrepancies and deviations from reality within the model. For that reason a simulation was run with the same conditions as the simulation shown in figure A.6, but all measurements were given a 1 s time delay. The results can be seen in figure A.8. Note that the temperatures shown in this graph are no longer the ”real temperatures”, but the measured temperatures including the 1 s delay. The output temperature now lies a little above the reference temperature as the temperature of the water tank rises towards that of the reference temperature.

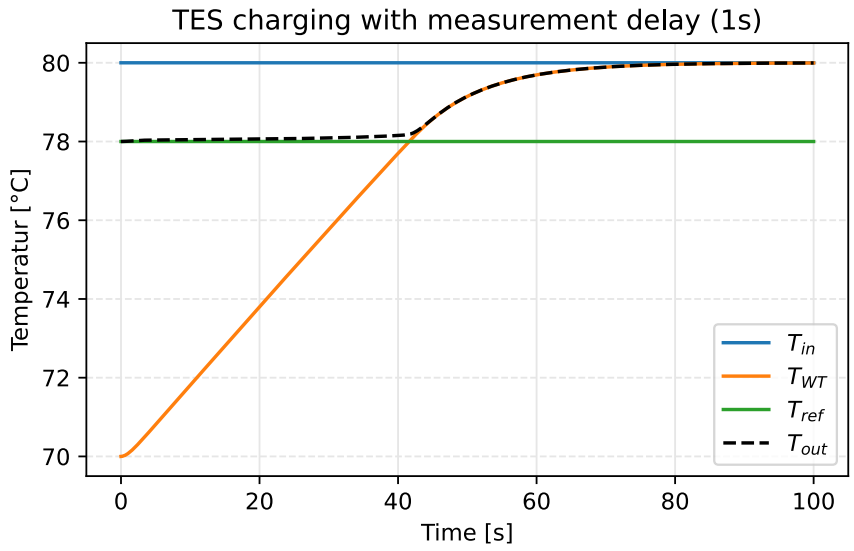


Figure A.8: Results from testing valve controller with a charging TES (water tank) where all measurements have a delay of 1 s. As one can see from the graph, the output temperature constantly lies a little bit above the reference temperature. This is caused by the rise in tank temperature between the measurement value and the actual value, so the controller is constantly mixing wrong by sending too much water through the water tank.

A.2.3 Controller for exhaust gas boiler

In order to test the controller for the Exhaust Gas Boiler (EGB) a Modelica model was created, which can be seen in figure A.9. The model closely resembles that of the EGB model (see section B.1.2). This model is quite straight-forward: Gas, moist air to be exact, flows from left to right in the figure; from the overdetermined boundary condition on the left to the underdetermined on the right. The overdetermined boundary condition has a constant mass flow $\dot{m} = 50\,\mathrm{kg/s}$ and step function for input temperature, which is described by

$$T_{in}(t) = 400\,^{\circ}\mathrm{C} - \Theta(t - 10\,\mathrm{s}) \cdot 20\,^{\circ}\mathrm{C}. \tag{A.18}$$

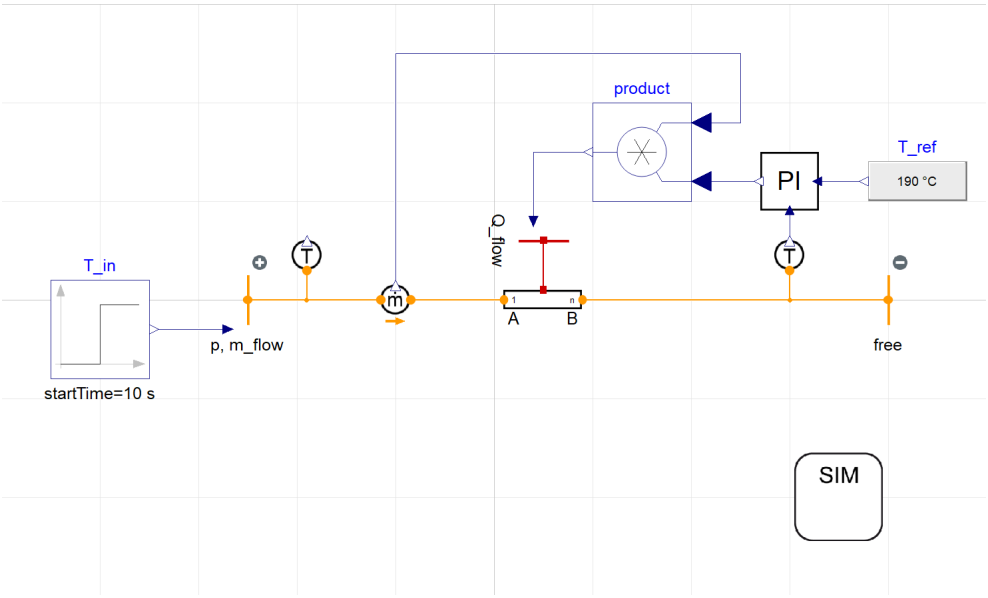


Figure A.9: Model for testing EGB controller. Moist air flows from the overdetermined boundary on the left to the underdetermined boundary on the right. Heat is extracted from the gas via the tube in the middle, the amount of heat is the product of the mass flow and the output from the PI-controller.

Heat is extracted from the gas using a tube connected to a heat boundary. The amount of heat extracted is the product of the mass flow, measured via the sensor to the left of the tube, and the output from the PI-controller.

There are basically two types of behaviour which are interesting to test when it comes to input temperature: Behaviour when the controller goes from off to on at a given temperature and behaviour when controller is on and input temperature changes. For the second case, it is in particular important to check what happens when the input temperature is lowered. Assuring that the controller quickly stabilizes after this is critical as output temperatures under 180 °C in the EGB can lead to corrosion ([2], p. 9). Both of these behaviours where tested in the same simulation. The controller is turned on at $t = 0$ s, it stabilizes quite fast, and after 10 s the step function for the input temperature activated and the controller needs to stabilize once again. The results from the simulation, with the final tuning parameters, is shown in figure A.10.

This is not a thesis on control theory, despite the high usage of controllers. It is, for that reason, of minimal interest in this thesis to study in detail the process for finding correct tuning parameters. The general technique used was trial and error: Test values until the behaviour is ”OK”, then adjust one parameter at a time until something that behaves well is found. For this controller it was important to not have an underdamped system, as temperatures far below the reference temperature can, as mentioned earlier, lead to corrosion problems. The results from the tuning are the following: $K_p = 10^3$ and $T_i = 0.4$ s.

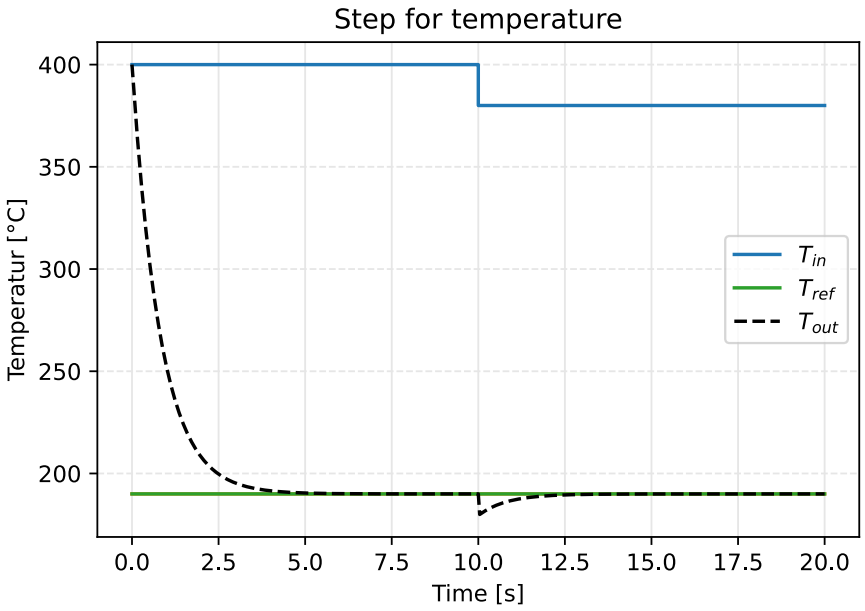


Figure A.10: Results from testing EGB controller with a step function for input temperature. The controller stabilizes quickly both when turned on and when the input temperature changes, without exhibiting underdamped behaviour. This is vital: The output temperature should not fall significantly below the reference temperature of 190 °C as temperatures under 180 °C can lead to corrosion ([2], p. 9).

Another aspect of the controller which is interesting to check is its independence of mass flow. As mentioned further up, the output from the PI-controller is multiplied with the measured mass flow. In theory this should remove almost all dependence on mass flow, but it is always good to test the theory. A model close to the one shown in figure A.9 was created, but the step function for temperature was replaced with a constant value of 400 °C, and a step function was added for mass flow (at the overdetermined boundary). The step function followed the behaviour

$$\dot{m}(t) = 50\,\mathrm{kg/s} - \Theta(t - 10\,\mathrm{s}) \cdot 25\,\mathrm{kg/s}. \tag{A.19}$$

The results from this test can be seen in figure A.11. As one can easily see there is little to no effect on the output temperature when the mass flow changes.

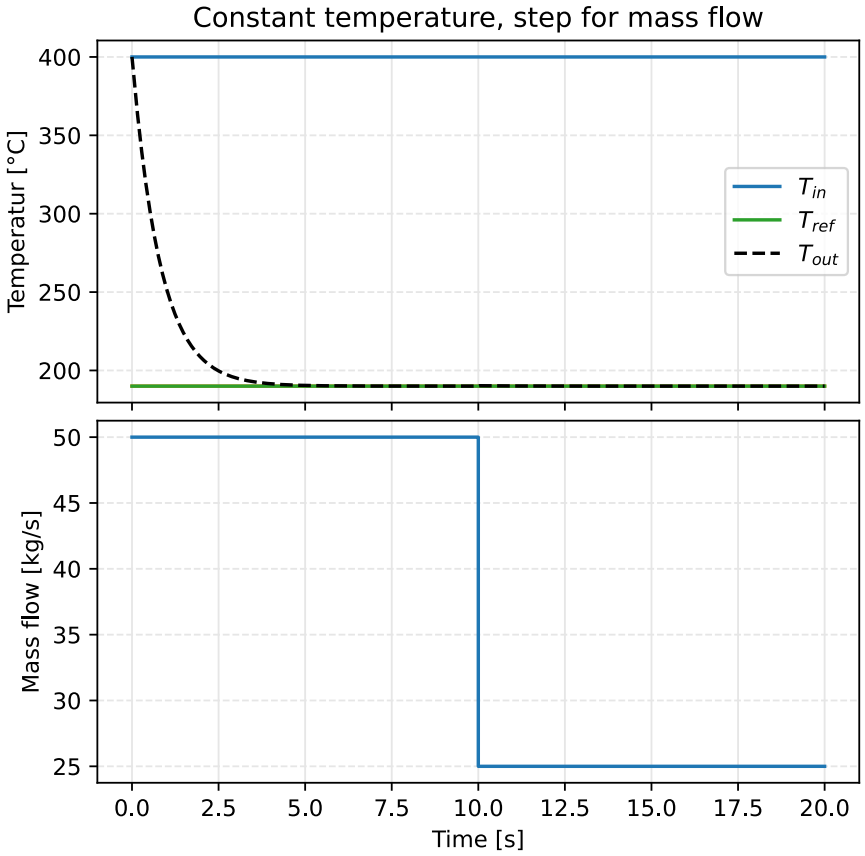


Figure A.11: Results from testing EGB controller with a step function for mass flow. The controller stabilizes the output temperature at the reference in about 5 seconds, and at the 10 s mark the mass flow changes. It is clear from the graph that this change in mass flow has little to no effect on the output temperature. This is expected as the heat extracted is proportional to the mass flow, but it is nevertheless good to assure that it is the case.

A.2.4 Other controllers

Three additional controllers have been used: One for the heat pump used in solution 3 (4.4), one for potable water heating and one for the evaporation in the steam cycle using the heat from the EGBs. The controller used for potable water heating was identical to that of the heat pump, so it will not be mentioned further. Tests for the heat pump controller was done with the model shown in figure A.12.

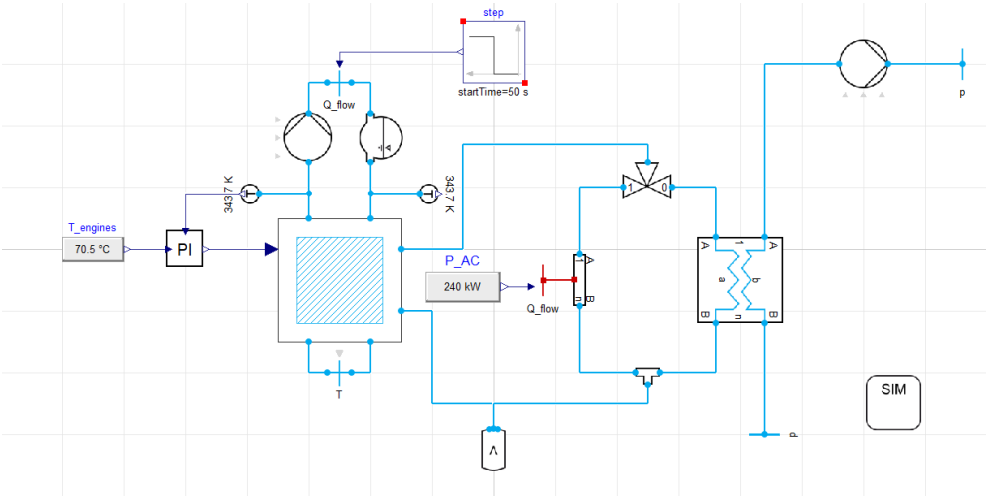


Figure A.12: Testing model for heat pump controller. The model is based on the model used in solution 3, see figure 4.14. The difference from that model is that this one has switched out the power supply model (which has dynamical behaviour based on load profiles) with a controlled heat flow boundary. A step function is then applied to this boundary.

For the heat pump controller it was quickly discovered that a large temperature difference showed up when a step function of typical values was applied. In order to counteract this, inertia was added to the system using a small water tank with volume $1\,\mathrm{m}^3$. This acts as a buffer, giving the controller enough time to stabilize. Results without buffer can be seen in figure A.13, whilst the results with the buffer can be seen in figure A.14. The final tuning parameters of the controller were $K_p = 10^4$ and $T_i = 0.1\,\mathrm{s}$.

The controller for the mass flow through steam evaporation was not put under extensive testing on its own, and will not be discussed in detail here. The tuning parameter for this controller was: $K_p = 1$ & $T_i = 0.5\,\mathrm{s}$.

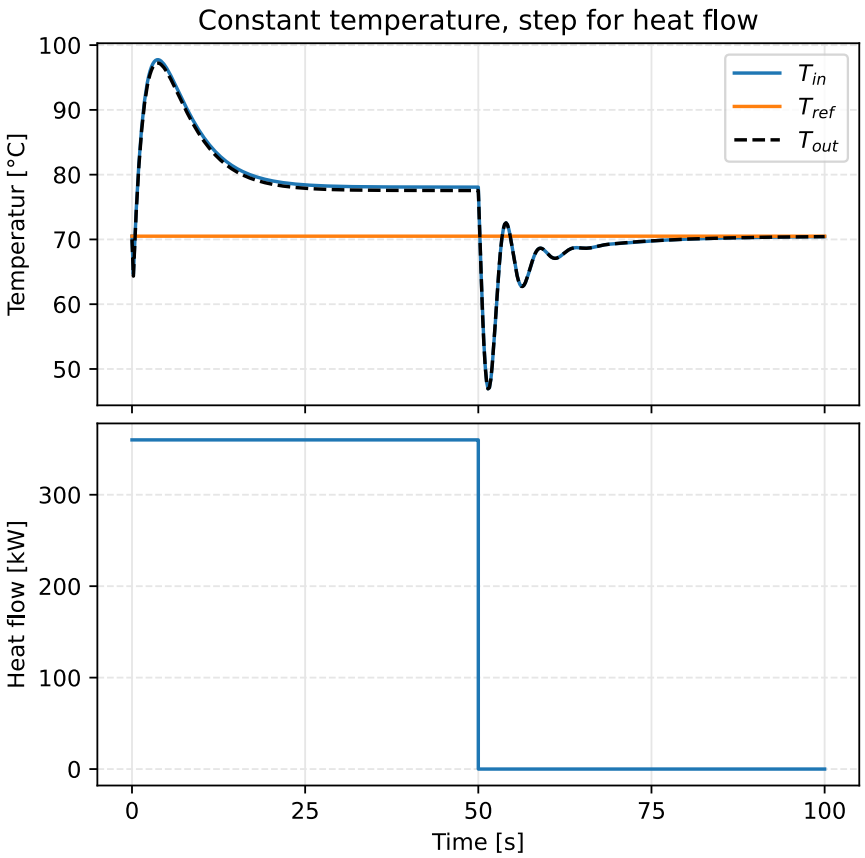


Figure A.13: Results from testing HP controller with a step function for heat flow.

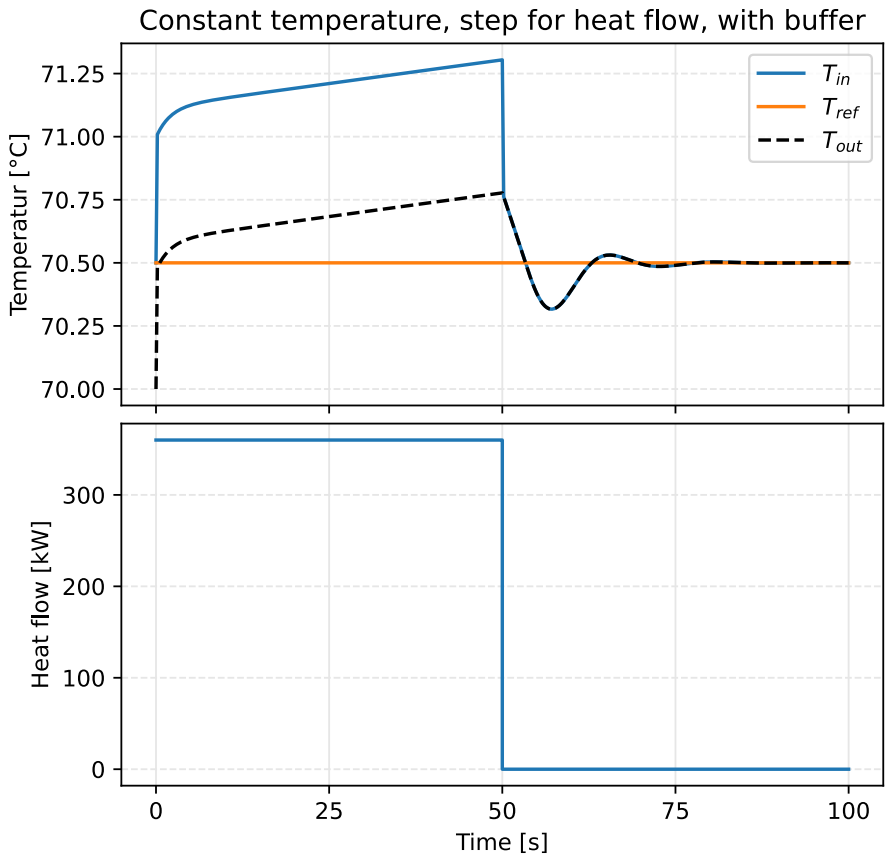


Figure A.14: Results from testing HP controller with a step function for heat flow. Here a buffer volume has been added, to add inertia to the system. The volume of the buffer tank is 1 m^3. This buffer volume gives the controller time to stabilize without major changes in temperature output.

Appendix B

Models

This appendix includes graphical view and code for the models at the core of the simulations used in this thesis. The graphical model for the power supply system will be presented, along with its sub-models: The engine model and the EGB model. After that there will be a discussion on some more minor models: This includes the model for the heat pump, the helper model for the HT recovery system and the steam cycle. Following this will be a section showing the code for the various models, where necessary. Some of the models contained little to no code, these will not be shown.

B.1 Graphical models

This section will feature a thorough description of the engine system, including the models for the DGs and EGBs that are used in the complete model for the power supply. Following this will be the model for the heat pump, the HT recovery system helper model and the steam cycle.

B.1.1 Diesel generator

The Diesel Generator model was implemented to be in as much accordance as possible with the data given in the datasheets [15] and [16]. The most important contents of these are discussed in detail in section C.1. The graphical part of the implemented model can be seen in figure B.1. Related code can be found in section B.2.1.

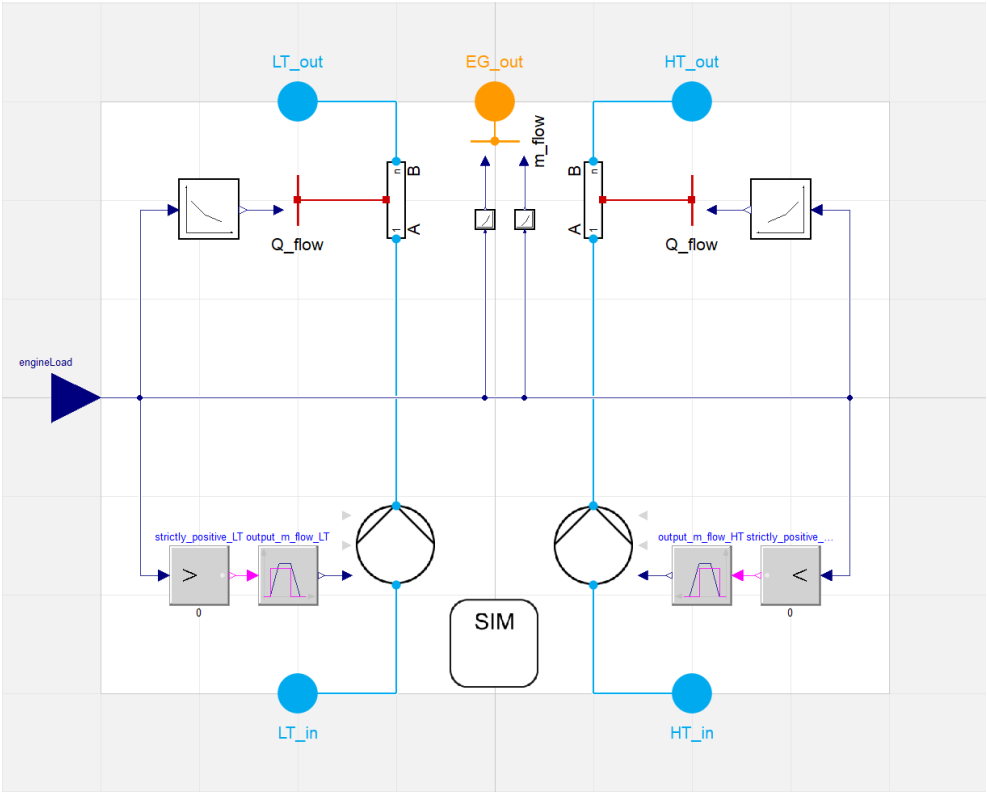


Figure B.1: Graphical model of diesel generator. Low Temperature- and High Temperature water is pumped through the engine and heated by an amount given by the engine load using interpolation of known datapoints. The exhaust gas output mass flow and temperature is also given by the engine load via interpolation.

There are two liquid flows: LT- and HT water, to the left and right in the figure, respectively. Both flows are pumped by a pump that can be seen in the lower half of

the figure, and heated by a tube in the upper half of the figure. The pumps are set to be ideal, meaning they do not contribute to the total energy balance of the system. They pump water at a constant rate when at nominal speed, 30 kg/s and 50 kg/s respectively. They are turned on when the engine load, which comes in to the model via the input on the left, is higher than zero. They take some time to reach nominal speed, their windup time, and similarly they take some time to turn off, their wind down time. In the model these are set to be the same, referred to as the delay time of the pump. The delay time for the two pumps are set to 25 s. This value is referenced in the datasheet for the engine (new version) as the time it typically takes for the Diesel Generator to reach nominal speed ([16], p. ”2-3”)[1]).

The heating of the two water flows, via the tubes, is given by interpolation blocks. These blocks are built using an interpolation function from the Modelica Standard Library, and take in two arrays defining the datasets used for interpolation. The block will not be discussed in further details here, see section B.2.7 for the code. The data for the interpolation blocks, as well as the graphs they are based on, can be found in section C.1. It is important to note that there are only known datapoints for engine loads in the range 50-100%. It is, however, within reason to assume that for an engine load of 0% the heating is equal to zero. It is, of course, wild speculation that the engine behaves linearly in the range 0-50%, but further information is sadly not available.

The exhaust gas output flow can be seen in the top middle of the figure, and is determined by a boundary condition marked ”m_flow”. The values of this boundary condition for mass flow and temperature are given, similar to the heating of the two water flows, by interpolation of known data given an engine load. Once again, the data and interpolation points can be found in section C.1.

B.1.2 Exhaust gas boiler

The model for the Exhaust Gas Boiler, which can be seen in figure B.2 (code is found in section B.2.2), is quite simple in design. It takes in exhaust gas to the left (EG_in), extracts energy from this gas (when active), and outputs the heat extracted in watts to the right (P_out). The heat is extracted using a simple tube in the middle of the gas flow; the heat extracted is controlled by a PI-controller, the tuning of which was explained in section A.2.3. This PI-controller is active so long as the engineLoad, input on the bottom of the model, is above a threshold value. This value is currently set to 0%.

An important thing to note with this model is that the heat extraction, in principle, disregards entropy; the extracted heat is output purely as a number of watts. In reality this heat should be transferred from one fluid, the exhaust gas, to another fluid, the steam in a heat exchange. This necessitates that the exhaust gas has a higher

[1]This datasheet chooses to abstain from the common notation of absolute page numbers, and uses instead the page number within each chapter. So when this citation is referencing the datasheet page ”2-3”, this is the third page of the second chapter. This is not to be confused with pages 2 through 3 nor section 2.3, both of which would be wrong.

temperature than the steam, such that the process leads to an increase in entropy. If this does not hold, e.g. the steam has a higher temperature than the exhaust gas, then the opposite will happen: Heat will flow from steam to exhaust gas. Disregarding the use of the heat, by extracting it and outputting it as a number, is the same as ignoring these fundamental principles of thermodynamics. It is then the job of the user of the model to assure that the use of the heat is realistic, and doesn't break the second law of thermodynamics (see 3.1.1).

This has been checked for the use case here, which can be seen in section B.2.3, and the use of the heat is well within reason. The heat is extracted from the exhaust gas by cooling it from approximately 400 °C to 190 °C. It is used to evaporate water at 8 bar, where the temperature for water is about 170 °C.

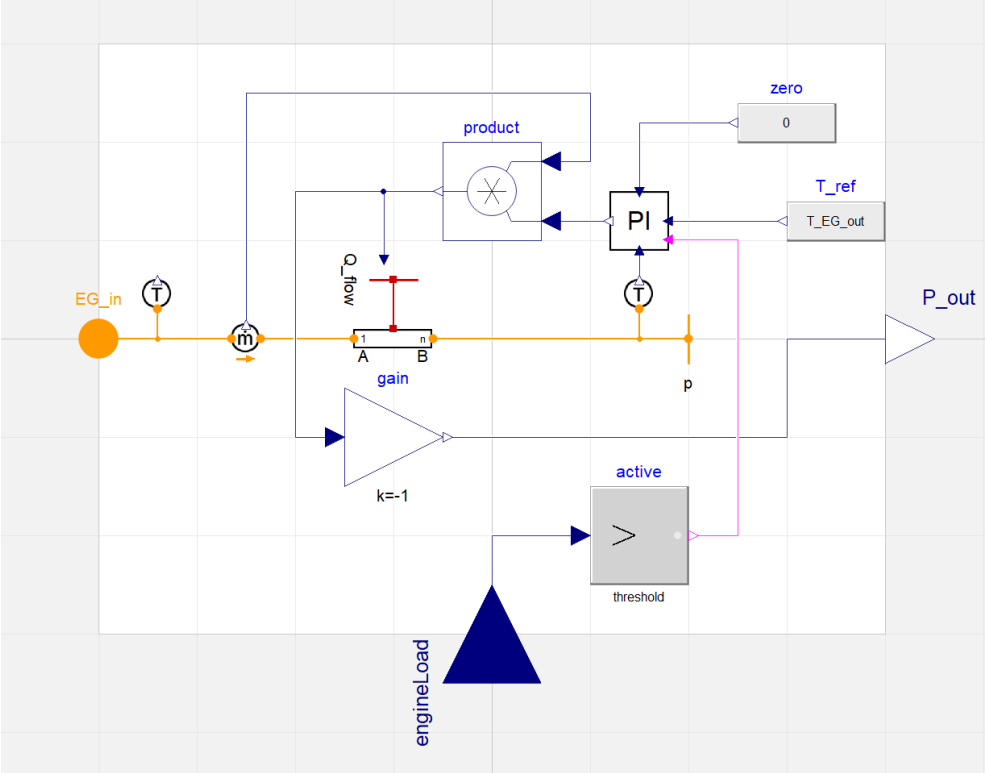


Figure B.2: Graphical model of exhaust gas boiler. Exhaust gas flows in from the left, heat is extracted via the tube in the middle using a PI-controller and the exhaust gas is released. The engine load, input at the bottom, determines if the PI-controller extracting the heat is active or if no heat should be extracted. Heat is only extracted if the engine load is above the given threshold value (currently set to 0%). The extracted heat is output to the right of the model (P_out), in watts.

B.1.3 Power supply

The model for the power supply is shown in figure B.3, the code can be found in section B.2.3. This model is quite complex, and will therefore be discussed in great detail in this section.

The model, in principle, follows much of concepts shown in figure 2.2 in section 2.1. To some extent the model also follows the setup shown in figure 2.2, with one DG ”triplet” on each side in the upper part of the model. The two DG triplets consist of DGs 1, 2 & 3 and DGs 4, 5 & 6, which in the model show up in the right and left corners respectively.

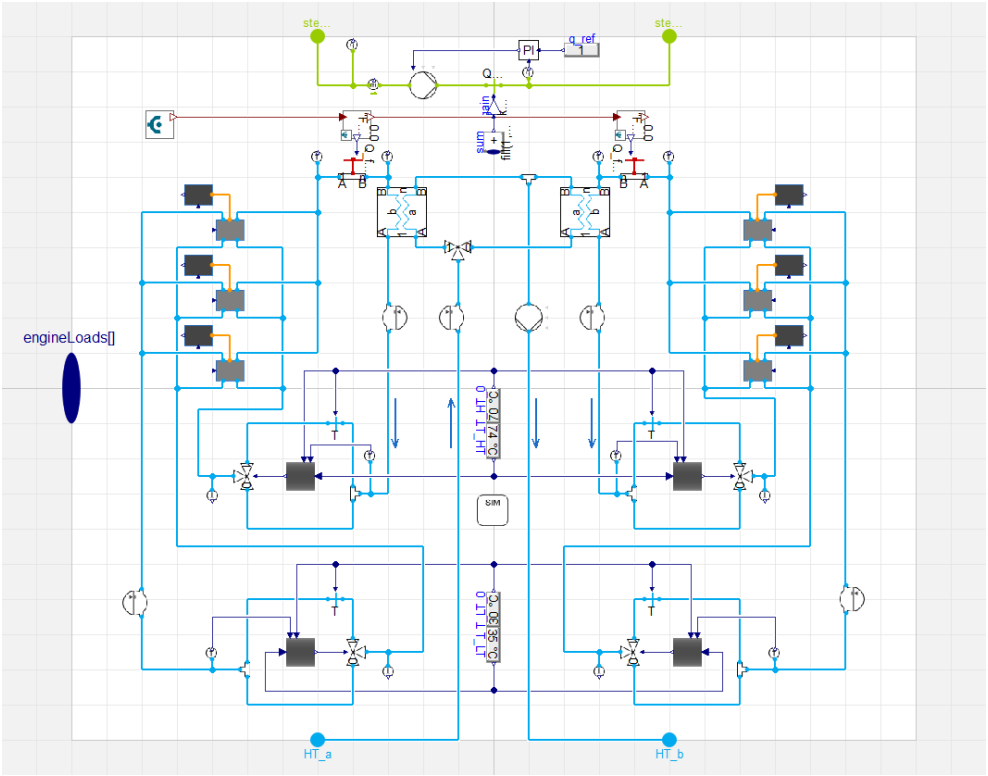


Figure B.3: Model of the power supply. Two triplets of Diesel Generator (DG)s, all connected to an Exhaust Gas Boiler (EGB), can be seen to the upper left and right of the figure. The two triplets are both connected to their own cooling systems, one for LT water and one for HT water. The HT water cooling system also exchanges heat with FWG and the HT recovery system. The HT recovery system can be seen in the middle of the model, shaped almost like a ”T”, except at the bottom it has an input in the left half of the model (HT_a) and an output in the right half (HT_b). Heat extracted from the EGBs are output in the top middle of the model to the steam. The steam is input via the left connector and output via the right connector; a PI-controller controls the mass flows. Values for the engine loads of the six DGs come in as input on the left.

The Diesel Generators, shown as light grey rectangles, are all connected to an Exhaust Gas Boiler, shown as black rectangles. The output from the EGBs is summed in the top of the figure, and as input to the heating of the steam (more on that further down). The HT recovery system can be seen in the middle of the figure, shaped like a ”T”. It has input/output in the lower side of the figure: To the left is the input (HT_a) and to the right is the output (HT_b). Each DG triplet is connected to one HT flow for heat exchange with FWG, the HT recovery system and a cooler. The water flows from the engines, through these three exchanges (in the order they were mentioned), before returning to the engines. The water flow in the DG triplet loop is determined by the number of DGs that are running in that triplet. The water flow of the HT recovery system is on the other hand a constant independent of the number of running DGs.

It is important to note that while figure 2.2 includes the users of the HT recovery system, namely potable water and air conditioning, the power supply model does not. The power supply model has instead an input and output interface for the HT recovery system so the users can be placed outside.

The steam is input to the model via the upper left VLE fluid port, and output to the upper right. The steam section within this model is quite simple. As discussed further up the energy extracted from the exhaust gas is summed in the block in the upper middle, marked by a plus-sign. This is used as input to a heat flow boundary in the steam. In addition to this, there is a pump (slightly to the left) and a PI-controller with a sensor (slightly to the right) controlling this pump. The controller was briefly discussed in section A.2.4.

B.1.4 Heat pump

The model for the heat pump, which can be seen in figure B.4, was modelled after an ideal heat pump. See section 3.1.4 for the theory behind ideal heat pumps. The power to the heat pump is input via a connector to the left, or given by a fixed value used in the block to the right labelled ”power_default”. The two liquid flows can be seen in the bottom and top of the model, where they are heated via tubes. The heating is determined by the power and CoP in accordance with the theory on ideal heat pumps (again, see section 3.1.4). The code for this model can be found in section B.2.4.

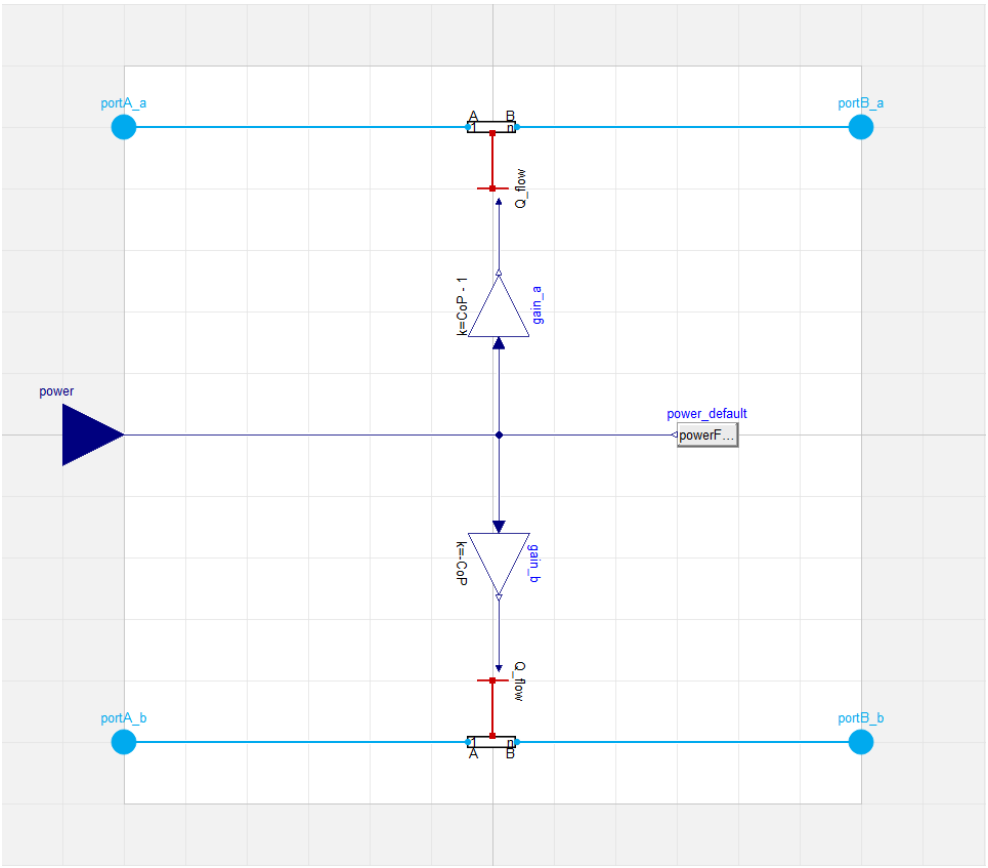


Figure B.4: Model for the heat pump. The two liquid flows, flow a and b, can be seen flowing from left to right in the top and bottom of the model, respectively. The power is input to the left, or alternatively using a fixed value to the right. The heat extracted from the two tubes is calculated using the Coefficient of Performance (CoP). See B.2.4 for the related code.

B.1.5 HT recovery system helper model

The HT recovery system helper model was created in order to test various solutions for meeting HT water demand in port. In chapter 4 it was used to run simulations for all three solutions for HT water (4.2, 4.3 & 4.4), but during testing it was also used to test many of the solutions that did not prove sufficient (e.g. 5.2.1). Figure B.5 shows the graphical view of the model.

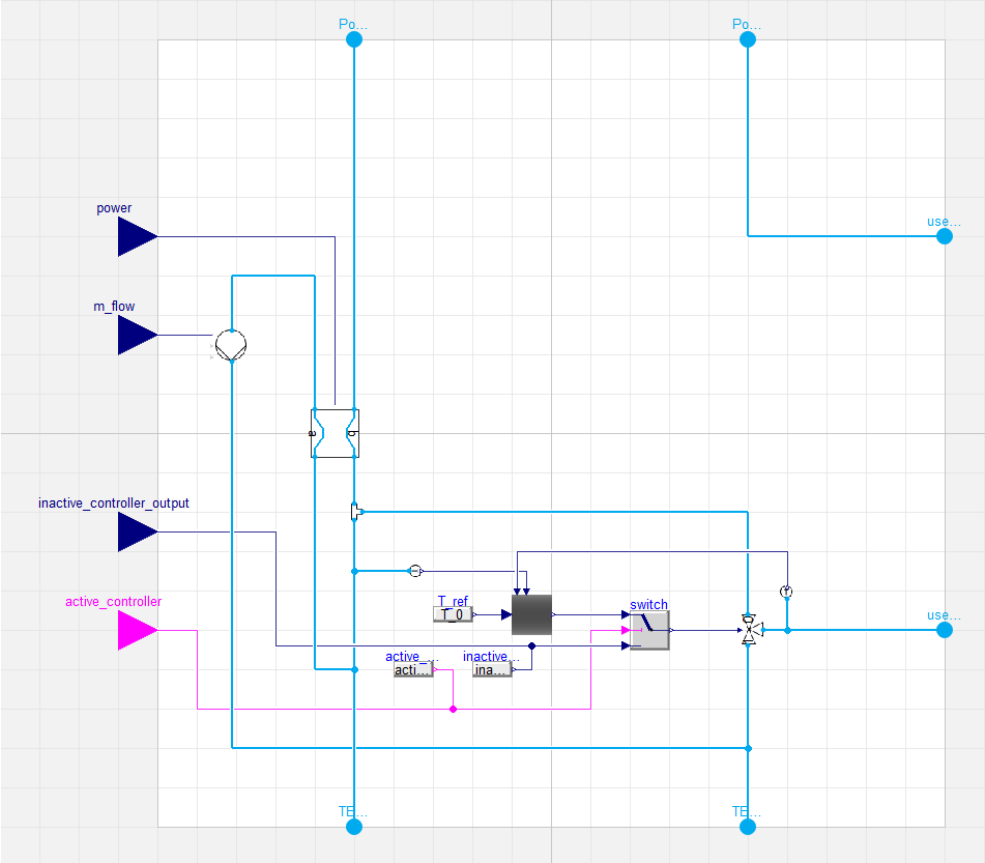


Figure B.5: This figure shows the helper model for the HT recovery system. In the top is the input/output to the power supply model, to the right is the input/output to the users and in the bottom is the input/output to the TES (if activated). The output to the TES is controlled by the ideal valve controller, black box in the lower middle of the model. See section A.2.1 for more info on this. At the left border of the model is input from users, which can be activated/deactivated by the user using the GUI of the model (see section B.2.5 for the code related to this). These include values related to the controller and heat pump, the last of which can be seen slightly to the left in the middle of the figure.

In the top of the figure one can see the connectors to the power supply model, output to the left and input on the right. The input flows, as one can see in the top right corner,

directly to the users on the right hand side. To the lower right the water returns from the users, and is split by a controlled valve (see section A.2.1 for the implementation of this controller). The controller receives user input from the left on whether or not to be active, and potentially what the output to the valve is when the controller is inactive. The reference temperature to the controller is set to be ”T_0”, which is set by the user in the GUI of the model. In the bottom of the model there is input and output to the TES. The heat pump, along with input to this, can be seen in the upper left section of the model, along with a pump for mass flow through the lower temperature flow in the heat pump. All sections of this model is further explained in the code for the model, which can be seen in B.2.5. This code is quite extensive.

B.1.6 Steam cycle

The model for the steam cycle, which can be seen in figure B.6, has taken a significant amount of the time spent on this master thesis. And still, it is the most incomplete of the models. As was discussed in section 4.5 the current model leads to many bad points during simulations. This is in large part caused by problems with stabilizing mass flows after changes in the heat needed/available. To assure that all mass flows are correct there are several logic blocks in the middle of the figure doing calculations, but this grants a less robust simulation than if these mass flows were found implicit by the dynamics of the system.

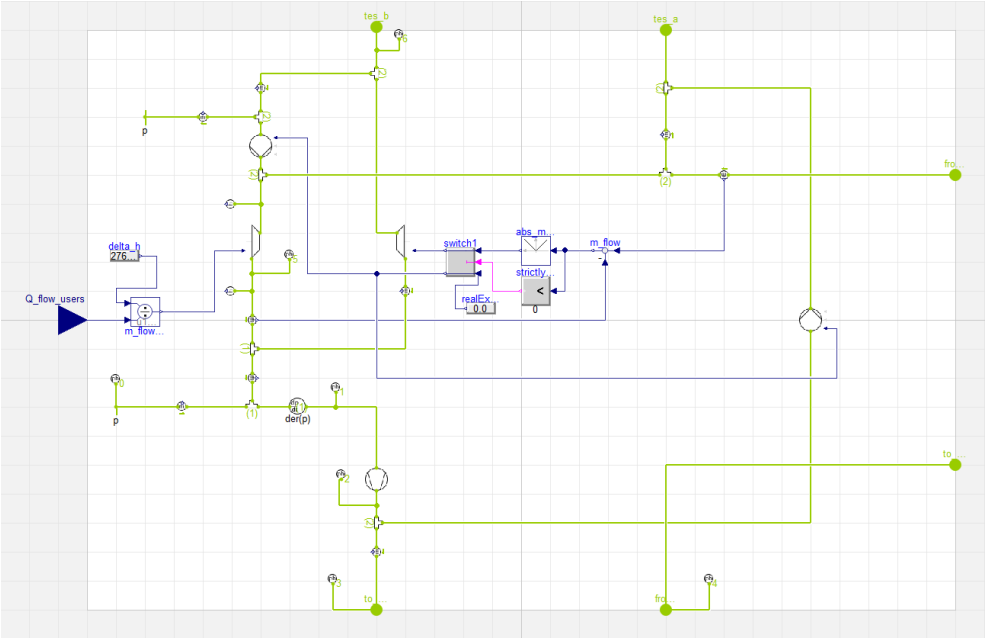


Figure B.6: Figure showing the graphical model for the steam cycle. The connectors at the bottom of the figure connect the steam cycle to the evaporation in the power supply model. The connectors to the right connect to the separator (which seperates the multiphase fluid into pure vapour) and pressure state (which defines the pressure in many of the blocks at the high pressure). The connectors at the top connect to heating in some form, in the simulations run this was a boundary condition representing an electric boiler, but this was planned to function with a TES. To the left is the input for the user need, and in the middle is logic blocks ensuring the mass flows through the various pipes are correct. To the upper and lower left there are two boundaries, the upper is used for stabilizing and the lower is used as a water source.

B.2 Code

This section covers the Modelica code of all models made for the models used in this thesis. In code listings, note that ”@” denotes an annotation, hidden for convnience. These usually include information about placement in graphical interface and information about placement of parameters in GUI. The notation ”{...}” has been chosen to indicate the "collapse" of code. If nothing is specified this code is equivalent to content from the graphical interface including the GUI of the various blocks/models used in the graphical interface.

B.2.1 Diesel Generator model

Code listing B.1: Code for the Diesel Generator (DG) model. This defines the parameters available to the user of the model. This consists of the nominal mass flow, minimum mass flow and delay time for the pumps of the two water flows: LT- and HT water.

```
model DG
  import SI = Modelica.SIunits;

  parameter SI.MassFlowRate min_m_flow_LT = 0.0 "
     Minimum mass flow of LT water" @;
  parameter SI.MassFlowRate min_m_flow_HT = 0.0 "
     Minimum mass flow of HT water" @;
  parameter SI.MassFlowRate min_m_flow_EG = 0.0 "
     Minimum mass flow of exhaust gas" @;

  parameter SI.Time delay_pump_LT = 0 "Delay time for
     LT pumps" @;
  parameter SI.Time delay_pump_HT = 0 "Delay time for
     HT pumps" @;

  parameter SI.MassFlowRate m_flow_LT = 50 "Mass flow
     of LT water" @;
  parameter SI.MassFlowRate m_flow_HT = 75 "Mass flow
     of HT water" @;

  {...}
equation
  {...}

  @
end DG;
```

B.2.2 Exhaust Gas Boiler model

Code listing B.2: Code for the Exhaust Gas Boiler (EGB) model. ”T_EG_out” is the reference temperature for the PI-controller of the model, the value it aims at when extracting heat from the exhaust gas. The variable ”threshold” is the value for which the EGB turns off. The parameter ”gasType” is a variable which is used as the gas type input for all gas blocks in the model, which makes it possible for the user to choose the gas type used in the model.

```
model EGB
  "Exhaust Gas Boiler - takes in exhaust gas from
     diesel generator and outputs heat extracted"
  parameter Modelica.SIunits.Temperature T_EG_out "
     Output temperature of EG";
  parameter Real threshold = 0 "Engine load for which
     boiler turns off";

  inner parameter TILMedia.GasTypes.BaseGas gasType "
     Gas type" @;

  {...}
equation
  {...}

  @
end EGB;
```

B.2.3 Power supply model

Code listing B.3: Code for the power supply model. The variables in this code is just a handy way of setting the parameter values for all 6 DGs and EGBs from the power supply GUI. The code below equation is the connectors that are hidden from the graphical view of the model. The first two segments are the connectors between the engine load input to the power supply and their corresponding input to the DGs/EGBs. The segment below is the connectors connecting the output from the EGBs to the sum outputting the total heat extracted from the EGBs in the power supply.

```
model PowerSupply "Generates electricity, which in
   turn creates exhaust gas and waste water. Exhaust
   gas can create steam."
  import CruiZE;

  import SI = Modelica.SIunits;

  parameter Boolean include_steam = false;

  parameter SI.MassFlowRate min_m_flow_LT "Minimum
     mass flow of LT water from the 6 DGs" @;
  parameter SI.MassFlowRate min_m_flow_HT "Minimum
     mass flow of HT water from the 6 DGs" @;
  parameter SI.MassFlowRate min_m_flow_EG "Minimum
     mass flow of exhaust gas from the 6 DGs" @;

  parameter SI.Time delay_pump_LT "Delay time for LT
     water pump of the 6 DGs" @;
  parameter SI.Time delay_pump_HT "Delay time for HT
     water pump of the 6 DGs" @;

  parameter SI.Temperature T_EG_out "Output
     temperature from EGBs" @;
  parameter Real threshold_engine_load "Engine load
     for which EGBs are turned off" @;

  parameter SI.MassFlowRate m_flow_steam_init = 0.005
     "Initial mass flow for steam" @;

  {...}
equation
  connect(engineLoads[1], DG1.engineLoad);
  connect(engineLoads[2], DG2.engineLoad);
  connect(engineLoads[3], DG3.engineLoad);
  connect(engineLoads[4], DG4.engineLoad);
  connect(engineLoads[5], DG5.engineLoad);
```

```
  connect(engineLoads[6], DG6.engineLoad);

  connect(engineLoads[1], EGB1.engineLoad);
  connect(engineLoads[2], EGB2.engineLoad);
  connect(engineLoads[3], EGB3.engineLoad);
  connect(engineLoads[4], EGB4.engineLoad);
  connect(engineLoads[5], EGB5.engineLoad);
  connect(engineLoads[6], EGB6.engineLoad);

  connect(EGB1.P_out, sum.u[1]);
  connect(EGB2.P_out, sum.u[2]);
  connect(EGB3.P_out, sum.u[3]);
  connect(EGB4.P_out, sum.u[4]);
  connect(EGB5.P_out, sum.u[5]);
  connect(EGB6.P_out, sum.u[6]);

  {...}

  if include_steam then
    connect(sensor_m_flow_highP.portB,pump_steam.
       portA) @;
    connect(pump_steam.portB,bc_egb. portA) @;
    connect(sensor_q_post_heat.port,bc_egb. portB) @;
    connect(sensor_q_post_heat.sensorValue,controller.
        u_m) @;
    connect(controller.u_s,q_ref. y) @;
    connect(controller.y,pump_steam. m_flow_in) @;
    connect(sensor_q_pre_heat.port,
       sensor_m_flow_highP.portA) @;
    connect(sensor_m_flow_highP.portA, steam_a) @;
    connect(sum.y, gain.u) @;
    connect(bc_egb.heatFlowRateInput, gain.y) @;
    connect(bc_egb.portB, steam_b) @;
  end if;

  @
end PowerSupply;
```

B.2.4 Heat pump

Code listing B.4: Code for the liquid to liquid heat pump, which is modelled after an ideal heat pump. The power to the heat pump can either be provided by an input, if ”use_powerInput” is set to true, or by a fixed value ”powerFixed”. The Coefficient of Performance (CoP) is given by ”CoP”, and must in this model be a constant. There are also parameters for initializing the temperature in the two liquid flows, as well as defining the liquid types (e.g. water).

```
model LiquidLiquidHP "A simple liquid to liquid heat
    pump"
  import Modelica.Units.SI;

  parameter Boolean use_powerInput = true "= true, if
     power defined by input" @;
  parameter SI.Power powerFixed = 0 "Fixed value for
     heat pump power" @;

  parameter Real CoP "Coefficient of performance" @;

  parameter SI.Temperature T_init_a = 298.15 "Initial
     temperature of liquid a" @;
  parameter SI.Temperature T_init_b = 298.15 "Initial
     temperature of liquid b" @;

  parameter TILMedia.LiquidTypes.BaseLiquid
     liquidType_a "Liquid type a" @;
  parameter TILMedia.LiquidTypes.BaseLiquid
     liquidType_b "Liquid type b" @;

  {...}
equation
  {...}

  if use_powerInput then
    connect(power, gain_a.u) @;
    connect(power, gain_b.u) @;
  else
    connect(power_default.y, gain_a.u) @;
    connect(power_default.y, gain_b.u) @;
  end if;

  {...}
  @
end LiquidLiquidHP;
```

B.2.5 HT recovery system helper model

Code listing B.5: Code for the helper model for the HT recovery system. This model is quite complex, but is based on a simple principle: The user should be able to control whether or not the HT recovery system contains a TES and/or heat pump. Currently, there is not an option to have a heat pump without a TES, due to time constraints this was not finished. A workaround has been used for solution 3 where ”include_TES” is set to true, but using a boundary condition where one would normally put the TES. The system also contains many parameters for controlling the behaviour of these two, such as the power input to the heat pump or the controller behaviour for the TES. There is also significant logic to ensure the model is dynamic to the user input.

```
model HTRecoverySystem "Flexible model for HT recovery
    system of a cruise ship"
  import Modelica.Units.SI;

  // General
  parameter Boolean include_TES = false "Include ports
      for thermal energy storage" @;
  parameter Boolean include_HP = false "Include heat
     pump for TES cycle" @;

  parameter SI.Temperature T_0 = 298.15 "Lower bound
     for output temperature" @;


  // SIM
  parameter TILMedia.LiquidTypes.BaseLiquid liquidType
      "Liquid type" @;


  // Thermal energy storage
  parameter Boolean use_activeControllerInput = false
     "= true, if active_controller is defined by input
     " @;
  parameter Boolean active_controllerFixed = true "=
     true, if controller is active" @;
  parameter Boolean use_inactiveControllerOutputInput
     = false "= true, if inactive_controller_output is
      defined by input" @;
  parameter Real inactive_controller_outputFixed = 0 "
     output from controller when inactive" @;


  // Heat pump
```

```
  parameter Real CoP = 2.0 "Coefficient of Performance
      for heat pump" annotation(Dialog(enable=
     include_HP, tab = "HP", group="Heat pump
     functionality"));

  parameter Boolean use_powerInput = false "= true, if
      m_flow defined by input" annotation(Dialog(
     enable=include_HP, tab = "HP", group="Power"));
  parameter SI.Power powerFixed(displayUnit="kJ") =
     100e3 "Fixed value for heat pump power"
     annotation(Dialog(enable=(not use_powerInput and
     include_HP), tab = "HP", group="Power"));

  parameter Boolean use_massFlowRateInput = false "=
     true, if m_flow defined by input" annotation(
     Dialog(enable=include_HP, tab = "HP", group="Mass
      Flow Rate"));
  parameter SI.MassFlowRate m_flowFixed = 10 "Fixed
     value for pump mass flow rate" annotation(Dialog(
     enable=(not use_massFlowRateInput and include_HP)
     , tab = "HP", group="Mass Flow Rate"));


  // Initialization
  parameter SI.Temperature T_init = 298.15 "Initial
     temperature" @;


  // Conncectors
  TIL.Connectors.LiquidPort PowerSupply_a(liquidType=
     liquidType) @;
  TIL.Connectors.LiquidPort PowerSupply_b(liquidType=
     liquidType) @;

  TIL.Connectors.LiquidPort users_a(liquidType=
     liquidType) @;
  TIL.Connectors.LiquidPort users_b(liquidType=
     liquidType) @;

  TIL.Connectors.LiquidPort TES_a(liquidType=
     liquidType) if include_TES @;
  TIL.Connectors.LiquidPort TES_b(liquidType=
     liquidType) if include_TES @;
```

```
Modelica.Blocks.Interfaces.BooleanInput
   active_controller if include_TES and
   use_activeControllerInput @;
Modelica.Blocks.Interfaces.RealInput
   inactive_controller_output if include_TES and
   use_inactiveControllerOutputInput @;
Modelica.Blocks.Interfaces.RealInput power if
   include_HP and use_powerInput @;
Modelica.Blocks.Interfaces.RealInput m_flow if
   include_HP and use_massFlowRateInput @;

TIL.LiquidComponents.Valves.DirectionalControlValve
   valve_TES(
  liquidType=liquidType,
  use_switchingPositionInput=true,
  switchingPositionFixed=0,
  m_flowStart=1e-6,
  TStart=T_init,
  effectiveFlowArea0=0.01) if include_TES @;

TIL.LiquidComponents.JunctionElements.VolumeJunction
    junction(liquidType=liquidType, TInitial=T_init)
    if include_TES @;
TIL.LiquidComponents.Pumps.SimplePump pump(
   liquidType=liquidType,presetVariableType="m_flow"
   ,use_massFlowRateInput=
   use_massFlowRateInput,m_flowFixed=m_flowFixed,eta
   =1,etaDrive=1) if include_HP @;
Blocks.HeatPumps.LiquidLiquidHP HP(use_powerInput=
   use_powerInput,powerFixed=powerFixed,CoP=
   CoP,T_init_a=T_init,T_init_b=T_init,liquidType_a=
   liquidType, liquidType_b=liquidType) if
   include_HP @;

Blocks.ideal_valve_controller controller if
   include_TES @;
Blocks.Temperature T_ref(T(displayUnit="degC") = T_0
   ) if include_TES @;
TIL.LiquidComponents.Sensors.Sensor_T sensor_T_0(
   sensorValue(unit="K")) if include_TES @;
TIL.LiquidComponents.Sensors.Sensor_T sensor_T_in(
   sensorValue(unit="K")) if include_TES @;
```

```
  Modelica.Blocks.Logical.Switch switch if include_TES
      @;
  Modelica.Blocks.Sources.BooleanExpression
     active_controller_default(y=
     active_controllerFixed) if include_TES and not
     use_activeControllerInput @;
  Modelica.Blocks.Sources.RealExpression
     inactive_controller_output_default(y=
     inactive_controller_outputFixed) if include_TES
     and not use_inactiveControllerOutputInput @;
equation
  connect(PowerSupply_b, users_a) @;

  // Connect thermal energy storage
  if include_TES then
    // Liquid connectors
    connect(users_b, valve_TES.portA) @;
    connect(valve_TES.portB, junction.portA) @;
    connect(valve_TES.portC, TES_a) @;
    connect(TES_b, junction.portC) @;
    if not include_HP then
      connect(junction.portB, PowerSupply_a);
    end if;


    // Controller input
    connect(T_ref.T, controller.T_ref) @;
    connect(sensor_T_0.port, junction.portC) @;
    connect(sensor_T_0.sensorValue, controller.T_0) @;
    connect(sensor_T_in.port, valve_TES.portA) @;
    connect(sensor_T_in.sensorValue, controller.T_in)
       @;


    // Controller output
    connect(controller.u, switch.u1) @;
    connect(switch.y, valve_TES.switchingPosition_in)
       @;
    if use_activeControllerInput then
      connect(active_controller, switch.u2) @;
    else
      connect(active_controller_default.y, switch.u2)
         @;
    end if;
```

```
    if use_inactiveControllerOutputInput then
      connect(inactive_controller_output, switch.u3) @
          ;
    else
      connect(inactive_controller_output_default.y,
          switch.u3) @;
    end if;

  end if;


  // Connect heat pump
  if include_HP then
    connect(HP.portA_b, PowerSupply_a) @;
    connect(HP.portA_a, pump.portA) @;
    if not include_TES then
      connect(users_b, HP.portB_b);
      connect(pump.portB, HP.portB_a);
    end if;

    if use_powerInput then
      connect(power, HP.power) @;
    end if;

    if use_massFlowRateInput then
      connect(m_flow, pump.m_flow_in) @;
    end if;

  end if;

  if include_TES and include_HP then
    connect(pump.portB, TES_a) @;
    connect(TES_b, HP.portB_a) @;
    connect(HP.portB_b, junction.portB) @;
  end if;

  if not include_TES and not include_HP then
    connect(users_b, PowerSupply_a);
  end if;

  @
end HTRecoverySystem;
```

B.2.6 Steam cycle

Code listing B.6: Code for the steam cycle. This mostly defines the two pressures, ”p_atm” and ”p_high”, and the initial mass flow rate. There is also a parameter for VLE fluid type.

```
model SteamCycle "Model of steam cycle with
   implementation for various forms of heating (
   including TES)"
  import Modelica.Units.SI;

  constant SI.Pressure p_atm = 1.013e5 "Atmospheric
     pressure = 1.013 bar";
  parameter SI.Pressure p_high = 8e5 "Pressure after
     compression";

  parameter SI.MassFlowRate m_flow_init = 1;

  inner parameter TILMedia.VLEFluidTypes.BaseVLEFluid
     vleFluidType "VLE fluid type" @;

  {...}
equation
  {...}

  @
end SteamCycle;
```

B.2.7 Additional code

Code listing B.7: Code for ideal valve controller. The controller takes in two measurement temperature, T_{in} & T_0, and a reference temperature T_{ref}. The controller checks if the denominator is different from zero, which is the same as checking if T_0 is equal to T_{in}. This has a somewhat unconventional syntax: The "<>" operator, which basically means "(larger than or equal) and (smaller than or equal)". If the denominator is different from zero the output is calculated using the formula $u_{temp} = (T_{ref} - T_{in})/(T_0 - T_{in})$, if not then u_{temp} is set to zero. The output u is then equal to the value of u_{temp} bounded between zero and one.

```
model ideal_valve_controller
  Real u_temp(unit="1");

  Modelica.Blocks.Interfaces.RealInput T_in(unit="K")
     @;
  Modelica.Blocks.Interfaces.RealInput T_0(unit="K") @
     ;
  Modelica.Blocks.Interfaces.RealInput T_ref(unit="K")
      @;
  Modelica.Blocks.Interfaces.RealOutput u(unit="1") @;
equation
  if noEvent(T_0 <> T_in) then
    u_temp = (T_ref - T_in)/(T_0 - T_in);
  else
    u_temp = 0;
  end if;

  u = min(max(u_temp,0),1);

  @
end ideal_valve_controller;
```

Code listing B.8: Code for interpolation block. The two arrays ”u_arr” and ”x_arr” represent the dataset used for the interpolation, whereas ”u” is the input to the interpolation block and ”x” is the output from the interpolation function. The two parameters ”gain” and ”offset” are used to give the output ”y”, given by the formula $y = \text{gain} \cdot x + \text{offset}$. This makes it easy to give the dataset in a something else than the base SI-units, and still keep the output a SI-unit. For example if the dataset has ”x_arr” given in degrees Celsius then an offset of 273.15 gives an output in Kelvin.

```
block Interpolate
  "Takes input and gives output based on interpolation
      of given dataset"
  parameter Real u_arr[:] "Array of values
     corresponding to input" @;
  parameter Real x_arr[:] "Array of values
     corresponding to output" @;
  Real x = Modelica.Math.Vectors.interpolate(u_arr,
     x_arr, u);

  parameter Real gain = 1 @;
  parameter Real offset = 0 @;

  Modelica.Blocks.Interfaces.RealInput u @;
  Modelica.Blocks.Interfaces.RealOutput y = gain * x +
      offset @;

  @;
end Interpolate;
```

Code listing B.9: Code for a conversion block from a set of integers to a set of boolean values, used to convert enumerated state to a boolean. The code takes in the boolean values corresponding to the inputs as a parameter. During the simulation it checks that the input is within the defined bounds, and maps to the output.

```
block PositiveIntegerToBoolean "Assigns n boolean
   values to the n first positive integers"
  parameter Boolean output_values[:] = false;

  {...}
protected
  Integer length = size(output_values, 1);
equation
  if 0 < u and u <= length then
    y = output_values[u];
  else
    y = false;
  end if;

  @
end PositiveIntegerToBoolean;
```

Code listing B.10: Code for a conversion block from a set of integers to a set of real values, used to convert enumerated state to a real. The code takes in the real values corresponding to the inputs as a parameter. During the simulation it checks that the input is within the defined bounds, and maps to the output.

```
block PositiveIntegerToReal "Assigns n real values to
   the n first positive integers"
  parameter Real output_values[:] = 0;

  {...}
protected
  Integer length = size(output_values, 1);
equation
  if 0 < u and u <= length then
    y = output_values[u];
  else
    y = 0;
  end if;

  @
end PositiveIntegerToReal;
```

Appendix C

Additional data

This appendix contains additional data needed for the engine model. The engine used by the case ship, Wärtsilä 12V46C, will be explained in detail in the first section. This includes the raw data as well as the datapoints chosen for the model.

C.1 Engine - Wärtsilä 12V46C

This section will cover the Diesel Generator (DG) used in the case ship: Wärtsilä 12V46C. The section is based on info taken in combination from sources [15] and [16]. The first being a project guide describing how to implement the engine Wärtsilä 46 in marine application, which in this version is the Wärtsilä 46C engine. The second is the product guide for Wärtsilä 46F, notice the ”F” as this is not the product guide for the engine used by the cruise ship, but a later model. Source [16] will for that reason only be used as a supplementary source when [15] lacks the necessary information. Both guides include the 12V-version, which is the one used by the case ship.

In order for the engine to function properly the engine and its subsystems must be kept at the right temperatures. This is done by utilizing two cooling water flows: LT- and HT water. The LT water must be kept at a temperature between 25 °C and 38 °C at input, similarly the HT water should be kept at a temperature of about 74 °C ([16] p. ”3-15”. In order to accurately model the engine, and the subsequent systems built on it, it is important to know about the heat dissipated to these water flows by the engine. These values are related to the engine load. These relationships can be seen, for engine loads in the range 50-100%, in figure C.2 and C.1 respectively ([15] p. 38-39). The pressure difference for the two water flows in the engine will be ignored in this model, as they are ignored accross the board. It is, wrongly, assumed that all water flows remain at 1 atm at all times. This is done for simplicity.

Another aspect of the engine which is important for the modelling of these systems is the exhaust gas. For the exhaust gas there are two values that depend on the engine load: The mass flow and temperature. The relationships for these values can be seen in figure C.3 and C.4, respectively ([15] p. 38 & 41). In addition it is assumed that the exhaust gas exits the engine at 1 atm.

The interpolation data, which can be seen in table C.1, C.2, C.3 and C.4, is the data that is actually used in the model as interpolation points for calculation of these values. These points have been taken from figure C.1, C.2, C.3 and C.4. In the first three of these an additional point is added, the zero point. These are added as it is assumed that both LT- & HT-heating, as well as exhaust gas mass flow, will be zero for zero load. However, no assumption like this can be made for the exhaust gas output temperature.

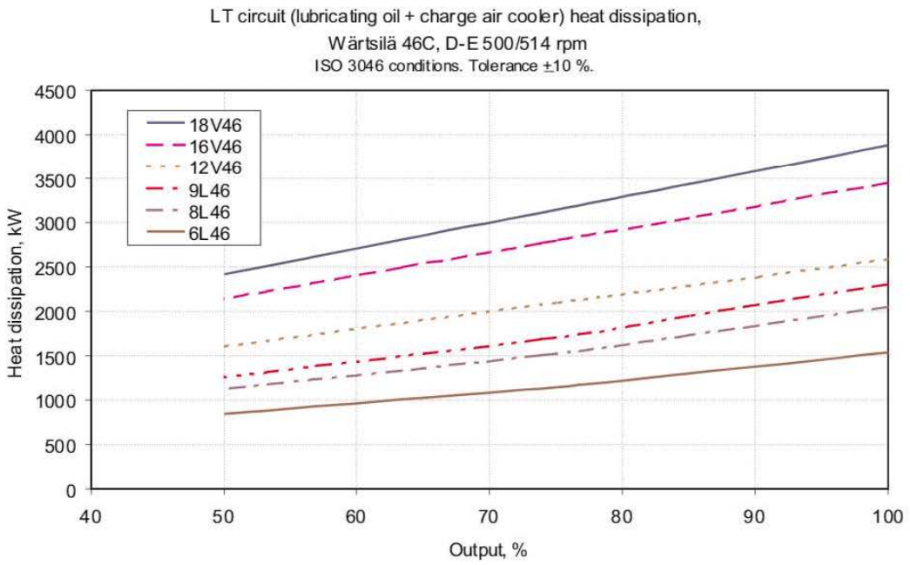


Figure C.1: Correspondence between engine load in percentage and heat dissipation in LT water for the engine Wärtsilä 12V46C, the engine used in the case.

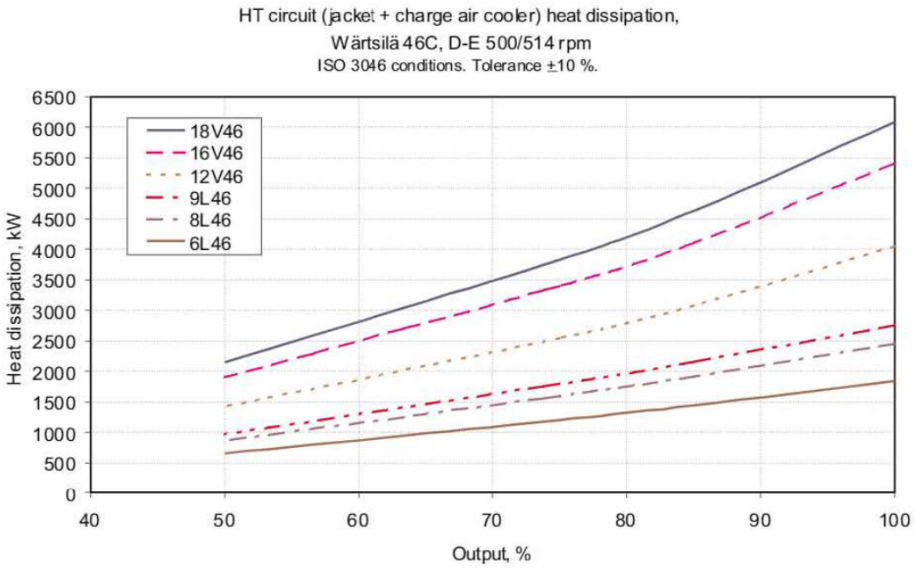


Figure C.2: Correspondence between engine load in percentage and heat dissipation for HT water in the engine Wärtsilä 12V46C, the engine used in the case.

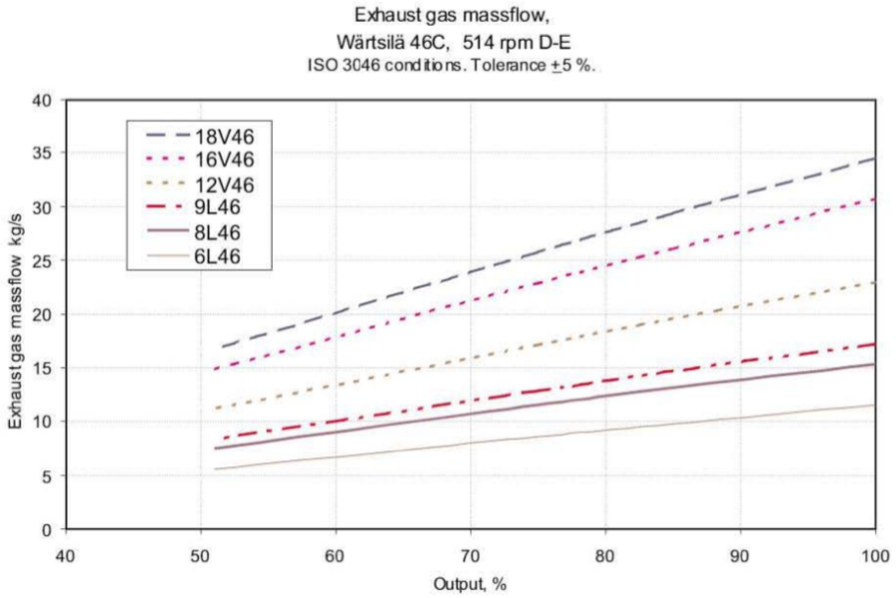


Figure C.3: Correspondence between engine load in percentage and the mass flow of exhaust gas for the engine Wärtsilä 12V46C, the engine used in the case.

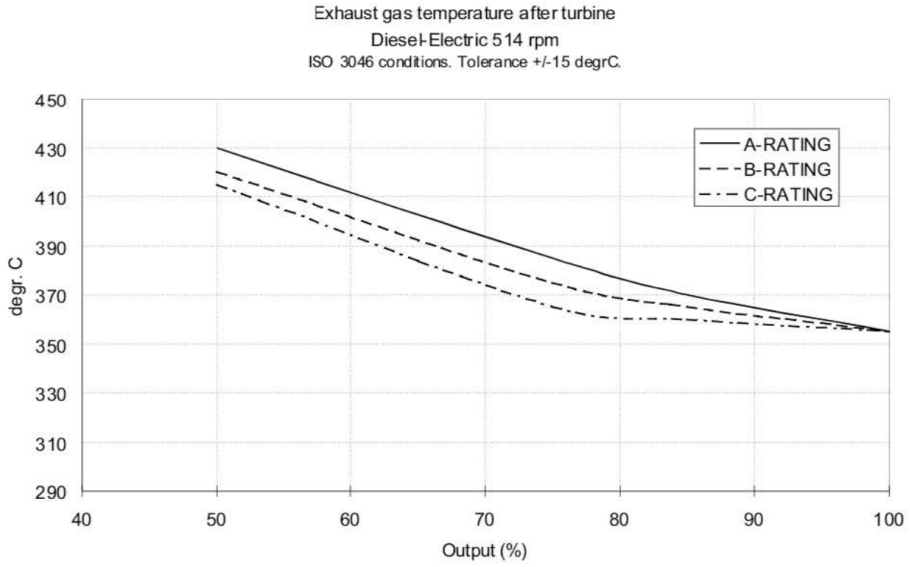


Figure C.4: Correspondence between engine load in percentage and the temperature of exhaust gas for the engine Wärtsilä 12V46C, the engine used in the case.

Table C.1: Data used in interpolation block for heating of LT water in the DG model, values taken from figure C.1. The zero point is assumed.

Engine load	LT heating [kW]
100%	2600
70%	2000
50%	1600
0%	0

Table C.2: Data used in interpolation block for heating of HT water in the DG model, values taken from figure C.2. The zero point is assumed.

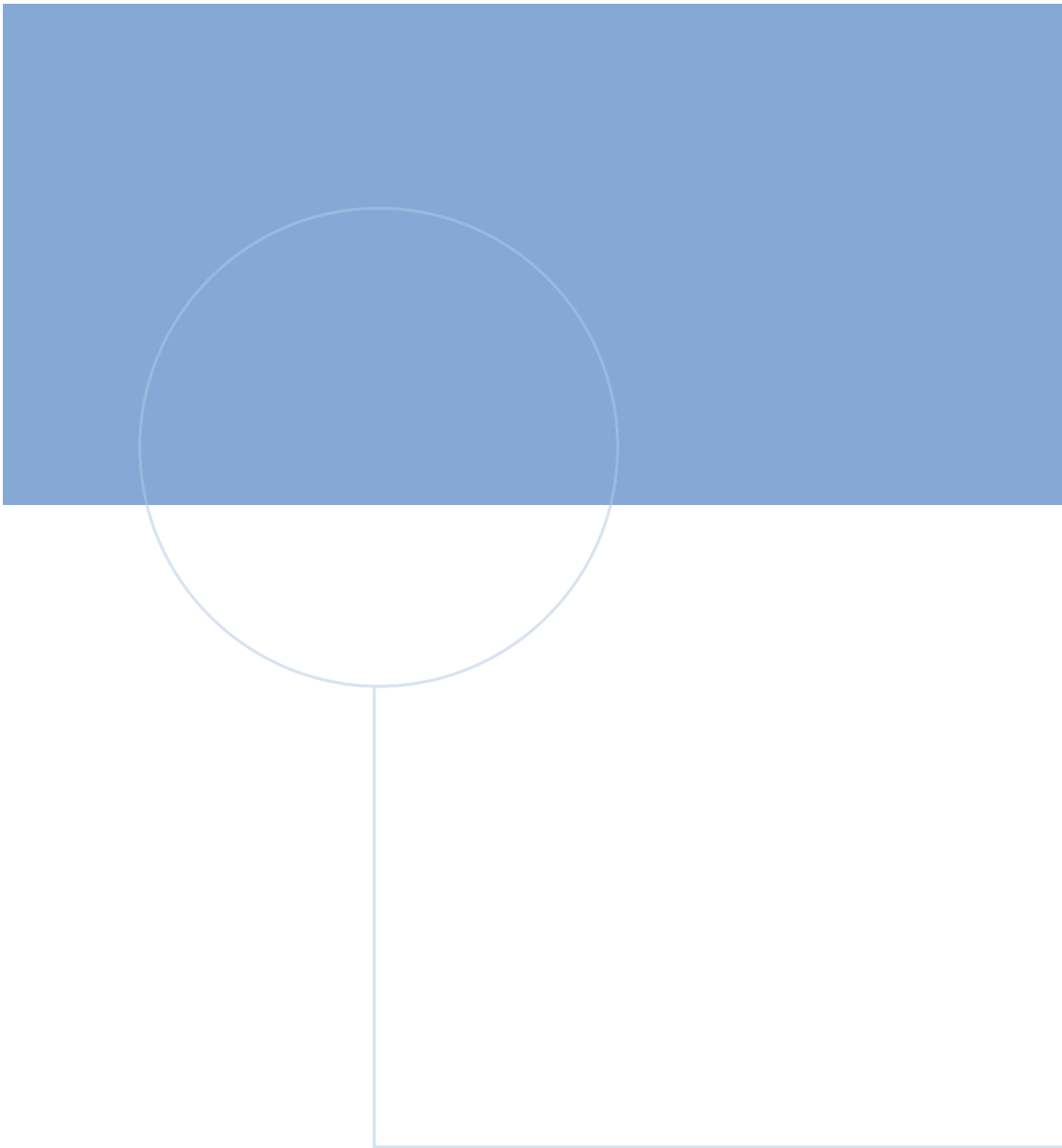
Engine load	HT heating [kW]
100%	4000
80%	2750
50%	1500
0%	0

Table C.3: Data used in interpolation block for mass flow of exhaust gas in the DG model, values taken from figure C.3. The zero point is assumed.

Engine load	EG mass flow [kg/s]
100%	23
65%	15
0%	0

Table C.4: Data used in interpolation block for temperature of exhaust gas in the DG model, values taken from figure C.4. No zero point can be assumed in this case, so values for engine loads lower than 50% is extrapolated from the known data shown here.

Engine load	EG temperature [C]
100%	355
78%	360
50%	415



NTNU

Norwegian University of
Science and Technology