William Davidsen

Modeling and Control of Steam Bottoming Cycles with Moving Point of Vaporization

Master's thesis in Industrial Chemistry and Biotechnology
Supervisor: Sigurd Skogestad
Co-supervisor: Lucas Ferreira Bernardino
June 2025

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Chemical Engineering

NTNU
Norwegian University of
Science and Technology

William Davidsen

Modeling and Control of Steam Bottoming Cycles with Moving Point of Vaporization

Master's thesis in Industrial Chemistry and Biotechnology
Supervisor: Sigurd Skogestad
Co-supervisor: Lucas Ferreira Bernardino
June 2025

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Chemical Engineering

NTNU
Norwegian University of
Science and Technology

ABSTRACT

Petroleum-related activities account for nearly a quarter of Norway's total greenhouse gas emissions, posing a significant challenge to the climate targets under the Paris Agreement. Reducing emissions from the oil and gas sector is therefore critical. Combined cycle power plants offer a promising alternative for enhancing fuel efficiency and reducing emissions in offshore installations. In this thesis, a dynamic model of a steam bottoming cycle for offshore implementations was developed in CasADi/MATLAB and solved using the IDAS solver from SUNDIALS. The model was based on simplifying assumptions such as ideal gas and constant heat capacities. It demonstrated the ability to change equations used in the solver based on the vapor quality, allowing the point of vaporization to move along the once-through steam generator (OTSG). Several control structures for control of the superheated steam and operational strategies for power were tested and compared to relevant studies.

Simulations showed nonlinearities and instabilities of the open-loop model, and dynamics related to moving point of vaporization. Further, the simulations showed benefits of introducing a flow controller for linearization, improved dynamic response and decoupling of controlled variables. The work confirmed the advantages seen in related studies of a combined feedforward and feedback control based on a steady-state energy balance over the OTSG, compared to feedback control.

A case study with two different operational objectives for power control was done. The combined cycle case showed that the valve position controller effectively minimized steady-state throttling losses, however improved steam temperature dynamics from constant steam pressure operation was not observed. The steam cycle case, showed throttling losses for constant pressure operation with larger loss at lower loads.

SAMMENDRAG

Petroleumsrelaterte aktiviteter står for nesten en fjerdedel av Norges totale klimagassutslipp, noe som utgjør en betydelig utfordring for å nå klimamålene i Parisavtalen. Det er derfor avgjørende å redusere utslippene fra olje- og gassektoren. Kombinerte sykluser er et lovende alternativ for å øke drivstoffeffektiviteten og redusere utslippene på offshore-installasjoner.

I denne masteroppgaven ble en dynamisk modell av en dampsyklus for offshore installasjon utviklet i CasADi/MATLAB og løst ved hjelp av IDAS fra SUNDIALS. Modellen var basert på antakelser, som ideell gass og konstante varmekapasiteter. Den viste evne til å endre ligningene brukt i løseren basert på gassfraksjonen, noe som gjorde det mulig å flytte fordampningspunktet langs en-passasje-dampgeneratoren (OTSG). Flere reguleringsstrukturer for kontroll av damptemperatur og trykk, samt strategier for effektstyring, ble testet og sammenlignet med relevante studier.

Simuleringene viste ulineariteter og ustabiliteter i modellen, samt dynamikk relatert til flytting av fordampningspunktet. Videre viste simuleringene fordeler ved å innføre kontroll av fødestrøm for linearisering, forbedret dynamisk respons og økt uavhengighet av kontroll variabler. Arbeidet bekreftet fordelene som er vist i lignende studier ved bruk av kombinert forover- og tilbakekoblet regulering basert på energibalanse over OTSG-en, sammenlignet med kun tilbakekobling.

En case-studie med to ulike operasjonelle mål for effektstyring ble gjennomført. I casen med kombinert syklus viste at ventilposisjonsregulatoren effektivt reduserte effekttapet ved stillstand, men forbedredet damptemperaturdynamikk ved konstant trykk ble ikke observert. I damp-syklus casen ble det observert effekttap ved konstant trykk, med større tap ved lavere pådrag.

PREFACE

This thesis was written as part of the course *TKP4900 Chemical Process Technology, Master's Thesis* at the Department of Chemical Engineering at Norwegian University of Science and Technology (NTNU). The work was done in collaboration with SINTEF Energy Research as a part of the Petroleum Research Centre (PETROSENTER) programme.

I would like to express my gratitude to my supervisor Sigurd Skogestad for his invaluable insights, incredible intuition within process control and foreseeing issues weeks before I encountered them. I am also deeply thankful to my co-supervisor Lucas F. Bernardino for always being available and incredibly helpful. For the amazing collaboration on the summer and specialization projects, which really encouraged me to pursue this path for the thesis.

Further, I would like to extend a thanks to Rubén M. Montañés and the SINTEF Gas Technology Group for allowing me to be a part of the project.

I would like to thank my family for all the support and help you have given me over the past five years, this would not have been possible without you. Lastly I would like to thank my fellow students for the incredible working atmosphere and all the memories along the way.

Trondheim
11th of June 2025

William Davidsen

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

CV	*Controlled variable*
DAE	*Differential algebraic equation*
DV	*Disturbance variable*
FOPDT	*First order plus dead time*
FPSO	*Floating Production Storage and Offloading vessel*
FVM	*Finite volume method*
GT	*Gas turbine*
MB	*Moving boundary*
MV	*Manipulated variable*
NTNU	*Norwegian University of Science and Technology*
ODE	*Ordinary differential equation*
OTSG	*Once-through steam generator*
PDE	*Partial differential equation*
PID	*Proportional integral derivative*
RGA	*Relative gain array*
SIMC	*Simple internal model control*
ST	*Steam turbine*

CHAPTER

ONE

INTRODUCTION

In 2023, the total greenhouse gas emissions from Norwegian petroleum-related activities amounted to 11.6 million tons of CO_2 equivalents [1]. This represented approximately 24.8% of Norway's total greenhouse gas emissions to air. The petroleum-related emissions were 40% higher than in 1990, mainly due to increased production [1].

The dominant source of emissions was gas turbines responsible for power and heat production with 82.91% of the petroleum-related emissions [2]. Given their large share, reducing emissions from gas turbines would significantly impact total emissions.

In addition to being an important part of Norway's commitment to the Paris agreement, emission reduction also presents economic incentives for the oil and gas sector [3]. The European Union Emissions Trading System is a key tool in EU's strategy to reduce greenhouse gas emissions cost-effectively [4]. In 2023, the carbon tax was set to 761 NOK per ton of CO_2 equivalents, and is expected to rise to around 2000 NOK per ton by 2030 [2, 3]. A 1% reduction in turbine-related emissions could have saved the oil and gas sector approximately 70 million NOK in avoided taxes in 2023. The tax reduction comes in addition to potential savings from reduced fuel consumption.

However, not all emission reduction measures are easily implemented offshore. For example, electrification is a significant challenge for certain installations such as Floating Production Storage and Offloading vessels (FPSOs). Additionally, electrification has lately been heavily debated due to its potential impact on onshore electricity price and demand [5].

Another emission reduction measure is turbine efficiency improvement. Fuel efficiency can be improved by utilizing the hot flue gas from the gas turbine to produce additional power in a steam cycle ('bottoming cycle'). Combined cycle power plants, which integrate gas turbines with a bottoming cycle, are standard practice in onshore installations but are not wide spread offshore due to size and weight limitations [6]. A few combined cycle systems were installed in the early 2000s on the Oseberg, Eldfisk and Snorre B platforms with the aim of reducing the fuel consumption with 25% [7].

1.1 Previous Work and Scope

This work is a part of a case study in the LowEmission subproject SP1. The objectives of this project include increasing the overall efficiency of gas turbines and enabling more wide-spread implementation by optimizing weight and developing effective control strategies [8]. A lot of research has been conducted in the project since the project start-up in 2019: Montañés et al. (2021) [9] and Mazzetti et al. (2021) [10] optimized the geometry and minimized the weight of the once-through steam generator (OTSG). Zotică et al. (2022) [11] proposed nonlinear input and output transformations for feedforward control of heat to power cycles. Montañés et al. (2024) [6] studied the performance, operation and control of a combined cycle for offshore operation using a high-fidelity model, to mention a few. Another relevant study is Zotică et al. (2020) [12] which studied operation and control of drum-based heat to power cycles, with fixed point of vaporization.

This thesis is a continuation and improvement of the model development in the specialization project Davidsen (2024) [13]. The project developed a discretized model of the OTSG with moving point of vaporization. The scope of this work is to develop a model of the bottoming cycle including the developed OTSG model to study performance and operational strategies for offshore implementations with moving point of vaporization. The results will be compared with relevant studies, such as the high-fidelity model in Montañés et al. (2024) [6] and the fixed point of vaporization model in Zotică et al. (2020) [12].

1.2 Thesis Structure

The thesis is divided into 10 chapters. Chapter 1 gives a brief motivation and overview of previous work. Chapter 2 presents some basic concepts within modeling and process control. Chapter 3 describes the bottoming cycle process studied in this project. Chapter 4 presents the mathematical model formulation of the bottoming cycle model with the underlying assumptions, while the model implementation is described in Chapter 5. Chapter 6 elaborate on the control structures developed on top of the bottoming cycle model which were studied in this work. The simulation results and discussion are divided into two chapters, where the simulations only involving the OTSG are given in Chapter 7 and the bottoming cycle simulations are given in Chapter 8. Additional discussion is included in Chapter 9, while the conclusion and further work are given in Chapter 10.

CHAPTER

TWO

BASIC CONCEPTS

2.1 Spacial Discretization

Two of the most common methods for discretization in thermo-fluid simulations are moving boundary (MB) and finite volume method (FVM). In the MB method, the heat exchanger is divided into three uneven volumes for liquid, two-phase and gas. The size of each volume change with the operating conditions within the heat exchanger. In contrast, the FVM-method divide the heat exchanger into evenly sized volumes. Each control volume can exist in a liquid state, a gas state, or a two-phase state, and the fluid state changes with the operating conditions. [14]

Both methods transforms the partial-differential-equations (PDE) into ordinary differential equations (ODE) by discretizing the system along the one-dimensional spatial domain. The transformed equations are easier to solve numerically [15]. Mass and energy balances are applied on the discretized volumes, and the formulation of both discretization methods imply that the mass and energy are conserved over each volume and the total volume [16]. Figure 2.1.1 illustrates how these methods can be applied to model a liquid-to-gas phase transition in a heat exchanger tube.

Despite their similarities, the methods differ in metrics such as accuracy, computational cost and complexity. The FVM method is generally less complex to implement than the MB method, as it use fixed boundaries. Depending on the discretization resolution, FVM may offer higher accuracy than MB. However, FVM tends to be more computationally intensive due to the greater number of equations that need to be solved. [17]

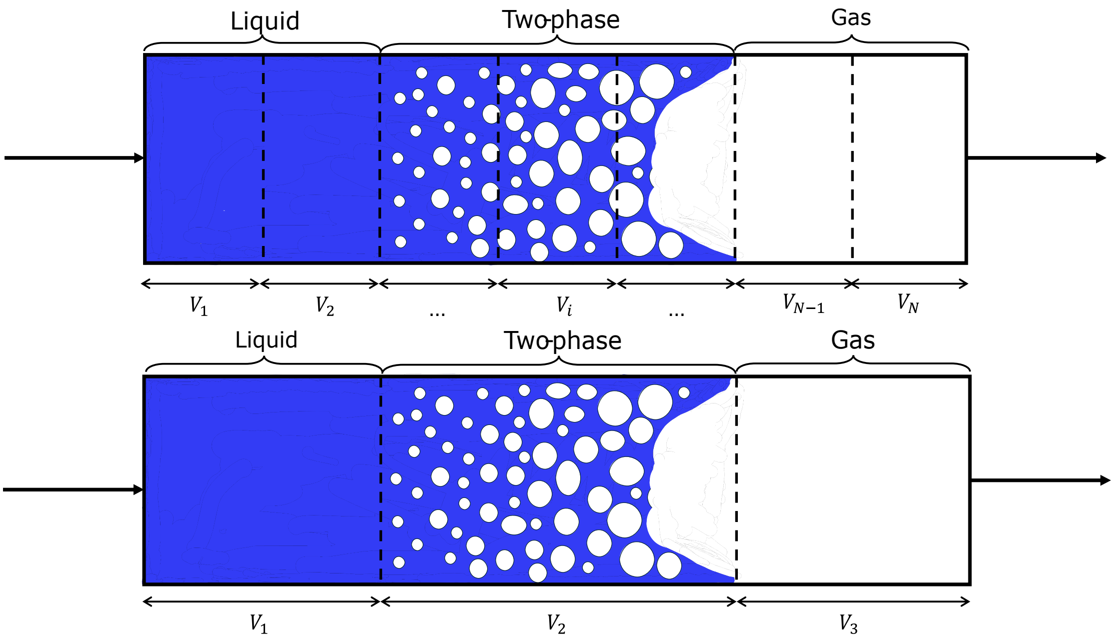


Figure 2.1.1: Discretization methods FVM (on top) and MB (on bottom) for a liquid to gas phase transition in a tube.

2.2 Dynamic Systems

In the context of dynamic processes, system variables are generally classified into three types. Manipulated variables (MV, u), are those that can be adjusted by the control system. An example of a manipulated variable is a valve position. Disturbance variables (DV, d) affect the process but cannot be directly manipulated. Controlled variables (CV, y) are the process variables that the system aims to regulate to a desired setpoint. This classification forms the basis for the analysis and design of control systems. A block diagram of a general dynamic system is shown in Figure 2.2.1. [18]

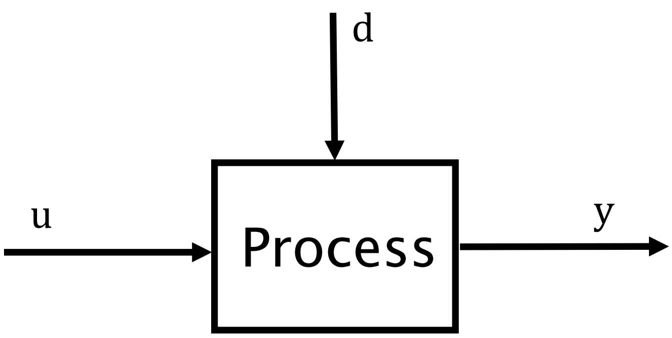


Figure 2.2.1: Block diagram of a general dynamic system.

2.3 Process Control

A process control system can be decomposed into several layers, based on time-scale separation and physical distance. In this work, the focus is on the regulatory control layer, whose primary objective is to prevent the process from deviating from its designated setpoint over short time scales [19].

Several control strategies are available to achieve the objectives of regulatory control. One common approach is feedback control, which utilize the deviation between the CV and its setpoint to apply corrective action using the MV. An important advantage of feedback control is that it can correct for unmeasured disturbances. Feedforward control is another control strategy that use a model to correct measured disturbances before the CV is affected. However, feedforward control has some drawbacks as the system is not able to correct for unmeasured disturbances. Therefore, these two methods are often combined for industrial applications to leverage the strength of both strategies. [18]

Proportional–integral–derivative(PID)-controllers are widely used in the industry for feedback control. The controller consists of three terms: the proportional part adjust the output based on the size of the error, the integral part removes the steady-state offset by integration and the derivative part predict the future trend. [20]

The derivative term of the PID-controller is often neglected as it is sensitive to measurement noise and good control can often be achieved without it [21]. This results in the PI-controller equation as shown in Eq. 2.1.

$$u = u_0 + K_c(y_{SP} - y) + \frac{K_c}{\tau_I}\int_0^t (y_{SP} - y)dt \tag{2.1}$$

Here, y is the controlled variable with the setpoint y_{SP}, u represents the control output with bias u_0, and K_c and τ_I are tuning parameters. The actual control output applied to the process may be subject to physical constraints. For instance, a valve cannot open beyond its fully open position or close beyond fully shut. This valve saturation can be expressed as shown in Eq. 2.2, where $\tilde{u}$ is the actual control output.

$$\tilde{u} = max(0, min(u, 1)) \tag{2.2}$$

The PI-control loop can be visualized in a block diagram as shown in Figure 2.3.1.

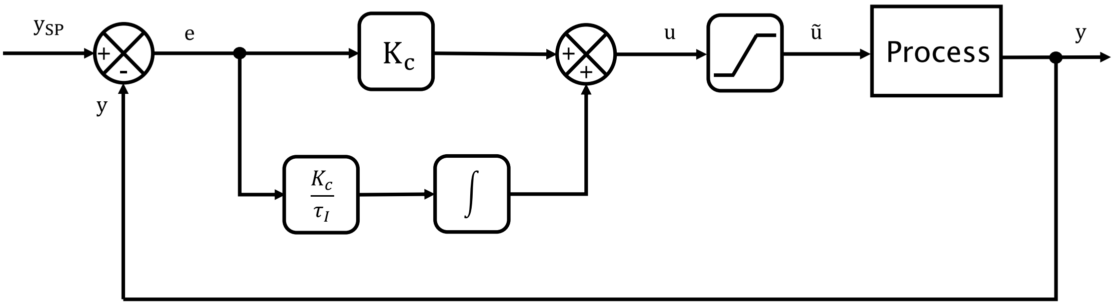


Figure 2.3.1: Block diagram for PI-controller with valve saturation. [19]

Anti-windup is important for controllers with integral action as valves may saturate. If a valve is saturated over a long period of time, the integral error becomes very large and an equally large input with opposite sign is needed to counteract the error. The windup can therefore disable the controller over a substantial period of time. Anti-windup is used to disable the integral action when the valve is saturated [19]. The PI-controller equation with anti-windup is shown in Eq. 2.3, and in a block diagram in Figure 2.3.2.

$$u = u_0 + K_c e + \frac{K_c}{\tau_I}\int_0^t e\ dt + \frac{1}{\tau_T}\int_0^t e_T\ dt \qquad (2.3)$$
where: $e = y_{SP} - y$, $e_T = \tilde{u} - u$

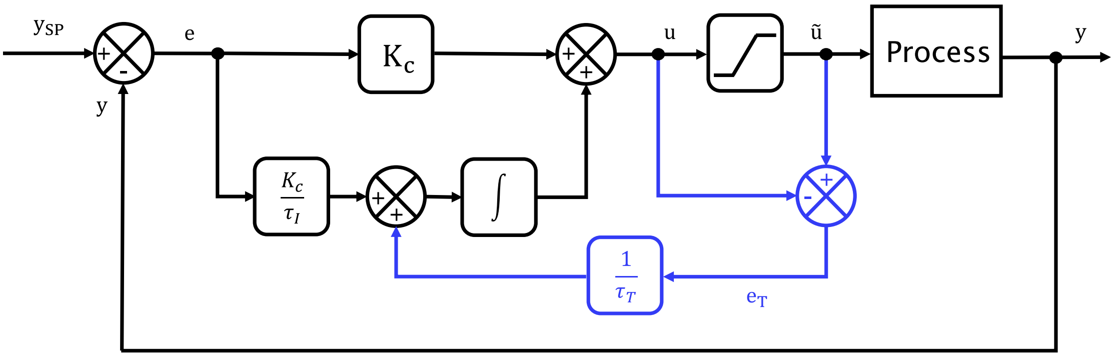


Figure 2.3.2: Block diagram for PI-controller with valve saturation and Anti-windup. [19]

2.3.1 SIMC Tuning

The tuning coefficients K_c, τ_I and τ_T can be calculated with SIMC [22] tuning rules based on the model parameters k, τ and θ, and closed-loop time constant τ_c. k is the steady-state gain, τ is the time constant and θ is the delay. The model parameters can be found by applying step changes to the system. This section describe the tuning models used in this work.

2.3.1.1 FOPDT Model

The first order plus dead time (FOPDT) model is a useful approximation for most real processes, and Skogestad's Half-rule can be used to approximate higher order dynamics [19]. The model parameters can be found from the system response as shown in Figure 2.3.3.

The SIMC tuning rules used for a FOPDT process are shown in Eq. 2.4. For anti-windup, τ_T is assumed to be equal to τ_I.

$$\begin{aligned} K_c &= \frac{1}{k}\frac{\tau}{\tau_c + \theta} \\ \tau_I &= min(\tau, 4(\tau_c + \theta)) \end{aligned} \tag{2.4}$$

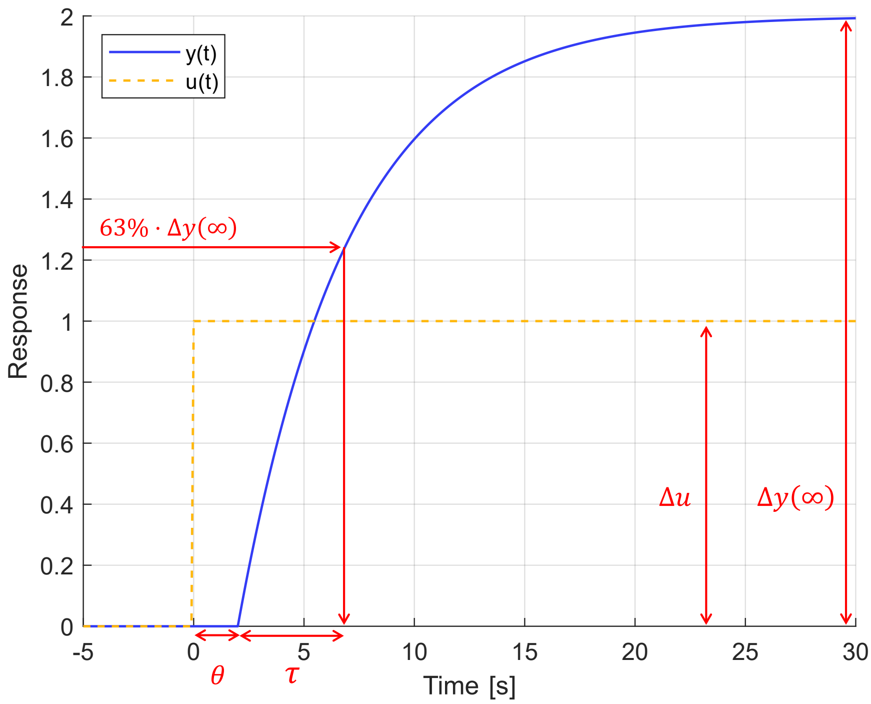


Figure 2.3.3: Example on how model parameters can be found from a step change for a FOPDT response.

2.3.1.2 Instantaneous Processes

For processes where the change is instantaneous ($\tau = 0$s) a pure I-controller can be used [19]. This could for example be pressure related dynamics which can be very fast. For instantaneous processes, the SIMC tuning rules becomes as shown in Eq. 2.5.

$$\begin{aligned} \tau_I &= \tau \\ K_I &= \frac{K_c}{\tau_I} = \frac{1}{k}\frac{1}{\tau_c + \theta} \end{aligned} \tag{2.5}$$

Since τ is zero, only the integral term of the PI-equation remains as shown in Eq. 2.6.

$$u = u_0 + K_I \int_0^t e \text{ dt} \tag{2.6}$$

2.3.1.3 P-controller Tuning

P-controllers were tuned as the proportional gain of a SIMC PI-controller as shown in Eq. 2.7.

$$K_p = \frac{1}{k}\frac{1}{\tau_c + \theta} \tag{2.7}$$

This leaves only the proportional part of the PI-controller equation as shown in Eq. 2.8.

$$u = u_0 + K_p e \tag{2.8}$$

2.3.2 Relative Gain Array (RGA)

RGA is a measure of interaction for multivariable input-output systems [23]. G is an n $\times$ n non-singular steady-state gain matrix, as shown in Eq. 2.9. Each element g_{ij} represents the steady-state gain from manipulated variable j to controlled variable i.

$$G = \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \tag{2.9}$$

The RGA matrix can then be calculated with Eq. 2.10 where $\times$ denote element-by-element multiplication.

$$RGA = G \times (G^{-1})^T \tag{2.10}$$

The RGA matrix can also be used to decide on y_i-u_j pairings based on the following RGA pairing rules listed in Skogestad (2005) [23]:

- Pairing rule 1: Prefer pairings such that the rearranged system, with selected pairings along the diagonal, has an RGA matrix close to identity at frequencies around the closed-loop bandwidth.
- Pairing rule 2: Avoid (if possible) pairing on negative steady-state RGA elements.

CHAPTER

THREE

PROCESS DESCRIPTION

The combined cycle consists of a gas turbine and connected steam bottoming cycle. The bottoming cycle is a simple steam cycle, and is conceptually the same as presented in Zotică et al. (2022) [11]. The main units in the bottoming cycle are a pump, steam generator, steam turbine and condenser, as shown in Figure 3.0.1. The combustion in the gas turbine (GT) produce power and hot flue gas that contain excess energy in the form of heat. The flue gas energy can be recovered and used to generate superheated steam in the OTSG. The superheated steam is used to generate additional power in the steam turbine (ST) increasing the fuel efficiency of the combined cycle. The steam is then condensed in a condenser and recycled in the closed-loop system by the feedwater pump.

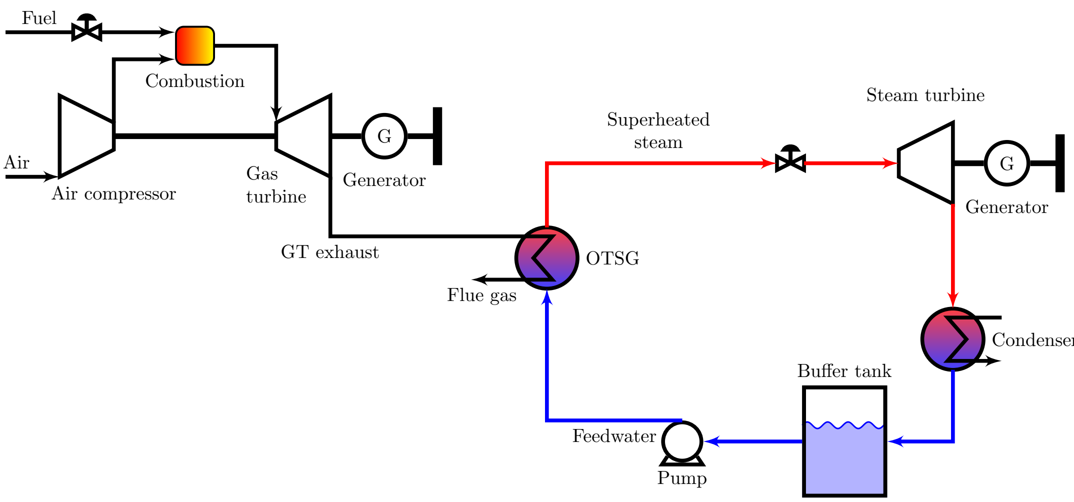


Figure 3.0.1: Process flowsheet of the combined cycle showing the GT and main process components of the bottoming cycle such as pump, OTSG, ST and condenser. Component framework for flowsheets developed by Zotică et al. (2022) [11].

3.1 Heat-to-Work Cycles

The bottoming cycle is a heat-to-work cycle which consists of four main steps:

1. Pump - Isenthalpic compression of the working fluid from the saturation pressure to OTSG inlet pressure.
2. OTSG - Super-heated steam generation with pressure drop.
3. Turbine - One stage isentropic expansion of steam.
4. Condenser - Isobaric condensation of fluid to saturated liquid.

These steps are illustrated in a p-H-diagram in Figure 3.1.1. The diagram is a visual representation and does not depict the operating conditions with complete accuracy. As shown in the figure, step 3 corresponds to the energy which is converted to work by the steam turbine, while step 4 represents the energy that is transferred to the bottoming cycle, but not converted into useful work. The figure indicates that approximately one-third of the energy transferred to the bottoming cycle in Step 2 is converted into work by the cycle.

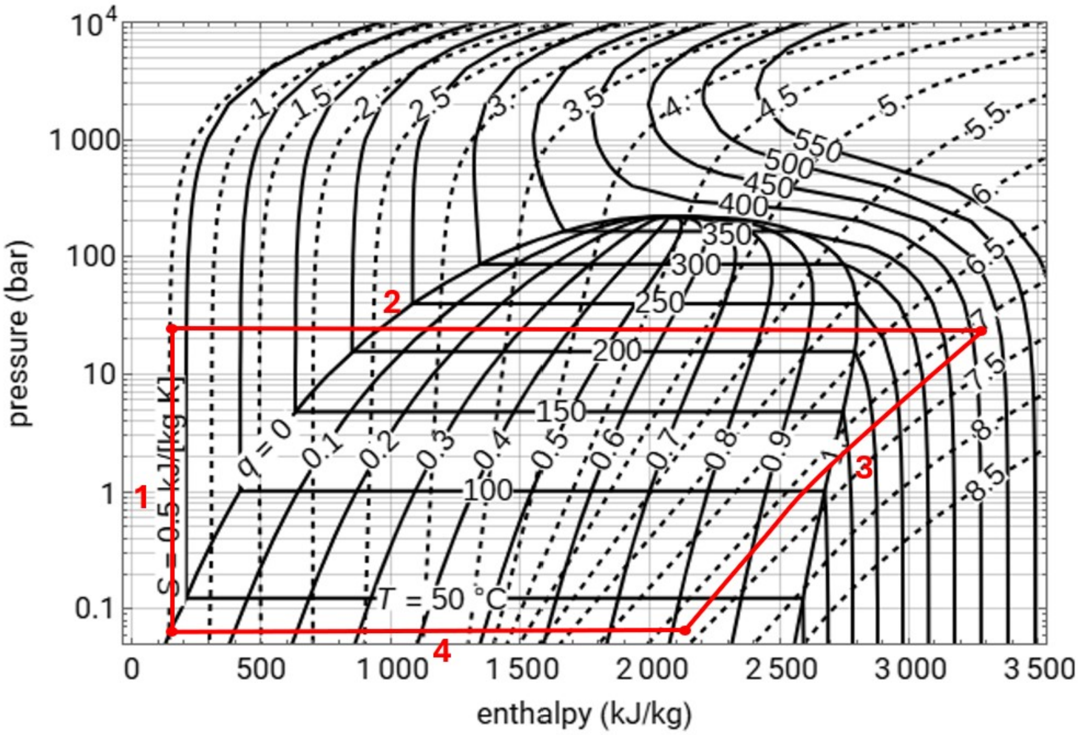


Figure 3.1.1: p-H diagram [24] showing the main steps of the heat-to-work cycle.

For optimal steam turbine operation, the liquid fraction at the turbine outlet should not exceed 0.13–0.16, as higher fractions can cause erosion and damage to the turbine [21]. To maximize efficiency, the outlet pressure should be kept as low as possible, typically between 0.03 and 0.07 bar [21]. However, if the liquid is sub-cooled, the efficiency is reduced as more energy is required to heat the fluid in step

2. The pressure range corresponds to a saturation temperature in the condenser of approximately 24–39°C, based on the Antoine equation. Cooling water suitable for maintaining these condenser operating conditions is readily available in arctic regions such as Norway.

3.1.1 Boundary Conditions

Boundary conditions are important for defining the limits of the system model. In this work the focus is on the bottoming cycle and the gas turbine is not modeled, but its effect on the bottoming cycle is captured through the flue gas mass flow (m_g^0) and temperature (T_g^0). These variables make up the boundary conditions of the model on the exhaust gas side.

The bottoming cycle is modeled using pressure driven flow with pressure boundaries on both sides of the pump. At the OTSG cold side inlet, the pump acts as the boundary, with fixed pump pressure (p_p) and temperature (T_p). At the end of the cycle, the condenser pressure (p_c) is used as a boundary condition, based on the assumption of perfect cold side mass flow control. The nominal boundary conditions used in the simulations are given in Appendix A.3.

CHAPTER

FOUR

MATHEMATICAL FORMULATION

This chapter aims to show how the bottoming cycle was formulated mathematically and the underlying assumptions of the model. Simplified models are often used for control purposes. The model is based on assumptions, including the ideal gas law, constant heat capacity and homogeneous two-phase flow. Modeling principles such as conservation of mass and energy were applied. Each process unit will be further described in the following sections. The complete set of equations, assumptions and parameters used in the model are given in Appendix A.

4.1 OTSG Model

The OTSG model was developed in the specialization project Davidsen (2024) [13]. It was modeled as an ideal counter current heat exchanger with phase change from liquid to steam on the cold side and no phase change on the hot side. The total volume and area of the OTSG were equally distributed over N segments using FVM discretization. The FVM method was chosen due to its straightforward implementation and the computational expense is considered negligible as the overall system of equations remains relatively small. Each segment contains a hot and a cold side with connecting inflows and outflows. Mass and energy balances were applied on each segment. A visualization of the OTSG model is shown in Figure 4.1.1. Assumptions and equations are summarized in Appendix A.1 and A.2, and only changes to the model developed in the specialization project will be described in detail.

The specialization project highlighted some model related issues for further work. The following section aims to resolve some of these issues and elaborate on additional development of the model before the system was extended with the remaining process units.

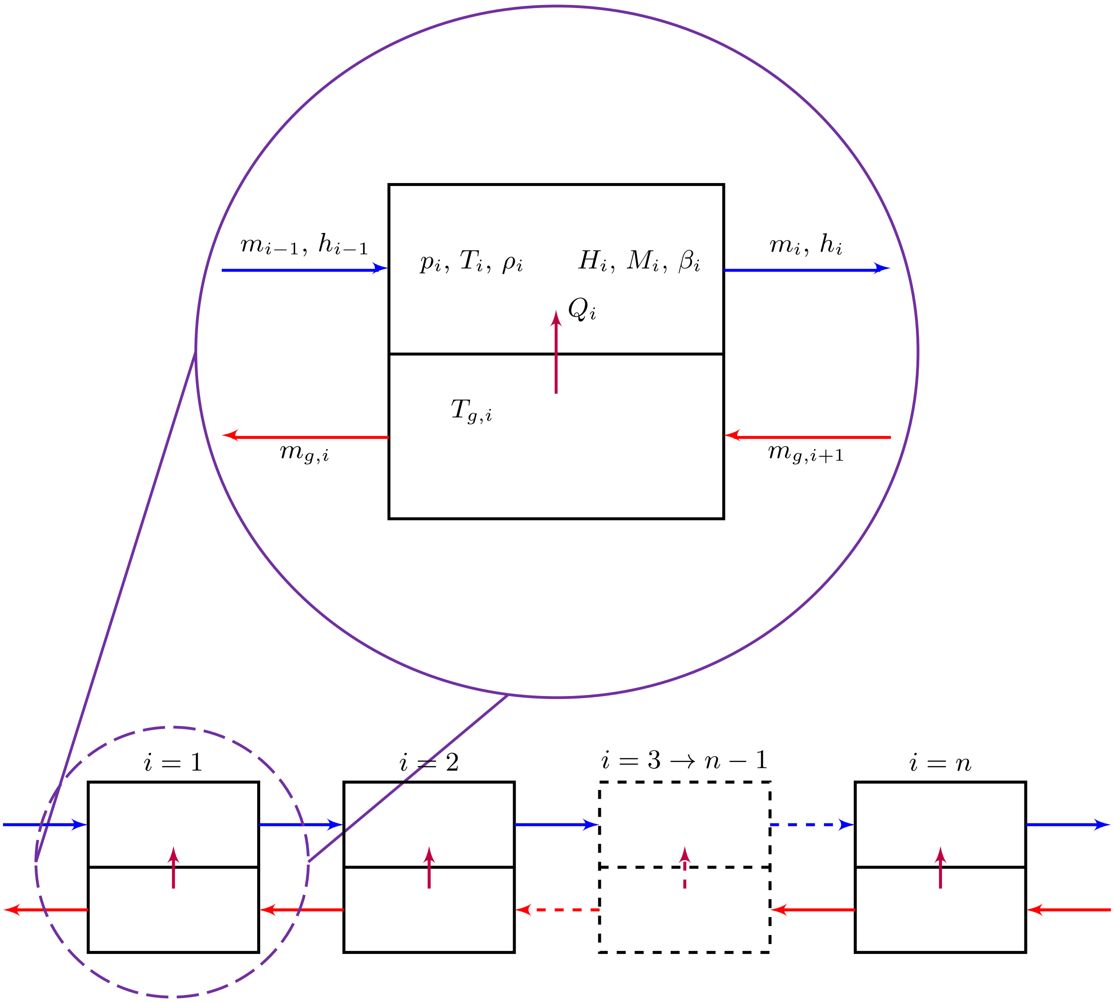


Figure 4.1.1: OTSG model developed in Davidsen (2024) [13] discretized with N elements. Each segment has an inlet and outlet flow for each side and heat transfer from the hot to cold side.

4.1.1 Model Revisions Compared to Specialization Project

One of the issues presented in Davidsen (2024) [13] was related to the heat transfer equation and its dependency on the discretization resolution. The specialization project proposed changing the modeling of the temperature driving force (ΔT) used in the heat transfer equation shown in Eq. 4.1. The temperature driving force was changed to arithmetic as shown in Eq. 4.2 to minimize this effect in the revised model.

$$Q_i = U_i A \Delta T_i \tag{4.1}$$

$$\begin{aligned} \Delta T_i &= T_{g,i} - T_i && \text{(Original)} \\ \Delta T_{i,Ar} &= \frac{T_{g,i+1} - T_i}{2} + \frac{T_{g,i} - T_{i-1}}{2} && \text{(Revised)} \end{aligned} \tag{4.2}$$

Another issue presented was solver breakdown during reversed flow caused by larger step changes such as -50% cold side flow. The project found that the breakdown was caused by a rise in temperature followed by a change of state in the first segment but not the consecutive segments. However, the cause of the increased temperature was not identified. The model is not designed to handle reversed flow. While the solver is able process such conditions, the results are not realistic because the underlying equations are invalid. For example, the cold-side specific enthalpy (h_i) is calculated using the temperature of segment i. Since the equations are not changed during reversed flow, this gives wrong segment temperature in enthalpy calculation. This is particularly an issue for the first segment as the pump temperature (T_p) is fixed. Resulting in an energy build-up in segment 1 as a fixed amount of energy is leaving the segment for reversed flow as shown from the energy balance in Eq. 4.3.

$$\frac{dH_1}{dt} = m_p h_p - m_1 h_1 + Q_1 \tag{4.3}$$

This inconsistency is assumed to be the root cause of the solver breakdown. Even though reversed flow could be realistic for short periods of time, it was decided to not accommodate for this phenomena in the model. Instead, very large step changes were avoided to prevent reversed flow from occurring.

4.1.2 Additional Model Revisions

4.1.2.1 OTSG Heat Transfer

In the previous model, the overall heat transfer coefficient (U) was constant for all cold side fluid phases. The change of phase on the cold side was assumed to have small influence on U as the low pressure flue gas on the hot side has the largest resistance for the overall heat transfer. This assumption does not hold in this work as the OTSG geometry is significantly different. Due to mounted fins on the hot side, the effective hot side area is much larger, reducing the hot side resistance. Consequentially, U has a higher dependency on the cold side heat transfer coefficient and the phase change cannot be neglected.

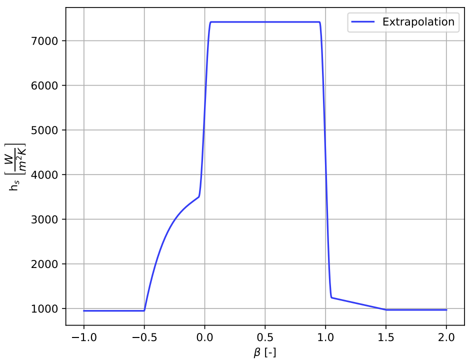
To capture the changes in U better, a polynomial extrapolation was made based on provided OTSG data from Montañés et al. (2021) [9] on cold side heat transfer coefficient (h_s) and hot side heat transfer coefficient (h_g).

On the cold side, an extrapolation for each state as a function of vapor quality (β) was made. A lower boundary on h_s was used outside the range $\beta \in [-0.5, 1.5]$ ($T \in [39°C, 505°C]$) for numerical stability. Additionally, two transition polynomials were made around the switching points to make one continuous function. The transition polynomials were made on a domain ± 0.05 around the switching points. The polynomial extrapolation as a function of β is shown in Figure 4.1.2a.

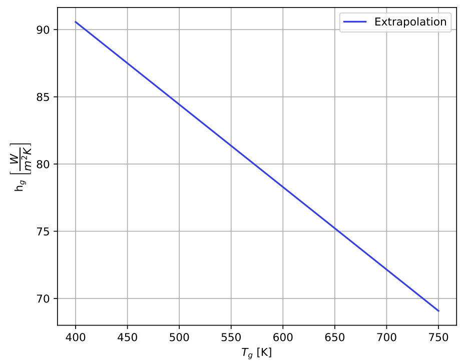
On the hot side, a single extrapolation as a function of flue gas temperature was made, as there is no phase change on this side. The polynomial extrapolation is shown for the range $T_g \ \epsilon \ [400K, 750K]$ in Figure 4.1.2b, which is expected to capture the operational range of the OTSG.

The overall heat transfer coefficient was then calculated for each segment as shown in Eq. 4.4. The effects of increased heat transfer on the hot side due to geometry are captured in the variable A_{corr}. Note that thermal resistance in the wall is not considered in this work, and is neglected in the calculation of U.

$$U_i = \frac{1}{\dfrac{1}{h_s(\beta_i)} + \dfrac{1}{h_g(T_{g,i}) \cdot A_{corr}}} \tag{4.4}$$



(a) h_s



(b) h_g

Figure 4.1.2: Polynomial extrapolation for h_s and h_g used in calculation of U.

4.1.2.2 Pump Implementation

The OTSG model was developed with pressure boundaries at the inlet and outlet of the cold side. It was later discovered that this may have caused the 'Spring effect' during transitions seen in Davidsen (2024) [13], which caused mass and pressure oscillations making the system difficult to control. The feedwater pump equation at the OTSG cold side inlet was changed to include a valve so that the pump mass flow could be used as a manipulated variable, as shown in Eq. 4.5.

$$\begin{aligned} m_p &= k_{OTSG}(p_p - p_1) \quad \text{(Previous pump equation)} \\ m_p &= z_m k_m (p_p - p_1) \qquad \text{(new pump equation)} \end{aligned} \tag{4.5}$$

The valve was assumed isenthalpic and with linear valve characteristics. It was designed with a pressure drop of 5 bar, and a nominal valve opening $z_{m,0} = 0.5$. The pressure drop help prevent reversed flow when the OTSG pressure vary. The pump, valve and OTSG with relevant system variables are shown in Figure 4.1.3. Simulations of different pump equations for the OTSG are shown in Chapter 7.

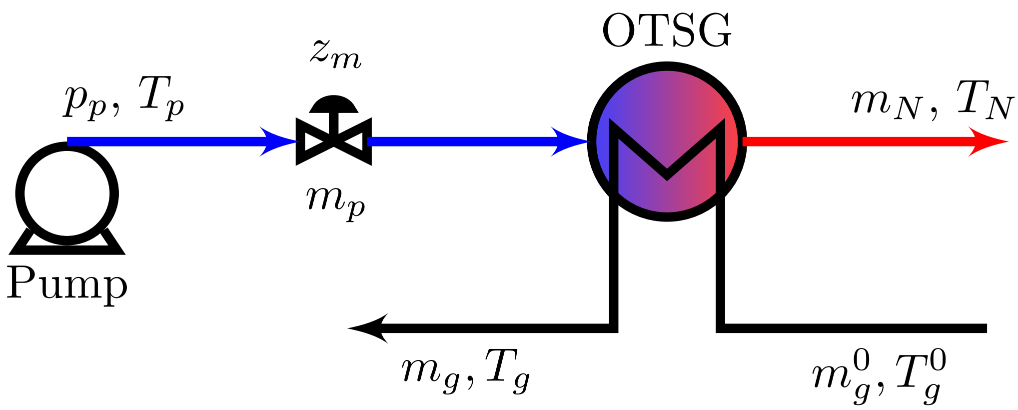


Figure 4.1.3: Pump with pressure boundary and valve for mass flow control (subscript p). The hot flue gas vaporize the working fluid making superheated steam in the OTSG. Component framework: [11]

4.2 Superheated Steam Section

The superheated steam section is the part of the bottoming cycle where the working fluid exists as steam. This section includes the process units: steam valve, steam turbine, and two steam holdups. The units are illustrated in Figure 4.2.1.

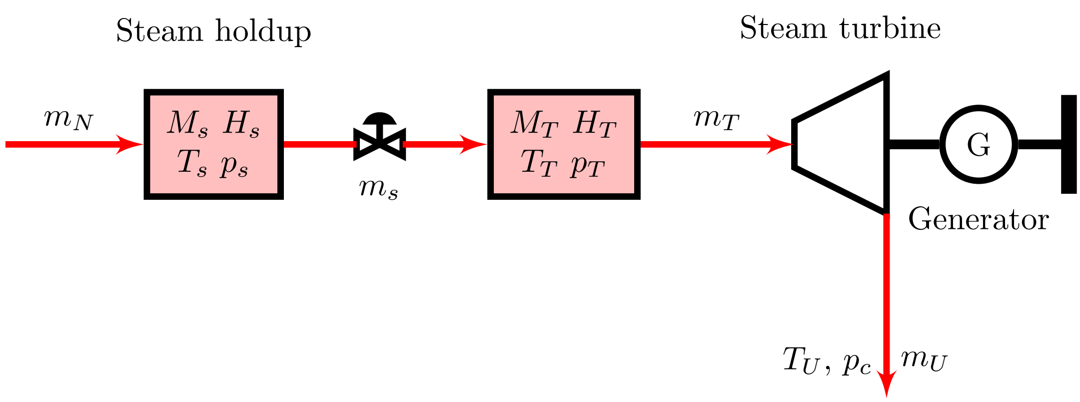


Figure 4.2.1: Main components of the superheated steam section. The superheated steam from the OTSG outlet (subscript N) enter the first steam holdup (subscript s) followed by the isenthalpic valve and the second steam holdup (subscript T). Further, the fluid is expanded in the steam turbine where the mechanical energy is converted to electrical energy in the generator. The post-turbine flow (subscript U) is then condensed in the condenser (subscript c). Component framework: [11].

4.2.1 Steam Holdups

A volume of superheated steam was added between the OTSG and superheated steam valve which represent the pipe volume in this section. Additionally, a similar volume was added between the valve and turbine to represent the steam holdup of the turbine. These volumes were modeled the same way as an OTSG cold side volume with dynamic mass and energy balances, but without heat transfer (Q=0). The equations, assumptions and parameters regarding the steam holdups are summarized in Appendix A.

4.2.2 Steam Valve

The superheated steam valve is one of the available manipulated variables of the system. The valve was assumed to be isenthalpic and have linear valve characteristics. The valve equation is shown in Eq. 4.6.

$$m_s = z_v k_v (p_s - p_T) \tag{4.6}$$

z_v is the valve opening and k_v is the valve coefficient. The superheated steam valve was designed with 1 bar pressure drop and a nominal valve opening $z_{v,0} = 0.9$.

4.2.3 Steam Turbine

The steam turbine was designed as a singe stage turbine. Here static mass balance was assumed, giving the outlet flow (m_U) equal to the inlet flow (m_T) as shown in Eq. 4.7. Note that the steam holdup of the turbine is modeled in the preceding volume.

$$m_U = m_T \tag{4.7}$$

Assuming isentropic expansion of an ideal gas with constant heat capacity within the turbine, the outlet temperature (T_U) can be related to the inlet temperature (T_T) through the pressure ratio across the turbine, as given in Eq. 4.8. Here, R is the gas constant, C_p^S is the heat capacity of steam, and M_w is the molecular weight of the water.

$$T_U = T_T \left(\frac{p_c}{p_T} \right)^{\frac{R}{C_p^S M_w}} \tag{4.8}$$

From the static energy balance, assuming adiabatic operation, the work output of the turbine (W) is equal to the change in enthalpy across the turbine. This is expressed in Eq. 4.9, where η is the polytropic efficiency.

$$W = -\eta m_T C_p^s (T_T - T_U) \tag{4.9}$$

The power (P) represents the base load and is defined as the negative of the work, assuming no loss in conversion from mechanical to electrical energy as shown in Eq. 4.10.

$$P = -W \tag{4.10}$$

The mass flow coefficient (ϕ_d) can be used for gas turbine design to relate the operating conditions to a design point as shown in Eq. 4.11. The flow coefficient was set based on the nominal operating conditions, as no industrial turbine model was used.

$$\phi_d = \frac{m_T \sqrt{T_T}}{p_T} \tag{4.11}$$

The assumption of isentropic expansion for the turbine implies that the fluid remains in gas phase, which is not the case as the outflow of the turbine may contain condensate. As a result, the temperature and vapor quality therefore have to be recalculated for the post-turbine flow to account for phase change. The specific enthalpy at the turbine outlet (h_U) is calculated based on the outlet temperature using Eq. 4.12.

$$h_U = C_p^w (T_U^{sat} - T_{Ref}) + \Delta H_{vap} + C_p^s (T_U - T_U^{sat}) \tag{4.12}$$

Due to the two-phase outflow assumption, the real temperature (T_U^{real}) is equal to the saturation temperature (T_U^{sat}) as shown in Eq. 4.13.

$$T_U^{real} = T_U^{sat} \tag{4.13}$$

The saturation temperature was calculated using Antoine equation as shown in Eq. 4.14, where A_c, B_c and C_c are the Antoine coefficients.

$$log_{10}(p_c) = A_c - \frac{B_c}{T_U^{sat} + C_c} \tag{4.14}$$

The enthalpy and saturation temperature can then be used to recalculate vapor fraction as shown in Eq. 4.15.

$$h_U = C_p^w (T_U^{sat} - T_{Ref}) + \beta_U \Delta H_{vap} \tag{4.15}$$

4.3 Condenser and Buffer Tank

The condenser and buffer tank make up the remaining part of the bottoming cycle. The process units are illustrated in Figure 4.3.1. The post-turbine flow is cooled to saturated liquid in the condenser and stored in the buffer tank.

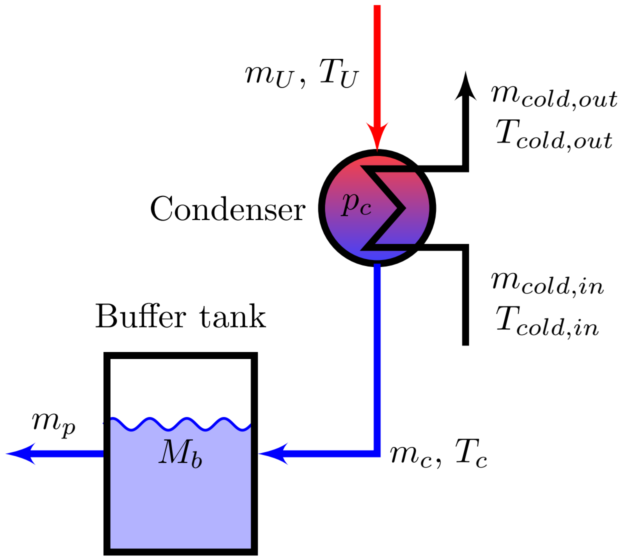


Figure 4.3.1: Condenser and buffer tank making up the remaining section of the bottoming cycle. The two-phase post-turbine flow is condensed to saturated liquid in the condenser and stored in the buffer tank. Component framework: [11].

4.3.1 Condenser

The condenser was modeled with a single hot side (process side) volume. It is assumed that the heat is removed with 'perfect' cold side mass flow control. Due to this assumption the condenser saturation pressure is constant. The hot side was modeled with static mass and energy balances as shown in Eq. 4.16.

$$\begin{aligned} m_c &= m_U \\ Q_c &= m_U h_U - m_c h_c \end{aligned} \tag{4.16}$$

In the condenser it was assumed two-phase inlet flow and saturated liquid outlet flow on the hot side. This gives the following relations for the condenser outlet temperature (T_c) and enthalpy (h_c).

$$\begin{aligned} T_c &= T_U^{sat} \\ h_c &= C_p^w (T_c - T_{Ref}) \end{aligned} \tag{4.17}$$

4.3.2 Buffer Tank

To close the loop, a buffer tank was added to the cycle. The saturated liquid is fed into the tank while the feedwater pump removes the needed amount of liquid from the tank. The buffer tank was modeled with a dynamic mass balance as shown in Eq. 4.18.

$$\frac{dM_b}{dt} = m_c - m_p \tag{4.18}$$

Since the saturation temperature in the condenser is the same as the pump fluid temperature, the energy balance over the buffer tank is closed, but not explicitly modeled.

CHAPTER

FIVE

MODEL IMPLEMENTATION

The system of equations contains first order differential equations (ODE) and algebraic equations and is therefore a system of differential-algebraic equations (DAE). The general form of the DAE is shown in Eq. 5.1, where x denotes differential states, z denotes algebraic states and t is time.

$$\begin{aligned} \dot{x} &= f(x, z, t) \\ 0 &= g(x, z, t) \end{aligned} \tag{5.1}$$

The DAE system was implemented in the symbolic framework for numerical optimization and optimal control, CasADi (v.3.6.6) [25]. CasADi supports integration with both Python and Matlab. For this work, Matlab (v.R2023a) was selected as the implementation environment.

5.1 Detecting Phase Change

The working fluid in the OTSG undergoes a phase change. The fluid states are physically different, and are thus governed by different equations. Since the state of a volume changes with time, the model has to be able to detect and change the equations used in the solver during simulation. The vapor quality (β) was used to detect such changes: When allowing β to be any real value ($\mathbb{R}$), it can be interpreted as sub-cooled liquid for $\beta \leq 0$, two-phase fluid for $0 < \beta < 1$ and super-heated steam for $\beta \geq 1$. Note that the limits are carefully chosen for numerical reasons: including $\beta = 0$ in the two-phase region could result in zero pressure. This yields two logical conditions for the points at which the governing equations change ('switching points'):

$$\text{Condition 1: } \beta \geq 1$$
$$\text{Condition 2: } \beta \leq 0$$

CasADi contains functions for evaluating logical conditions containing symbolic variables, and in this case the *if_else* function is particularly useful. The two conditions can be arranged in a nested if-else-statement, as shown in Eq. 5.2, where *Gas*, *Liquid* and *Two-phase* represents the algebraic equation for the given state. The nested statement was passed to the solver for each set of changing equations allowing the solver to switch equation based on the fluid state at each step in time.

$$\text{if\_else}(\text{Condition 1}, \mathit{Gas}, \text{if\_else}(\text{Condition 2}, \mathit{Liquid}, \mathit{Two\text{-}phase})) \tag{5.2}$$

The implementation of detecting and switching equations was developed for the OTSG in Davidsen (2024) [13]. In this work, the logic was extended for changing U as described in Chapter 4.1.2.1, and other process units such as the superheated steam volumes.

5.2 Controller Implementation

The integration terms within the control equations were treated as additional states, $\hat{e}$ and $\hat{e}_T$, which were evaluated by giving the errors, e and e_T, as differential equations to the solver, returning the integrated error as shown Eq. 5.3.

$$\begin{aligned} \hat{e} &= \int e \text{ dt} \Leftrightarrow \frac{d\hat{e}}{dt} = e \\ \hat{e}_T &= \int e_T \text{ dt} \Leftrightarrow \frac{d\hat{e}_T}{dt} = e_T \end{aligned} \tag{5.3}$$

This gives the rewritten version of the PI-equation with anti-windup as shown in Eq. 5.4. u was then evaluated at each timestep and used in differential and algebraic equations to manipulate the system.

$$u = u_0 + K_c e + \frac{K_c}{\tau_I}\hat{e} + \frac{1}{\tau_T}\hat{e}_T \tag{5.4}$$

5.3 Numerical Solver

The DAE was solved from the initial condition over the specified length in time using the IDAS solver from the SUNDIALS library [26]. IDAS is a general purpose solver for DAE initial value problems [27]. For the simulations, the default IDAS solver was used: the variable order, variable step BDF integrator automatically choose the time step which gives smaller steps in regions with rapid dynamics and long steps in smooth regions [27]. The nonlinear equations at each time step were solved using a newton iteration, and the linear systems within were solved using a dense, direct method. The parameters used in the solver are listed in Table 5.3.1.

Table 5.3.1: IDAS solver parameters used in the simulations.

Parameter	Value
reltol	10^{-9}
abstol	10^{-9}
max_num_steps	10000
tf [s]	1
newton_scheme	direct (default)
linear_solver	qr (default)

5.4 Simulator Initialization

The bottoming cycle units, especially turbine and condenser, were sensitive to the initial guess. Therefore, a reasonable initial guess had to be set for successful initialization of the model in order to find a steady-state.

The OTSG model developed in the specialization project was able to initialize with liquid state in all segments. This model was therefore used to provide an approximate initial guess for the OTSG. This state was then loaded with guesses for the remaining system variables for initialization of the bottoming cycle close to the nominal steady-state.

CHAPTER

SIX

CONTROL

This chapter aims to elaborate on the control structure design and objectives. The chapter is divided into two main parts, and will first describe the bottoming cycle controllers and then the operational strategies for power developed on top of the control layer of the bottoming cycle. Dynamic responses used for tuning and example calculations of tuning parameters are given in Appendix B. The tuning of the bottoming cycle controllers are summarized in Table 6.2.1 and additional tuning for the operational strategies are summarized in Table 6.3.1.

6.1 Variable Identification

In the bottoming cycle described in Chapter 3, there are three degrees of freedom: feedwater pump valve (z_m), steam valve (z_v), and condenser cold side mass flow ($m_{cold,in}$). Further, four available controlled variables in the bottoming cycle were considered: steam temperature (T_s), condenser pressure (p_c), steam pressure (p_s), and steam turbine work (W). The flue gas temperature (T_g^0), mass flow (m_g^0) and pump temperature (T_p) were considered the most important disturbances for the bottoming cycle. T_g^0 and m_g^0 were assumed independent variables, but are in a combined cycle configuration correlated as they are influenced by the gas turbine load. An overview of the variable classification is given in Table 6.1.1.

In some cases presented in this work, m_g^0 was assumed to be an extra available degree of freedom for operational strategies on W, this is further described in Chapter 6.3. p_c was assumed to be constant by perfect control of $m_{cold,in}$, as detailed in Chapter 4.3.1, and will therefore not be considered further in control structure design. This leaves T_s and p_s as CVs, and z_v and z_m as MVs available in the bottoming cycle. The following pairings were selected for the remaining variables: z_m-T_s and z_v-p_s. The steam temperature controller was implemented as a cascade controller as developed in Zotică et al. (2022) [11] with a flow controller as inner loop. The chosen pairings and the necessity for flow controller will be further discussed in Chapter 8.2.2 and 8.2.3.

Table 6.1.1: Variable classification showing available MVs, CVs and DVs in the bottoming cycle. $^*m_g^0$ was treated as an extra degree of freedom and not a disturbance in some cases.

MV	CV	DV
z_m	T_s	T_g^0
z_v	p_s	m_g^{0*}
$m_{cold,in}$	p_c	T_p
	W	

6.2 Bottoming Cycle Pairing and Tuning

6.2.1 Flow Controller (C1)

A flow controller was implemented to regulate the feedwater mass flow, and used the pump valve position as manipulated variable. The pressure dynamics of the system are fast and the mass flow is influenced by the pressure drop across the pump valve.

The controller was designed as a pure I-controller, with a closed-loop time constant $\tau_c = 5s$. This resulted in an integral gain of $K_I = 0.0097$. The flow controller was tuned first because of its good effect on the system dynamics and serves as the inner loop of the cascade control structure for temperature controller C3.

6.2.2 Steam Pressure Controller (C2)

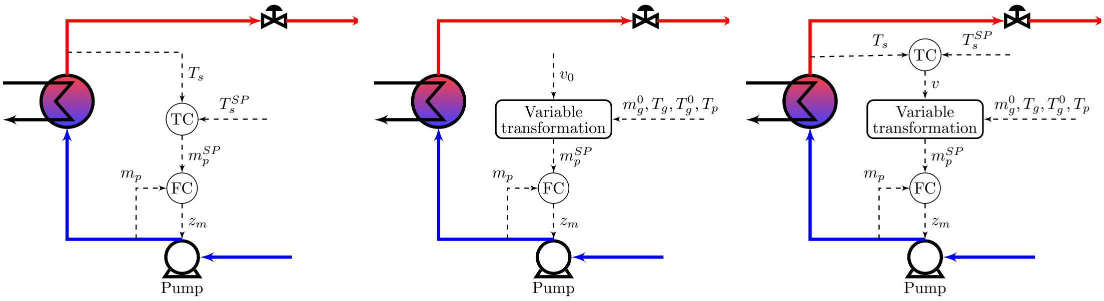
The steam valve was used to control the steam pressure, measured upstream of the valve. Note that the downstream pressure (p_T) cannot be used as this would set the flow rate twice in the bottoming cycle, and therefore be inconsistent [28].

The pressure controller was implemented as a PI-controller with anti-windup. The controller was tuned with the flow controller C1 active, and $\tau_c = 5s$, which gave $K_c = -2.4528$ and $\tau_I = 13s$ as tuning parameters.

6.2.3 Steam Temperature Controller (C3)

To control the steam temperature (T_s), several control structures were evaluated as the outer control loop for the flow controller: A feedback controller, a feedforward controller and a combined feedback and feedforward controller. The control structures are shown in Figure 6.2.1.

The controllers were tuned separately for two operating modes: constant pressure mode, where the steam pressure controller C2 was active, and floating pressure mode, where it was inactive.



(a) Feedback **(b)** Feedforward **(c)** Feedback and feedforward

Figure 6.2.1: Control structures showing the three implementations for controller C3. Component framework: [11].

6.2.3.1 Feedback Controller (C3.1)

The feedback controller regulate T_s by manipulating the setpoint for the flow controller (m_p^{SP}). The controller implementation is shown in Figure 6.2.1a. It was designed as a PI-controller and with a closed-loop time constant of 100s due to system dynamics and sufficient timescale separation for the inner loop. The resulting tuning parameters are as follows: $K_c = -0.03036$ and $\tau_I = 236s$ for floating pressure and $K_c = -0.02718$ and $\tau_I = 199s$ for constant pressure.

6.2.3.2 Combined Feedforward and Feedback (C3.2)

A nonlinear feedforward and feedback controller for control of steam bottoming cycles was developed in Zotică et al.(2022) [11]. It utilized the steady-state energy balance over the OTSG for feedforward disturbance rejection. The controlled variable was steam enthalpy (h_s), and the manipulated variable was m_p^{SP}. In the paper, h_s depended on both T_s and p_s, but since the enthalpy is assumed to be independent of pressure in this work, it is more convenient to control T_s directly.

The non-linear input and output transformations were therefore derived from the OTSG energy balance for T_s instead of h_s and are shown in Eq. 6.1. v_0 is the static transformed input equal to the nominal steam temperature, v is the transformed controller output, and u is equal to m_p^{SP}. The combined controller is shown in Figure 6.2.1c, and a detailed derivation of the equations is provided in Appendix D.

$$\begin{aligned}
e &= T_s^{SP} - T_s \\
v_0 = T_s &= T_p - \frac{\Delta H_{vap}(T_p)}{C_p^s} + \frac{m_g^0 C_p^g}{m_p C_p^s}(T_g^0 - T_g) \\
v &= v_0 + K_c e + \frac{K_c}{\tau_I}\int_0^t e \ \mathrm{dt} \\
u = m_p^{SP} &= \frac{m_g^0 C_p^g (T_g^0 - T_g)}{\Delta H_{vap}(T_p) + C_p^s(v - T_p)}
\end{aligned} \tag{6.1}$$

The feedback controller was tuned as a PI-controller based on a step in the transformed variable v, and with a closed-loop time constant of 100s. This resulted in $K_c = 4.0069$ and $\tau_I = 790s$ for floating pressure, and $K_c = 3.6902$ and $\tau_I = 730s$ for constant pressure mode.

6.2.3.3 Feedforward Controller (C3.3)

The feedforward controller has essentially the same implementation as C3.2, but the feedback loop was deactivated with $K_c = 0$. The controller was therefore only using the inverted energy balance to change m_p^{SP}. The controller is illustrated in Figure 6.2.1b.

6.2.4 Tuning Summary

The controller tunings and pairings of the bottoming cycle controllers are summarized in Table 6.2.1 and 6.2.2 respectively.

Table 6.2.1: Controller tuning parameters for bottoming cycle controllers.

Controller	τ_c [s]	K_c	τ_I [s]	K_I
C1	5	-	-	0.0097
Floating pressure mode				
C3.1	100	-0.03036	236	-
C3.2	100	4.0069	790	-
C3.3	-	0	-	-
Constant pressure mode				
C2	5	-2.4528	13	-
C3.1	100	-0.02718	199	-
C3.2	100	3.6902	730	-
C3.3	-	0	-	-

Table 6.2.2: Controller pairings for the bottoming cycle controllers.

Controller	MV	CV
C1	z_m	m_p
C2	z_v	p_s
C3	m_p^{SP}	T_s

6.3 System Operational Objectives

Two cases with different operational objectives for the bottoming cycle were considered in this work: 'steam cycle' and 'combined cycle'. In the steam cycle case the system has to satisfy power demand. The objective of the power controller was therefore to perform rapid power control. In combined cycle case, the power demand was handled by the gas turbines. Here, the objective of the steam turbine was therefore maximizing power for higher combined cycle efficiency.

6.3.1 Steam Cycle Case

In the steam cycle case, the objective was to satisfy the power demand (W^{SP}) which could for example be set by a controller measuring the grid frequency. The steam cycle case assumes that m_g^0 was an available MV. Since there are no gas turbines in this configuration, the setup more closely resembles a coal-fired power plant.

To maintain the steam cycle objective, a parallel controller with multiple inputs and a single output was implemented. The parallel controller was developed based on using one MV for fast response and another for long term response, as presented in Zotică et. al (2020) [12]. Two sub-classes were developed for the steam cycle case: one for floating pressure mode and one for constant pressure mode.

In the floating pressure case, the steam valve was used for rapid power control and flue gas mass flow was used for long term control. The steam valve has a fast, but temporary effect on power. The flue gas mass flow has a good steady-state effect on power, but the delay to power is somewhat longer. The control structure used in the floating pressure case is shown in Figure 6.3.1.

In the constant pressure case, the fast controller did not act directly on the valve. Instead, it adjusts the setpoint of the steam pressure controller (p_s^{SP}). The long term controller manipulate the flue gas flow as for the floating pressure case. The control structure for the constant pressure case is shown in Figure 6.3.2.

It is important that only one of the parallel controllers have integral action [19]. For both cases, the long term controller was tuned as a PI-controller, while the fast controller was tuned as a P-controller. The P-controller has a bias u_0, and resets u to u_0 at steady-state. Additionally, the cases differ in timescale separation.

In the constant pressure case the P-controller was tuned with $\tau_c = 20s$, and the PI-controller with $\tau_c = 80s$. While in the floating pressure case, the P-controller was tuned with $\tau_c = 5s$ and PI-controller with $\tau_c = 40s$. The controller tunings are given in Table 6.3.1.

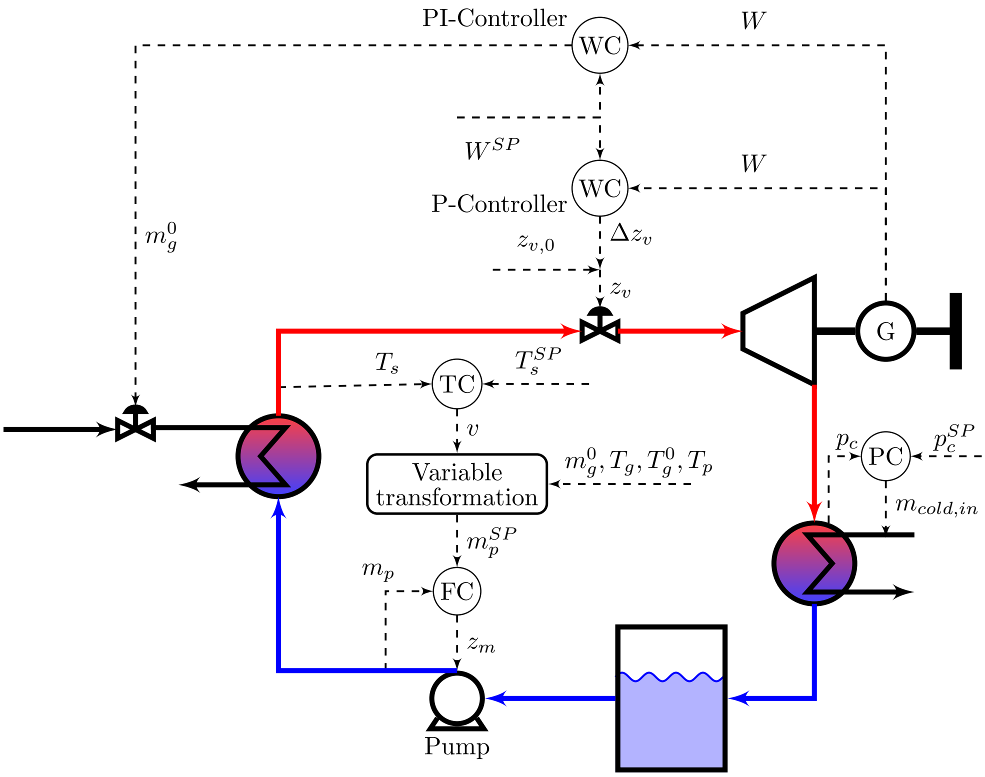


Figure 6.3.1: Control structure for controllers used in the floating pressure case, showing the bottoming cycle controllers and the parallel controller for power. Component framework: [11].

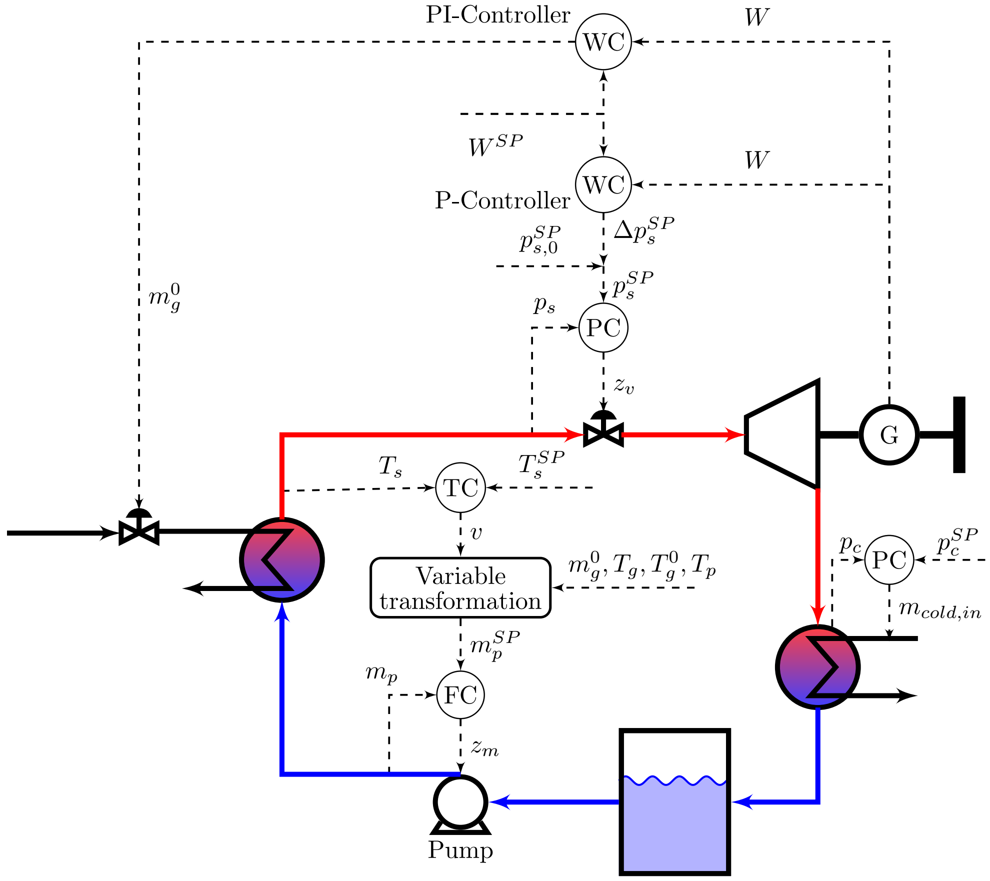


Figure 6.3.2: Control structure for controllers used in the constant pressure case, showing the bottoming cycle controllers and the parallel controller for power. Component framework: [11].

6.3.2 Combined Cycle Case

In the combined cycle case, the flue gas mass flow was treated as a disturbance and therefore not an available MV. The objective of this case was to maximize power, and can be achieved by letting p_s vary. However, constant pressure operation may improve response in steam temperature for large and rapid flue gas changes [6].

A cascade control structure for stabilizing control with resetting of one MV as in Storkaas and Skogestad [29] was made. The structure use controller C2 for constant pressure operation in dynamic phases and a valve position controller (VPC) to minimize steady-state throttling losses. The steam valve position was taken back to the nominal position ($z_{v,0}$) by the VPC. This is done by changing the steam pressure setpoint. The VPC was tuned as a PI-controller with anti-windup and $\tau_c = 200s$ for slowly resetting valve position.

Additionally, the steam pressure has to be kept above a minimum pressure (p_s^{min}) to avoid turbine trip and below a maximum pressure (p_s^{max}) to avoid high mechanical stress. The pressure constraints were set to $p_s^{SP} \pm 1$ bar to show the dynamics of activating these additional pressure constraints. The combined cycle control structure is shown in Figure 6.3.3.

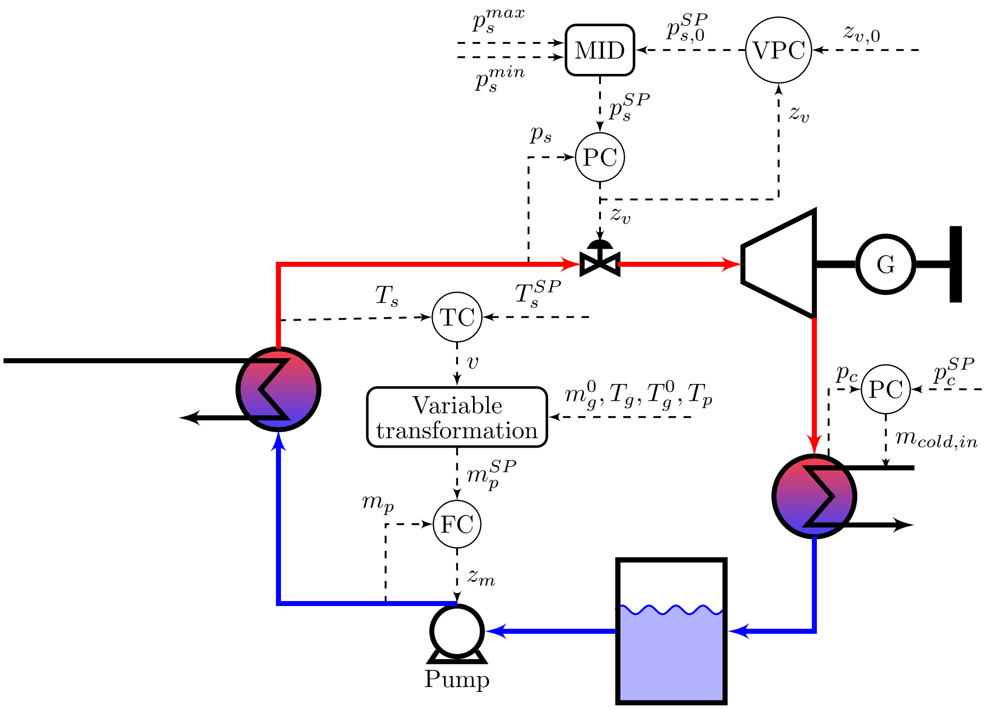


Figure 6.3.3: Control structure used in the combined cycle case, showing the bottoming cycle controllers and the VPC controller for resetting the valve opening. Component framework: [11].

6.3.3 Tuning Summary

The controller tunings for additional controllers for operational objectives on power are summarized in Table 6.3.1.

Table 6.3.1: Additional controller tuning parameters for the operational objective cases.

Case	τ_c [s]	K_c	τ_I [s]	K_p
Steam cycle floating pressure				
P-controller	5			-0.0005
PI-controller	40	-0.00116	3	
Steam cycle constant pressure				
P-controller	20			0.00066
PI-controller	80	-0.00051	3	
Combined cycle				
Valve position controller	200	-0.0047	1	

CHAPTER

SEVEN

OTSG SIMULATIONS

The OTSG simulations were carried out before the remaining process units were added to the cycle. To run the simulations, the model made in the specialization project Davidsen (2024) [13] was used. These simulations are given in this chapter.

7.1 Pump Equation

There are several ways to model a pump, and a few options were evaluated in this work. One could for example have a fixed pump pressure (p_p) and mass flow given with pressure driven flow, or fixed mass flow. These options are shown in Eq. 7.1.

$$\begin{aligned} m_p &= k_{OTSG}(p_p - p_1) \qquad &\text{(Pressure equation)} \\ m_p &= m_p^0 \qquad &\text{(Mass flow equation)} \end{aligned} \tag{7.1}$$

The effect of different pump equations were compared for OTSG outlet temperature (T_{37}) for a -10K step change in flue gas temperature (T_g^0) as shown in Figure 7.1.1. The figure shows that the pressure driven flow gave oscillations, while the fixed mass flow (m_p^0) had no oscillations. One can clearly see that the controllability of the system greatly improves with fixed mass flow.

An intermediate effect can be achieved for a fixed pump pressure with a valve after the pump with floating valve position. This implementation is shown in Eq. 4.5 in Chapter 4.1.2.2. This option is a trade-off for the options given above as the mass flow can be controlled, but it has a fixed pressure boundary. The performance will depend on how tight the controller is tuned. For example: tight tuning will give dynamics similar to mass flow equation, while slow controller will give dynamics similar to the pressure equation. A valve was therefore implemented after the pump for mass flow control.

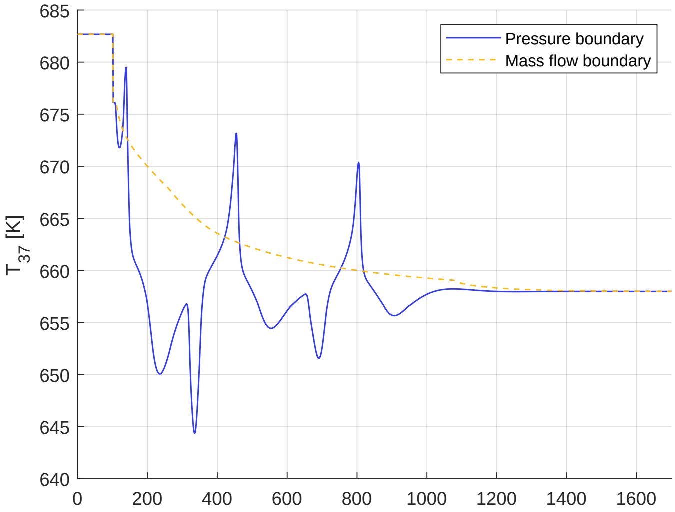


Figure 7.1.1: Comparison of OTSG outlet temperature (T_{37}) for pressure and mass flow pump equation to a -10K step in T_g^0.

CHAPTER

EIGHT

BOTTOMING CYCLE SIMULATIONS

This chapter contains the simulations done on the bottoming cycle model presented in Chapter 4 and control structures described in Chapter 6. The code used in the simulations is given in Appendix E.

8.1 Simulation Conditions

The constants and fluid properties used in the bottoming cycle simulations are listed in Appendix A.3. The nominal boundary conditions, obtained from Montañés et al. (2024) [6], are also provided in Appendix A.3. While the referenced study considered two OTSGs and GTs operating in parallel, this work focuses on a single OTSG and GT. Consequently, the mass flow rates were halved. The flue gas variables correspond to the gas turbine operating at 90% load. The degree of freedom specifications for the simulations are summarized in Appendix C.

8.2 Open-loop Simulations

The open simulations were performed without any active controllers. The nominal steady-state of selected system variables and specifications are given in Table 8.2.1. The nominal steady-state mass flow (m) was close to the design value of 10.95 kg/s. The deviation is due to rounding applied during the estimation of ϕ_d, which affects the mass flow through the turbine.

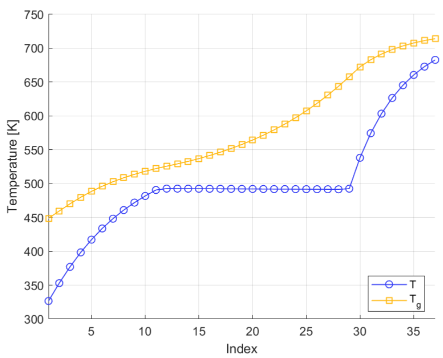
The superheated steam temperature (T_s) and pressure (p_s) were approximately 682.8 K and 23 bar. The steam temperature was approximately 50 K higher, while the flue gas outlet temperature was 6 K higher than that reported in Montañés et al. (2024) [6]. Since both temperatures are higher, this difference may arise from the assumption of constant heat capacities and the estimations of them.

At the nominal steady-state, the liquid fraction in the turbine outlet (β_U) was ca. 13%. This is close to the limit compared to the recommended value as described in Chapter 3.1. The liquid fraction can be reduced by for example running the system with a higher condenser pressure which would make the turbine less efficient, or increasing the degree of overheating by running the gas turbine at a higher load.

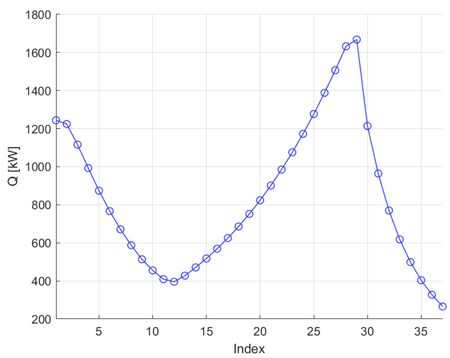
The temperature and heat transfer profiles of the OTSG are shown in Figure 8.2.1. The figure shows that the heat transfer was highest around the switching point from two-phase to steam. The temperature profile shows that the cold side fluid was liquid up to segment 12, two-phase up to segment 30 and gas for the remaining segments.

Table 8.2.1: Nominal steady-state for selected system variables. *Degree of freedom specifications

Variable	Value
m	10.9461 kg/s
T_g	448.76 K
T_s	682.83 K
p_s	23.002 bar
p_T	22.002 bar
P	11523 kW
T_U	201.48 K
β_U	0.8726
p_c	0.0358 bar
T_c	300.12 K
z_m	0.5*
z_v	0.9*
m_g^0	112.75 kg/s*



(a) T and T_g profile



(b) Q profile

Figure 8.2.1: Nominal steady-state temperature and heat transfer along the length of the OTSG.

8.2.1 Disturbance Rejection

Step changes in the main disturbances were applied to the system to show the dynamics of the open-loop system. The disturbances and their magnitudes are given in Table 8.2.2. All steps were applied at t = 200s.

Table 8.2.2: Magnitude of applied step changes for open-loop disturbances.

Disturbance Variable	Step size
T_g^0	±10 K
m_g^0	$\pm10\%$
T_p	±10 K

8.2.1.1 Step in m_g^0

This section shows the system response to a negative step in m_g^0. The applied step change in m_g^0 is shown in Figure 8.2.2.

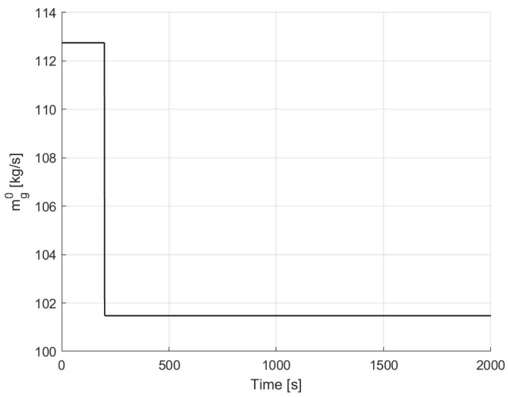
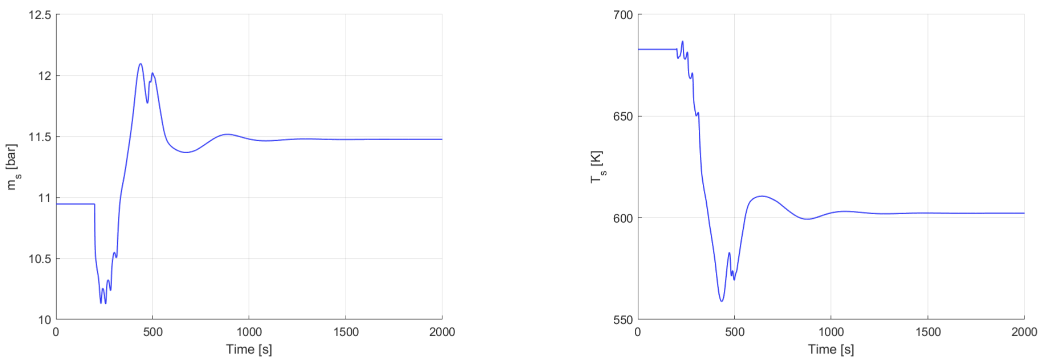


Figure 8.2.2: Negative step change in m_g^0 applied on the open-loop system.

Figure 8.2.3 shows the response in m_s and T_s. The figure shows that when the flue gas mass flow was reduced, there was a delay of approximately 60-70s in the temperature response. This delay was caused by a temporary reduction in m_s due to segment switching which increased the liquid holdup in the OTSG. The reduced mass flow kept the temperature high even though the flue gas mass flow was reduced, since the heat was distributed over a smaller mass flow. Once the change in holdup was complete, the increase in m_s resulted in a drop in steam temperature, as the same amount of heat was then distributed over a larger mass flow. The dynamics associated with OTSG segment switching will be examined in more detail in the following section.

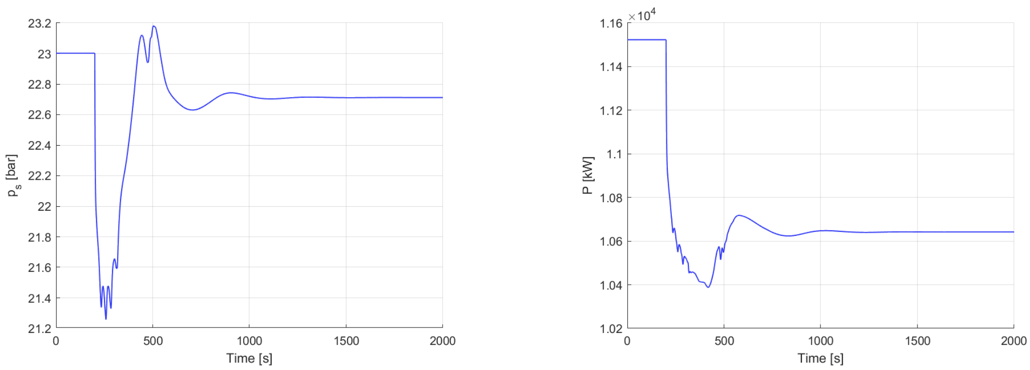


(a) m_s **(b)** T_s

Figure 8.2.3: Open-loop response in m_s and T_s for a negative step in m_g^0.

Figure 8.2.4 shows the response in p_s and power (P). Initially, the pressure dynamics were rapid compared to the temperature dynamics, and the dynamics of m_s and p_s were closely related. Since temperature and pressure are inherently linked via the ideal gas law, a disturbance that reduces temperature also results in a reduction in pressure. Because the pump pressure is constant, a decrease in OTSG pressure increases the pressure drop across the pump valve for fixed valve opening, which in turn increases the mass flow through the system. This increased mass flow amplify the temperature drop: greater flow on the cold side and reduced flow on the hot side both contribute to a decrease in superheated steam temperature.

Further, Figure 8.2.4 shows that the response in power was fast. This is beneficial as a power controller can be fast even though other dynamics such as temperature are slow.



(a) p_s **(b)** P

Figure 8.2.4: Open-loop response in p_s and P for the applied step change in m_g^0

8.2.1.2 Step in T_g^0

This section shows the open-loop response for a positive step in T_g^0. The applied step is shown in Figure 8.2.5.

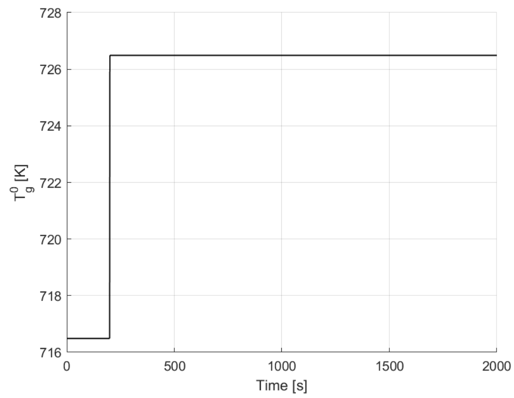
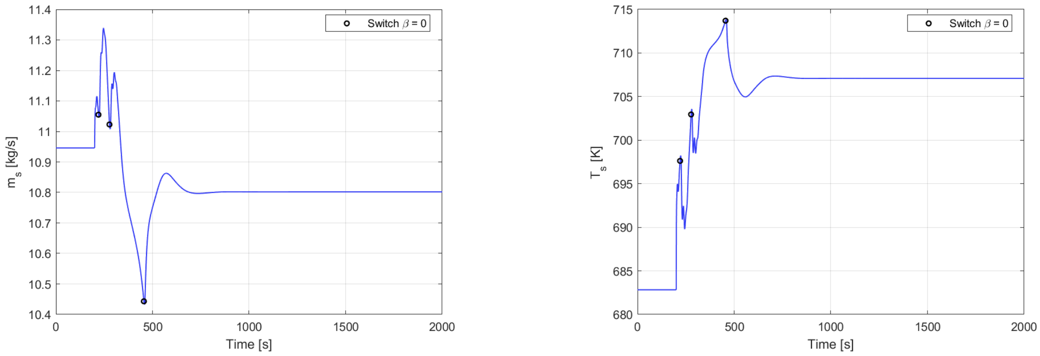


Figure 8.2.5: Positive step change in T_g^0 applied on the open-loop system.

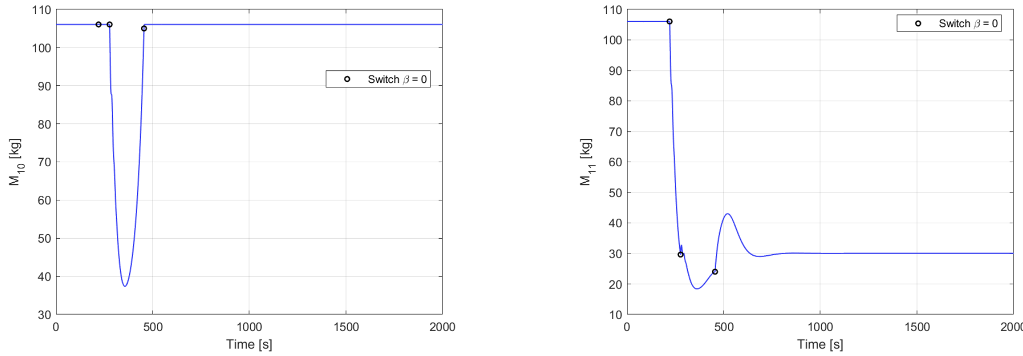
Figure 8.2.6 shows the response of m_s and T_s. As for the negative step in m_g^0, the system transition was dominated by switching related dynamics. To show this more clearly, the liquid switching events were added to the figures. There were three liquid switching events for this disturbance: the first two events were segment 10 and 11 switching from liquid to two-phase, while the third event was segment 10 switching back from two-phase to liquid. Since steam has a much lower density than liquid water, the mass holdup in a segment decreases significantly when steam is present in the segment. The change of state can be seen from the changes in holdup in Figure 8.2.7. Comparing Figure 8.2.6 and Figure 8.2.7 shows that m_s increased during holdup reduction of segment 10 and 11, and was reduced for the consecutive holdup increase of segment 10 prior to the third switching event.

Further, the figures shows that the noisy transition in T_s was caused by the switching from liquid to two-phase of segment 10 and 11. The figure also shows that the increase in temperature around the third event was caused by the mass flow reduction.



(a) m_s **(b)** T_s

Figure 8.2.6: Open-loop response in m_s and T_s, with marked liquid switching events, for the +10K step change in T_g^0.



(a) M_{10} **(b)** M_{11}

Figure 8.2.7: Open-loop response in segment holdup M_{10} and M_{11}, with marked liquid switching events, for the +10K step change in T_g^0.

8.2.1.3 Step in T_p

The applied positive step change in cold side inlet temperature (T_p) is shown in Figure 8.2.8.

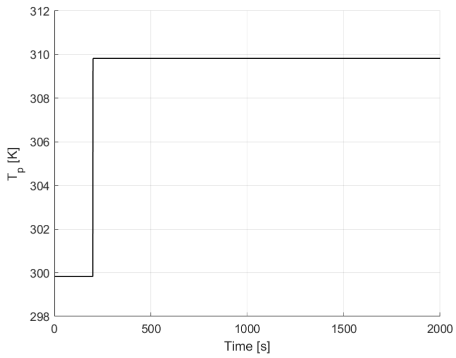
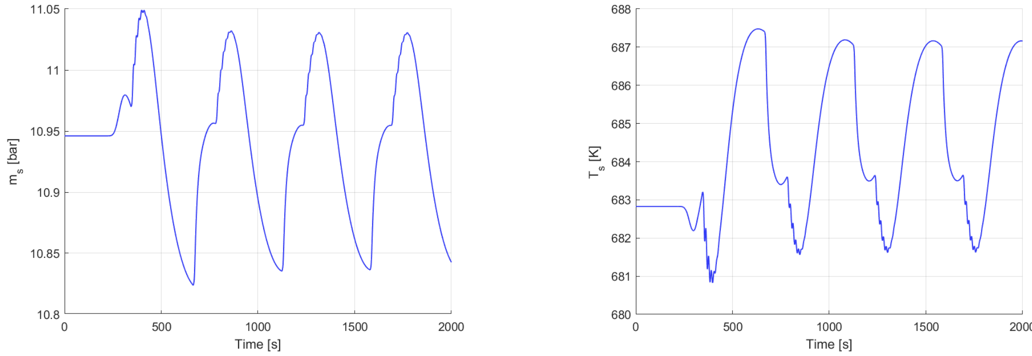


Figure 8.2.8: Positive step change in T_p applied on the open-loop system.

The response of m_s and T_s to the positive step in T_p is shown in Figure 8.2.9. The figure shows that the system was unable to find a stable steady-state and has sustained oscillations over the simulated time. The oscillations were observed throughout the system. This phenomena is not unique to this disturbance, and will be further investigated in the following section.



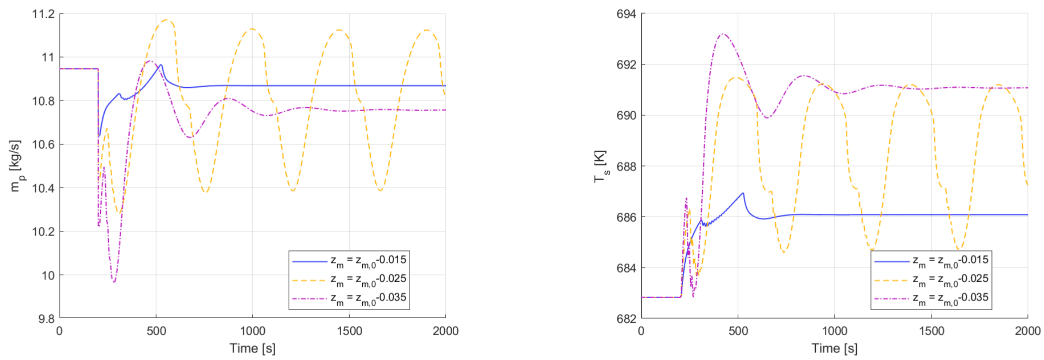
(a) m_s **(b)** T_s

Figure 8.2.9: Sustained oscillations in m_s and T_s as a result of the positive step change in T_p.

8.2.2 Sustained Oscillations

To investigate the sustained oscillations and the need for a stabilizing controller in the system, a series of step changes were applied in pump valve position (z_m).

Figure 8.2.10 shows the change in pump mass flow (m_p) and T_s for different steps in z_m. The figure shows that the dynamic response varied a lot for different step changes. The most noticeable response was the sustained oscillations for -2.5% change in valve position, while -1.5% and -3.5% change in valve position were able to reach a new steady-state. The same behavior was also observed for positive step changes, for example $+5\%$ step change in valve position, suggesting that phenomena occurs for several step changes.

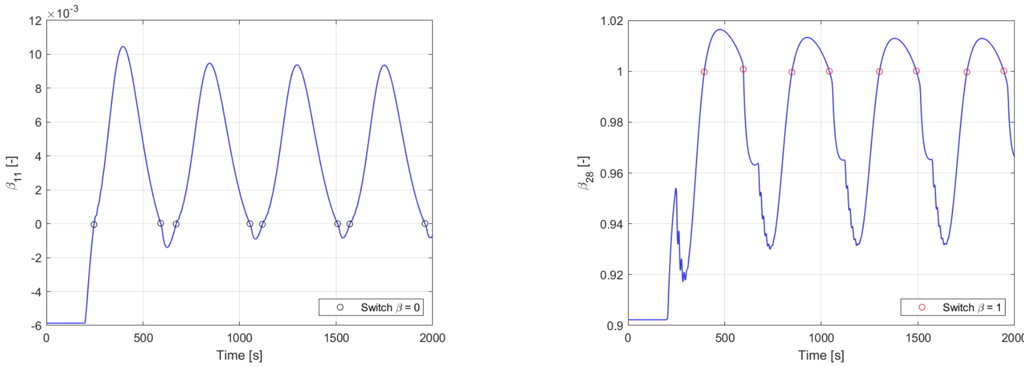


(a) m_p **(b)** T_s

Figure 8.2.10: Response in m_p and T_s for applied step changes in z_m.

The sustained oscillations were reproduced for a -0.5% step change from a steady-state with $z_m = 0.485$ instead of 0.5. This shows that the phenomena is likely not related to the magnitude of the step itself, but rather related to specific states of the system.

One common feature for all cases of sustained oscillations was that the OTSG has a segment with β alternating between positive and negative triggering a liquid switching event, as shown in Figure 8.2.11. It is therefore believed that when the solver tried to find a steady-state with β close to zero on the positive side, the increased mass flow from the switching to two-phase reduced the temperature for β to go just below zero. The switch event from two-phase back to liquid triggered decrease of mass flow, before switching back to two-phase again. In these situations, the oscillations became sustained and not able to find a stable steady-state.



(a) β_{11} (b) β_{28}

Figure 8.2.11: Sustained oscillations in β_{11} and β_{28} with marked switching points for the -2.5% step change in z_m.

Further, the sustained oscillations were not observed in simulations of only the OTSG or turbine, where the pressure was fixed at both boundaries of the unit. This may suggest that it may be related to an OTSG-turbine interaction, where the steam pressure is allowed to vary.

The oscillations were reproduced for systems with 25 and 55 OTSG segments instead of 37. This may indicate that the phenomena is not numeric, but rather related to the model equations and assumptions. The fluid model in this work is simple, for example fluid acceleration was not considered resulting in instantaneous changes in mass flows to changes in pressure. It is therefore believed that the oscillations would not have been observed in a system with a more extensive fluid model.

To conclude, the sustained oscillations were assumed to be caused by the system fluid modeling and an OTSG-turbine interaction, and triggered by switching events. The oscillations were slow with a period around 500 s so any controller should be able to stabilize the system. A solution is keeping the pump valve position floating using a flow controller.

8.2.3 Steady-state RGA

To see whether or not the system interactions changed by implementation of the flow controller (C1), a steady-state RGA analysis was done. Two RGA matrices were made: Case 1 with MV = $[z_m, z_v]$, and Case 2 where C1 was active with MV = $[m_p^{SP}, z_v]$. The CVs were the same for both cases: CV = $[T_s, p_s]$.

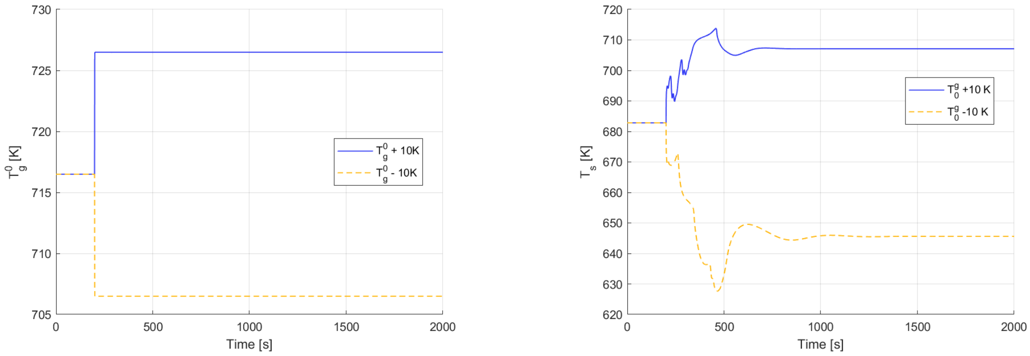
The gain and RGA matrices for both cases are shown in Table 8.2.3. The RGA matrix for Case 1 shows that the off-diagonal pairing (z_m-p_s and z_v-T_s) was preferred based on the RGA pairing rules presented in Chapter 2.3.2. Since none of the RGA pairings were negative, both diagonal and off-diagonal pairing could work well for Case 1. On the other hand, the RGA matrix for Case 2 shows that the diagonal pairing (m_p^{SP}-T_s and z_v-p_s) was the preferred pairing. The introduction of a flow controller in Case 2 made T_s and p_s more decoupled as the diagonal elements are close to the identity, and this was seen as favorable in terms of control of the bottoming cycle.

Table 8.2.3: Gain and RGA matrices for the two cases.

Case 1	**Case 2**
MV = $[z_m, z_v]$	MV = $[m_p^{SP}, z_v]$
$\begin{bmatrix} z_m - T_s & z_m - p_s \\ z_v - T_s & z_v - p_s \end{bmatrix}$	$\begin{bmatrix} m_p^{SP} - T_s & m_p^{SP} - p_s \\ z_v - T_s & z_v - p_s \end{bmatrix}$
$G = \begin{bmatrix} -294.800 & -22.800 \\ 6.870 & -0.300 \end{bmatrix}$	$G = \begin{bmatrix} -52.400 & 2.900 \\ -1.250 & -1.050 \end{bmatrix}$
$RGA = \begin{bmatrix} 0.361 & 0.639 \\ 0.639 & 0.361 \end{bmatrix}$	$RGA = \begin{bmatrix} 0.938 & 0.062 \\ 0.062 & 0.938 \end{bmatrix}$

8.2.4 System Nonlinearities

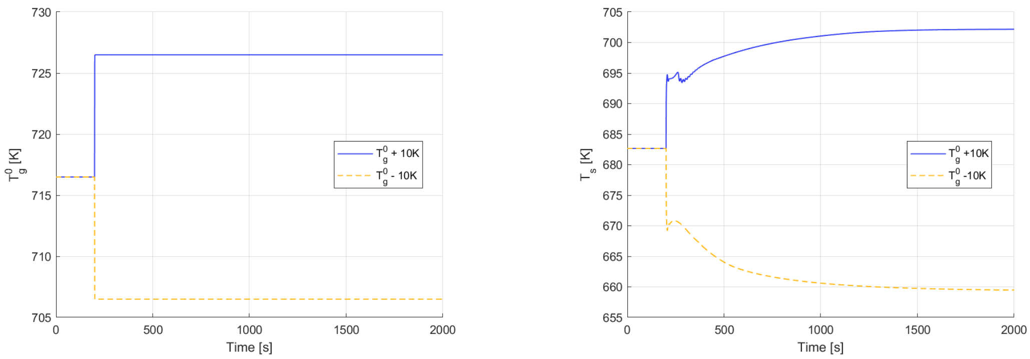
The open-loop disturbance simulations showed that the system response was nonlinear. This can be seen in the response of T_s for the applied step changes in T_g^0 as shown in Figure 8.2.12. The figure shows that the steady-state change in T_s was different for an equal change in T_g^0 in opposite directions. The change in T_s was approximately 50% larger for the negative step, than the positive step in T_g^0. Additionally, the figure shows that the responses were dynamically different. This can be explained by that the switching dynamics are directionally dependent.



(a) T_g^0 (b) T_s

Figure 8.2.12: Step changes in T_g^0 and following nonlinear responses in T_s for the open-loop system.

To see whether or not a flow controller could linearize the system behavior, the same step changes were applied to the system with flow controller C1 active. Figure 8.2.13 shows that with the flow controller active, the change in T_s was approximately 20% larger for the negative step than for the positive step, compared to a 50% difference without the flow controller. Furthermore, the dynamic response was smoother, with less noise and reduced reversed temperature response to switching events. The responses were also more symmetrical in both directions. The reduction in nonlinearity is likely due to the flow controller mitigating the amplifying effect discussed in Chapter 8.2.1.1.



(a) T_g^0 (b) T_s

Figure 8.2.13: Step changes in T_g^0 and following nonlinear responses in T_s with flow controller C1 active.

8.2.5 Summary

The open-loop simulations showed that the system dynamics were heavily influenced by OTSG segment switching events. It also showed that the system response was nonlinear and that the system could end up in sustained oscillations related to the system modeling. Introduction of a flow controller in the system would be preferred to remove oscillations, reduce nonlinear behavior, improve dynamic response and decouple controlled variables.

8.3 Temperature Controller Comparison

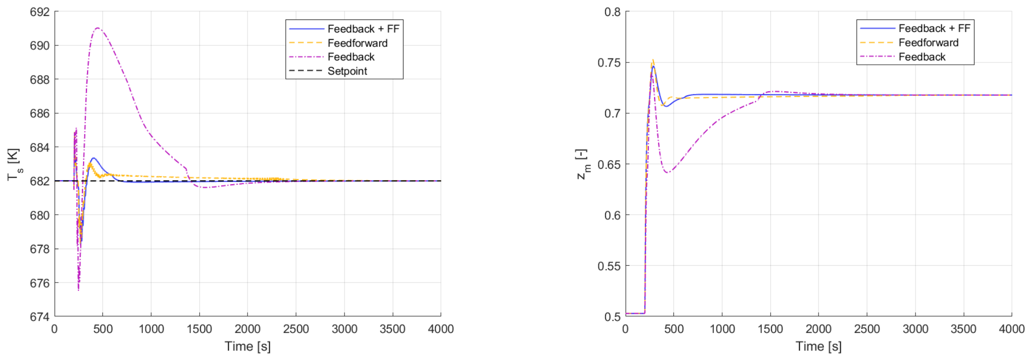
The control structures for the superheated steam temperature controller (C3) described in Chapter 6.2.3 were compared in a disturbance rejection test in floating pressure mode. The simulations were based on a steady-state with $z_{v,0}$, m_g^0 and T_s^{SP} as specifications for the available degrees of freedom. The step changes were applied at t = 200 s and are listed in Table 8.3.1.

Table 8.3.1: Magnitude of applied step changes for temperature controller comparison.

Disturbance Variable	Step size
T_g^0	± 10 K
m_g^0	± 10 %

8.3.1 Step in m_g^0

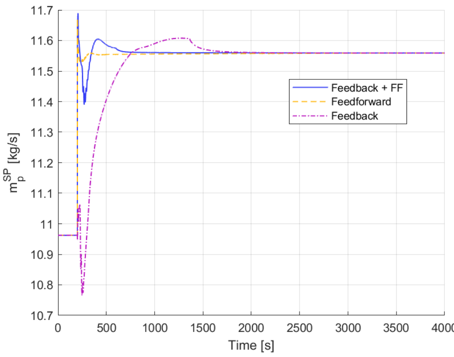
Figure 8.3.1 shows the response in T_s and z_m, while Figure 8.3.2 shows the response in m_p^{SP} and m_p for the suggested control structures for a positive step in m_g^0. The figures shows that the feedback controller has a larger overshoot in T_s compared to the other structures. Further, the feedforward controller has the smallest overshoot which suggests that the derived feedforward model proactively captured the flue gas disturbance. However, it takes around 2000s to remove the offset in temperature with the feedforward implementation. The combined feedback and feedforward controller seems to perform the best with small overshoot and fast removal of offset.



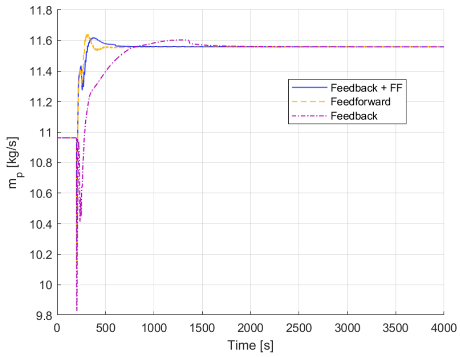
(a) T_s **(b)** z_m

Figure 8.3.1: Response in T_s and z_m for the proposed C3 controllers to a step change in m_g^0.

The same trend can also be seen for the mass flow: the controllers with feedforward were the fastest at finding the new steady-state mass flow. Figure 8.3.2 shows that the mass flows have some deviation from setpoint in the initial dynamic response, which is due to the closed time constant of 5s for the controller, and a tighter tuning would have given more similar response. However, even though tighter tuning is possible in the model it is seen as unrealistic as physical valves need time to actuate.



(a) m_p^{SP}

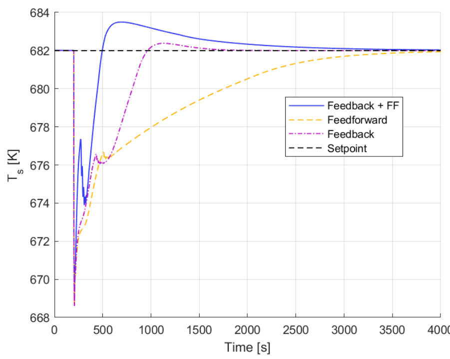


(b) m_p

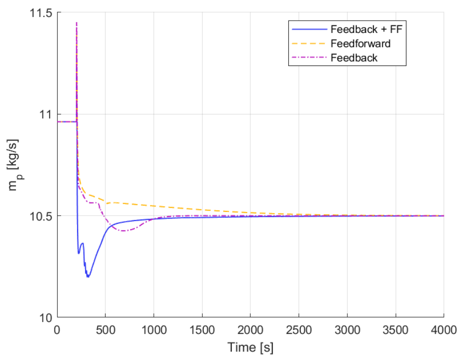
Figure 8.3.2: Response in m_p and m_p^{SP} for the proposed C3 controllers to a step change in m_g^0.

8.3.2 Step in T_g^0

Figure 8.3.3 shows the dynamic response to the negative step in T_g^0. The settling time in T_s for the controllers with feedforward action were longer for this disturbance, compared to the step in m_g^0. The feedback controller performed better here compared to the feedforward controller and was the fastest to reach the new steady-state. The combined feedback and feedforward controller has a larger overshoot than the feedback controller. However, the combined feedback and feedforward controller has the smallest deviation from setpoint.



(a)



(b)

Figure 8.3.3: Response in T_s and m_p for the proposed C3 controllers to a step change in T_g^0.

8.3.3 Summary

Based on the disturbance rejection tests, the combined feedback and feedforward controller had the best performance in terms of least deviation from the setpoint. For the disturbance in m_g^0 the feedforward action give a major advantage, but this was not so clear for the disturbance in T_g^0. The combined feedback and feedforward controller (C3.2) was therefore used as steam temperature controller in the remaining simulations in this work. The combined feedback and feedforward controller was also the preferred control structure in Zotică et al. (2022) [11].

8.4 Case study on Power Control

This section will review the control structures developed for power demand and power maximization described in Chapter 6.3.

8.4.1 Steam Cycle Case

For the steam cycle case, two control structures were compared: floating pressure case and constant pressure case. The control structures are shown in Figure 6.3.1 and 6.3.2 respectively.

8.4.1.1 Steady-state

The two steam cycle cases use different control strategies and have different specifications, which lead to minor differences in steady-state as shown in Table 8.4.1. Variables with identical setpoints, such as T_s and P, have equal steady-state for the two cases. However, differences appear in p_s and z_v: in the floating pressure case, the P-controller resets z_v to its setpoint, while in the constant pressure case it resets p_s. The difference in specifications gave throttling losses for the constant pressure case, as indicated by a larger flue gas flow. Note that m_g^0 was treated as a MV and not a DV for this control strategy, as explained in Chapter 6.3. The larger flue gas flow and equal temperature setpoint in turn gave a larger m_p for the constant pressure case.

Table 8.4.1: Steady-state for selected variables in floating and constant pressure cases. *Nominal controller setpoints/specifications.

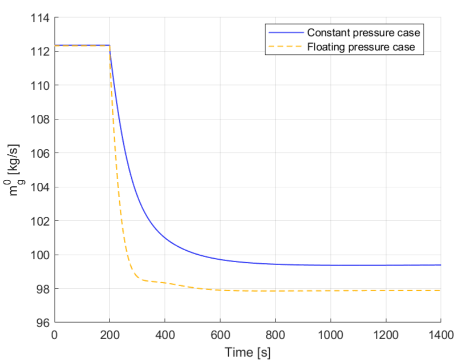
Variable	Floating pressure case	Constant pressure case
m_p	10.93858 kg/s	10.93861 kg/s
z_m	0.497	0.499
z_v	0.900*	0.876
p_s	22.97 bar	23.00 bar*
T_s	682.00 K*	682.00 K*
P	11500 kW*	11500 kW*
m_g^0	112.33 kg/s	112.36 kg/s

8.4.1.2 Changes in Power Demand (W^{SP})

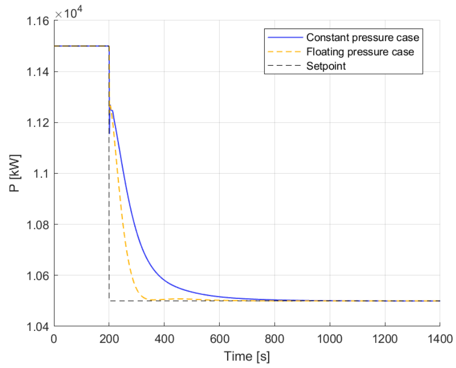
The overall goal of these control structures is to satisfy power demand and therefore rapid changes in power demand has to be handled. This was tested with step changes in power demand of ±1 MW from the steady-states given above.

Setpoint reduction in power

Figure 8.4.1 shows the -1 MW step change in power demand and following response in m_g^0 and P. The figure shows that the floating pressure case has better steady-state efficiency, as less flue gas was needed for same amount of power. The difference in efficiency comes from throttling losses when not resetting the valve position as shown in Figure 8.4.2a, but rather keeping the pressure at its setpoint as shown in Figure 8.4.2b.

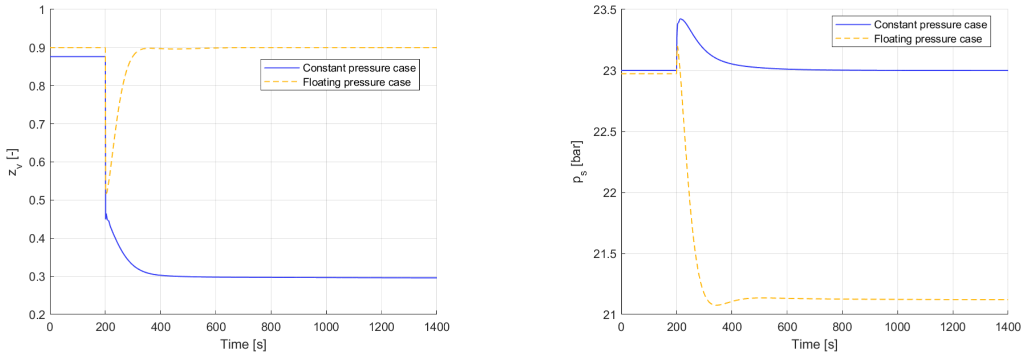


(a) m_g^0



(b) P

Figure 8.4.1: Reduction in power setpoint and following response in m_g^0 and P for the steam cycle control structures.



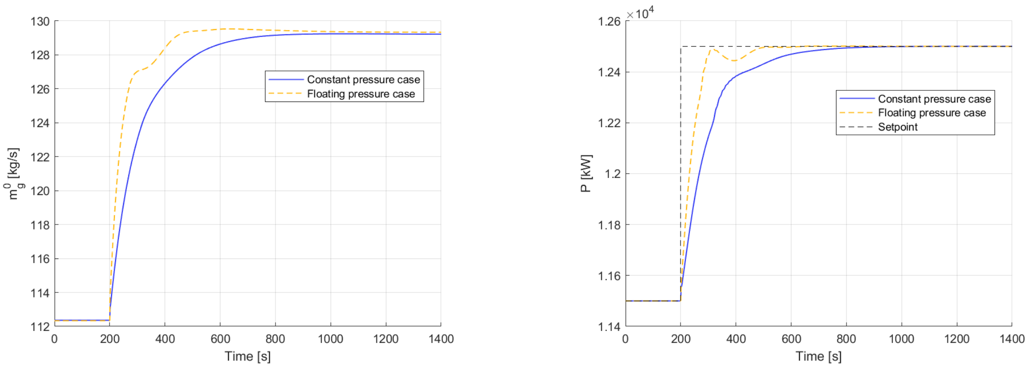
(a) z_v **(b)** p_s

Figure 8.4.2: Response in z_v and p_s to the negative setpoint change in power for the steam cycle control structures.

The floating pressure case was faster at reaching the new power setpoint. This was expected since the floating pressure case was tuned with smaller τ_c than the constant pressure case. The initial response in power was nearly instantaneous for both cases. This was caused by the P-controller acting on the superheated steam valve, either directly or by adjusting the pressure setpoint, as illustrated in Figure 8.4.2a.

Setpoint increase in power

Figure 8.4.3 shows the +1 MW change in power demand and following response in P and m_g^0. As for the negative step, the floating pressure case was faster due to differences in tuning.



(a) **(b)**

Figure 8.4.3: Increase in power setpoint and following response in m_g^0 and P for steam cycle control structures.

Figure 8.4.4 shows the response in z_v and p_s, and shows that the superheated steam valve saturated for both cases. The initial jump in power was small due to the valve saturation. The P-controller in the floating pressure case brought the valve position out of saturation and back to its setpoint when power reached its setpoint. For constant pressure case, the pressure setpoint was unobtainable and the valve position remained saturated. Consequentially, the constant pressure case had better steady-state efficiency.

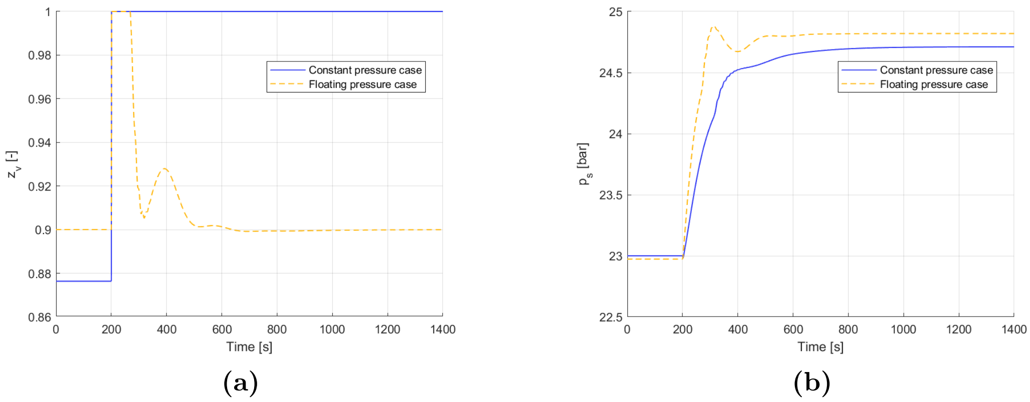


Figure 8.4.4: Response in z_v and p_s to the increase in power setpoint for the steam cycle control structures.

8.4.1.3 Disturbance Rejection

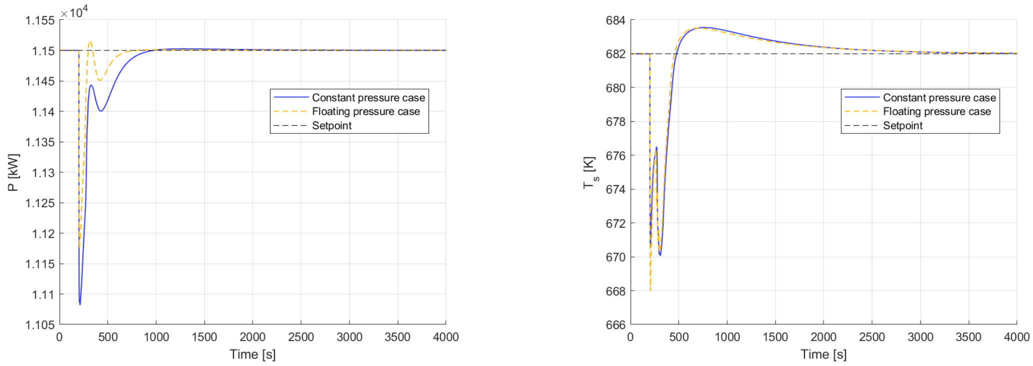
The control structures for the steam cycle case were compared in a disturbance rejection test. The applied step changes are shown in Table 8.4.2.

Table 8.4.2: Step changes in disturbances applied on the steam cycle cases.

Disturbance Variable	Step size
T_p	± 10 K
T_g^0	± 10 K

Step in T_g^0

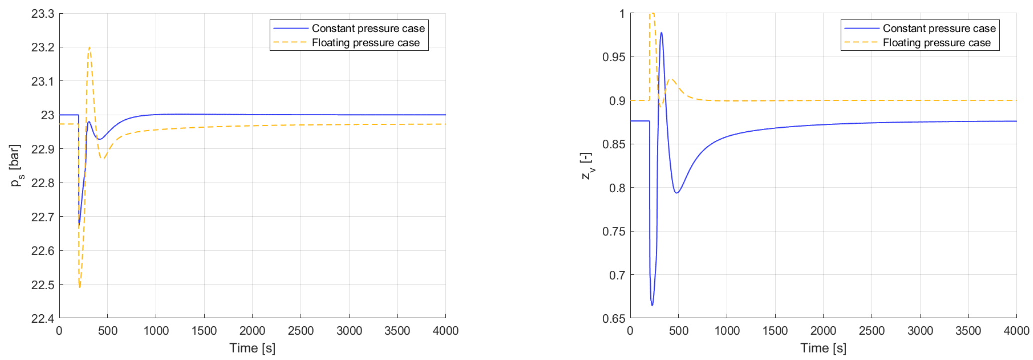
Figure 8.4.5 shows the response in P and T_s to a -10 K step change in T_g^0. The figure shows that the floating pressure case was faster than the constant pressure case at rejecting the disturbance and bringing the power back to its setpoint. The temperature response was more or less the same for both control structures.



(a) P **(b)** T_s

Figure 8.4.5: Response in P and T_s for a -10 K step change in T_g^0 for the steam cycle control structures.

Figure 8.4.6 shows the response in p_s and z_v for the step change in T_g^0. The figure shows that the response for z_v in constant pressure case looks counterintuitive: when there was a drop in power, the pressure controller reduced the steam valve position which throttled the flow, further reducing power. The P-controller probably should have been tuned more aggressively so that p_s^{SP} was reduced more in the initial dynamic response, which would have resulted in a larger valve opening for z_v, giving a smaller spike in power. The difference in valve actuation was likely why the spike in power was larger for the constant pressure case than the floating pressure case.

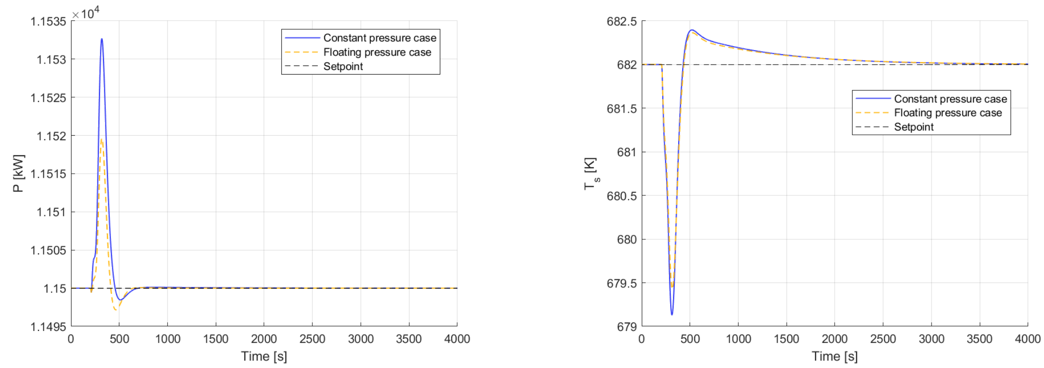


(a) **(b)**

Figure 8.4.6: Response in p_s and z_v for a -10 K step change in T_g^0 for the steam cycle control structures.

Step in T_p

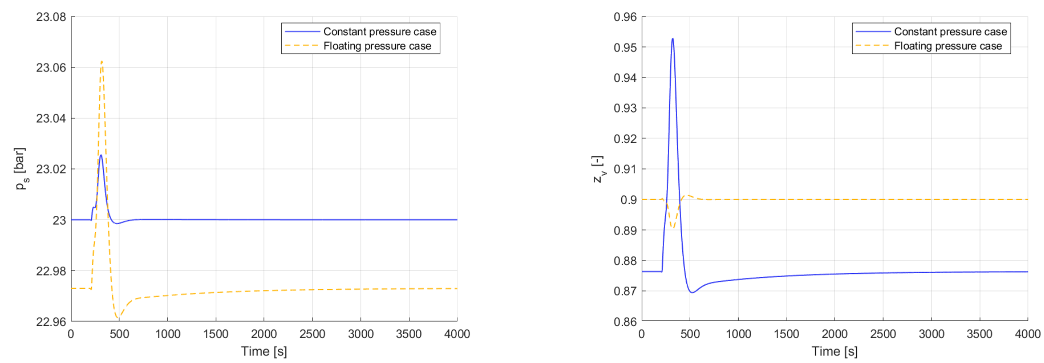
Figure 8.4.7 shows the response in P and T_s for a +10K step change in T_p. The figure shows that the spike in power was larger for constant pressure case as seen for the disturbance in T_g^0.



(a) P **(b)** T_s

Figure 8.4.7: Response in P and T_s for the +10K step change in T_p for the steam cycle control structures.

The response in p_s and z_v are shown in Figure 8.4.8, which shows that the P-controller in floating pressure case reduced the valve opening z_v, thus reducing the spike in power. Further, the P-controller in constant pressure case increased p_s^{SP} to counteract the increase in power. However, like for the step change in T_g^0, the pressure setpoint was not increased enough, which increased the valve position and not throttling the flow.



(a) p_s **(b)** z_v

Figure 8.4.8: Response in p_s and z_v for the +10K step change in T_p for the steam cycle control structures.

8.4.1.4 Summary

The case comparison for steam cycle cases showed that the throttling losses were larger for the constant pressure case and with larger difference at lower loads. The behavior of the steam valve in constant pressure case was counterintuitive, assumed due to P-controller tuning. Further, the simulations showed that the power controllers were able to adjust power according to demand and reject disturbances. However, the power controllers should have been tuned more aggressively for better setpoint tracking. The dynamic response in power for changes in power demand were similar to what was observed for parallel control of a drum based system in Zotică et al. (2020) [12].

8.4.2 Combined Cycle Case

One control structure was developed for the combined cycle case, and is shown in Figure 6.3.3. The objective of the bottoming cycle in the combined cycle case is to maximize power, while dynamically operate at constant pressure for improved steam temperature dynamics.

8.4.2.1 Steady-state

The steady-state for selected system variables for the combined cycle case is given in Table 8.4.3. The degree of freedom specifications for the combined cycle case were: m_g^0, $z_{v,0}$ and T_s^{SP}.

Table 8.4.3: Steady-state of the combined cycle case. *Nominal controller setpoints/specifications.

Variable	Value [unit]
z_m	0.503
m_p	10.9624 kg/s
z_v	0.900*
p_s	23.023 bar
P	11527 kW
T_s	682.00 K*
m_g^0	112.75 kg/s*

8.4.2.2 Dynamic Simulations

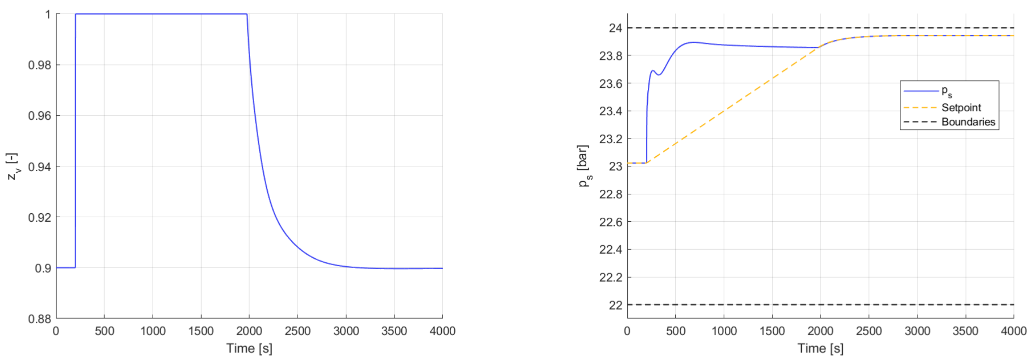
The disturbances and step sizes applied to the open-loop are given in Table 8.4.4 and were applied at t = 200 s.

Table 8.4.4: Applied step changes in the disturbance variables.

Disturbance Variable	Step size
T_g^0	± 10 K
m_g^0	± 10 %
T_p	± 10 K

Step in T_g^0

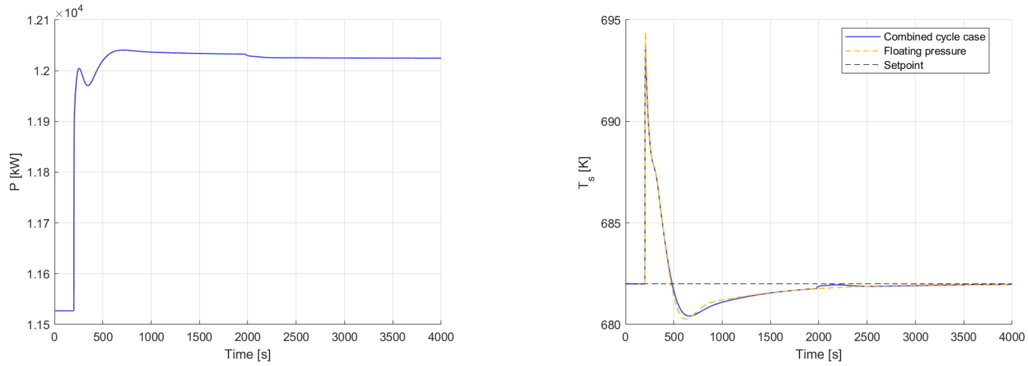
Figure 8.4.9 shows the response in z_v and p_s for the +10K step change in T_g^0. The figure shows that the steam valve saturated at fully open. Consequentially, the pressure controller was not able to maintain its setpoint for p_s. The figure also shows that the VPC slowly shifted the pressure setpoint until the steam valve was no longer saturated (at t = 2000s), and z_v was brought back to its nominal setpoint. The pressure boundaries were not activated for the step change in T_g^0.



(a) z_v (b) p_s

Figure 8.4.9: Response in z_v and p_s to the step change in T_g^0 for the combined cycle case.

Figure 8.4.10 shows the response in P and T_s. The valve saturation maximize power during the transition, and the power output was reduced when the valve position was brought back to its nominal value around t = 2000s. To evaluate whether or not constant steam pressure improved steam temperature dynamics, the temperature response for the same disturbance, but with floating steam pressure was used for comparison. It shows that the temperature response was more or less the same, which is reasonable as the steam pressure was essentially floating in the dynamic phase due to steam valve saturation.



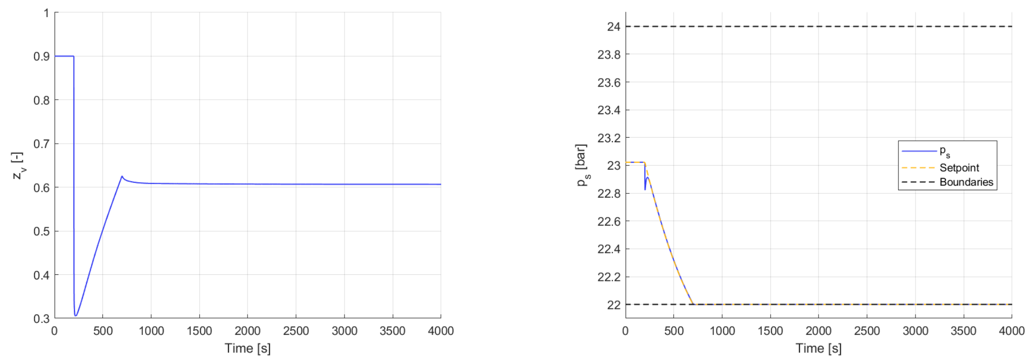
(a) P **(b)** T_s

Figure 8.4.10: Response in P and T_s to the step change in T_g^0 for the combined cycle case.

Step in m_g^0

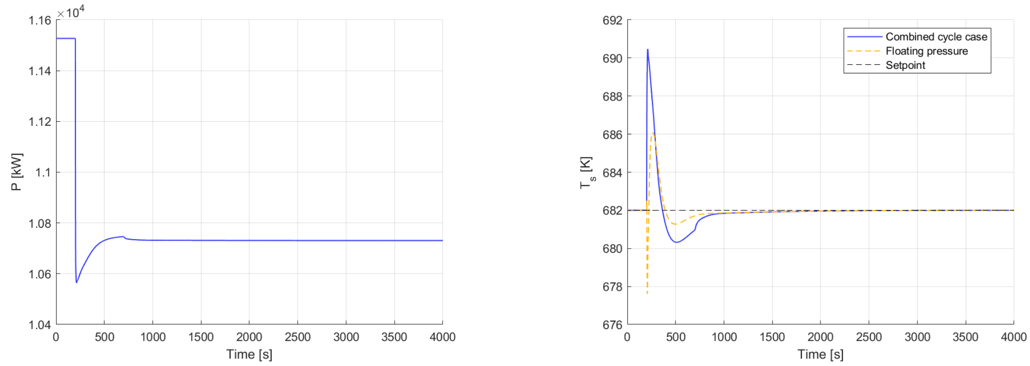
Figure 8.4.11 shows the response in z_v and p_s for a -10% step change in m_g^0. The figure shows that the pressure controller initially has a minor overshoot. The VPC reduced p_s^{SP} to bring the valve back to nominal position, but the minimum pressure constraint became active and the valve position was therefore not brought back to its nominal point.

Figure 8.4.12 shows the response in P and T_s. For this disturbance, the power was not maximized dynamically as maintaining the pressure setpoint increased the throttling losses. The VPC maximized steady-state power, by opening the valve position as much as possible for the active pressure constraint. The temperature response was compared with floating pressure, and shows that the floating pressure controller has smaller deviation from setpoint, but it has an initial negative spike in temperature. Constant pressure operation did not improve the steam temperature dynamic response.



(a) z_v **(b)** p_s

Figure 8.4.11: Response in z_v and p_s for the -10% step change in m_g^0 for the combined cycle case.



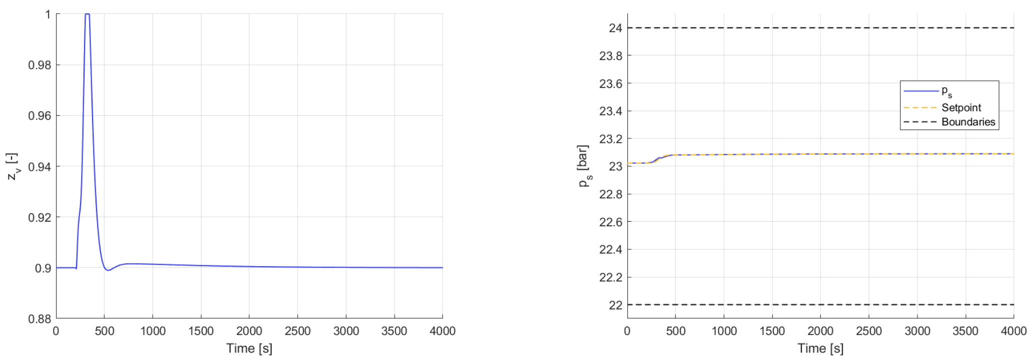
(a) P (b) T_s

Figure 8.4.12: Response in P and T_s to the -10% step change in m_g^0 for the combined cycle case.

Step in T_p

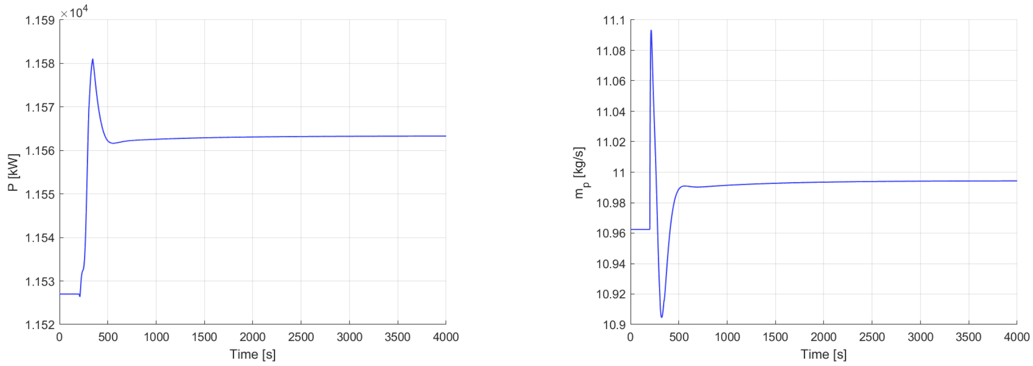
Figure 8.4.13 shows the response in z_v and p_s for the +10K step change in T_p. This is the same disturbance that caused stable oscillations for the open-loop simulations in Chapter 8.2. The figure shows that z_v saturated and was brought back to the setpoint by the VPC.

Figure 8.4.14 shows the response in P and m_p and shows that the steady-state cold side mass flow increased when T_p increased. Since the temperature controller has a fixed setpoint more fluid was needed to remove the heat in the OTSG to maintain the setpoint. The increased mass flow gave increased steady-state power as a result of the disturbance.



(a) (b)

Figure 8.4.13: Response in z_v and p_s for the +10K step change in T_p for combined cycle case.



(a) **(b)**

Figure 8.4.14: Response in P and m_p to the $+10$K step change in T_p for combined cycle case.

8.4.2.3 Summary

The VPC worked well for power maximization at steady-state, also with additional pressure constraints. Constant steam pressure operation did not improve steam temperature dynamics for the applied disturbances in the combined cycle case.

CHAPTER

NINE

DISCUSSION

9.1 Comments on Power Objective Cases

The combined cycle case is relevant for power plants where fuel efficiency is important. The improved steam temperature response from constant pressure operation found in Montañés et al. (2024) [6] was not seen in these simulations. This could for example be related to differences in OTSG temperature modeling such as no pressure dependence on enthalpy and no thermal inertia in OTSG tube wall. Further, it could be related to differences in tuning, either between the studies or between cases presented here. The throttling losses being larger at lower loads were observed as reported in the paper.

The steam cycle case is mostly relevant to biomass or coal-fired power plants where one would bypass surplus flue gas to satisfy the specific demand of power. For such systems, the steam valve should have been designed larger, with the nominal valve opening of 0.5 instead of 0.9, to allow larger movement in upward direction. This is in contrast to the combined cycle case where running with large valve opening is important for efficiency. Additionally, smaller τ_c probably could have been used in the steam cycle cases, as the power response was likely not fast enough to maintain power demand in practice. As for the combined cycle case, the constant pressure did not improve steam temperature dynamics and the cascade on steam pressure control therefore only introduced more delay for the P-controller than necessary.

9.2 Comments on Modeling Assumptions

Simplifying assumptions were applied to the model for control purposes. Some assumptions may have made system dynamics faster than what is realistic. For example the fluid model was simple and did not consider fluid acceleration. This made mass flows change instantly to changes in pressure. Further, assumption of no thermal inertia in the OTSG tube wall and fairly small steam holdups gave fast temperature dynamics with small delay from flue gas disturbances to the superheated steam.

Switching related dynamics were dependent on discretization resolution. More segments would give less holdup in each segment and the change in mass flow to a switching event would have been smaller. However, it would also resulted in more switching events as the overall change in OTSG holdup due to a disturbance would be the same.

Despite the potential model improvements, it is not believed that changing these assumptions would effect the overall approximation of the model and the assumptions are therefore seen as reasonable.

9.3 Comments on Code Organization and Solver

CasADI/IDAS implementation proved very useful for making it possible to dynamically change equations in the solver in this work. However, the run time was fairly long, and the solver parameters and code were not optimized for computational efficiency.

For a 2000 s simulation, the run time was around 1250 s, but the average time spent on solving each time step was approximately 3 ms. Investigations showed that the model spent around 99% of the time building the DAE, which was done at each step length (tf). Building the DAE at each step is unnecessary and major computational improvements can therefore be achieved by restructuring the code so that the DAE is built once and with the applied steps changed directly in the solver using the CasADi input parameter 'p'. Subsequent simulations demonstrated that by restructuring the code, the run time could be reduced dramatically, from approximately 1250 s to just 10 s.

The default newton method was compared to other solver methods such as sparse newton and preconditioned Krylov, which are available through the solver parameters *newton_scheme* and *linear_solver*. Subsequent simulations were without notable improvement in run time, and the default dense/direct newton method therefore proved to efficiently solve the given DAE.

CHAPTER

TEN

CONCLUSIONS

In this thesis, a dynamic model of a steam bottoming cycle was built. The model showed the ability to change equations used in the solver based on the vapor quality, allowing the point of vaporization to move along the OTSG. Boundary conditions and OTSG geometry for offshore implementation were used. Several control structures for steam temperature and pressure control, and operational strategies for power were developed and compared to relevant studies.

The simulations showed nonlinearities and instabilities of the open-loop system. The dynamic response was influenced by segment switching events. Introduction of a flow controller was beneficial for linearization, improved dynamic response and decoupling of controlled variables. Further, the simulations confirmed the advantages seen in relevant studies of a combined feedforward and feedback control, based on a steady-state energy balance over the OTSG.

Control structures with two different operational objectives for power control were made. The combined cycle case used a valve position controller to effectively minimize steady-state throttling losses, however improved steam temperature dynamics from constant steam pressure operation were not observed. The steam cycle case showed throttling losses for constant pressure with larger difference at lower loads. Further, the simulations showed that more aggressive tuning is desirable and that the P-controller tuning could have been improved.

10.1 Further Work

The implementation and code could be subject to major improvements in computational efficiency by simple modifications such as only constructing the DAE once, instead of at each time step.

The complexity of the model could also be enhanced. This includes incorporating a more complex fluid model, accounting for OTSG wall resistance and thermal inertia, and developing a more detailed condenser and turbine model.

In the context of the simulation results, several areas could be explored further. These include a deeper investigation into why improved steam temperature dynamics from constant pressure operation were not observed, improved controller tuning of P-controllers, and an analysis on the dynamic effects of a finer discretization resolution of the OTSG.

REFERENCES

[1] Lene Skyrudsmoen and Berit Storbråten. *Laveste klimagassutslipp siden 1990*. Statistics Norway. Accessed 16.01.2025. 2024. URL: https://www.ssb.no/natur-og-miljo/forurensning-og-klima/statistikk/utslipp-til-luft/artikler/laveste-klimagassutslipp-siden-1990.

[2] Norwegian petroleum (Ministry of Energy). *Emissions to air*. Accessed 16.01.2025. 2024. URL: https://www.norskpetroleum.no/en/environment-and-technology/emissions-to-air/.

[3] Norwegian Ministry of Climate and Environment. *Norway's Climate Action Plan for 2021–2030*. Accessed 22.05.2025. URL: https://www.regjeringen.no/contentassets/a78ecf5ad2344fa5ae4a394412ef8975/en-gb/pdfs/stm202020210013000engpdfs.pdf.

[4] European Commission. *EU Emissions Trading System (EU ETS)*. No date, accessed: 2025-01-17. URL: https://climate.ec.europa.eu/eu-action/eu-emissions-trading-system-eu-ets_en.

[5] Geir Molnes. *Åtte spørsmål og svar om elektrifisering av sokkelen*. Faktisk. Accessed 16.01.2025. 2022. URL: https://www.faktisk.no/artikkel/atte-sporsmal-og-svar-om-elektrifisering-av-sokkelen/112801.

[6] Rubén M. Montañés, Cristina Zotică, and Adriana Reyes-Lúa. "Operation and control of compact offshore combined cycles for power generation". In: *Energy* 290 (2024), p. 130315. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2024.130315. URL: https://www.sciencedirect.com/science/article/pii/S0360544224000860.

[7] Offshore. *Combined cycle plant to power Snorre production platforms*. 2000. URL: https://www.offshore-mag.com/home/article/16763315/combined-cycle-plant-to-power-snorre-production-platforms?utm_source=chatgpt.com.

[8] SINTEF Energy Research. *Efficiency enhancement of gas turbines*. LowEmission project website: Accessed 07.05.2025. URL: https://lowemission.no/research/efficiency-enhancement-of-gas-turbines-sp1.

[9] Rubén M. Montañés et al. "Compact Steam Bottoming Cycles: Minimum Weight Design Optimization and Transient Response of Once-Through Steam Generators". In: *Frontiers in Energy Research* 9 (2021). DOI: https://doi.org/10.3389/fenrg.2021.687248. URL: https://www.frontiersin.org/journals/energy-research/articles/10.3389/fenrg.2021.687248/full.

[10] Marit J. Mazzetti et al. "Achieving 50% weight reduction of offshore steam bottoming cycles". In: *Energy* 230 (2021), p. 120634. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2021.120634. URL: https://www.sciencedirect.com/science/article/pii/S0360544221008835.

[11] Cristina Zotică et al. "Control of steam bottoming cycles using nonlinear input and output transformations for feedforward disturbance rejection". In: *IFAC-PapersOnLine* 55.7 (2022). 13th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems DYCOPS 2022, pp. 969–974. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2022.07.570. URL: https://www.sciencedirect.com/science/article/pii/S2405896322009764.

[12] Cristina Zotică et al. "Optimal operation and control of heat to power cycles: A new perspective from a systematic plantwide control approach". In: *Computers & Chemical Engineering* 141 (2020), p. 106995. ISSN: 0098-1354. DOI: https://doi.org/10.1016/j.compchemeng.2020.106995. URL: https://www.sciencedirect.com/science/article/pii/S0098135419311342.

[13] William Davidsen. *Modeling of once-through steam generators*. Specialization project. 2024.

[14] Erik Salazar-Herran et al. "Flexible dynamic model of PHEX for transient simulations in Matlab/Simulink using finite control volume method". In: *International Journal of Refrigeration* 110 (2020), pp. 83–94. ISSN: 0140-7007. DOI: https://doi.org/10.1016/j.ijrefrig.2019.11.003. URL: https://www.sciencedirect.com/science/article/pii/S0140700719304682.

[15] F. Moukalled, L. Mangani, and M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab®*. Vol. 113. Fluid Mechanics and Its Applications. Cham: Springer, 2016. ISBN: 978-3-319-16873-9. DOI: 10.1007/978-3-319-16874-6. URL: https://link.springer.com/book/10.1007%2F978-3-319-16874-6.

[16] Suhas V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Washington, D.C.: Hemisphere Publishing Corporation, 1980. ISBN: 978-0891165224. URL: https://catatanstudi.wordpress.com/wp-content/uploads/2010/02/numerical-heat-transfer-and-fluid-flow.pdf.

[17] Bin Xu, Adamu Yebi, and Zoran Filipi. "Modeling for Organic Rankine Cycle Waste Heat Recovery System Development". In: *Organic Rankine Cycle Technology for Heat Recovery*. Ed. by Enhua Wang. Rijeka: IntechOpen, 2018. Chap. 4. DOI: 10.5772/intechopen.78997. URL: https://doi.org/10.5772/intechopen.78997.

[18] Dale E Seborg. *Process dynamics and control*. eng. Hoboken, N.J, 2004.

[19] Sigurd Skogestad. “Advanced control using decomposition and simple elements”. In: *Annual Reviews in Control* 56 (2023), p. 100903. ISSN: 1367-5788. DOI: https://doi.org/10.1016/j.arcontrol.2023.100903. URL: https://www.sciencedirect.com/science/article/pii/S1367578823000676.

[20] Cheng-Ching Yu. *Autotuning of PID Controllers: Relay Feedback Approach*. eng. 2nd Edition. London: Springer Verlag London Limited, 2006. ISBN: 1846280362.

[21] Sigurd Skogestad. *Prosessteknikk : masse- og energibalanser*. nob. Trondheim, 2009.

[22] Sigurd Skogestad. “Simple analytic rules for model reduction and PID controller tuning”. In: *Journal of Process Control* 13.4 (2003), pp. 291–309. ISSN: 0959-1524. DOI: https://doi.org/10.1016/S0959-1524(02)00062-8. URL: https://www.sciencedirect.com/science/article/pii/S0959152402000628.

[23] Sigurd Skogestad. *Multivariable feedback control : analysis and design*. eng. Chichester, 2005.

[24] Rachael L. Baumann and John L. Falconer. *Pressure-Enthalpy Diagram for Water*. Wolfram Demonstrations Project. 2015. URL: https://demonstrations.wolfram.com/PressureEnthalpyDiagramForWater/.

[25] Joel A E Andersson et al. “CasADi – A software framework for nonlinear optimization and optimal control”. In: *Mathematical Programming Computation* (2018).

[26] David J Gardner et al. “Enabling new flexibility in the SUNDIALS suite of nonlinear and differential/algebraic equation solvers”. In: *ACM Transactions on Mathematical Software (TOMS)* 48.3 (2022), pp. 1–24. DOI: 10.1145/3539801.

[27] Radu Serban et al. *User Documentation for IDAS*. v6.1.1. 2024. URL: https://sundials.readthedocs.io/en/latest/idas.

[28] Elvira Marie B. Aske and Sigurd Skogestad. “Consistent Inventory Control”. In: *Industrial & Engineering Chemistry Research* 48.24 (2009), pp. 10892–10902. DOI: 10.1021/ie801603j. URL: https://doi.org/10.1021/ie801603j.

[29] Espen Storkaas and Sigurd Skogestad. “Cascade Control of Unstable Systems with Application to Stabilization of Slug Flow”. In: *IFAC Proceedings Volumes* 37.1 (2004). 7th International Symposium on Advanced Control of Chemical Processes (ADCHEM 2003), Hong-Kong, 11-14 January 2004, pp. 335–340. ISSN: 1474-6670. DOI: https://doi.org/10.1016/S1474-6670(17)38754-2. URL: https://www.sciencedirect.com/science/article/pii/S1474667017387542.

[30] Hugh D Young. *Sears and Zemansky’s university physics : with modern physics*. eng. Essex, 2016.

[31] Chia-Tsun Liu and William T. Jr. Lindsay. “Vapor pressure of deuterated water from 106 to 300.deg.” In: *Journal of Chemical & Engineering Data* 15.4 (1970), pp. 510–513. DOI: 10.1021/je60047a015. URL: https://doi.org/10.1021/je60047a015.

[32] Daniel R. Stull. "Vapor Pressure of Pure Substances. Organic and Inorganic Compounds". In: *Industrial & Engineering Chemistry* 39.4 (1947), pp. 517–550. DOI: 10.1021/ie50448a022. URL: https://doi.org/10.1021/ie50448a022.

APPENDICES

APPENDIX

A

MODEL

A.1 Assumptions Summary

General assumptions

- Ideal gas
- Constant C_p
- Homogeneous flow
- No reaction
- Linearized equation of state neglecting temperature effects on ρ
- Neglected fluid acceleration
- Saturation pressure follows Antoine equation
- Equivalent pressure in steam and liquid for two-phase
- Neglected enthalpy dependency on pressure
- No heat loss to environment
- No negative flow

Pump

- Constant pump pressure

OTSG

- Ideal counter current flow
- Dynamic mass and energy balances
- No thermal resistance or inertia in wall
- No work

Superheated steam holdups

- Dynamic mass and energy balances
- No work

Valves

- Linear valve characteristics
- Isenthalpic

Turbine

- Static mass and energy balance
- Isentropic expansion
- Polytropic efficiency
- Two-phase outflow
- Perfect conversion in generator

Condenser

- Static mass and energy balances
- No thermal resistance in wall
- Perfect temperature control giving saturated liquid outflow and constant condenser pressure

A.2 Equation Summary

Table A.2.1: Derived model equations for bottoming cycle.

Process unit	Equation
Pump	$m_p = z_m k_m (p_p - p_1)$
	$h_p = C_p^w (T_p - T_{Ref})$
OTSG	$\dfrac{dH_i}{dt} = m_{i-1}h_{i-1} - m_i h_i + Q_i$
$i \in \{1, 2, 3, ..., N\}$	$\dfrac{dM_i}{dt} = m_{i-1} - m_i$
$\beta_i \in \mathbb{R}$	$Q_i = m_{g,i} C_p^g (T_{g,i+1} - T_{g,i})$
	$Q_i = U_i A \left(\frac{T_{g,i+1} - T_i}{2} + \frac{T_{g,i} - T_{i-1}}{2} \right)$
	$m_{g,i} = m_{g,i+1}$
	$m_i = k_{OTSG}(p_i - p_{i+1})$
	$H_i = M_i h_i$
	$h_i = C_p^w (T_i^{sat} - T_{Ref}) + \beta_i \Delta H_{vap,i}$
	$\Delta H_{vap,i} = \Delta H_{vap}(T_{Ref}^{sat}) + (C_p^w - C_p^s)(T_{Ref}^{sat} - T_i^{sat})$
	$log_{10}(p_i) = A - \dfrac{B}{T_i^{sat} + C}$
	$p_i = \dfrac{1}{k_p \cdot \rho_{Ref}}(\rho_i - \rho_{Ref}) + p_{Ref}$
	$U_i = f(\beta_i, T_{g,i})$
Liquid phase	$h_i = C_p^w (T_i - T_{Ref})$
$\beta_i \leq 0$	$\rho_i = \frac{M_i}{V}$
Two phase	$T_i = T_i^{sat}$
$0 < \beta_i < 1$	$p_i = \dfrac{\beta_i M_i R T_i}{(V - V_L) M_w}$, with $V_L = \dfrac{(1 - \beta_i) M_i}{\rho_i}$
Gas phase	$h_i = C_p^w (T_i^{sat} - T_{Ref}) + \Delta H_{vap,i} + C_p^S (T_i - T_i^{sat})$
$\beta_i \geq 1$	$p_i = \dfrac{M_i R T_i}{V M_w}$

Superheated steam volume	$\frac{dH_j}{dt} = m_{j,in}h_{j,in} - m_j h_j$
	$\frac{dM_j}{dt} = m_{j,in} - m_j$
$j \in \{S, T\}$	$H_j = M_j h_j$
$\beta_i \in \mathbb{R}$	$h_j = C_p^w(T_j^{sat} - T_{Ref}) + \beta_j \Delta H_{vap,j}$
	$\Delta H_{vap,j} = \Delta H_{vap}(T_{Ref}^{sat}) + (C_p^w - C_p^s)(T_{Ref}^{sat} - T_j^{sat})$
	$log_{10}(p_j) = A - \frac{B}{T_j^{sat} + C}$
	$p_j = \frac{1}{k_p \cdot \rho_{Ref}}(\rho_j - \rho_{Ref}) + p_{Ref}$
Liquid phase	$h_j = C_p^w(T_j - T_{Ref})$
$\beta_j \leq 0$	$\rho_j = \frac{M_j}{V}$
Two phase	$T_j = T_j^{sat}$
$0 < \beta_j < 1$	$p_j = \frac{\beta_j M_j R T_j}{(V - V_L)M_w}$, with $V_L = \frac{(1-\beta_j)M_j}{\rho_j}$
Gas phase	$h_j = C_p^w(T_j^{sat} - T_{Ref}) + \Delta H_{vap,j} + C_p^S(T_j - T_j^{sat})$
$\beta_j \geq 1$	$p_j = \frac{M_j R T_j}{V_j M_w}$
Pre-turbine valve	$m_s = z_v k_v(p_s - p_T)$
Turbine	$p_T = \frac{m_T\sqrt{T_T}}{\phi_d}$
	$T_U = T_T \left(\frac{p_c}{p_T}\right)^{\frac{R}{C_p^s M_w}}$
	$W = \eta m_T C_p^s(T_T - T_U)$
	$P = -W$
	$m_U = m_T$

Post valve flow	$T_U^{real} = T_U^{sat}$ $h_U = C_p^w(T_U^{sat} - T_{Ref}) + \Delta H_{vap} + C_p^s(T_U - T_U^{sat})$ $h_U = C_p^w(T_U^{sat} - T_{Ref}) + \beta_U \Delta H_{vap}$ $log_{10}(p_c) = A_c - \frac{B_c}{T_U^{sat} + C_c}$ $\Delta H_{vap} = \Delta H_{vap}(T_{Ref}^{sat}) + (C_p^w - C_p^s)(T_{Ref}^{sat} - T_U^{sat})$
Condenser	$m_c = m_U$ $T_c = T_U^{sat}$ $h_c = C_p^w(T_c - T_{Ref})$ $Q_c = m_U h_U - m_c h_c$
Buffer tank	$\frac{dM_b}{dt} = m_c - m_p$

A.3 Parameters

Table A.3.1: Constants and fluid properties used in the simulations.

Constant	Description	Value	Unit
Thermodynamic and fluid property			
R	Gas constant	$8.314 \cdot 10^{-5}$	m^3bar/(K mol)
C_p^w	Heat capacity water	4.24	kJ/(kg K)
C_p^s	Heat capacity steam	2.43	kJ/(kg K)
C_p^g	Heat capacity flue gas	1.02	kJ/(kg K)
T_{Ref}	Reference temperature	0	K
T_{Ref}^{sat}	Reference boiling point temperature	576.15	K
$\Delta H_{vap}(T_{Ref}^{sat})$	Reference energy of vaporization [12]	1382	kJ/kg
M_w	Molar weight water	$18 \cdot 10^{-3}$	kg/mol
ρ_{Ref}	Reference density water	1000	kg/m^3
p_{Ref}	Reference pressure water	1	bar
k_p	Compressibility factor water [30]	$4.58 \cdot 10^{-5}$	1/bar
A_{anto}	Antoine coefficient (379-573K) [31]	3.55959	-
B_{anto}	Antoine coefficient (379-573K) [31]	643.748	K
C_{anto}	Antoine coefficient (379-573K) [31]	-198.043	K
$A_{anto,c}$	Antoine coefficient (255.9-373K) [32]	4.6543	-
$B_{anto,c}$	Antoine coefficient (255.9-373K) [32]	1435.264	K
$C_{anto,c}$	Antoine coefficient (255.9-373K) [32]	-64.848	K
Pump			
$z_{m,0}$	Nominal valve opening	0.5	-
p_{pump}	Pump pressure	29	bar
k_m	Valve coefficient	4.3571	kg/(bar s)
OTSG			
N	Discretization resolution	37	-
k_{OTSG}	Segment mass flow friction factor	10.95*(N+1)	kg/(bar s)
V_{Tot}	Total OTSG volume	3.92	m^3
A_{Tot}	Total OTSG cold side area	739.4	m^2
A_{Corr}	OTSG hot side geometry correction factor	7.2894	-
Superheated steam volumes			
V_s	Holdup volume	0.5	m^3
V_T	Holdup volume	0.5	m^3
Superheated steam valve			
$z_{v,0}$	Nominal valve opening	0.9	-
k_v	Valve coefficient	$12.1\bar{6}$	kg/(bar s)

Constant	Description	Value	Unit
Turbine			
ϕ_d	Turbine mass flow coefficient	13	kgK$^{0.5}$/(s bar)
η	Turbine efficiency factor	0.9	-
Condenser			
p_c	Condenser pressure	0.0358	bar
Buffer tank			
M_b	Buffer tank mass (initial)	10000	kg

Table A.3.2: Nominal boundary conditions of the system model.

Variable	Value [unit]
m_g^0 [6]	112.75 kg/s
T_g^0 [6]	443.3 $°C$
T_p [6]	26.7 $°C$
p_p	29 bar
p_c	0.0358 bar

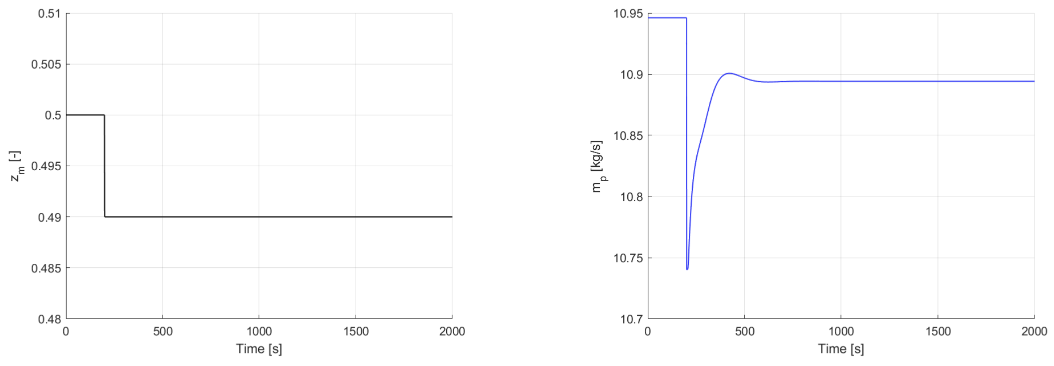
APPENDIX

B

TUNING

B.1 Flow Controller (C1)

Figure B.1.1 shows flow controller tuning profile from an open-loop simulation with step in z_m. The flow controller was tuned fast ($\tau_c = 5s$), and the identified model was therefore integrating process. The controller was tuned as a pure I-controller.



(a) z_m **(b)** m_p

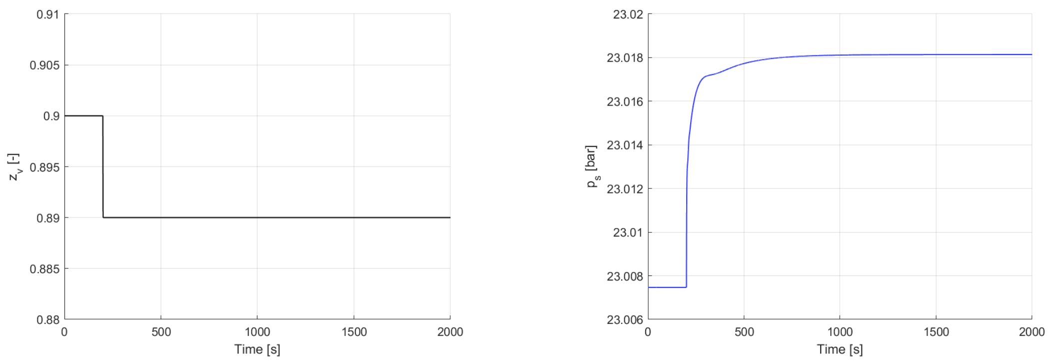
Figure B.1.1: Step in z_m and response in m_p used for flow controller tuning.

B.1.1 Example Calculation I-Controller

$$
\begin{aligned}
\theta &= 0s \\
\tau &= 0s \\
k &= \frac{\Delta y}{\Delta u} = \frac{10.7399 - 10.9461}{-0.01} = 20.62 \\
\tau_c &= 5s \\
\tau_I &= \tau \\
K_I &= \frac{K_c}{\tau_I} = \frac{1}{k}\frac{1}{\tau_c + \theta} = \frac{1}{20.62}\frac{1}{5+0} = 0.0097
\end{aligned}
\tag{B.1}
$$

B.2 Pressure controller (C2)

Figure B.2.1 shows pressure controller tuning profile on system with C1 controller active. The identified process here was FOPDT model and the controller was tuned as a PI-controller with anti-windup.



(a) z_v **(b)** p_S

Figure B.2.1: Step in z_v and response in p_s used for steam pressure controller tuning.

B.2.1 Example Calculation PI-Controller with Anti-Windup

$$
\begin{aligned}
\theta &= 0s \\
\tau &= 13s \\
k &= \frac{\Delta y}{\Delta u} = \frac{23.0181 - 23.0075}{-0.01} = -1.06 \\
\tau_c &= 5s \\
K_c &= \frac{1}{k}\frac{\tau}{\tau_c + \theta} = \frac{1}{-1.06}\frac{13}{5+0} = -2.4528 \\
\tau_I &= min(\tau, 4(\tau_c + \theta)) = min(13, 20) = 13s \\
\tau_T &= \tau_I = 13s
\end{aligned}
\tag{B.2}
$$

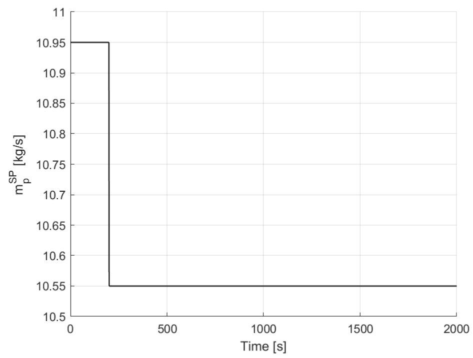
B.3 Temperature Controller (C3)

B.3.1 Constant Pressure Mode

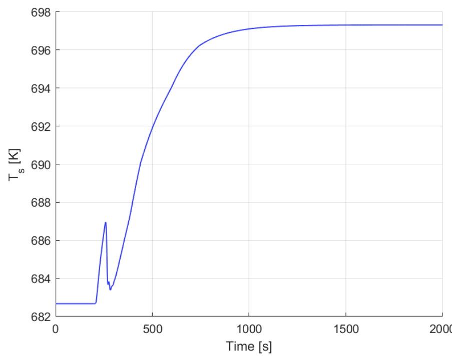
In constant pressure mode, the steam temperature controllers were tuned with C1 and C2 controllers active.

B.3.1.1 Feedback

Figure B.3.1 shows feedback temperature controller tuning for constant superheated steam pressure. The tuning was based on a step in m_s^{SP}. The identified model was FOPDT and the controller was tuned as a PI-controller. The controller was tuned with 100s delay to ignore the initial dynamics caused by switching.



(a) m_p^{SP}

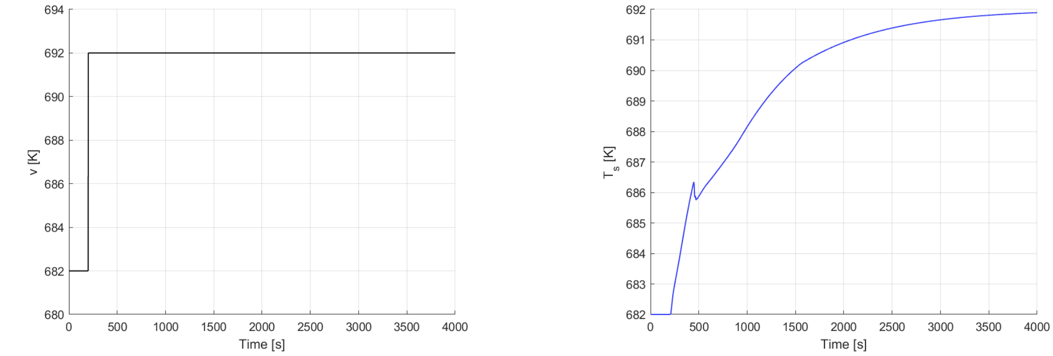


(b) T_S

Figure B.3.1: Step in m_p^{SP} and response in T_s for temperature feedback controller tuning in constant pressure mode.

B.3.1.2 Combined Feedforward and Feedback

Figure B.3.2 shows feedback controller tuning profile based on a step in transformed variable v. The identified model was FOPDT and the controller was tuned as a PI-controller.



(a) v **(b)** T_S

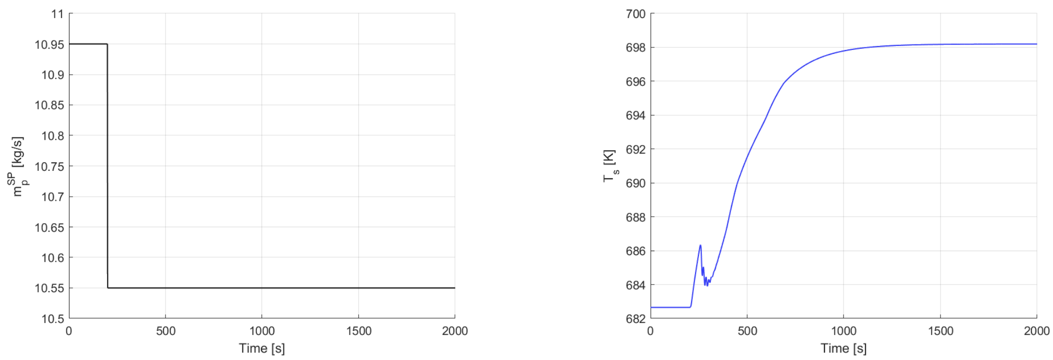
Figure B.3.2: Step in transformed variable v and response in T_s for temperature feedback controller tuning in constant pressure mode.

B.3.2 Floating Pressure Mode

In floating pressure mode, the steam temperature controllers were tuned with C1 controller active.

B.3.2.1 Feedback

Figure B.3.3 shows feedback temperature controller tuning based on a step in m_s^{SP}. The identified model here was FOPDT and the controller was tuned as a PI-controller. The controller was tuned with 100s delay to ignore the initial dynamics caused by switching dynamics.

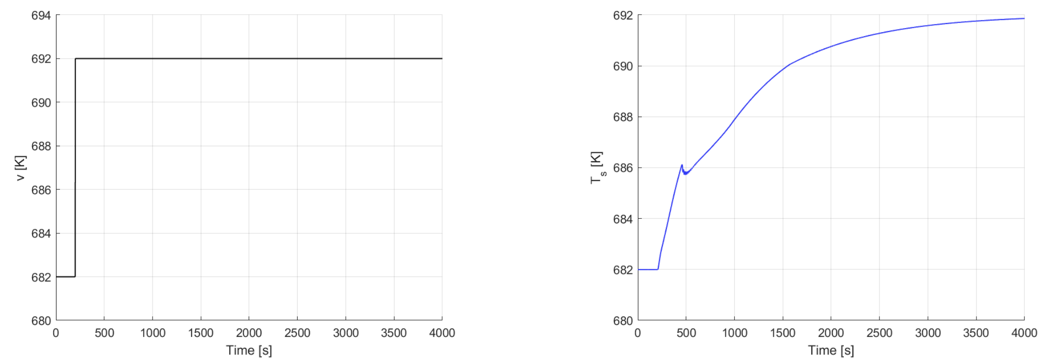


(a) m_p^{SP} **(b)** T_S

Figure B.3.3: Step in m_p^{SP} and response in T_s for temperature feedback controller tuning in floating pressure mode.

B.3.2.2 Combined Feedforward and Feedback

Figure B.3.4 shows combined feedforward and feedback temperature controller tuning based on a step in v. The identified model was FOPDT and the controller was tuned as a PI-controller.



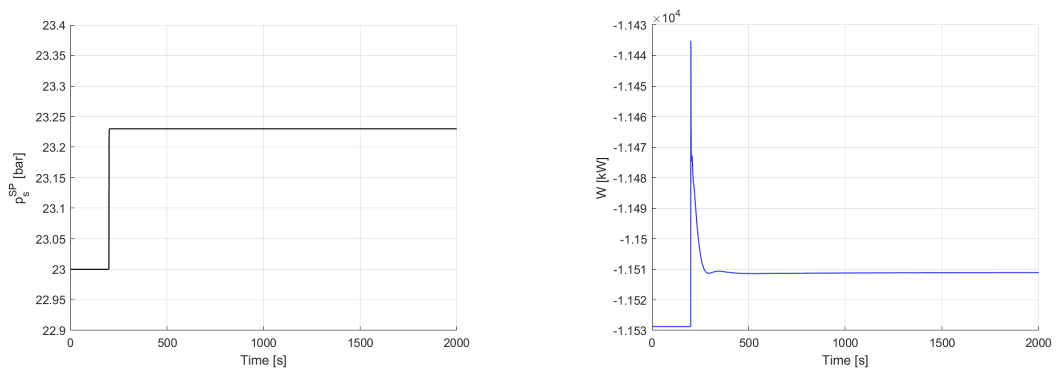
(a) v **(b)** T_S

Figure B.3.4: Step in v and response in T_s for temperature feedback controller tuning in floating pressure mode.

B.4 Steam Cycle Case

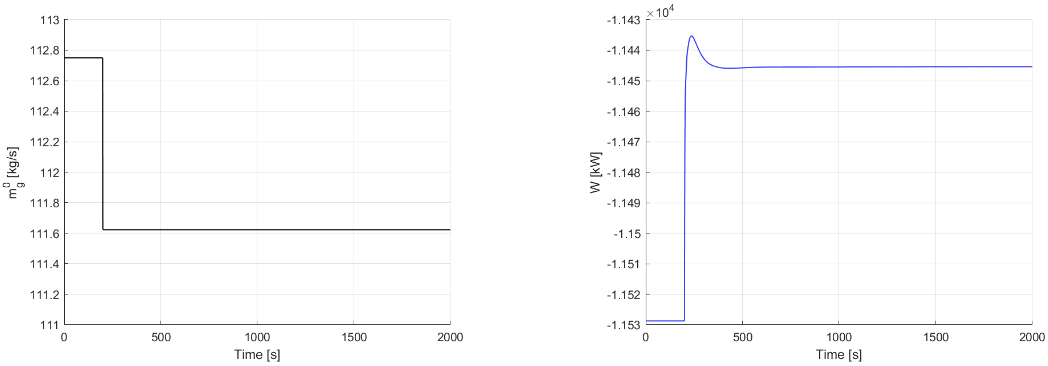
B.4.1 Constant Pressure Case

The constant pressure case was tuned with C1, C2 and C3 controllers active. Figure B.4.1 shows the tuning profile for P-Controller based on a step in p_s^{SP}, and Figure B.4.2 shows tuning profile for PI-controller tuning based on step in m_g^0.



(a) p_s^{SP} **(b)** W

Figure B.4.1: Step in p_s^{SP} and response in W used for P-controller tuning in constant pressure case.

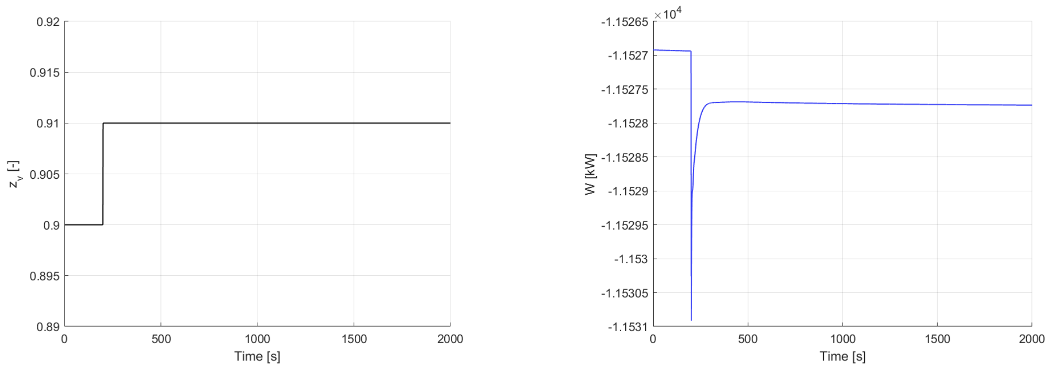


(a) m_g^0 **(b)** W

Figure B.4.2: Step in m_g^0 and response in W used for PI-controller tuning in constant pressure case.

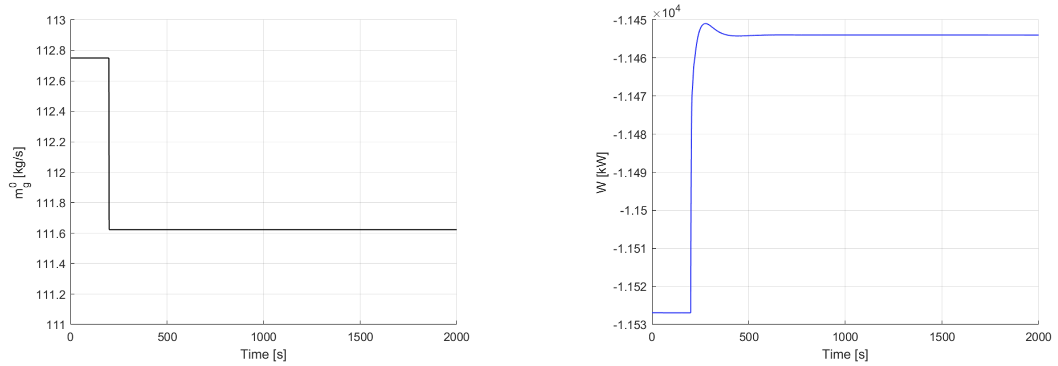
B.4.2 Floating Pressure Case

In floating pressure case, the controllers were tuned with C1 and C3 controllers active. Figure B.4.3 shows the tuning profile used for P-controller tuning based on a step in z_v. Figure B.4.4 shows the tuning profile used for PI-controller tuning based on a step in m_g^0.



(a) z_v **(b)** W

Figure B.4.3: Step in z_v and response in W used for P-controller tuning in floating pressure case.

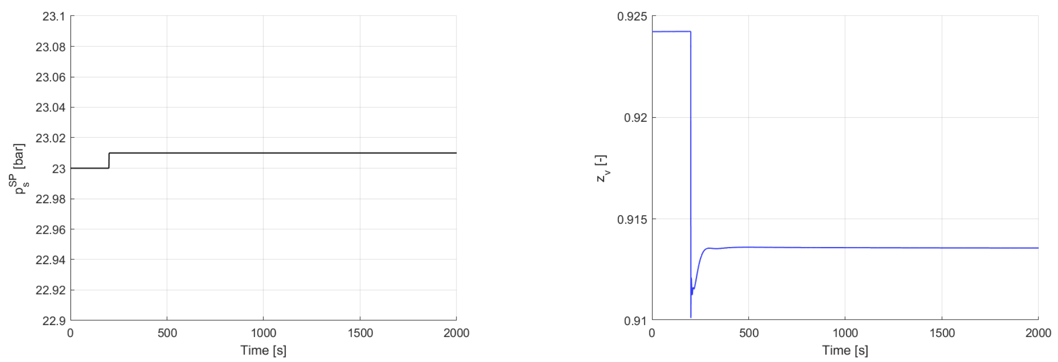


(a) m_g^0 **(b)** W

Figure B.4.4: Step in m_g^0 and response in W used for PI-controller tuning in floating pressure case.

B.5 Combined Cycle Case

In the combined cycle case, the VPC was tuned with C1, C2 and C3 controllers active. Figure B.5.1 shows the step in p_s^{SP} and response in z_v. The VPC was tuned with $\tau_c = 200s$.



(a) p_s^{SP} **(b)** z_v

Figure B.5.1: Step in p_s^{SP} and response in z_v for VPC tuning.

APPENDIX

C

SIMULATIONS

C.1 Controlled Variables and Specifications

Table C.1.1: Controlled variables and specifications for the available degrees of freedom used in the simulations. Floating/constant denote whether or not the pressure controller C2 was used.

Description	m_p^{SP}	T_s^{SP}	v_0	W^{SP}	$p_{s,0}^{SP}$	$z_{v,0}$	$z_{m,0}$	m_g^0
Open-loop Chapter 8.2	-	-	-	-	-	0.9	0.5	112.75
Flow controller C1 Chapter 8.2.2	10.95	-	-	-	-	0.9	-	112.75
C3 controller comparison Feedback, Chapter 8.3	-	682	-	-	-	0.9	-	112.75
C3 controller comparison Feedforward, Chapter 8.3	-	-	682	-	-	0.9	-	112.75
C3 controller comparison Combined, Chapter 8.3	-	682	-	-	-	0.9	-	112.75
Steam cycle case, floating Chapter 8.4.1	-	682	-	-11500	-	0.9	-	-
Steam cycle case, constant Chapter 8.4.1	-	682	-	-11500	23	-	-	-
Combined cycle case Chapter 8.4.2	-	682	-	-	-	0.9	-	112.75

APPENDIX

D

FEEDFORWARD TRANSFORMATION DERIVATION

Starting from the steady-state energy balance over the OTSG, as shown in Eq. D.1 assuming no heat flow or work across the system boundary.

$$E_{in} = E_{out} \tag{D.1}$$

The total energy E is assumed to be the sum of internal energy U, potential energy E_p and kinetic energy E_k. The kinetic and potential energy is neglected, leaving only the internal energy as shown in Eq. D.2.

$$\begin{aligned} E &= U + E_k + E_p \\ E &= U \end{aligned} \tag{D.2}$$

From the definition of enthalpy, and neglecting the change in internal energy from volume and pressure changes we get the following relation for U.

$$\begin{aligned} dH &= dU + d(PV) \\ U &= H \end{aligned} \tag{D.3}$$

By substituting Eq. D.3 into Eq. D.2, we get Eq. D.4 where m is the mass flow and h is the specific enthalpy.

$$E = H = mh \tag{D.4}$$

The OTSG has two inlet flows (m_p, m_g^0) and two outlet flows (m_s, m_g) which gives the following energy balance from Eq. D.1.

$$m_p h_p + m_g^0 h_g^0 = m_s h_s + m_g h_g \tag{D.5}$$

Assumption of steady-state gives equivalent inlet and outlet flows flows: $m_g = m_g^0$ and $m_s = m_p$.

$$\begin{aligned} m_p h_p - m_p h_s &= m_g^0 h_g - m_g^0 h_g^0 \\ m_p(h_p - h_s) &= m_g^0(h_g - h_g^0) \end{aligned} \tag{D.6}$$

Assumption of ideal gas makes enthalpy independent of pressure, as shown in Eq.D.7 derived from the total differentials of enthalpy.

$$\begin{aligned} dh &= C_p dT + \left(V - T \left(\frac{\partial V}{\partial T} \right)_p^{IG} \right) dp \\ \left(\frac{\partial V}{\partial T} \right)_p^{IG} &= \frac{nR}{p} \\ T \left(\frac{\partial V}{\partial T} \right)_p^{IG} &= \frac{nRT}{p} = V \\ dh &= C_p dT + (V - V)\, dp \\ dh &= C_p dT \end{aligned} \tag{D.7}$$

Since enthalpy is a reference state we can assume a reference point $h_{Ref} = h_p = 0$. We can also choose to theoretically vaporize the fluid at T_p and then heat it to its final temperature T_s which will give the following enthalpy relations for the cold side

$$\begin{aligned} & h_p = 0 \\ & h_s = \Delta H_{vap}(T_p) + C_p^s(T_s - T_p) \end{aligned} \tag{D.8}$$

where,

$$\Delta H_{vap}(T_p) = \Delta H_{vap}(T_{Ref}^{sat}) + (C_p^w - C_p^s)(T_{Ref}^{sat} - T_p) \tag{D.9}$$

Likewise we get the following relation for the hot side change in enthalpy as there are no phase change:

$$h_g - h_g^0 = C_p^g(T_g - T_g^0) \tag{D.10}$$

Substituting the enthalpy equations into Eq. D.6 gives the following energy balance over the OTSG:

$$-m_p(\Delta H_{vap}(T_p) + C_p^s(T_s - T_p)) = m_g^0 C_p^g(T_g - T_g^0) \tag{D.11}$$

Rearrange the equation for m_p as this is the variable we want to manipulate with the feedforward transformation. The negative sign in the numerator can be eliminated by inverting the temperature difference.

$$m_p = \frac{m_g^0 C_p^g (T_g^0 - T_g)}{\Delta H_{vap}(T_p) + C_p^s (T_s - T_p)} \tag{D.12}$$

The static transformed input v_0 is defined as shown in D.13

$$v_0 = T_s = T_p - \frac{\Delta H_{vap}(T_p)}{C_p^s} + \frac{m_g^0 C_p^g}{m_p C_p^s}(T_g^0 - T_g) \tag{D.13}$$

The transformed input v is given by the PID-controller equation as shown in Eq. D.14.

$$\begin{aligned} v &= v_0 + K_c e + \frac{K_c}{\tau_I} \int_0^t e \text{ dt} \\ \text{where, } e &= T_s^{SP} - T_s \end{aligned} \tag{D.14}$$

Inserting the transformed controller output v for T_s in equation D.12 gives the physical output u as shown in Eq. D.15

$$u = m_p^{SP} = \frac{m_g^0 C_p^g (T_g^0 - T_g)}{\Delta H_{vap}(T_p) + C_p^s (v - T_p)} \tag{D.15}$$

APPENDIX

E

CODE

Several versions of the model were made for different control structures. The code for the steam cycle case with constant pressure is provided in full (model.m and init.m). Since the models are fairly large and only have minor differences in what controllers that were active, the other code files will be available through a GitHub repository and a .zip file together with simulation result files and plots used in this thesis.

init.m and model.m contains a side-equation for grid frequency (ω) with some related constants (L, M_g, D_g and ω_0) that were not used in the model, this is marked in the code.

E.1 GitHub Repository

Code, simulation result files and plots are provided in the GitHub repository below.

www.github.com/williadav/Thesis

E.2 Model Initialization (init.m)

```
clear
clc
addpath('C:\casadi-3.6.6') % Import Casadi path here

%% Run multiple steps in series
save = {'test_1.csv', 'test_2.csv'};

% Make step profile
T0g = 443.338+273.15;
Tg_step_N = [(T0g)*ones(1,200), (T0g-10)*ones(1,3800)];
Tg_step_P = [(T0g)*ones(1,200), (T0g+10)*ones(1,3800)];

% Apply step profile to constant or control array
applied_step = {Tg_step_N, Tg_step_P};
step_index  = {9, 9};

% loop counter for number of runs
for y=1:2
% Set filepath for storage, use false to discard results
filepath = save{y};

% load state from .csv file, use false if no load file
load_from_file = 'load_file.csv';

% if loaded state contains OTSG only, use true
OTSG_only = false;

% Set number of OTSG segments
n = 37;
```

```
%% Constants
% Thermodynamic and fluid properties
R          = 8.314462618*1e-5; % [m3*bar/K/mol] gas constant
cpS        = 2.43;             % [kJ/kg/K] heat capasity steam
cpW        = 4.24;             % [kJ/kg/K] heat capasity water
cpG        = 1.02;             % [kJ/kg/K] heat capasity gas
T_Ref      = 0;                % [K] Reference temperature
TB_Ref     = 303+273.15;       % [K] Reference boiling point temperature
dHvap0     = 1382;             % [kJ/kg] Reference energy of vaporization
Mw         = 18*10^(-3);       % [kg/mol] molar weight water
rho_Ref    = 1000;             % [kg/m3] Reference density water
p_Ref      = 1;                % [bar] Reference pressure water
k_p        = 4.58*10^(-5);     % [1/bar] Compresibility factor water
A_anto     = 3.55959;          % [-] Antoine Coeff (379 to 573K)
B_anto     = 643.748;          % [K] Antoine Coeff (379 to 573K)
C_anto     = -198.043;         % [K] Antoine Coeff (379 to 573K)
A_anto_c   = 4.6543;           % [-] Antoine Coeff (255.9 to 373K)
B_anto_c   = 1435.264;         % [K] Antoine Coeff (255.9 to 373K)
C_anto_c   = -64.848;          % [K] Antoine Coeff (255.9 to 373K)

% Design pressure and mass flow for valve design
p_in     = 24;               % [bar] OTSG cold side inlet pressure
pS       = 23;               % [bar] OTSG cold side outlet pressure
pT       = 22;               % [bar] Turbine inlet pressure
mP       = 21.9/2;           % [kg/s] Nominal mass flow

% OTSG pressure drop discretization
p_guess = linspace(p_in, pS, n+2);
p_guess = p_guess(2:n+1);

% Pump
p_p      = 29;                       % [bar] Pump pressure
Tp       = 26.6769+273.15;           % [K]   Pump temperature
zm       = 0.5;                      % [-] Nominal valve opening
k_m      = mP/(zm*(p_p-p_guess(1))); % [kg/bar] pump valve coeff

% OTSG
T0g      = 443.338+273.15;       % [K] OTSG hot side inlet temperature
mG       = 225.5/2;              % [kg/s] OTSG hot side mass flow
V_tot    = 3.92;                 % [m3] OTSG volume
A_tot    = 739.4;                % [m2] OTSG Area
A_corr   = 5389.79/A_tot;        % [-]  OTSG hot side fin correction
k_OTSG   = mP/((p_in-pS))*(n+1); % [kg/bar] OTSG valve coeff

% Super heated steam section
Vs       = 0.5;              % [m3] Steam buffer volume
zv       = 0.9;              % [-] Nominal valve opening
kv       = mP/(zv*(pS-pT)); % [kg/bar] valve coeff 1 bar pressure drop
VT       = 0.5;              % [m3]  Pre-turbine volume

% Turbine
phiD     = 13;          % [kgK^(1/2)/(s bar)] Turbine massflow coefficient
Mg       = 2.5022e+03;  % Frequency calculation parameter, not used
Dg       = 661.9580;    % Frequency calculation parameter, not used
w0       = 50;          % Frequency calculation parameter, not used
L        = 11;          % Frequency calculation parameter, not used
eta      = 0.9;         % [-] Turbine efficency factor

% Condenser
p_c      = 0.0358;      % [bar] Condenser pressure

% Make constants array
constants = [R, cpS, cpW, cpG, T_Ref, TB_Ref, T0g, Tp, mG, dHvap0, ...
    Mw, k_OTSG, V_tot, rho_Ref, p_Ref, k_p, A_anto, B_anto, C_anto,...
    A_tot, A_corr, kv, Vs, phiD, Mg, Dg, w0, L, ...
    A_anto_c, B_anto_c, C_anto_c, eta, VT, k_m, p_p, p_c];
```

```
%% Control
% Flow controller (C1)
m_in_SP_0 = 10.95;
KI_m_in   = 0.009699;

% Pressrue controller (C2)
pS_SP     = pS;
zv_0      = zv;
Kc_pS     = -2.4528;
tauI_pS   = 13;
tauT_pS   = tauI_pS;

% Combined FB + FF temperature controller (C3)
Ts_SP     = 682;
Kc_Ts     = 3.6902;
tauI_Ts   = 730;
v_0       = 682;        % FF bias

% Power controller. Case: Standalone, constant pressure
W_SP      = -11500;
Kc_W1     = 0.00066;
Kc_W2     = -0.0005075;
tauI_W2   = 3;

% Make control array
control = [zm; KI_m_in; Ts_SP; Kc_Ts; tauI_Ts; v_0; pS_SP; ...
    zv_0; Kc_pS; tauI_pS; tauT_pS; W_SP; Kc_W1; Kc_W2; tauI_W2];

%% Set initial guess
% OTSG (Only used if load_file = false)
beta_guess = -0.4;             % [-]
M_guess    = V_tot*1000/(n);   % [kg]
T_guess    = Tp;               % [K]
h_guess    = cpW*(Tp-T_Ref);   % [kJ/kg]
H_guess    = M_guess*h_guess;  % [kJ]
m_guess    = mP;               % [kg/s]
Tg_guess   = T0g;              % [K]
mg_guess   = mG;               % [kg/s]
U_guess    = 0.4;              % [kW/K]
Q_guess    = U_guess*A_tot/n*(Tg_guess-T_guess);   % [kW]
rho_guess  = 1000;             % [kg/m3]
TB_guess   = TB_Ref;             % [K]

% Super heated steam (s) (Used if OTSG_only = true)
m_s_guess    = mP;
Ts_guess     = 683;            % [K]
TBs_guess    = TB_Ref;         % [K]
hs_guess     = cpW*(TBs_guess-T_Ref) + dHvap0 ...
+ cpS*(Ts_guess-TBs_guess); % [kJ/kg]
beta_s_guess= (hs_guess-cpW*(TBs_guess-T_Ref))/dHvap0; % [-]
rho_s_guess = rho_guess;       % [kg/m3]
Ms_guess     = pS*Vs*Mw/(R*Ts_guess);   % [Kg]
Hs_guess     = Ms_guess*hs_guess;       % [kJ]

% Pre-turbine (T) (Used if OTSG_only = true)
m_T_guess    = mP;             % [kg/s]
TT_guess     = Ts_guess;       % [K]
TBT_guess    = TB_Ref;
hT_guess     = cpW*(TBT_guess-T_Ref) + dHvap0 + ...
cpS*(TT_guess-TBT_guess); % [kJ/kg]
beta_T_guess= (hT_guess-cpW*(TBT_guess-T_Ref))/dHvap0; % [-]
rho_T_guess = rho_guess;
MT_guess     = pT*VT*Mw/(R*TT_guess);   % [Kg]
HT_guess     = MT_guess*hT_guess;       % [kJ]

% Turbine (Used if OTSG_only = true)
W_guess      = -16500/2;       % [kW]
P_guess      = -W_guess;       % [kW]
w_guess      = 50;             % [Hz]
```

```
m_U_guess      = mP;
TU_guess       = 240;
TU_real_guess = 273.15+27;                                   % [K]
TBU_guess      = TU_real_guess;                               % [K]
hU_guess       = cpW*(TBU_guess-T_Ref) + dHvap0 ...
+ cpS*(TU_guess-TBU_guess); % [kJ/kg]
beta_U_guess   = 0.9;

% Condenser (c) (Used if OTSG_only = true)
Tc_guess      = TU_real_guess;                               % [K]
m_c_guess     = mP;                                          % [kg/s]
pc_guess      = 0.0358;                                      % [bar]
hc_guess      = cpW*(Tc_guess-T_Ref);                        % [kJ/kg]
Qc_guess      = m_U_guess*hU_guess - m_c_guess*hc_guess;     % [kW]
Mb_guess      = 10000;                                       % [kg] buffer tank

% Control variables (Used if OTSG_only = true)
% Dummies are constants/control variables added to solver for file storage
dummy_zm_guess     = zm;
dummy_m_in_SP      = m_in_SP_0;
dummy_v_guess      = v_0;
dummy_zV_guess     = zv;
dummy_mg_guess     = mG;
dummy_pS_SP_guess = pS;

interrmIn_guess = 0;
interrTs_guess  = 0;
interrpS_guess  = 0;
interrzV_guess  = 0;
interrW_guess   = 0;

% Three possible ways to initalize system using if-else
% Load only OTSG from file
if ~isempty(load_from_file) && OTSG_only
    loadTable = readtable(load_from_file);
    lastRow = loadTable{end, :};  % Load last instance in time

    % OTSG variables
    num_x = 2*n;     % Number of x variables
    num_z = 10*n+1; % Number of z variables

    % Reorder: x and z are swapped in key_names
    reordered_data = [lastRow(num_z+1:end), lastRow(1:num_z)];

    % Assign to initial conditions
    x_0 = reordered_data(1:num_x);
    z_0 = reordered_data(num_x+1:end)';

    % Initialize model variables after OTSG
    x_0 = [x_0, Ms_guess, Hs_guess, MT_guess, HT_guess, w_guess, ...
        Mb_guess];

    z_0 = [z_0; m_s_guess; Ts_guess; TBs_guess; pS; hs_guess; ...
        beta_s_guess; rho_s_guess; m_T_guess; TT_guess; TBT_guess; ...
        pT; hT_guess; beta_T_guess; rho_T_guess; W_guess; P_guess; ...
        m_U_guess; TU_guess; TU_real_guess; TBU_guess; hU_guess; ...
        beta_U_guess; Tc_guess; m_c_guess; hc_guess; Qc_guess];

    % Initialize control variables
    z_0 = [z_0; dummy_zm_guess; dummy_m_in_SP; dummy_v_guess; ...
        dummy_zV_guess; dummy_mg_guess; dummy_pS_SP_guess];

    x_0 = [x_0, interrmIn_guess, interrTs_guess, interrpS_guess,...
        interrzV_guess, interrW_guess];
```

```
% Load all variables from file
elseif ~isempty(load_from_file) && ~OTSG_only
    loadTable = readtable(load_from_file);
    lastRow = loadTable{end, :};

    % System variables
    num_x = 2*n+11;     % Number of x variables
    num_z = 10*n+1+32; % Number of z variables (hex + inlet + other)

    reordered_data = [lastRow(num_z+1:end), lastRow(1:num_z)];

    % Assign to initial conditions
    x_0 = reordered_data(1:num_x);
    z_0 = reordered_data(num_x+1:end)';

% Not load from file
else
    % OTSG variables
    x_0 = repmat([M_guess; H_guess], n, 1);
    z_0 = [m_guess];
    for k=1:n
        z_0 = [z_0; m_guess; T_guess; h_guess; Tg_guess; mg_guess; ...
            Q_guess; p_guess(k); rho_guess; beta_guess;TB_guess];
    end

    % Initialize model variables after OTSG
    z_0 = [z_0; m_s_guess; Ts_guess; TBs_guess; pS; hs_guess; ...
        beta_s_guess; rho_s_guess; m_T_guess; TT_guess; ...
        TBT_guess; pT; hT_guess; beta_T_guess; rho_T_guess; ...
        W_guess; P_guess; m_U_guess; TU_guess; TU_real_guess; ...
        TBU_guess; hU_guess; beta_U_guess; Tc_guess; m_c_guess; ...
        hc_guess; Qc_guess];

    x_0 = [x_0, Ms_guess, Hs_guess, MT_guess, HT_guess, w_guess, ...
         Mb_guess];

    % Initialize control variables
    z_0 = [z_0; dummy_zm_guess; dummy_m_in_SP; dummy_v_guess; ...
        dummy_zV_guess; dummy_mg_guess; dummy_pS_SP_guess];

    x_0 = [x_0, interrmIn_guess, interrTs_guess, interrpS_guess,...
        interrzV_guess, interrW_guess];
end


%% Run
T = 4000; % [s] Total simulation time
N = T;    % [-] Number of steps

% Inital conditions
x = x_0;
z = z_0;

% Make result arrays
x_save = zeros(N+1, length(x_0));
z_save = zeros(N+1, length(z_0));
x_save(1, :) = x_0;
z_save(1, :) = z_0;

time_save = [0];
running_time = 0;
```

```
for i = 1:N
    % Apply step profile in either 'control' or 'constants'
    % comment out for no step
    constants(step_index{y}) = applied_step{y}(i);

    % Evaluate system (Use .m file for system model)
    [xf_keys, zf_keys, result] = Model(n, x, z, constants, control);
    x = result.xf;
    z = result.zf;

    % Save state
    x_save(i+1, :) = full(x);
    z_save(i+1, :) = full(z);

    % Update time counter and display time
    running_time   = running_time + T/N;
    time_save = [time_save, running_time];
    disp(running_time);
end

%% Make result arrays

% Convert keys to strings
zf_keys = cellfun(@(var) char(var.name()), ...
    num2cell(zf_keys), 'UniformOutput', false);
xf_keys = cellfun(@(var) char(var.name()), ...
    num2cell(xf_keys), 'UniformOutput', false);

% Initialize cell arrays for results
key_names = [zf_keys(:); xf_keys(:)]; % Convert to column vectors
values = [z_save, x_save];

% For terminal display
for i = 1:length(key_names)
    data = full(values(:, i));
    data = data(end);
    disp([key_names{i}, ': ', mat2str(round(data,4))])
end

%% Store result as CSV

if filepath
    % Create a cell array for storing data
    numKeys = length(key_names);         % Total number of keys
    maxRows = size(values, 1);           % Assuming same number of rows
    tableData = cell(maxRows + 1, numKeys); % +1 for the header

    % Store the keys as headers in the first row
    for i = 1:numKeys
        tableData{1, i} = key_names{i};
    end

    % Store the values under each corresponding header
    for i = 1:numKeys
        dataArray = values(:, i);
        for j = 1:maxRows
            % Store values starting from row 2
            tableData{j + 1, i} = dataArray(j);
        end
    end

    % Convert the cell array to a table
    csvTable = cell2table(tableData(2:end, :), ...
        'VariableNames', tableData(1, :));

    % Write the table to a CSV file
    writetable(csvTable, filepath);
end
end
```

E.3 Model (Model.m)

```
function [x_keys, z_keys, sol] = ...
    Model(n, x_0, z_0, constants, control)
    import casadi.*

    %% constants
    % Unpack constants
    constants = num2cell(constants);
    [R, cpS, cpW, cpG, T_Ref, TB_Ref, T0g, Tp, mG_0, dHvap0, ...
     MW, k_OTSG, V_tot, rho_Ref, p_Ref, k_p, A_anto, B_anto, ...
     C_anto, A_tot, A_corr, kv, Vs, phiD, Mg, Dg, w0, L, ...
     A_anto_c, B_anto_c, C_anto_c, eta, VT, k_m, p_p, pc] ...
     = deal(constants{:});

    % Unpack controller constants
    control = num2cell(control);
    [z_m_0, KI_m_in, Ts_SP, Kc_Ts, tauI_Ts, v_0, pS_SP_0, zV_0, ...
     Kc_pS, tauI_pS, tauT_pS, W_SP, Kc_W1, Kc_W2, tauI_W2] ...
     = deal(control{:});

    % Segment dependent constants
    V      = V_tot/n;
    A      = A_tot/n;

    %% Solver initialization
    % Define variables as CasADi symbolic variables
    % OTSG
    m_in    = SX.sym('m_in');
    m       = SX.sym('m', n);
    T       = SX.sym('T', n);
    h       = SX.sym('h', n);
    M       = SX.sym('M', n);
    H       = SX.sym('H', n);
    Tg      = SX.sym('Tg', n);
    mg      = SX.sym('mg', n);
    Q       = SX.sym('Q', n);
    p       = SX.sym('p', n);
    rho     = SX.sym('rho', n);
    beta    = SX.sym('beta', n);
    TB      = SX.sym('TB', n);

    % Super heated steam (s)
    Ms      = SX.sym('Ms');
    Hs      = SX.sym('Hs');
    m_s     = SX.sym('m_s');
    Ts      = SX.sym('Ts');
    TBs     = SX.sym('TBs');
    pS      = SX.sym('pS');
    hS      = SX.sym('hS');
    beta_s = SX.sym('beta_s');
    rho_s  = SX.sym('rho_s');

    % Pre-turbine (T)
    MT      = SX.sym('MT');
    HT      = SX.sym('HT');
    m_T     = SX.sym('m_T');
    TT      = SX.sym('TT');
    TBT     = SX.sym('TBT');
    pT      = SX.sym('pT');
    hT      = SX.sym('hT');
    beta_T = SX.sym('beta_T');
    rho_T  = SX.sym('rho_T');

    % Turbine
    w       = SX.sym('w'); % Frequency calculation, not used
    W       = SX.sym('W');
    P       = SX.sym('P');
```

```matlab
    % Post-turbine (U)
    m_U     = SX.sym('m_U');
    TU      = SX.sym('TU');
    TU_real= SX.sym('TU_real');
    TBU     = SX.sym('TBU');
    hU      = SX.sym('hU');
    beta_U = SX.sym('beta_U');

    %Condenser and buffer tank
    Tc      = SX.sym('Tc');
    m_c     = SX.sym('m_c');
    Qc      = SX.sym('Qc');
    hc      = SX.sym('hc');
    Mb      = SX.sym('Mb');

    % Control
    dummy_zm        = SX.sym('dummy_zm');
    dummy_mg        = SX.sym('dummy_mg');
    dummy_pS_SP     = SX.sym('dummy_pS_SP');
    dummy_m_in_SP   = SX.sym('dummy_m_in_SP');
    dummy_v         = SX.sym('dummy_v');
    dummy_zV        = SX.sym('dummy_zV');

    interrmIn  = SX.sym('interrmIn');
    interrTs   = SX.sym('interrTs');
    interrpS   = SX.sym('interrpS');
    interrzV   = SX.sym('interrzV');
    interrW    = SX.sym('interrW');

    % Set states  x: differential, z: algebraic
    x = [];
    % OTSG
    for k=1:n
        x = [x; M(k); H(k)];
    end
    % Other system states
    x = [x; Ms; Hs; MT; HT; w; Mb];

    % Control
    x = [x; interrmIn;interrTs;interrpS;interrzV;interrW];

    % OTSG
    z = [m_in];
    for k=1:n
        z = [z; m(k);T(k);h(k);Tg(k);mg(k);Q(k);p(k);...
            rho(k);beta(k);TB(k)];
    end
    % Other system states
    z = [z; m_s; Ts; TBs; pS; hS; beta_s; rho_s; m_T; ...
        TT; TBT; pT; hT; beta_T; rho_T; W; P; m_U; TU; ...
        TU_real; TBU; hU; beta_U; Tc; m_c; hc; Qc];

    % Control
    z = [z; dummy_zm; dummy_m_in_SP; dummy_v; dummy_zV; ...
        dummy_mg; dummy_pS_SP];

    % Initialize equatuions
    Alg  = [];
    diff = [];

    %% OTSG inlet equations
    h_in  = cpW*(Tp-T_Ref);

    % Power control
    % WC1: P controller
    errW   = W_SP - W;
    pS_SP  = pS_SP_0 + Kc_W1*errW;

    % WC2: PI controller
    mG = mG_0 + Kc_W2*errW + Kc_W2/tauI_W2*interrW;
```

```
    % Steam temperature control
    % TC: FF + FB (C3)
    dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-Tp); % Energy of vaporization at Tp
    errTs = Ts_SP - Ts;                      % e: ySP-y
    v = v_0 + Kc_Ts*errTs + Kc_Ts/tauI_Ts*interrTs;
    m_in_SP = mG*cpG*(T0g-Tg(1))/(dHvap + cpS*(v-Tp));

    % FC: I Controller (C1)
    errmIn = m_in_SP - m_in;                    % e: ySP-y
    z_m = z_m_0 + KI_m_in*interrmIn;            % u = u0 + KI*integral(e)
    z_m_appl = fmax(0, fmin(1, z_m));           % u_bar = max(0, min(u, 1))

    init1 = m_in - z_m_appl*k_m*(p_p-p(1));
    Alg   = [Alg;init1];

    %% OTSG (i = 1)

    % Switch logic
    cond1 = beta(1) >= 1;
    cond2 = beta(1) <= 0;

    % 1: Steam, 2: Two-phase, 3: Liquid
    dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TB(1));
    Enth1 = cpW*(TB(1)-T_Ref) + dHvap + cpS*(T(1)-TB(1)) - h(1);
    Enth2 = T(1) - TB(1);
    Enth3 = cpW*(T(1)-T_Ref) - h(1);

    VL    = (1-beta(1))*M(1)/rho(1);
    Pres1 = M(1)*R*T(1)/(MW*V) - p(1);
    Pres2 = p(1)*(V-VL) - beta(1)*M(1)*R*T(1)/(MW);
    Pres3 = rho(1) - M(1)/V;

    % Use polynomial extrapolation to find U
    U = U_Calc(beta(1), Tg(1), A_corr);

    % Equations
    dMdt = m_in - m(1);
    dHdt = m_in*h_in - m(1)*h(1) + Q(1);

    alg1  = mg(1)*cpG*(Tg(2)-Tg(1)) - Q(1);
    alg2  = U*A*((Tg(2)-T(1))/2+(Tg(1)-Tp)/2) - Q(1);
    alg3  = m(1) - k_OTSG*(p(1)-p(2));
    alg4  = if_else(cond1, Enth1, if_else(cond2, Enth3, Enth2));
    alg5  = M(1)*h(1) - H(1);
    alg6  = mg(1) - mg(2);
    alg7  = if_else(cond1, Pres1, if_else(cond2, Pres3, Pres2));
    alg8  = p(1) - 1/(k_p*rho_Ref)*(rho(1)-rho_Ref) - p_Ref;
    alg9  = cpW*(TB(1)-T_Ref) + beta(1)*dHvap - h(1);
    alg10 = 10^(A_anto-(B_anto/(TB(1)+C_anto))) - p(1);

    Alg  = [Alg;alg1;alg2;alg3;alg4;alg5;alg6;alg7;alg8;alg9;alg10];
    diff = [diff;dMdt;dHdt];

    %% OTSG (i = 2 to n-1)
    for k=2:n-1

        % Switch logic
        cond1 = beta(k) >= 1;
        cond2 = beta(k) <= 0;

        dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TB(k));
        Enth1 = cpW*(TB(k)-T_Ref) + dHvap + cpS*(T(k)-TB(k)) - h(k);
        Enth2 = T(k) - TB(k);
        Enth3 = cpW*(T(k)-T_Ref) - h(k);

        VL    = (1-beta(k))*M(k)/rho(k);
        Pres1 = M(k)*R*T(k)/(MW*V) - p(k);
        Pres2 = p(k)*(V-VL) - beta(k)*M(k)*R*T(k)/(MW);
        Pres3 = rho(k) - M(k)/V;

        U = U_Calc(beta(k), Tg(k), A_corr);
```

```
        % Equations
        dMdt = m(k-1)-m(k);
        dHdt = m(k-1)*h(k-1) - m(k)*h(k) + Q(k);

        alg1  = mg(k)*cpG*(Tg(k+1)-Tg(k)) - Q(k);
        alg2  = U*A*((Tg(k+1)-T(k))/2+(Tg(k)-T(k-1))/2) - Q(k);
        alg3  = m(k) - k_OTSG*(p(k)-p(k+1));
        alg4  = if_else(cond1, Enth1, if_else(cond2, Enth3, Enth2));
        alg5  = M(k)*h(k) - H(k);
        alg6  = mg(k) - mg(k+1);
        alg7  = if_else(cond1, Pres1, if_else(cond2, Pres3, Pres2));
        alg8  = p(k) - 1/(k_p*rho_Ref)*(rho(k)-rho_Ref) - p_Ref;
        alg9  = cpW*(TB(k)-T_Ref) + beta(k)*dHvap - h(k);
        alg10 = 10^(A_anto-(B_anto/(TB(k)+C_anto))) - p(k);

        Alg  = [Alg;alg1;alg2;alg3;alg4;alg5;alg6;alg7;alg8;alg9;alg10];
        diff = [diff;dMdt;dHdt];
    end

    %% OTSG (i = n)

    % Switch logic
    cond1 = beta(n) >= 1;
    cond2 = beta(n) <= 0;

    dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TB(n));
    Enth1 = cpW*(TB(n)-T_Ref) + dHvap + cpS*(T(n)-TB(n)) - h(n);
    Enth2 = T(n) - TB(n);
    Enth3 = cpW*(T(n)-T_Ref) - h(n);

    VL    = (1-beta(n))*M(n)/rho(n);
    Pres1 = M(n)*R*T(n)/(MW*V) - p(n);
    Pres2 = p(n)*(V-VL) - beta(n)*M(n)*R*T(n)/(MW);
    Pres3 = rho(n) - M(n)/V;

    U = U_Calc(beta(n), Tg(n), A_corr);

    % Equations
    dMdt = m(n-1)-m(n);
    dHdt = m(n-1)*h(n-1) - m(n)*h(n) + Q(n);

    alg1  = mg(n)*cpG*(T0g-Tg(n)) - Q(n);
    alg2  = U*A*((T0g-T(n))/2+(Tg(n)-T(n-1))/2) - Q(n);
    alg3  = m(n) - k_OTSG*(p(n)-pS);
    alg4  = if_else(cond1, Enth1, if_else(cond2, Enth3, Enth2));
    alg5  = M(n)*h(n) - H(n);
    alg6  = mg(n) - mG;
    alg7  = if_else(cond1, Pres1, if_else(cond2, Pres3, Pres2));
    alg8  = p(n) - 1/(k_p*rho_Ref)*(rho(n)-rho_Ref) - p_Ref;
    alg9  = cpW*(TB(n)-T_Ref) + beta(n)*dHvap - h(n);
    alg10 = 10^(A_anto-(B_anto/(TB(n)+C_anto))) - p(n);

    Alg  = [Alg;alg1;alg2;alg3;alg4;alg5;alg6;alg7;alg8;alg9;alg10];
    diff = [diff;dMdt;dHdt];

    %% Super heated steam volume and valve (s)

    % Switch logic
    cond1 = beta_s >= 1;
    cond2 = beta_s <= 0;

    dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TBs);
    Enth1 = cpW*(TBs-T_Ref) + dHvap + cpS*(Ts-TBs) - hS;
    Enth2 = Ts - TBs;
    Enth3 = cpW*(Ts-T_Ref) - hS;

    VLs   = (1-beta_s)*Ms/rho_s;
    Pres1 = Ms*R*Ts/(MW*Vs) - pS;
    Pres2 = pS*(Vs-VLs) - beta_s*Ms*R*Ts/(MW);
    Pres3 = rho_s - Ms/Vs;
```

```
    % Control
    % PC: PI with Anti windup
    errpS = pS_SP - pS;
    zV = zV_0 + Kc_pS*errpS + Kc_pS/tauI_pS*interrpS + 1/tauT_pS*interrzV;
    zV_appl = fmax(0, fmin(1, zV));   % u_bar = max(0, min(u, 1))
    errzV = zV_appl - zV;

    % Equations
    dMsdt = m(n) - m_s;
    dHsdt = m(n)*h(n) - m_s*hS;

    alg1  = m_s - zV_appl*kv*(pS-pT);
    alg2  = if_else(cond1, Enth1, if_else(cond2, Enth3, Enth2));
    alg3  = Ms*hS - Hs;
    alg4  = if_else(cond1, Pres1, if_else(cond2, Pres3, Pres2));
    alg5  = pS - 1/(k_p*rho_Ref)*(rho_s-rho_Ref) - p_Ref;
    alg6  = cpW*(TBs-T_Ref) + beta_s*dHvap - hS;
    alg7  = 10^(A_anto-(B_anto/(TBs+C_anto))) - pS;

    Alg  = [Alg;alg1;alg2;alg3;alg4;alg5;alg6;alg7];
    diff = [diff;dMsdt;dHsdt];

    %% Pre-turbine volume (T)

    % Switch logic
    cond1 = beta_s >= 1;
    cond2 = beta_s <= 0;

    dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TBT);
    Enth1 = cpW*(TBT-T_Ref) + dHvap + cpS*(TT-TBT) - hT;
    Enth2 = TT - TBT;
    Enth3 = cpW*(TT-T_Ref) - hT;

    VLT   = (1-beta_T)*MT/rho_T;
    Pres1 = MT*R*TT/(MW*VT) - pT;
    Pres2 = pT*(VT-VLT) - beta_T*MT*R*TT/(MW);
    Pres3 = rho_T - MT/VT;

    % Equations
    dMTdt = m_s - m_T;
    dHTdt = m_s*hS - m_T*hT;

    alg1  = if_else(cond1, Enth1, if_else(cond2, Enth3, Enth2));
    alg2  = MT*hT - HT;
    alg3  = if_else(cond1, Pres1, if_else(cond2, Pres3, Pres2));
    alg4  = pT - 1/(k_p*rho_Ref)*(rho_T-rho_Ref) - p_Ref;
    alg5  = cpW*(TBT-T_Ref) + beta_T*dHvap - hT;
    alg6  = 10^(A_anto-(B_anto/(TBT+C_anto))) - pT;

    Alg  = [Alg;alg1;alg2;alg3;alg4;alg5;alg6];
    diff = [diff;dMTdt;dHTdt];

    %% Turbine
    % Equations
    dwdt      = 1/Mg*(P-L-Dg*(w-w0)); % Frequency calculation, not used

    TurbAlg1  = m_T*sqrt(TT) - phiD*pT;
    TurbAlg2  = TU - TT*(pc/pT)^((R*10^5)/(cpS*MW*10^3));
    TurbAlg3  = W + eta*m_T*cpS*(TT-TU);
    TurbAlg4  = P + W;

    Alg  = [Alg; TurbAlg1; TurbAlg2; TurbAlg3; TurbAlg4];
    diff = [diff; dwdt];
```

```
    %% Condenser inlet flow (U)
    dHvap = dHvap0 + (cpW-cpS)*(TB_Ref - TBU);
    InletAlg1 = m_U - m_T;
    InletAlg2 = cpW*(TBU - T_Ref) + dHvap + cpS*(TU - TBU) - hU;
    InletAlg3 = cpW*(TBU - T_Ref) + beta_U*dHvap - hU;
    InletAlg4 = 10^(A_anto_c-(B_anto_c/(TBU+C_anto_c))) - pc;
    InletAlg5 = TU_real - TBU;

    Alg = [Alg; InletAlg1; InletAlg2; InletAlg3; InletAlg4; InletAlg5];

    %% Condenser (c)
    CondAlg1 = m_U - m_c;
    CondAlg2 = Qc - m_U*hU + m_c*hc;
    CondAlg3 = hc - cpW*(Tc-T_Ref);
    CondAlg4 = Tc - TBU;

    Alg = [Alg; CondAlg1;CondAlg2;CondAlg3;CondAlg4];

    %% Buffer tank
    dMbdt = m_c - m_in;

    diff = [diff; dMbdt];

    %% Append controller eq.
    Dummy1 = dummy_zm - z_m_appl;
    Dummy2 = dummy_m_in_SP - m_in_SP;
    Dummy3 = dummy_v - v;
    Dummy4 = dummy_zV - zV_appl;
    Dummy5 = dummy_mg - mG;
    Dummy6 = dummy_pS_SP - pS_SP;

    Alg = [Alg; Dummy1; Dummy2; Dummy3; Dummy4; Dummy5; Dummy6];

    % Add controller integration terms to ode
    diff = [diff; errmIn; errTs; errpS; errzV; errW];

    %% Solver
    dae     = struct;
    dae.x   = x;     % Differential states
    dae.z   = z;     % Algebraic states
    dae.ode = diff;  % Differential equations
    dae.alg = Alg;   % Algebraic equations

    opts = struct('tf', 1, 'abstol', 1e-9, 'reltol', 1e-9, ...
        'max_num_steps', 10000);

    F = integrator('F', 'idas', dae, opts);

    sol = F('x0', x_0, 'z0', z_0);

    % Function output variables
    x_keys = x;
    z_keys = z;
end
```

E.4 U_Calc

```
function [U] = U_Calc(beta, Tg, A_corr)
    import casadi.*

    err1 = 0.05; % Transistion polynomial range around switching points

    % Define h_s function using CasADi if_else
    h_s = if_else(beta < -0.5, ... % h_s constant outside range
        3603.701367482919 + 2320.2251237389582*(-0.5) + ...
        4628.442414497457*(-0.5).^2 + 21226.60403391536*(-0.5).^3, ...
    if_else(beta < -err1, ... % Sub-cooled liquid
        3603.701367482919 + 2320.2251237389582 * beta + ...
        4628.442414497457*beta.^2 + 21226.60403391536*beta.^3, ...
    if_else((beta >= -err1) .* (beta <= err1), ...  % transistion polynomial
        -7614761.703970*beta.^3 - 11601.125619*beta.^2 + ...
        58270.825342*beta + 5487.306760, ...
    if_else((beta > err1) .* (beta < 1 - err1), ... % Two-phase polynomial
        7420, ...
    if_else((beta >= 1 - err1) .* (beta <= 1 + err1),...% transistion polynomial
        12297807.754146*beta.^3 - 36896455.689820*beta.^2 + ...
        36806951.316308*beta - 12203965.413641, ...
    if_else(beta < 1.5, ...
        1877.5815995356265 - 606.4854766082036*beta, ...  % gas phase
        1877.5815995356265 - 606.4854766082036*1.5)))))); % h_s outside range

    % Define h_g using CasADi if_else
    h_g = if_else(Tg < 400, ...
        115.12460346540009 - 0.06139204288200438*400, ...
    if_else(Tg < 700, ...
        115.12460346540009 - 0.06139204288200438*Tg, ...
        115.12460346540009 - 0.06139204288200438*700));

    % Compute U using h_s, h_g and A_corr
    U = 1 / (1 / (h_g * A_corr) + 1 / h_s)*1/1000; % [kW/K]
end
```

E.5 RGA

```
% RGA Calculation

% Define the process gain matrix G
% Case 1
%u1 = zm, u2 = zv
%y1 = Ts, y2 = ps
% g11 = zm-Ts, g12 = zv-Ts
% g21 = zm-ps, g22 = zv-ps

%Case 2
%u1 = mpSP, u2 = zv
%y1 = Ts,   y2 = ps
% g11 = mpSP-Ts, g12 = zv-Ts
% g21 = mpSP-ps, g22 = zv-ps

G = [-52.4, 2.9;
     -1.25,  -1.05];

% Compute the Moore-Penrose pseudoinverse of G
G_inv = pinv(G);

% Calculate the RGA
RGA = G .* G_inv.';

% Display the results
disp('Process Gain Matrix G:');
disp(G);

disp('Pseudoinverse of G:');
disp(G_inv);

disp('Relative Gain Array (RGA):');
disp(RGA);
```

NTNU

Norwegian University of
Science and Technology