TKP4580 - Specialization Project

Modeling of once-through steam generators

Author: William Davidsen

Trondheim, December 18, 2024

Supervisor: Sigurd Skogestad, IKP

Co-supervisor: Lucas F. Bernardino, SINTEF

Abstract

In this project, a discretized model of a once-through steam generator featuring dynamic phase boundary movement was developed. The model was implemented in Matlab/CasADi and solved using IDAS from SUNDIALS. Its primary application is to study the impact of size on heat recovery in combined cycle systems.

The geometry of the model was scaled with a reference system and tested with steady-state and dynamic simulations. These simulations showed that the model can switch set of equations in the segments, moving the phase boundaries. Step changes in relevant disturbances gave insight in model behavior. The simulations also highlighted issues with the model such as initial condition sensitivity, heat transfer modeling, and noise in mass and pressure.

Future work should address the model-related issues concerning the heat transfer. The correct geometry should be set for the model, and it should be tested against additional process data. Future work should also focus on implementing the other units in the steam cycle and control elements.

Preface

This report is submitted as a result of the specialization project at the Norwegian University of Science and Technology. This project was done in collaboration with SINTEF Energy Research as a part of the Petroleum Reasearch Centre (PETROSENTER) programme. I would like to acknowledge my supervisor Sigurd Skogestad and co-supervisor Lucas F. Bernardino for the invaluable contributions to this project.

Contents

List of Figures

List of Tables

1 Introduction

Offshore oil and gas production is an energy-intensive process. The largest contributor to the emissions is gas turbines for power production, which in 2023 were responsible for 82.91% of the greenhouse gas emissions from Norwegian petroleum activities. The aggregated greenhouse gas emissions from the petroleum sector accounted for about one quarter of Norway's total greenhouse gas emissions. [1]

Some oil and gas installations such as Floating Production, Storage and Offloading (FPSO) vessels are not easily adaptable for emission reduction measures such as electrification. Improvements in efficiency of gas turbines are therefore a viable option for reducing emissions.

One way of improving the gas turbine efficiency is to add a steam cycle to convert excess heat in the gas turbine exhaust to additional power. Combined cycle plants are standard practice in onshore installations but not a widespread technology offshore due to space and weight limitations. [2]

This report is built from previous work [2] [3] where the objectives were to optimize the geometry and minimize the weight of the once-through steam generator (OTSG) responsible for the heat recovery from the exhaust gas.

In this study the primary focus is on the development of a simplified OTSG model. The purpose of the model is to study the impact of size on the heat recovery. A discrete model is developed by segmenting the OTSG into smaller units, which are mathematically formulated with mass and energy balances and implemented in Matlab/CasADi. The system of equations is solved using the IDAS solver from the SUNDIALS suite. Figure 1 illustrates the general methodology for model formulation.

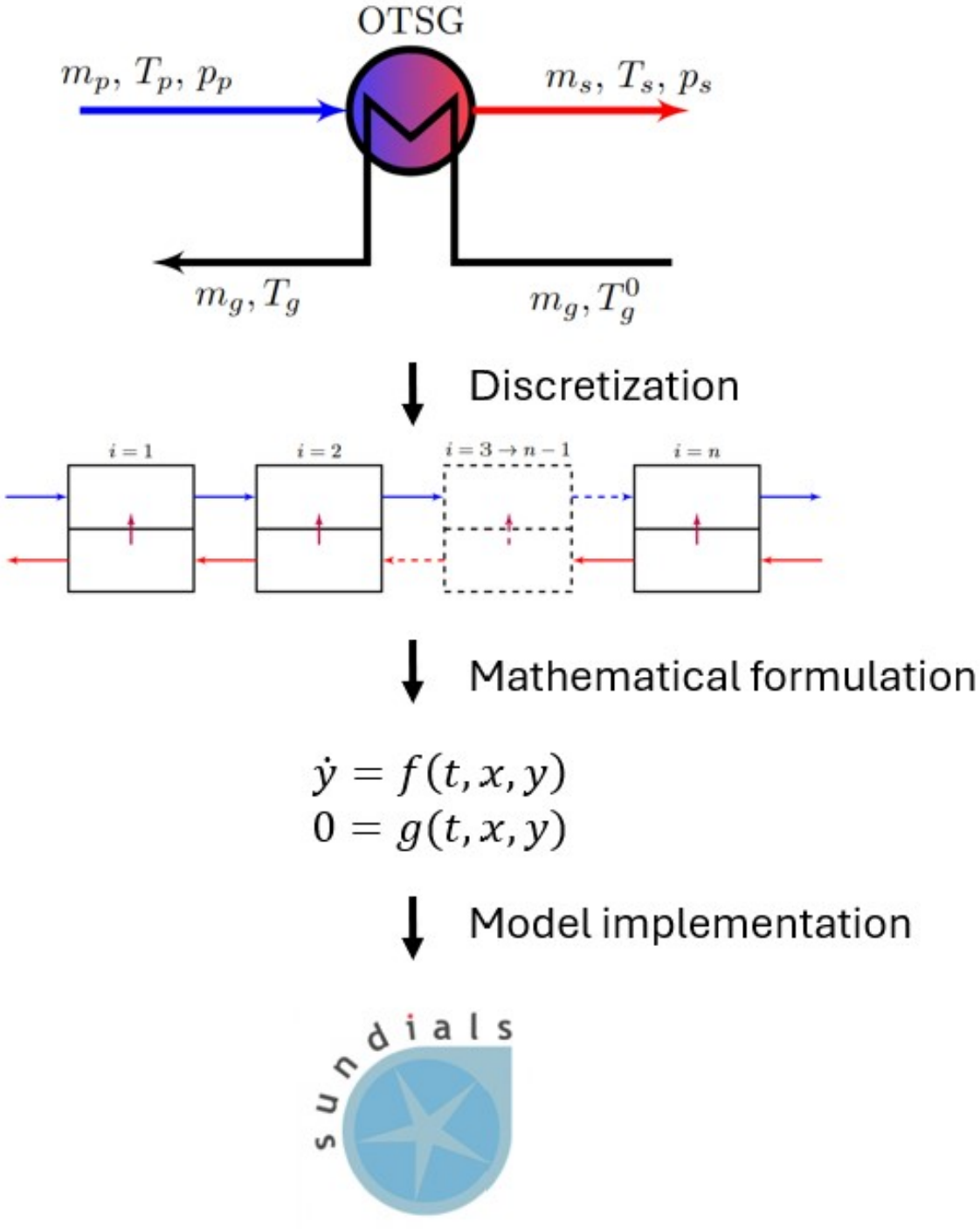


Figure 1: General procedure describing the steps for implementation of the model.

2 System description

The main units in the combined cycle are one Siemens SGT-750 gas turbine (GT), a once-through steam generator (OTSG), a steam turbine (ST) and a condenser. The heat from the GT flue gas is transferred to the steam cycle in the OTSG, which generate superheated steam. The superheated steam is then used to generate power in the ST, after which it is cooled in a condenser and recycled in a closed-loop system. For modeling purposes, however, the steam cycle is treated as an open-loop system. The process flowsheet is shown in Figure 2 and is based on the concept presented in Zotică (2022) [4].

The main focus in this study is on the OTSG unit. Therefore, the GT exhaust gas is considered the model boundary on the hot side of the OTSG. Changes to the GT can be simulated through changing the flue gas mass flow and temperature. The feedwater is considered the model boundary on the cold side, and cold side mass flow can be manipulated to simulate pump action.

Flue gas and cold side inlet mass flow and temperature are considered the main disturbances for the OTSG.

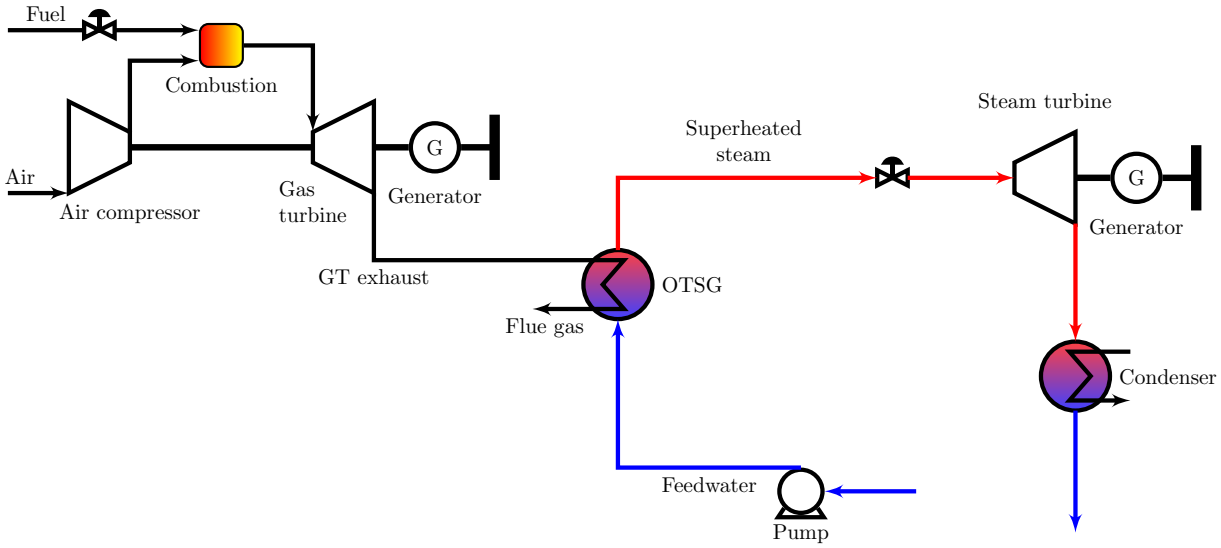


Figure 2: Flowsheet showing the main components of the system: GT, OTSG, ST and condenser

3 Model description

This section aims to explain the equations and assumptions used in the mathematical model of the system.

3.1 OTSG model

The OTSG is modeled by discretization into n equivalent segments. Each segment has a hot and cold side with inflows and outflows on each side. The total OTSG volume and area are equally distributed among the segments. An OTSG segment is illustrated in Figure 3. The segments are linked together with their neighboring segments through the inflows and outflows as shown in Figure 4.

It is assumed perfect mixing on each side in the segment which gives constant temperature within the hot and cold side. Further, it is assumed no heat loss to the environment and no thermal resistance in the tube walls.

Each OTSG segment is modeled with mass holdup M, energy holdup H, cold side temperature T, cold side pressure p, cold side liquid density ρ, cold side gas fraction β, hot side temperature T_g, and heat transfer Q. The flows between the segments are modeled with mass flow m and enthalpy flow h on the cold side, and mass flow m_g on the hot side.

In the OTSG it is a phase change from liquid to steam in the cold side, while the hot side is assumed to always be gas. The fluid in each segment can therefore be in one of the three following states: Liquid, two-phase or steam. The model must therefore contain different sets of equations for each state.

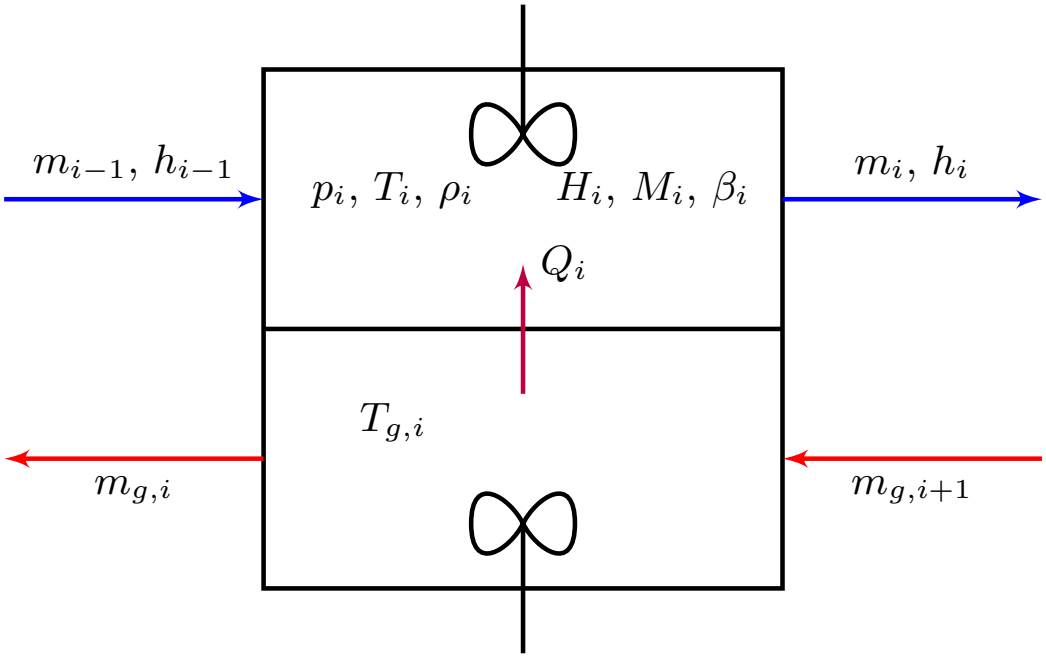


Figure 3: OTSG model segment with inflows and outflows.

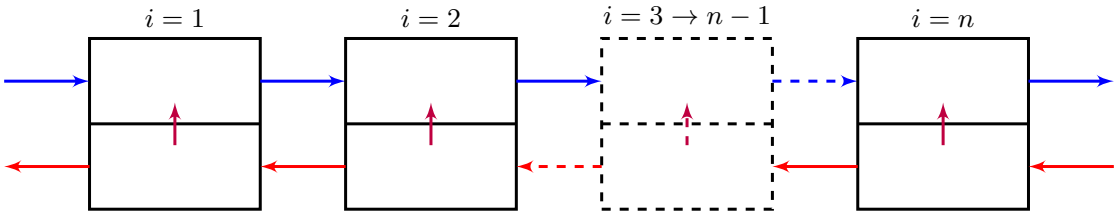


Figure 4: Discretized model of the OTSG showing n segments linked together with the inflows and outflows.

3.1.1 Heat transfer

Since the temperatures are assumed constant in the volumes on each side of the segment, the heat transfer equation becomes as shown in Eq. 1. Where Q_i is the heat transfer, U is the overall heat transfer coefficient, A is the heat transfer area, T_g is the hot side temperature and T is the cold side temperature.

$$Q_i = UA(T_{g,i} - T_i) \tag{1}$$

The overall heat transfer coefficient is limited by the hot side heat transfer coefficient of the OTSG due to the low pressure flue gas. Equal overall heat transfer coefficient for both liquid and steam on the cold side is therefore a reasonable assumption. The OTSG model is designed with equal area for each segment. Consequentially, it is reasonable to assume equal UA in both phases for these simulations. One could also argue that the overall heat transfer coefficient is higher in the two-phase area and that it could be described as a function of the gas fraction β. For simplicity it is assumed constant UA in all phases in these simulations. UA is given as UA_e, $g(\beta)$, and UA_s for liquid, two-phase and steam respectively in Eq. 2 to maintain the capability to implement different UA values for different phases in the model.

$$UA = \begin{cases} UA_e & \beta \leq 0 \\ g(\beta) & 0 < \beta < 1 \\ UA_s & \beta \geq 1 \end{cases} \tag{2}$$

3.1.2 Cold side

Mass balance

Since the cold side only contains pure water, we can assume no reaction. This gives the dynamic mass balance shown in Eq. 3, where M is the mass holdup and m is the mass flow.

$$\frac{dM_i}{dt} = m_{i-1} - m_i \tag{3}$$

Mass flow

The mass flow between the segments on the cold side is modeled as pressure driven flow assuming linear pressure drop as shown in Eq. 4. p is the segment pressure and C_{vd} is the mass flow coefficient. The acceleration term for the mass flow has been neglected, and the flow is considered homogeneous for two-phase flow.

$$m_i = C_{vd}(p_i - p_{i+1}) \tag{4}$$

Energy balance
For the energy balance it is assumed only inner energy as the energy of the system and that the effect of pressure/volume changes can be neglected. Further, it is assumed no work which gives Eq. 5 as dynamic energy balance for the cold side where H is the segment enthalpy and h is the enthalpy flow.

$$\frac{dH_i}{dt} = m_{i-1}h_{i-1} - m_i h_i + Q_i \tag{5}$$

Enthalpy
Due to the assumption of perfect mixing the enthalpy in the segment can be related to the enthalpy flow and the mass holdup of the segment as shown in Eq. 6.

$$H_i = M_i h_i \tag{6}$$

The state equation for enthalpy depends on β, which is unbounded and can be calculated with Eq. 7 for all states. This gives β less than 0 for sub-cooled liquid and larger than 1 for super-heated steam. Here it is assumed constant C_p and that the enthalpy does not depend on pressure.

$$h_i = C_p^w(T_i^{sat} - T_{Ref}) + \beta\Delta H_{vap,i} \tag{7}$$

β can then be used to determine the equation used for the cold side temperature T_i calculation shown in Eq. 8. Where T^{sat} is the saturation temperature, T_{Ref} is the reference temperature, ΔH_{vap} is the energy of vaporization, C_p^w and C_p^s are heat capacity of water and steam respectively.

$$\begin{aligned} h_i &= C_p^w(T_i - T_{Ref}) && \beta \leq 0 \\ T_i &= T_i^{sat} && 0 < \beta < 1 \\ h_i &= C_p^w(T_i^{sat} - T_{Ref}) + \Delta H_{vap,i} + C_p^S(T_i - T_i^{sat}) && \beta \geq 1 \end{aligned} \tag{8}$$

The energy of vaporization depends on the saturation temperature as shown in Eq. 9, where $\Delta H_{vap}(T_{Ref}^{sat})$ is the tabulated reference at the reference saturation temperature T_{Ref}^{sat}.

$$\Delta H_{vap,i} = \Delta H_{vap}(T_{Ref}^{sat}) + (C_p^w - C_p^s)(T_{Ref}^{sat} - T_i^{sat}) \tag{9}$$

Antoine equation
The Antoine equation can be used to describe the relationship between saturation temperature and pressure, as shown in Equation 10. In this formulation, A, B, and C are coefficients obtained from tabulated data.

$$log_{10}(p_i) = A - \frac{B}{T_i^{sat} + C} \tag{10}$$

Pressure and density
The linearized equation of state, neglecting change in density with temperature is shown in Eq. 11. k_p is the compressibility factor, ρ_{Ref} is the reference density and p_{Ref} is the reference pressure. The equation is used for all states, but whether it is used to calculate the pressure or liquid density depends on the segment state. In liquid phase the density is calculated with eq 12 and liquid pressure in eq 11.

In the two-phase it is assumed equivalent pressure in the gas and liquid phases. Eq 11 could then give the liquid density, and Eq 12 the pressure from ideal gas law utilizing the gas holdup and volume.

Further, in gas phase the liquid density does not have any physical meaning, but included to maintain consistent number of equations in solver. The gas pressure is calculated with ideal gas law from Eq. 12.

$$p_i = \frac{1}{k_p \cdot \rho_{Ref}}(\rho_i - \rho_{Ref}) + p_{Ref} \tag{11}$$

$$\begin{aligned} \rho_i &= \frac{M_i}{V} && \beta \leq 0 \\ p_i &= \frac{\beta M_i R T_i}{(V - V_L) M_w} && 0 < \beta < 1 \\ p_i &= \frac{M_i R T_i}{V M_w} && \beta \geq 1 \end{aligned} \tag{12}$$

where V_L is the liquid volume given as

$$V_L = \frac{(1-\beta) M_i}{\rho_i}$$

3.1.3 Hot side

Energy balance
The hot side is modeled with a static instead of dynamic energy balance. Additionally, the enthalpies are expressed in temperature which gives the relation shown in Eq. 13 as the hot side energy balance. m_g is the hot side mass flow, and C_p^g is the flue gas heat capacity.

$$0 = m_{g,i} C_p^g (T_{g,i+1} - T_{g,i}) - Q_i \tag{13}$$

Mass balance
On the hot side it is assumed fixed holdup which gives Eq. 14 as the static mass balance.

$$m_{g,i} = m_{g,i+1} \tag{14}$$

3.1.4 OTSG inlet conditions

The OTSG inlet to segment 1 on the cold side (m_0, T_0, p_0) are given by the flow from the pump (m_p, T_p, p_p). On the hot side the inlet flow in segment n ($m_{g,n}$, $T_{g,n}$) is given by the flue gas flow from the GT (m_g^0, T_g^0).

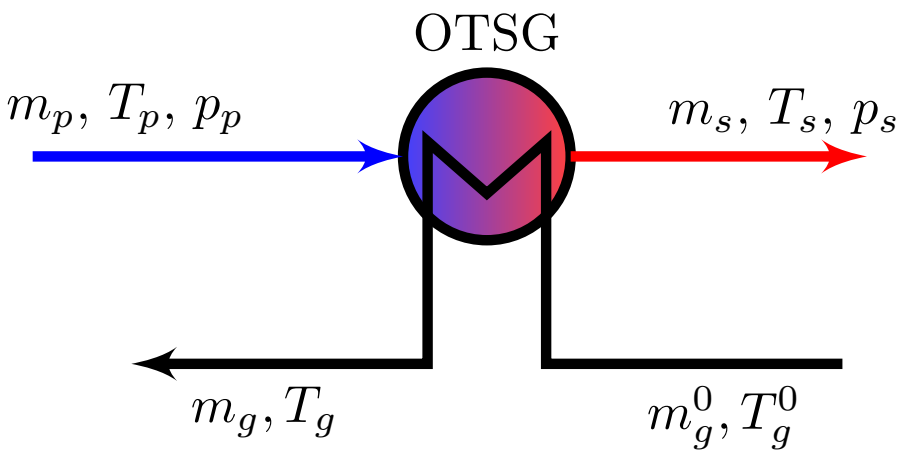


Figure 5: Inflows and outflows of the OTSG linking the discretized model to the rest of the system.

3.2 Summary model equations

The set of equations used in the OTSG model for each state are given in Table 1.

Table 1: OTSG Model equations

Phase	Equation
All phases	$\frac{dH_i}{dt} = m_{i-1}h_{i-1} - m_i h_i + Q_i$
	$\frac{dM_i}{dt} = m_{i-1} - m_i$
	$0 = m_{g,i}C_p^g(T_{g,i+1} - T_{g,i}) - Q_i$
	$Q_i = UA(T_{g,i} - T_i)$
	$m_{g,i} = m_{g,i+1}$
	$m_i = C_{vd}(p_i - p_{i+1})$
	$H_i = M_i h_i$
	$h_i = C_p^w(T_i^{sat} - T_{Ref}) + \beta_i \Delta H_{vap,i}$
	$\Delta H_{vap,i} = \Delta H_{vap}(T_{Ref}^{sat}) + (C_p^w - C_p^s)(T_{Ref}^{sat} - T_i^{sat})$
	$log_{10}(p_i) = A - \frac{B}{T_i^{sat} + C}$
	$p_i = \frac{1}{k_p \cdot \rho_{Ref}}(\rho_i - \rho_{Ref}) + p_{Ref}$
Liquid phase	$h_i = C_p^w(T_i - T_{Ref})$
$\beta_i \leq 0$	$\rho_i = \frac{M_i}{V}$
	$UA = UA_e$
Two phase	$T_i = T_i^{sat}$
$0 < \beta_i < 1$	$p_i = \frac{\beta_i M_i R T_i}{(V - V_L)M_w}$, with $V_L = \frac{(1-\beta_i)M_i}{\rho_i}$
	$UA = g(\beta)$
Gas phase	$h_i = C_p^w(T_i^{sat} - T_{Ref}) + \Delta H_{vap} + C_p^S(T_i - T_i^{sat})$
$\beta_i \geq 1$	$p_i = \frac{M_i R T_i}{V M_w}$
	$UA = UA_s$

3.3 Summary assumptions

- Ideal gas for steam
- Linearized equation of state neglecting temperature effects on ρ
- Neglected fluid acceleration
- Homogeneous flow
- Constant C_p
- Constant UA
- Neglect enthalpy dependency on pressure
- Reference points: $T_{Ref} = 0$ K and $T_{Ref}^{Sat} = 576$ K
- Linear pressure drop
- Perfect mixing
- No heat loss to environment
- No thermal resistance in wall
- Saturation pressure follows Antoine equation
- Equivalent pressure in steam and liquid for two-phase
- No reaction
- No work in OTSG
- Neglect potential and kinetic energy in energy balance
- Neglect effect of pressure/volume changes in energy balance

4 Model Implementation

The system of equations derived in Section 3 can be rewritten on the form:

$$\begin{aligned} \dot{y} &= f(t, x, y, z) \\ 0 &= g(t, x, y, z) \end{aligned} \tag{15}$$

This formulation is called first order differential-algebraic equation (DAE) and can be solved as an initial value problem given an initial condition for the variables. The DAE was implemented in the symbolic framework for numerical optimization tool CasADi in Matlab [5]. The DAE system was solved using the IDAS solver from the SUNDIALS library [6]. IDAS is specifically developed for solving initial value problems for DAE systems. The integration method in the solver is the variable-order, variable-coefficient Backward Differentiation Formula (BDF) [7]. The IDAS solver is designed for easy integration with CasADi and is distributed along with the CasADi package. The parameters used in the IDAS solver are listed in Table 2.

Table 2: Solver parameters used in the simulations.

Parameter	Value
Absolute tolerance	1e-9
Relative tolerance	1e-9
maximum iterations	10000
step length [s]	1

4.1 Building the DAE

The model dynamically appends equations for the OTSG based on desired number of segments (n). The first and last segment contains inlet conditions on cold and hot side respectively as stated in Section 3.1.4. These segments are therefore explicitly written in the model and always passed to the solver. For segment 2 to n-1 the model dynamically append the equations for each segment linking them to their neighboring segments.

Since the state of the fluid in each segment changes with time, the model must be able to dynamically change the set of equations for each segment. This was achieved with a switch on β. CasADi has the built-in function if_else (shown in Eq. 16) that allows for evaluation of logic statements containing symbolic variables. A logic statement containing β can therefore be used to change the set of equations given to the solver.

$$if\_else(\text{Condition},\ \text{Condition true},\ \text{Condition false}) \tag{16}$$

The derived model in Section 3 has 3 sets of equations which means that the system has two switching points corresponding to the dew and bubble point. The switching conditions on β listed in Eq. 17 can therefore be used to switch between the states.

$$\begin{aligned} \text{Condition 1} &= \beta \geq 1 \\ \text{Condition 2} &= \beta \leq 0 \end{aligned} \tag{17}$$

Two switching conditions can be handled with the following nested if_else statement shown in Eq 18, where Gas, Liquid and Two-phase represents set of equations for each state.

$$if\_else\,(\text{Condition 1},\ \text{Gas},\ if\_else(\text{Condition 2},\ \text{Liquid},\ \text{Two-phase})) \tag{18}$$

4.2 Solving the DAE

After the DAE is constructed, it is passed to the IDAS solver and solved over the given step in time. The model then return the value of the variables at the new step in time. The model is then called again with the new initial values to solve for the next step in time, and so on. The function used for building and solving the DAE is shown in Appendix C.1. The code used to run the model and store the results is shown in Appendix C.2.

5 Reference system validation

The system presented in Zotică (2020) [8] was reproduced in order to compare the developed model in this paper with an existing model. the reference system is modeled with economizer, drum and super-heater with fixed point of vaporization controlled with bypass of the economizer. The solver used in the paper and reproduced model are different: The reference used the built-in *ode15s* solver in MATLAB, while the reproduced model is using IDAS from the SUNDIALS suite.

During the implementation an error in the code was found. In the differential equation for drum temperature T_D (Eq. 19) derived from the combined heat and energy balance it should be C_p^w, not C_p^s. The error probably has no steady-state effect as these terms cancel anyways when $m_M = m_D$. However, this probably has some affect on the dynamics of the system. This was investigated by comparing an uncorrected and corrected model with the reference data.

$$\frac{dT_d}{dt} = \frac{1}{M_D c_p^w}(m_M(H_m - c_p^s T_D) - m_D(H_D - c_p^s T_D) + Q_D) \tag{19}$$

Further, some constants were different in the code and paper. For example, K_v, which was given as 2.32 kg/bar in the paper and 2.3624 kg/bar in the code. In these cases, the value given in the code was used.

5.1 Steady-state

5.1.1 Open-loop comparison

The system used for the open-loop comparison only contained the level controller for M_d to stabilize the inventories. This has no steady-state effect. The attemperator-bypass (m_{BE}) was fixed to its nominal value (0.6309 kg/s) in order to compare the system with the nominal state. The steady-state of the system with and without correction were compared to the nominal operating conditions as shown in Table 3. The table shows that the correction did not influence the steady-state, and that it is the same as the nominal point to the given precision of decimals.

Table 3: steady-state of the open-loop system with and without correction compared to the nominal operating point in the reference paper.

Variable	Reference [8]	Reproduced	Reproduced corrected
M_d [kg]	3000	3000.0000	3000.0000
T_d [K]	576	576.0136	576.0136
T_s [K]	868	868.0353	868.0353
T_e^g [K]	423	423.1089	423.1089
P [MW]	16.55	16.5526	16.5526

5.1.2 Closed-loop comparison

The system was also compared to the reference in turbine driven operation mode, where the pressure of the super-heated steam is used to control GT load and power is used to control the turbine inlet valve. The controllers were tuned with the same tuning as given in the reference.

The comparison is shown in Table 4. It shows that the correction did not influence the steady-state of the closed-loop system as for open-loop. The comparison also shows that the error relative to the reference is small.

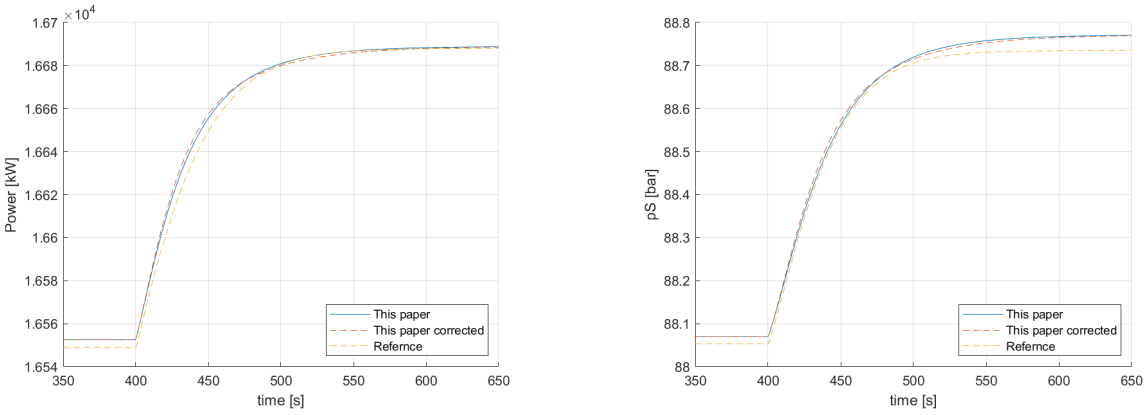
Table 4: Steady-state of the reproduced turbine driven system compared to the reference.

Variable	Reference [8]	Turbine driven	Turbine driven corrected
M_d [kg]	3000.0060	3000.0000	3000.0000
T_d [K]	575.9998	575.9999	575.9999
T_s [K]	868.0014	868.0669	868.0669
p_T [bar]	83.0530	83.0539	83.0539
T_e^g [K]	423.0043	423.0619	423.0619
P [kW]	16548.9597	16548.9507	16548.9507

5.2 Open-loop dynamic comparison

5.2.1 Step in m_g

The systems were tested with a 1% increase in flue gas mass flow to simulate increased GT load. The dynamic response in produced power and saturated steam pressure are shown in Figure 6. The figure shows that the difference between the reference and the reproduced model is small, and can be explained by a slight difference in steady-state. Further, the figure shows that the transients have similar behavior.



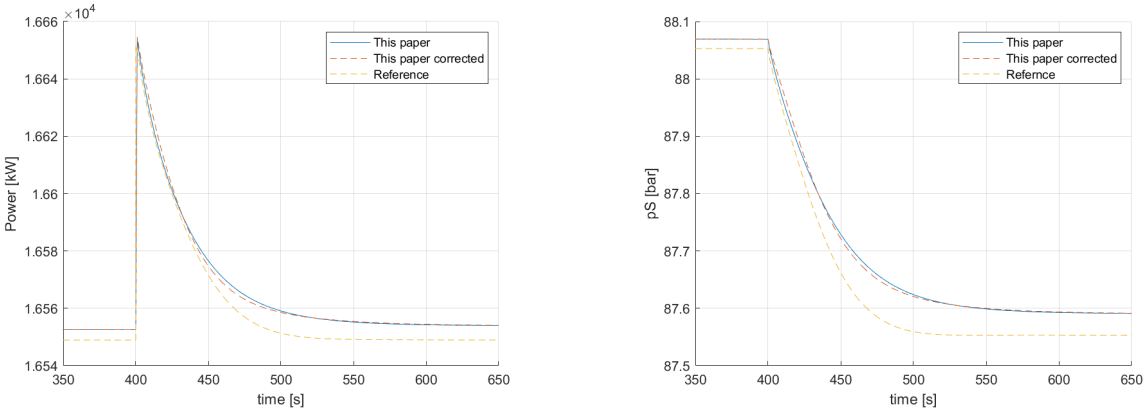
(a) Power produced

(b) super-heated steam pressure

Figure 6: Dynamic response of the reproduced systems and the reference for step change in m_g.

5.2.2 Step in z_v

The systems were also tested with a step change in turbine inlet valve opening from 0.9 to 1. The response compared to the reference is shown in Figure 7. The figure shows that reference and the reproduced models have a small difference, which can be explained with difference in initial value. The transients seems to have similar behavior. Further, comparing the corrected and uncorrected systems shows little to no difference.



(a) Power produced

(b) Super-heated steam pressure

Figure 7: Dynamic response of the systems and the reference for step change in z_V.

6 Simulations

The aim of this part is to provide an overview of the performance and limitations of the OTSG model. This will be done with steady-state comparison with the reference, dynamic simulations with step changes in relevant disturbances, and steady-state dependency on n.

6.1 Simulation conditions

The constants used in the simulations are listed in Appendix B, and are the same as in the reference paper [8], except for UA and V. Additionally, some constants used in this work were not used in the reference due to differences in modeling, for example k_p, p_{Ref} and ρ_{Ref}. These variables were obtained from other sources.

The UA constant was scaled linearly so that the heat transfer of the system became closer to the reference paper [8]. A linear scaling resulting in UA = 177 kW/K gave reasonable results which will become evident in Section 6.2. The linear scaling can be seen as changing the geometry of the OTSG so that it is matching the reference.

The reference paper did not provide the volume explicitly, and was set to 1 m^3. The volume does not have a steady-state effect, however it is important for the dynamics. Both V and A will in future work be given by provided geometry.

The simulations were performed with 37 nodes. The inlet and outlet conditions of the OTSG are given in Table 5. The conditions are the same as the reference paper [8] nominal operating point.

Table 5: Inlet and outlet conditions for the simulations.

Variable	Value	Unit
T_g^0	1273.15	K
T_p	318.15	K
m_g	31.4018	kg/s
p_{in}	89	bar
p_{out}	88	bar

6.2 OTSG steady-state simulations

The steady-state of the OTSG model was scaled to the reference system. The comparison of cold side mass flow and outlet temperatures on both sides are given in Table 6. It shows that the developed OTSG model has around 1 K higher outlet temperature on the cold side and 0.5 K lower outlet temperature on the hot side than the reference system. Since the cold side outlet is hotter and hot side outlet is colder than the reference, the difference could be reduced by reducing the linear scaling. However, the difference between the systems are considered small enough with the current scaling.

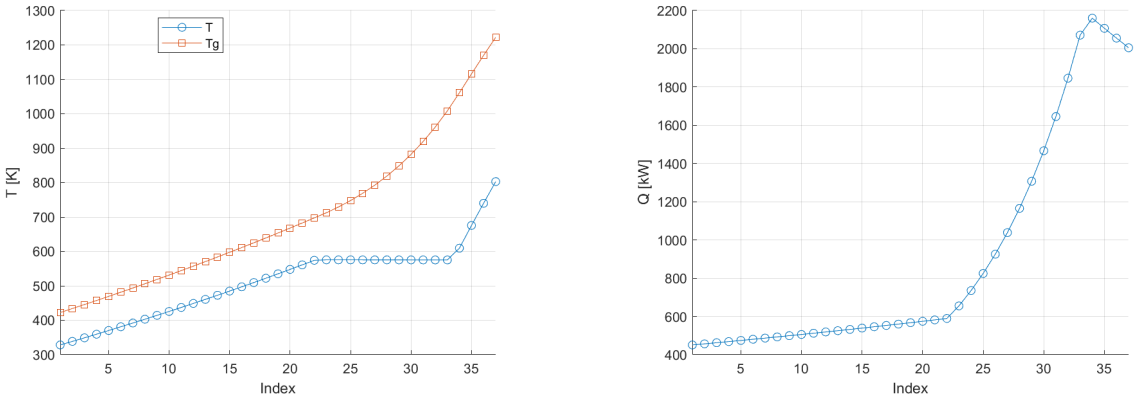
Figure 8 shows the temperature profiles and heat transfer along the segments. The figure shows that the OTSG, at the current conditions, is pinching at the cold side inlet which maximize the energy transfer. The heat transfer is highest for steam and late in the two-phase as the temperature gradient is highest here.

The figure also shows the switching of equations in segment 22 and 33. At index 22 the $T_i = T_i^{sat}$ equation becomes active which by the Antoine equation (Eq. 10) makes the temperature slowly decline with the pressure in the two-phase region. The switching points corresponds to where β becomes positive and larger than 1 in Figure 10. Since the model was initialized with liquid in all segments, this confirms that the model is able to switch set of equations from liquid to two-phase and two-phase to steam when converging to the steady-state.

Further, Figure 9 shows that the holdup in each segment is reduced with increased fraction of steam. This makes sense as steam displace more volume than liquid at the same pressure and temperature.

Table 6: OTSG model comparison with the reference model.

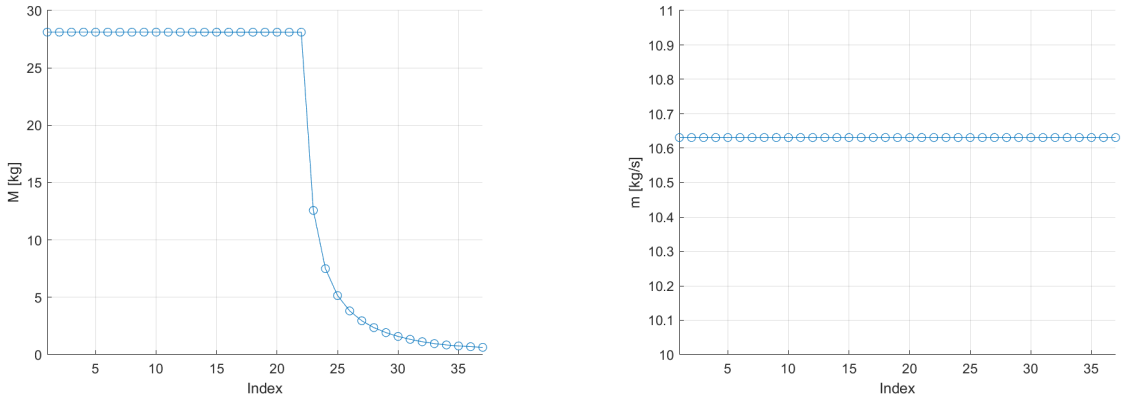
Variable	Reference [8]	OTSG model
m [kg/s]	10.6309	10.6309
T_s [K]	802	802.8858
T_g [K]	423	422.5514



(a) Temperature profile

(b) Heat transfer

Figure 8: OTSG temperature profiles and heat transfer along the segments.



(a) Mass holdup in segment

(b) Mass flow between segments

Figure 9: OTSG segment cold side holdup and mass flow.

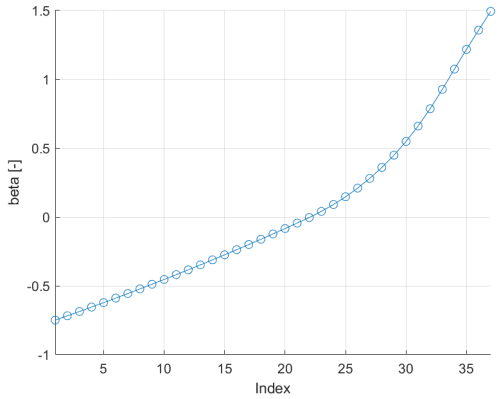


Figure 10: Gas fraction (β) along the OTSG segments.

6.3 Steady-state variation with n

The effect of the number of segments on the OTSG outlet temperatures is shown in Table 7. It shows that there is an effect on the steady-state values. It is also observed that the difference is smaller at larger n which suggests that it is converging at a slow rate. It is therefore important to use same n when comparing OTSG simulations.

The variation in outlet temperatures with n is assumed to be related to the heat transfer equation used for the system (Eq. 1) and will be further discussed in Section 7.3. Furthermore, it was not possible to apply any higher n to the solver at the current conditions.

Table 7: Effect of n on the steady-state value of outlet temperatures.

Variable	n = 30	n = 37	n = 45	n = 52	n = 59
T_s [K]	799.22	802.89	805.75	807.54	808.88
T_g [K]	425.53	422.55	420.22	418.78	417.68

6.4 Dynamic simulations

The OTSG model was tested with step changes in relevant parameters such as hot side inlet temperature and mass flow, to see how disturbances affect the system. Positive and negative steps of the same magnitude were applied to the system.

6.4.1 Step in m_g^0

The step changes in m_g^0 were set to $\pm 10\%$ of the nominal value. Figure 11 shows the step in m_g^0 applied to the system. The system response on the negative step change is shown in the figures below.

The nominal and new steady-state of the system are shown for cold side temperature and mass holdup in Figure 12. The figure shows that when m_g is reduced, the dew and bubble points are moved further down the OTSG. This makes sense as the heat transfer is reduced due to smaller temperature gradient across the segments. The figure also shows that the model is able to switch the set of equations for steam to two-phase and two-phase to liquid transitions.

Figure 13 shows the response in cold side mass flow and pressure for segment 21 which is close to the middle of the OTSG. The figure shows that the response has noise with same number of peaks as liquid segments switching to two-phase, before returning to the same steady-state. This behavior will be further investigated in Section 6.5.

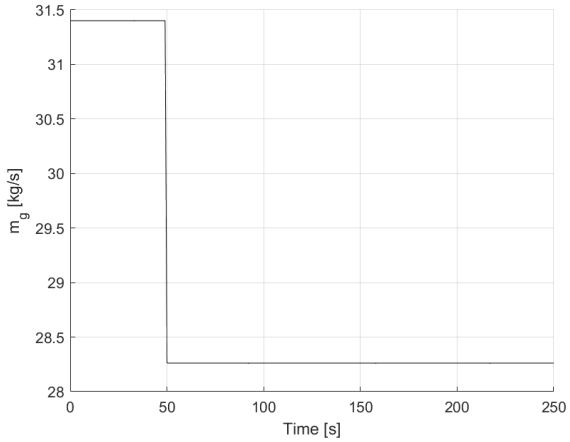
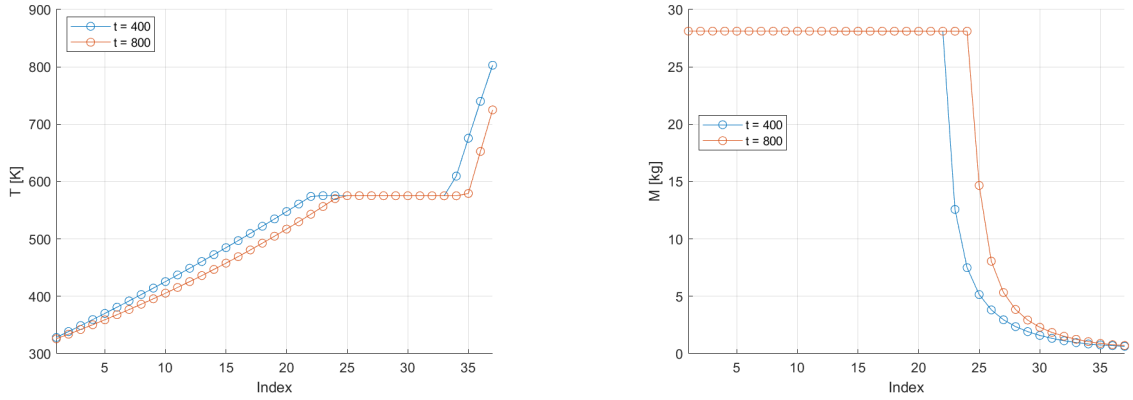


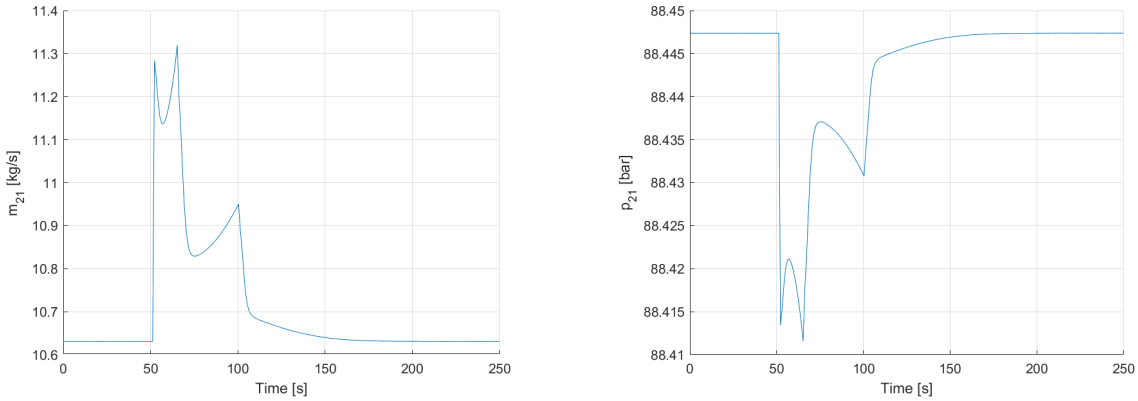
Figure 11: Negative step change in flue gas flow rate applied to the system.



(a) Cold side temperature

(b) Mass holdup

Figure 12: Cold side temperature and mass holdup profiles along the OTSG segments.



(a) Mass flow

(b) Pressure

Figure 13: System response in cold side mass flow and pressure for segment 21.

6.4.2 Step in T_g^0

The step changes in T_g^0 were set to $\pm 10\%$ of the nominal value. Figure 14 shows the applied positive step to the system.

Figure 15 shows the change of steady-state caused by the step change. The figure shows that when T_g^0 was increased the dew and bubble points were moved closer to the cold side inlet. The increased hot side temperature gave higher heat transfer, which resulted in higher cold side outlet temperature. This step change also shows that the model is able to switch the set of equations from liquid to two-phase and two-phase to steam.

As for the step in m_g the pressure and mass on the cold side had noise for changes in T_g^0. Additionally, the noise was also present in segments that were not switching equations as shown for segment 31 in Figure 16. This suggests that the noise is propagating through the system, and will be further looked into in Section 6.6.

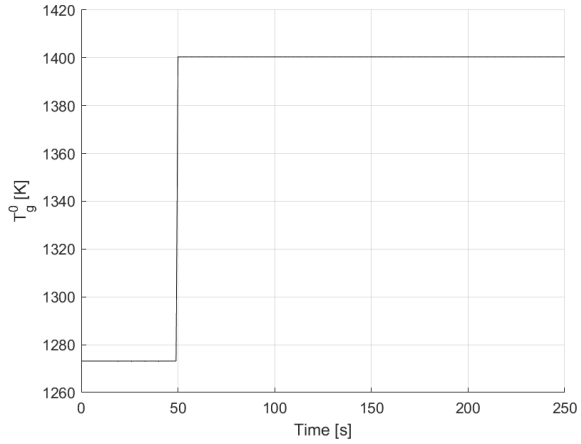
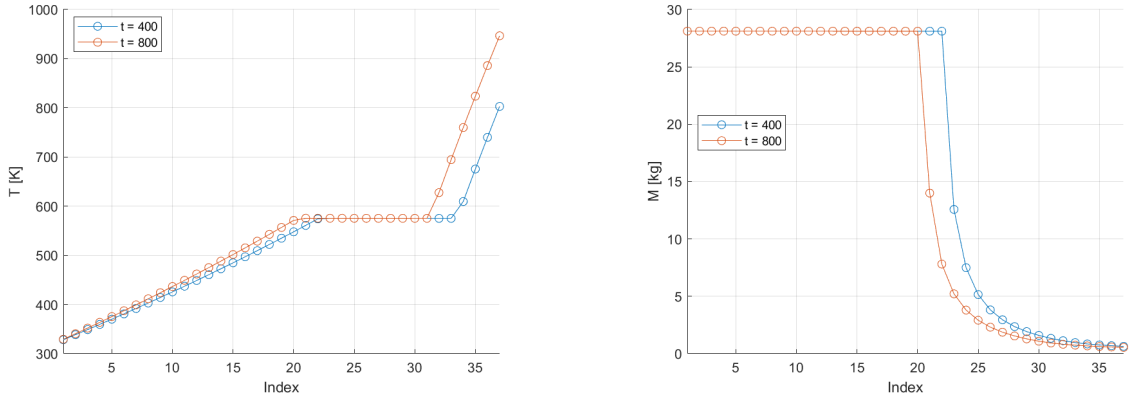


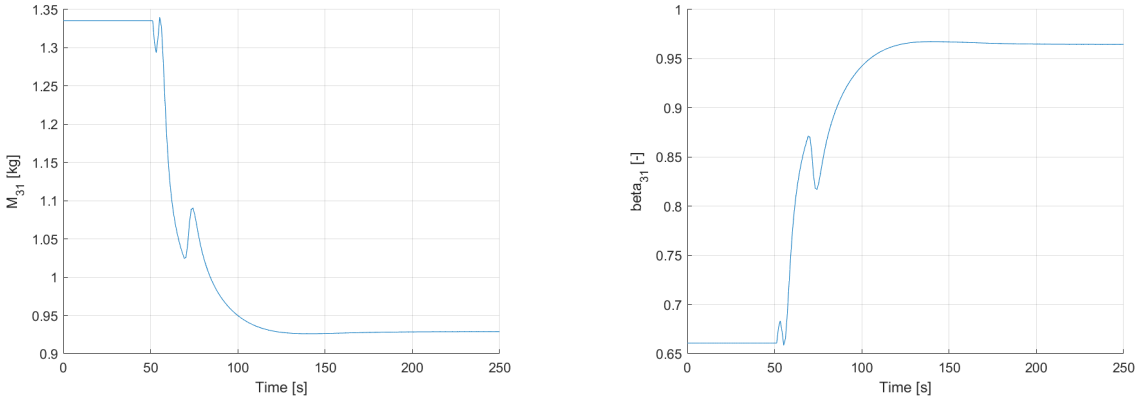
Figure 14: Negative step change in T_g^0 applied to the system.



(a) Temperature

(b) Mass holdup

Figure 15: Change in steady-state for cold side temperature and mass holdup.



(a) Mass holdup (b) β

Figure 16: Response in mass holdup and gas fraction β for segment 31.

6.4.3 Step in T_p

The step changes in cold side inlet temperature T_p were set to ± 10 K from the nominal point. The negative step change in temperature is shown in the figures below.

Figure 18 shows the change in steady-state for the applied step change. The figure shows that the system has a small change in β in all segments, and none of the switches were triggered. Further, the figure shows that there were small changes in mass holdup for liquid and steam, but the change in holdups with two-phase were larger. For example, in segment 23, the holdup had a notable increase.

Figure 19 and 20 shows the holdup and temperature in segment 11 and 37 respectively. The time profiles look smoother than for the other step changes described above, and could be due to no switching of equations in the segments.

The temperature profile in Figure 20 first shows an inverse response before converging to its new steady-state. This is likely caused by the holdup increase in the two-phase region temporary reducing the flow of cold fluid in the consecutive segments increasing the temperature of the reduced flow. The inverse response was not observed in liquid segments prior to two-phase segments such as segment 11 as shown in Figure 19.

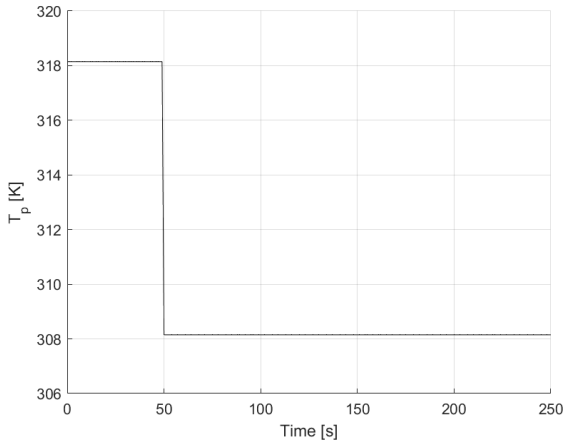
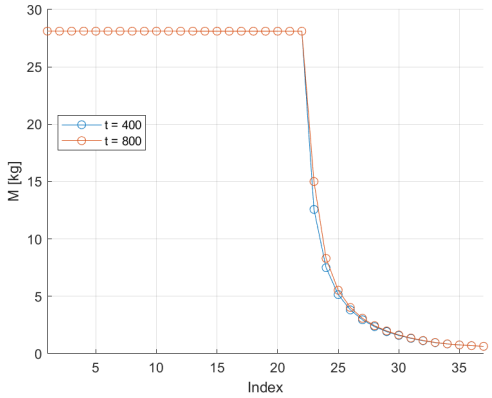
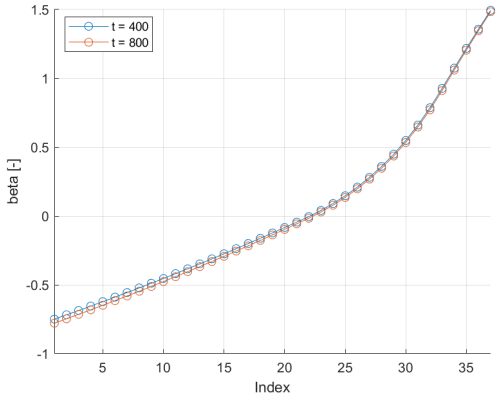


Figure 17: Applied negative Step change in T_p.

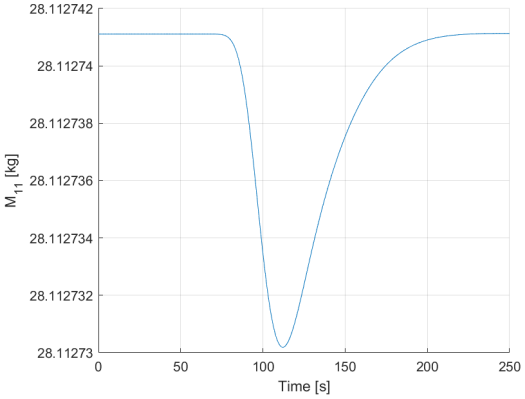


(a) Mass holdup

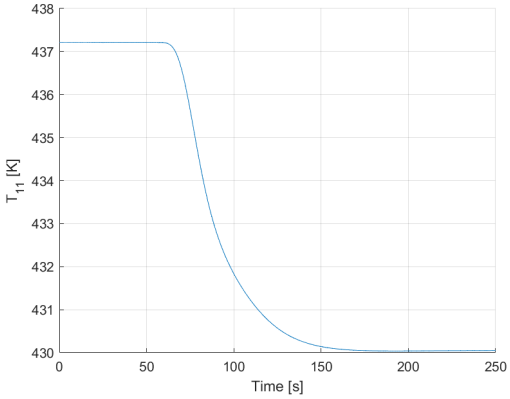


(b) β

Figure 18: Change in steady-state for cold side mass holdup and β.

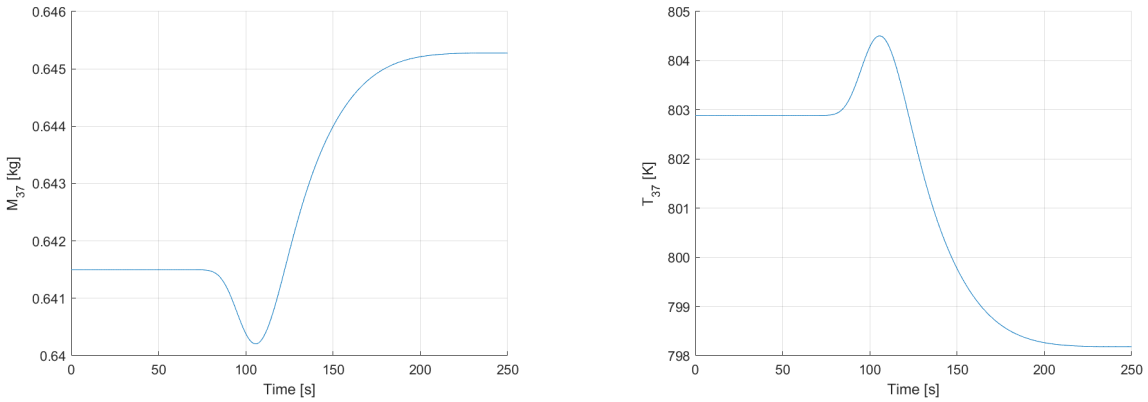


(a) Mass holdup



(b) Temperature

Figure 19: Response in cold side mass holdup and temperature for segment 11.



(a) Mass holdup

(b) Temperature

Figure 20: Response in cold side mass holdup and temperature for segment 37.

6.4.4 Step in p_p

A step change in inlet pressure of ±0.5 bar was applied to the system, resulting in 1.5 and 0.5 bar pressure drop over the OTSG respectively. The negative step on inlet pressure failed which will be discussed in Section 7.2. The magnitude of the step change was then reduced to -0.25 bar and the response is shown in the figures below.

Figure 22 shows the change in steady-state for the cold side temperature and pressure. The reduced pressure gradient gave a lower cold side mass flow through the system which increased the fluid temperature. This caused the fluid to evaporate earlier in the OTSG, and the pinch point was moved from the cold side inlet to the bubble point as shown in Figure 23.

Further, Figure 24 shows the response in cold side mass and pressure for segment 21. As for the other step changes, the transients are noisy here, matching the number of two-phase segments switching to liquid.

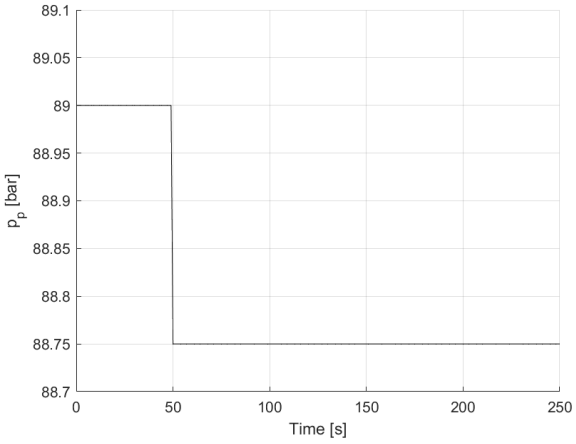
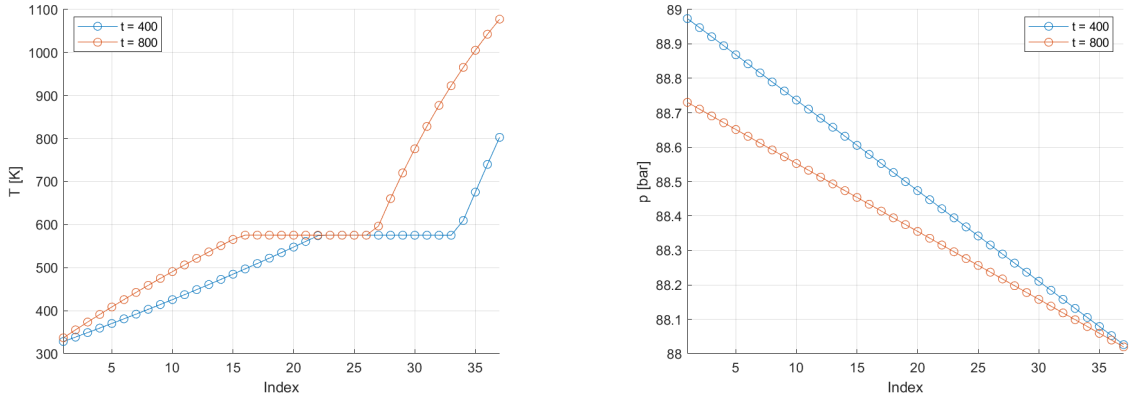


Figure 21: Step change in cold side inlet pressure p_p.



(a) Temperature (b) Pressure

Figure 22: Change in steady-state for cold side temperature and pressure.

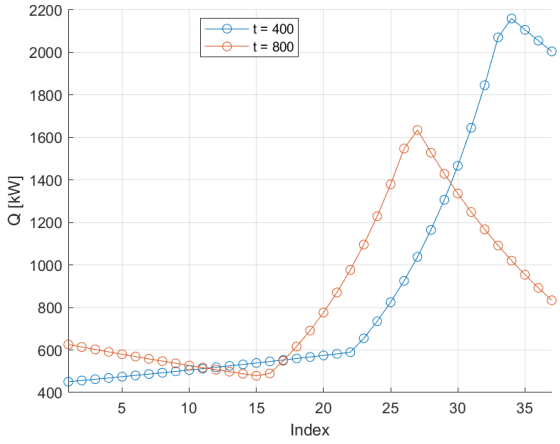
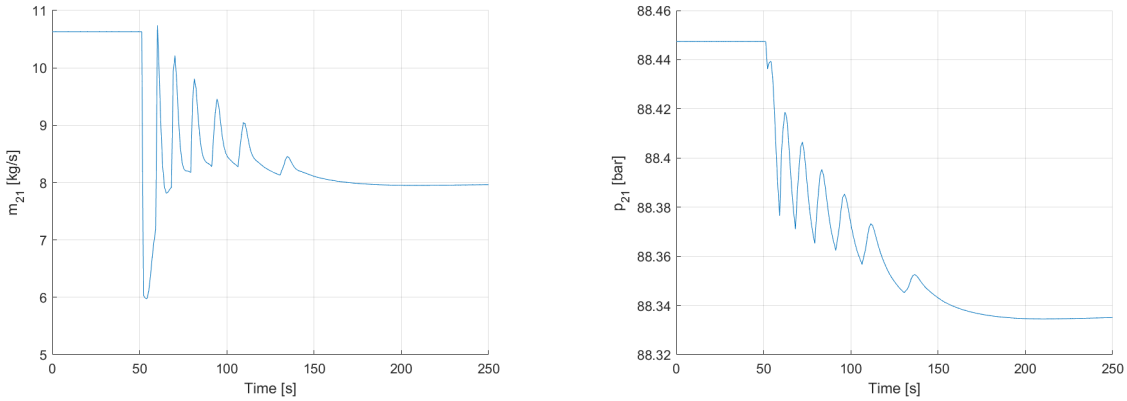


Figure 23: Change in steady-state for heat transfer.



(a) Mass flow (b) Pressure

Figure 24: Response in cold side mass flow and pressure for segment 21.

6.4.5 Step in p_s

The system was also tested with step changes in the outlet pressure of the OTSG. The step changes were applied with a magnitude of ± 0.5 bar. The solver was not able to handle a 0.5 bar increase, which is likely the same issue as for p_p as both steps resulted in a 0.5 bar pressure drop over the OTSG. The negative 0.5 bar step is shown in Figure 25, resulting in 1.5 bar pressure drop.

Figure 26 shows the steady-state temperature and pressure profile along the OTSG segments. It shows that increased pressure drop on the cold side gave increased mass flow which reduced temperature in the cold side outflow. The magnitude of the applied disturbance had a significant impact on the outlet temperature and resulted in a two-phase outflow.

Figure 27 shows the response in cold side temperature and pressure for segment 31. The figure shows that the response is the two variables are similar. This is due to the Antoine equation linking the temperature and pressure as the state is two-phase for the segment.

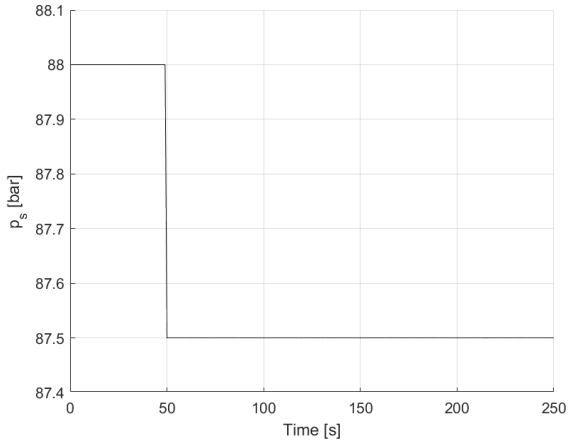
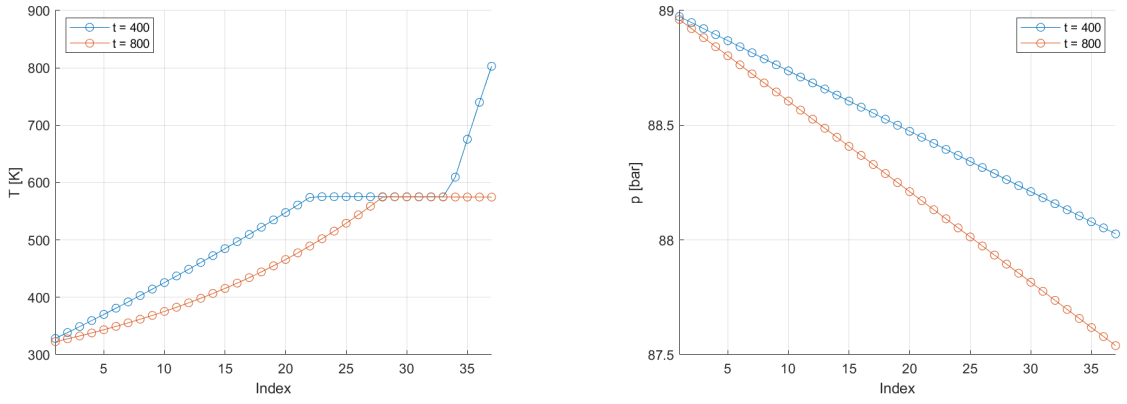


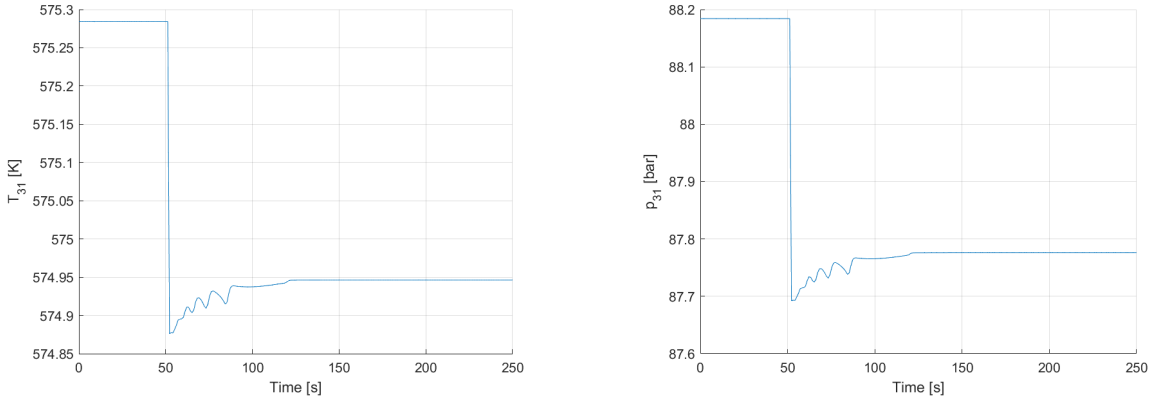
Figure 25: Negative step change in OTSG outlet pressure p_s.



(a) Temperature

(b) Pressure

Figure 26: Change in steady-state for cold side temperature and pressure.



(a) Temperature

(b) Pressure

Figure 27: Response in cold side temperature and pressure for segment 31.

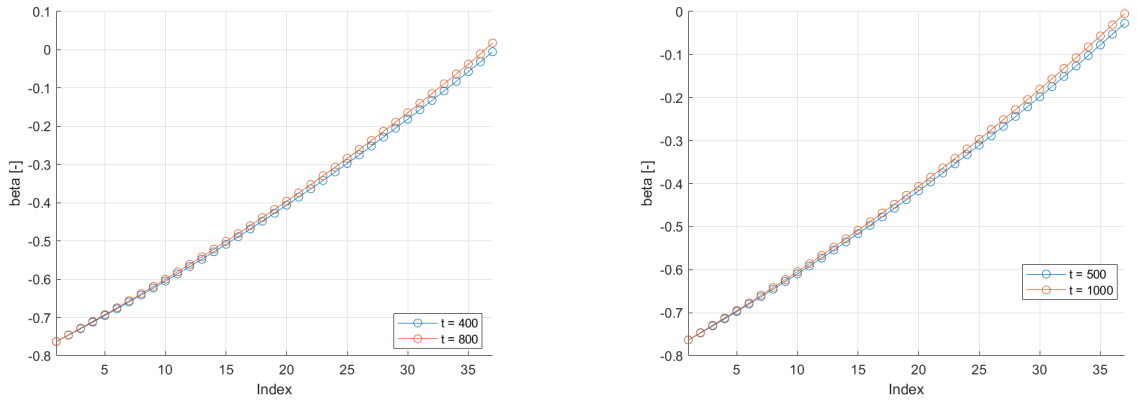
6.5 Switching point analysis

Based on the dynamic simulations, the switch logic generally seems to work well. However, as seen in multiple dynamic simulations there are sudden changes in mass flow and pressure during switching. It is therefore interesting to have a closer look on what happens on each switching point.

6.5.1 Liquid switching point

The model was brought to a steady-state with liquid in all segments. It was then applied two different step changes in hot side inlet temperature: One that caused one segment to switch state to two-phase and one step that did not cause any switch. Figure 28 shows the change in β for a step change giving a switch in one segment and a step change that did not cause a switch.

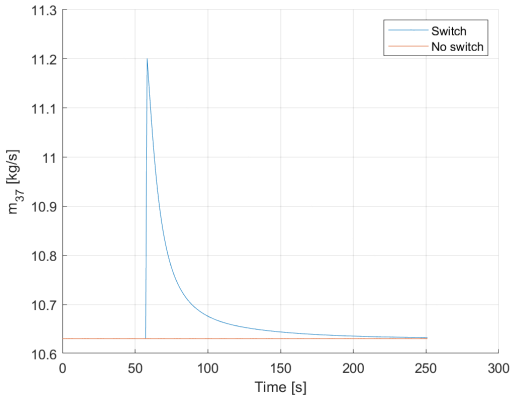
Figure 29 shows the difference in response for mass flow and pressure when the switch was triggered and not triggered. From the figure it is clear that switching from liquid to two-phase does cause a change in mass flow and pressure.



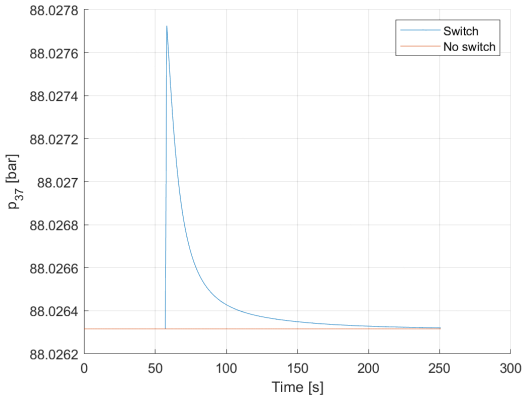
(a) Switch

(b) No switch

Figure 28: Change in steady-state profile of vapor fraction β for both step changes.



(a) Mass flow



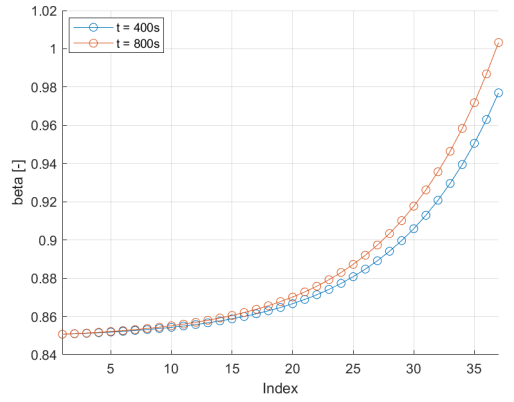
(b) Pressure

Figure 29: Response in mass flow and pressure for both the applied step changes to the system.

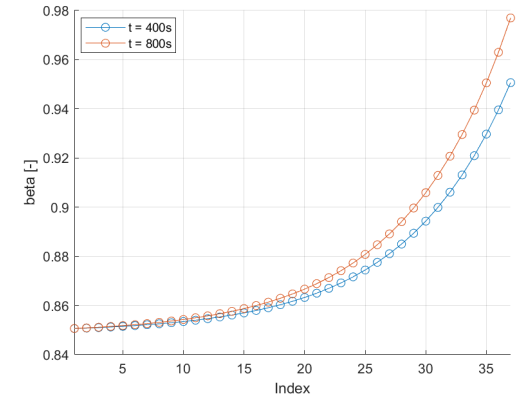
6.5.2 Gas switching point

The model was brought to a steady-state with two-phase and β close to 1 in all segments. It was then applied two step changes in hot side gas inlet temperature: One that caused a switch to gas phase in the last segment and one that did not cause a switch, as shown in Figure 30.

Figure 31 shows the mass flow and pressure response for both step changes. It shows that the change in mass flow and pressure are small and similar in both cases. This suggests that the change of equations from two-phase to gas does not cause noise in mass flow and pressure.

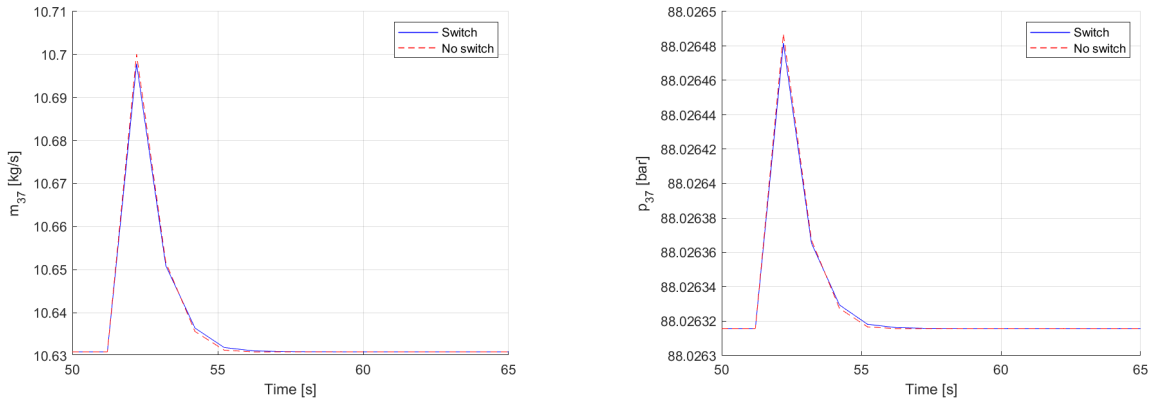


(a) Switch



(b) No Switch

Figure 30: Steady-state in vapor fraction before and after the step changes. The step change with switch is shown on the left, while the step change without switch is shown on the right.



(a) Mass flow

(b) Pressure

Figure 31: Response in mass flow and pressure to the step changes with and without switching.

6.5.3 Change of equations

The noise in mass and pressure propagating through the system due to liquid switching is assumed to be related to Eq. 12. Figure 32 shows the mass holdup as a function of β for constant p, ρ, and T at the bubble point. The figure shows that the change in mass holdup is large for small changes in β when it is close to 0. Further, the figure shows that the mass holdup for two-phase is converging towards both liquid and gas mass holdup at the switching points which makes it continuous.

The rapid change in mass holdup then cause rapid changes in mass flow and pressure through the system. These changes are fast since the flow is modeled with neglected acceleration term which would have dampened the rapid changes in flow. The changes are not observed in the liquid which can be explained by the assumption that density does not depend on the temperature. This suggests that the noise in mass and pressure are physical rather than numerical in the sense that it is induced by the modeled equations and assumptions.

The holdup-beta relation explains the behavior for the liquid switch and why it is not observed in the liquid phase. However, it does not explain why the disturbances are also present in the gas phase, which can be explained by a similar variation in mass holdup with temperature from Eq. 12.

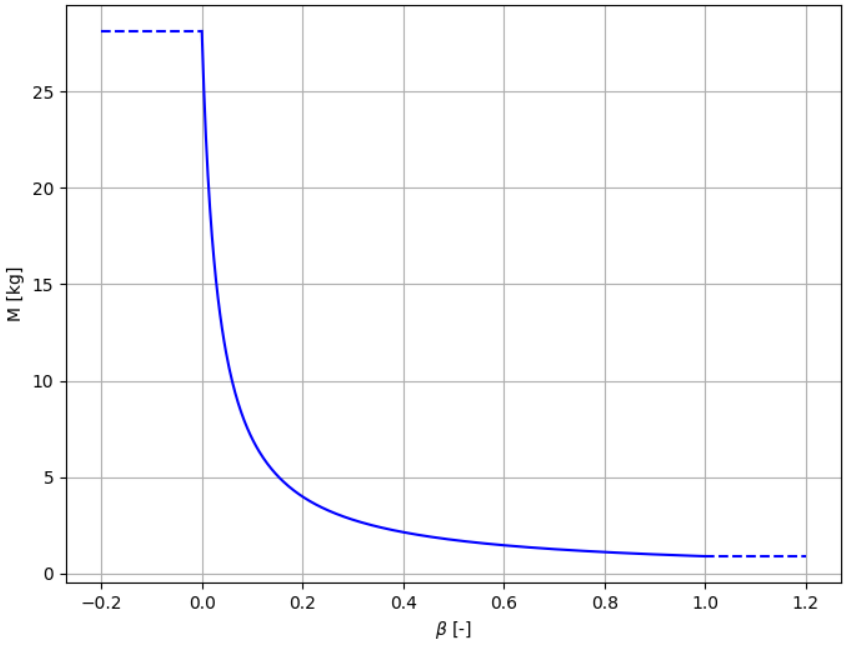


Figure 32: Mass holdup as a function of β for given p, ρ and T.

6.6 Mass and pressure propagation

Figure 33 shows cold side mass flow for selected segments in the OTSG during the step change in m_g^0 presented in Section 6.4.1. The figure shows that the liquid segments (1, 10 20) have an increase in mass flow during the transient, and the two-phase and steam segments (30, 37) have a decrease. For the liquid segments it was not observed any delay in propagation which makes sense as the liquid is considered air-free making pressure wave propagation fast.

Further, it was also not observed any delay for the steam segments. The propagation in gas systems can be shown through the time constant. Eq. 20 can be used to calculate the time constant for gas systems [9]. Since the residence time is small, and pressure drop is small relative to the absolute pressure, the time constant is small. At the simulated conditions, the time constant is approximately 0.2 ms which shows that the propagation is very fast.

$$\tau = \frac{1}{4}\frac{M}{q}\frac{p_{in} - p_{out}}{p} \tag{20}$$

The simulation results are only stored at each step length, and since this was set to 1 s it is not possible to differ responses faster than this. It is therefore not possible to detect the delay in propagation at the current simulation conditions. The solver has variable step size and will therefore average out the errors meaning that the simulations will still be correct on longer time scales.

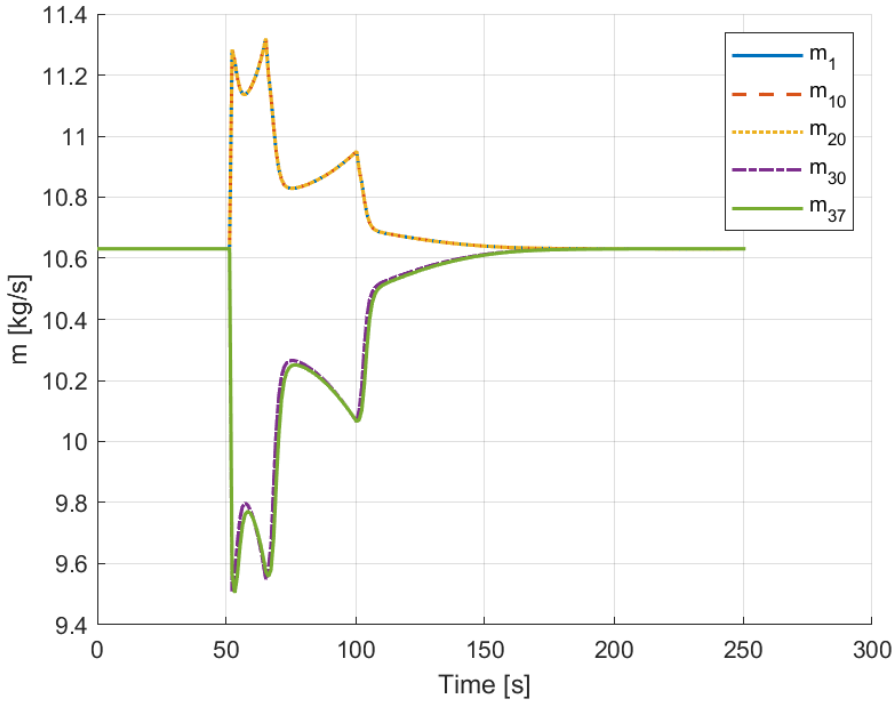


Figure 33: Mass flows from different segments in the OTSG to show that the propagation is very fast in all phases.

7 Discussion

Based on the dynamic simulations, the model seems to function as intended; however, during testing and implementation, a few challenges arose such as initial guess sensitivity and infeasible operating points. The model also showed room for improvements in steady-state variation with number of segments. These issues will be discussed in this section.

7.1 Initial guess sensitivity

7.1.1 Number of segments (n)

The solver seemed to struggle with initializing for a large number of segments (>60). This may be due to numerical problems caused by the discretization of the pressure drop, as it was able to initialize at a higher n with a larger pressure drop and other conditions were equal. However, simulating with a higher number of segments is not a goal by itself, as it would increase the computation time and may not give more realistic results.

7.1.2 k_p stiffness

The solver also had issues with stiffness in Eq. 11 as the system could not be initialized with $k_p = 4.58 \cdot 10^{-5}$ bar^{-1} which is the tabulated value for water [10]. However, the model was able to run at $4.58 \cdot 10^{-4}$ bar^{-1} and therefore this was used in the simulations. As a consequence, ρ is probably not realistic for the liquid phase at around 1040 kg m^{-3} at the cold side inlet (45°C, 89 bar) compared to the tabulated value 1000.91 kg m^{-3} (25°C, 89 bar) [11].

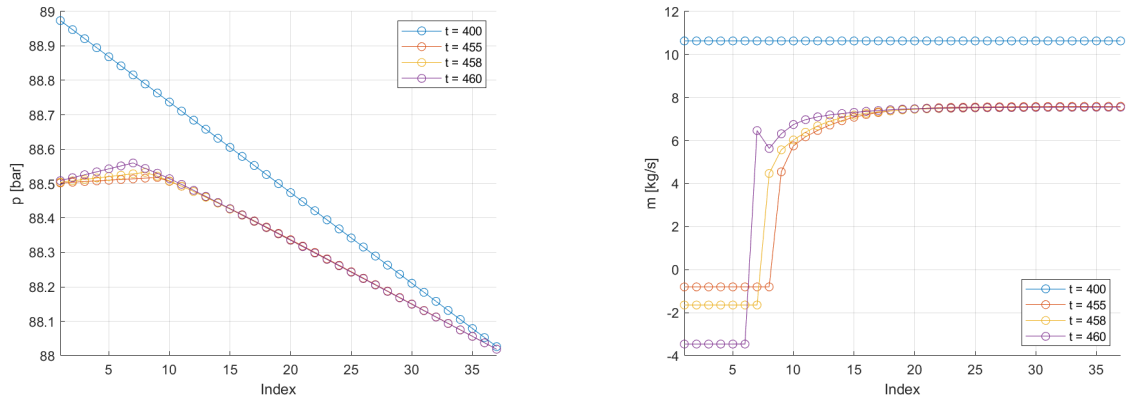
For two-phase the increased density may have given smaller liquid volume as the liquid was somewhat denser. Further, it would not affect the gas phase, as there is no liquid volume here, and the density is only included as a dummy variable for constant number of variables in the solver.

7.2 Small cold side mass flow

The step change in p_p resulting in a pressure drop of 0.5 over the OTSG failed for unknown reason. Later, it was tested with a ramp change of the same magnitude, which also failed. This suggests that the system reached an infeasible operating point. Even though such a step change may not be realistic, it is interesting to see what caused the solver to fail.

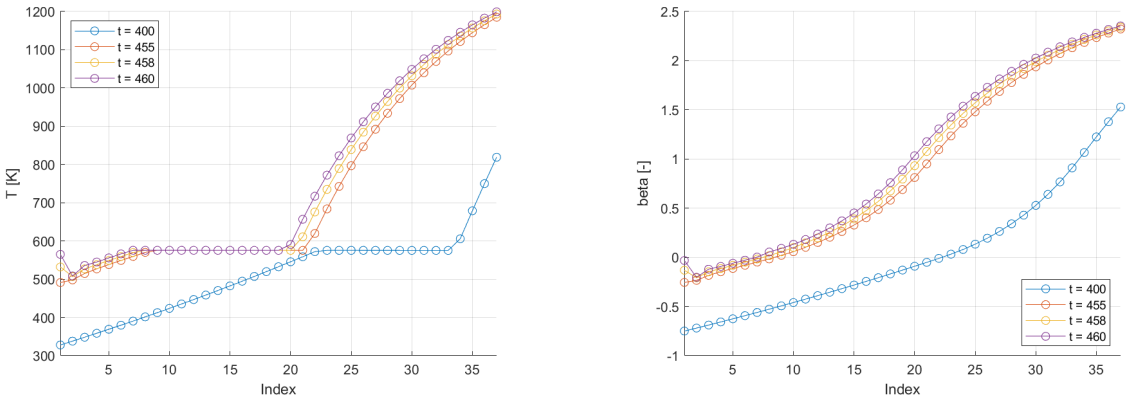
Figure 34 shows a reversed pressure gradient and mass flow in the liquid segments right before the solver failure at time 461 s. The reversed pressure gradient may be physical as the pressure is tied to the energy balance through the Antoine equation for the two-phase segments making the change in pressure slower. Nevertheless, the solver seems to be able to solve for multiple steps of negative flow, and is therefore probably not the cause of failure alone.

Figure 35 shows the cold side temperature and gas fraction. From the figure it can be observed that the temperature in the first segment increased fast approaching the bubble point prior to the solver breakdown. This may indicate that the cause of the failure is a switch of equations to two-phase in segment 1, while the consecutive segments did not switch.



(a) Pressure (b) Mass flow

Figure 34: Development in cold side mass flow and pressure as the system approach solver failure at t = 461 s.



(a) Temperature (b) Gas fraction

Figure 35: Development in cold side temperature and gas fraction as the system approach solver failure at t = 461 s.

7.3 Steady-state variation with n

The variation in steady-state temperatures with n was significant. An option to reduce this effect is to change the expression for the temperature driving force (ΔT) in Eq. 1. Instead by introducing the arithmetic mean (Eq. 21), the effect of n on steady-state temperatures was significantly reduced, as shown in Table 8.

$$\Delta T_{AM} = \frac{T_{g,i+1} - T_i}{2} + \frac{T_{g,i} - T_{i-1}}{2} \tag{21}$$

Figure 36 shows the comparison of the cold side outlet temperatures with the suggested expressions in the heat transfer equation. Extrapolation with an exponential model ($(1 - ae^{-bx}) + c$) showed that the outlet temperature for ΔT did not converge to the arithmetic mean outlet temperature, as it converged to 812.2 K. This suggests that there is something wrong with the implementation of the heat transfer and will require further investigation.

It is important to note that changing to arithmetic mean in the heat transfer equation would make the assumption of perfect mixing in each segment invalid as the temperature in each holdup is no longer constant.

Table 8: Steady-state variation with n using arithmetic mean.

n	30	37	45	52	59
T [K]	818.9871	818.9916	818.9885	818.9938	818.9816
T_g [K]	409.4690	409.4653	409.4679	409.4636	409.4735

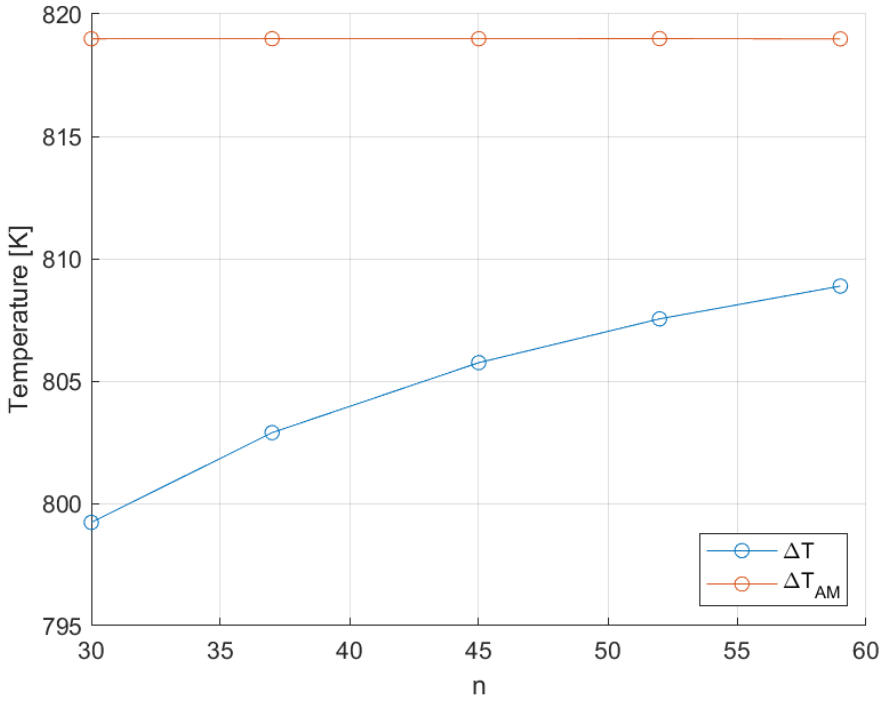


Figure 36: Comparison of steady-state variation in cold side outlet temperature T, with different expressions in heat transfer driving force, ΔT.

8 Conclusion

In this project a discretized model of an OTSG featuring dynamic phase boundary movement was developed based on simplified mass and energy balances. The model was implemented in Matlab/CasADi and solved using the IDAS solver.

The model was scaled and tested with steady-state and dynamic simulations. The simulations showed that the model dynamically switched set of equations in the segments which moved the phase boundaries. Step changes in relevant disturbances gave insight in model performance. The simulations also lead to detection of issues with the model such as initial condition sensitivity, heat transfer modeling, and noise in mass and pressure.

Future work should further investigate issues with heat transfer. The OTSG model should be set with the correct geometry and tested with additional process data. It should also focus on implementing the other units in the steam cycle and adding control elements to the steam cycle.

References

[1] Norwegian petroleum. Emissions to air. https://www.norskpetroleum.no/en/environment-and-technology/emissions-to-air/, 2024. Accessed: 07.10.2024.

[2] Marit J. Mazzetti, Brede A.L. Hagen, Geir Skaugen, Karl Lindqvist, Steinar Lundberg, and Oddrun A. Kristensen. Achieving 50 *Energy*, 230:120634, 2021. ISSN 0360-5442. doi: https://doi.org/10.1016/j.energy.2021.120634. URL https://www.sciencedirect.com/science/article/pii/S0360544221008835.

[3] Rubén M. Montañés, Brede Hagen, Han Deng, Geir Skaugen, Nicolas Morin, Marius Andersen, and Marit J. Mazzetti. Design optimization of compact gas turbine and steam combined cycles for combined heat and power production in a fpso system–a case study. *Energy*, 282:128401, 2023. ISSN 0360-5442. doi: https://doi.org/10.1016/j.energy.2023.128401. URL https://www.sciencedirect.com/science/article/pii/S0360544223017954.

[4] Cristina Zotică, Rubén M. Montañés, Adriana Reyes-Lúa, and Sigurd Skogestad. Control of steam bottoming cycles using nonlinear input and output transformations for feedforward disturbance rejection. *IFAC-PapersOnLine*, 55(7):969–974, 2022. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2022.07.570. URL https://www.sciencedirect.com/science/article/pii/S2405896322009764. 13th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems DYCOPS 2022.

[5] CasADi, 2024. URL https://web.casadi.org/.

[6] David J Gardner, Daniel R Reynolds, Carol S Woodward, and Cody J Balos. Enabling new flexibility in the SUNDIALS suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software (TOMS)*, 48(3):1–24, 2022. doi: 10.1145/3539801.

[7] Radu Serban, Cosmin Petra, Alan C. Hindmarsh, Cody J. Balos, David J. Gardner, Daniel R. Reynolds, and Carol S. Woodward. User documentation for idas. url-https://sundials.readthedocs.io/en/latest/idas, 2024. URL https://sundials.readthedocs.io/en/latest/idas. v6.1.1.

[8] Cristina Zotică, Lars O. Nord, Jenö Kovács, and Sigurd Skogestad. Optimal operation and control of heat to power cycles: A new perspective from a systematic plantwide control approach. *Computers & Chemical Engineering*, 141:106995, 2020. ISSN 0098-1354. doi: https://doi.org/10.1016/j.compchemeng.2020.106995. URL https://www.sciencedirect.com/science/article/pii/S0098135419311342.

[9] Sigurd Skogestad. Chemical and energy process engineering, 2009.

[10] Francis Weston Sears, Mark Waldo Zemansky, Hugh D. Young, and Roger A. Freedman. *University Physics*. Addison-Wesley, 10th edition, 2000.

[11] Chul Hee Cho, Jacob Urquidi, Surjit Singh, Seung C. Park, and G. Wilse Robinson. Pressure effect on the density of water. *The Journal of Physical Chemistry A*, 106(33):7557–7561, 2002. doi: 10.1021/jp0136260. URL https://doi.org/10.1021/jp0136260.

A Nomenclature

Table 9: Abbreviations

Abbreviation	Description
BDF	Backward Differentiation Formula
DAE	Differential-algebraic equation
FPSO	Floating production, storage and offloading vessel
GT	Gas turbine
OTSG	Once-through steam generator
ST	Steam turbine

Table 10: Latin symbols

Symbol	Unit	Description
C_p^g	[kJ/kg]	Heat capacity flue gas
C_p^s	[kJ/kg]	Heat capacity steam
C_p^w	[kJ/kg]	Heat capacity liquid
C_{vd}	[kg/bar]	Mass flow coefficient
H	[kJ]	Cold side segment enthalpy
h	[kJ/kg]	Cold side flow enthalpy
k_p	[1/bar]	Compressibility factor
M	[kg]	Segment holdup
M_w	[kg/mol]	Molar weight water
m	[kg/s]	Cold side mass flow
m_g	[kg/s]	Hot side mass flow
p	[bar]	Cold side pressure
p_p	[bar]	Cold side inlet pressure
p_s	[bar]	Cold side outlet pressure
Q	[kW]	Segment heat flow
R	[bar m3/mol/K]	Gas constant
T	[K]	Cold side temperature
T_p	[K]	Cold side inlet temperature
T_g	[K]	Hot side temperature
T_g^0	[K]	Hot side inlet temperature
UA	[kJ/kg/K]	Heat transfer coefficient
UA_e	[kJ/kg/K]	Heat transfer coefficient liquid
UA_s	[kJ/kg/K]	Heat transfer coefficient steam
V	[m3]	Segment volume
V_{tot}	[m3]	OTSG volume

Table 11: Greek symbols

Symbol	Unit	Description
β	[-]	vapor fraction
ρ	[kg/m3]	density

B Design parameters

Table 12: Constants used in the simulations

Parameter	Value	Unit
C_p^w	4.18	kJ/kg/K
C_p^s	3	kJ/kg/K
C_p^g	1.25	kJ/kg/K
T_{Ref}	0	K
T_{Ref}^{sat}	576.15	K
ρ_{Ref}	1000	kg/m^3
p_{Ref}	1	bar
R	$8.314 \cdot 10^{-5}$	m^3·bar/K/mol
M_w	18	g/mol
k_p	$4.58 \cdot 10^{-4}$	1/bar
A	5.11564	
B	1687.537	K
C	-42.98	K
ΔH_{vap}^0	1382	kJ/kg
C_{vd}	10.6309	kg/bar
UA_e	177	kW/K
UA_s	177	kW/K
V_{tot}	1	m^3

C Code

C.1 OTSG Model

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

    %% constants
    R        = constants(1);  % [m3*bar/K/mol]
    cpS      = constants(2);  % [kJ/kg/K]
    cpW      = constants(3);  % [kJ/kg/K]
    cpG      = constants(4);  % [kJ/kg/K]
    T_Ref    = constants(5);  % [K]
    TB_Ref   = constants(6);  % [K]
    T0g      = constants(7);  % [K]
    Tp       = constants(8);  % [K]
    mG       = constants(9);  % [kg/s]
    dHvap0   = constants(10); % [kJ/kg]
    UAs      = constants(11); % [kW/K]
    UAe      = constants(12); % [kW/K]
    Mw       = constants(13); % [kg/mol]
    Cvd      = constants(14); % [kg/bar]
    p_in     = constants(15); % [bar]
    p_out    = constants(16); % [bar]
    V_tot    = constants(17); % [m3]
    rho_Ref  = constants(18); % [kg/m3]
    p_Ref    = constants(19); % [bar]
    k_p      = constants(20); % [1/bar]
    A        = constants(21); % [-]
    B        = constants(22); % [K]
    C        = constants(23); % [K]

    % Segment dependent constants
    UAs_n = UAs/n;
    UAe_n = UAe/n;
    V     = V_tot/n;

    %% Solver initialization
    % Define variables
    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);
```

```
49
50      % Set states  x: differential, z: algebraic
51      x = [];
52      for k=1:n
53          x = [x; M(k); H(k)];
54      end
55
56      z = [m_in];
57      for k=1:n
58          z = [z;
                ↪ m(k);T(k);h(k);Tg(k);mg(k);Q(k);p(k);rho(k);beta(k);TB(k)];
59      end
60
61      % Initialize equatuions
62      Alg  = [];
63      diff = [];
64
65      % Set function output variables
66      x_keys = x;
67      z_keys = z;
68
69      %% Additional inlet equations
70      h_in  = cpW*(Tp-T_Ref); % Note: Assuming input is liquid state
71      init1 = m_in - Cvd*(p_in-p(1));
72      Alg   = [Alg;init1];
73
74      %% i = 1
75      dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TB(1));
76
77      % Switch logic
78      cond1 = beta(1) >= 1;
79      cond2 = beta(1) <= 0;
80
81      % 1: Steam, 2: Two-phase, 3: Liquid
82      Enth1 = cpW*(TB(1)-T_Ref) + dHvap + cpS*(T(1)-TB(1)) - h(1);
83      Enth2 = T(1) - TB(1);
84      Enth3 = cpW*(T(1)-T_Ref) - h(1);
85
86      Hex1 = UAs_n*(Tg(1)-T(1)) - Q(1);
87      Hex2 = (beta(1)*UAs_n + (1-beta(1))*UAe_n)*(Tg(1)-T(1)) - Q(1);
88      Hex3 = UAe_n*(Tg(1)-T(1)) - Q(1);
89
90      VL    = (1-beta(1))*M(1)/rho(1);
91      Pres1 = M(1)*R*T(1)/(Mw*V) - p(1);
92      Pres2 = p(1)*(V-VL) - beta(1)*M(1)*R*T(1)/(Mw);
93      Pres3 = rho(1) - M(1)/V;
```

```
    % 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  = if_else(cond1, Hex1, if_else(cond2, Hex3, Hex2));
    alg3  = m(1) - Cvd*(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-(B/(TB(1)+C))) - p(1);

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

    %% i = 2 to n-1
    for k=2:n-1
        dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TB(k));
        % Switch logic
        cond1 = beta(k) >= 1;
        cond2 = beta(k) <= 0;

        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);

        Hex1 = UAs_n*(Tg(k)-T(k)) - Q(k);
        Hex2 = (beta(k)*UAs_n + (1-beta(k))*UAe_n)*(Tg(k)-T(k)) - Q(k);
        Hex3 = UAe_n*(Tg(k)-T(k)) - Q(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;

        % 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  = if_else(cond1, Hex1, if_else(cond2, Hex3, Hex2));
        alg3  = m(k) - Cvd*(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-(B/(TB(k)+C))) - p(k);

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

```
151
152     %% i = n
153     dHvap = dHvap0 + (cpW-cpS)*(TB_Ref-TB(n));
154     % Switch logic
155     cond1 = beta(n) >= 1;
156     cond2 = beta(n) <= 0;
157
158     Enth1 = cpW*(TB(n)-T_Ref) + dHvap + cpS*(T(n)-TB(n)) - h(n);
159     Enth2 = T(n) - TB(n);
160     Enth3 = cpW*(T(n)-T_Ref) - h(n);
161
162     Hex1 = UAs_n*(Tg(n)-T(n)) - Q(n);
163     Hex2 = (beta(n)*UAs_n + (1-beta(n))*UAe_n)*(Tg(n)-T(n)) - Q(n);
164     Hex3 = UAe_n*(Tg(n)-T(n)) - Q(n);
165
166     VL    = (1-beta(n))*M(n)/rho(n);
167     Pres1 = M(n)*R*T(n)/(Mw*V) - p(n);
168     Pres2 = p(n)*(V-VL) - beta(n)*M(n)*R*T(n)/(Mw);
169     Pres3 = rho(n) - M(n)/V;
170
171     % Equations
172     dMdt = m(n-1)-m(n);
173     dHdt = m(n-1)*h(n-1) - m(n)*h(n) + Q(n);
174
175     alg1  = mg(n)*cpG*(T0g-Tg(n)) - Q(n);
176     alg2  = if_else(cond1, Hex1, if_else(cond2, Hex3, Hex2));
177     alg3  = m(n) - Cvd*(p(n)-p_out);
178     alg4  = if_else(cond1, Enth1, if_else(cond2, Enth3, Enth2));
179     alg5  = M(n)*h(n) - H(n);
180     alg6  = mg(n) - mG;
181     alg7  = if_else(cond1, Pres1, if_else(cond2, Pres3, Pres2));
182     alg8  = p(n) - 1/(k_p*rho_Ref)*(rho(n)-rho_Ref) - p_Ref;
183     alg9  = cpW*(TB(n)-T_Ref) + beta(n)*dHvap - h(n);
184     alg10 = 10^(A-(B/(TB(n)+C))) - p(n);
185
186     Alg  = [Alg;alg1;alg2;alg3;alg4;alg5;alg6;alg7;alg8;alg9;alg10];
187     diff = [diff;dMdt;dHdt];
188
189     %% Solver
190     dae     = struct;
191     dae.x   = x;      % Differential states
192     dae.z   = z;      % Algebraic states
193     dae.ode = diff;   % Differential equations
194     dae.alg = Alg;    % Algebraic equations
195
196     opts = struct('tf', 1, 'abstol', 1e-9, 'reltol', 1e-9, ...
197         'max_num_steps', 10000);
198
199     F = integrator('F', 'idas', dae, opts);
200
201     sol = F('x0', x_0, 'z0', z_0);
202 end
```

C.2 Main

```
clear
clc
addpath('C:\casadi-3.6.6')

% Set filepath for storage, use false to discard results
filepath = 'test.csv';

% Set number of segments
n = 37;

%% Constants
R        = 8.314462618*1e-5; % [m3*bar/K/mol] gas constant
cpS      = 3;                % [kJ/kg/K] heat capasity steam
cpW      = 4.18;             % [kJ/kg/K] heat capasity water
cpG      = 1.25;             % [kJ/kg/K] heat capasity gas
T_Ref    = 0;                % [K] Reference temperature
TB_Ref   = 303+273.15;       % [K] Reference boiling point temperature
T0g      = 1000+273.15;      % [K] Hot side inlet temperature
Tp       = 45+273.15;        % [K] Cold side inlet temperature
mG       = 31.4018;          % [kg/s] Hot side mass flow
dHvap0   = 1382;             % [kJ/kg] Reference energy of vaporization
UAs      = 177;              % [kW/K] Heat transfer coeff gas
UAe      = 177;              % [kW/K] Heat transfer coeff liquid
Mw       = 18*1e-3;          % [kg/mol] molar weight water
Cvd      = 10.6309*(n+1);    % [kg/bar]
p_in     = 89;               % [bar] Cold side inlet pressure
p_out    = 88;               % [bar] Cold side outlet pressure
V_tot    = 1;                % [m3] OTSG volume
rho_Ref  = 1000;             % [kg/m3] Reference density water
p_Ref    = 1;                % [bar] Reference pressure water
k_p      = 4.58*10^(-4);     % [1/bar] Compresibility factor water
A        = 5.11564;          % [-] Antoine Coeff
B        = 1687.537;         % [K] Antoine Coeff
C        = -42.98;           % [K] Antoine Coeff

constants = [R, cpS, cpW, cpG, T_Ref, TB_Ref, T0g, Tp, mG, dHvap0, ...
UAs, UAe, Mw, Cvd, p_in, p_out, V_tot, rho_Ref, p_Ref, k_p, A, B, C];


%% Set guesses
mP          = 10.6309;          % [kg/s]
beta_guess  = 0;                % [-]
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]
Q_guess     = 18000/n;          % [kW]
rho_guess   = 1000;             % [kg/m3]

p_guess   = linspace(p_in, p_out, n+2);
p_guess   = p_guess(2:n+1);     % [bar]
TB_guess  = TB_Ref;             % [K]
```

```
56
57
58  % Set order of guesses to match order in solver
59  x_0 = repmat([M_guess; H_guess], n, 1);
60  z_0 = [m_guess];
61  for k=1:n
62      z_0 = [z_0; m_guess; T_guess; h_guess; Tg_guess; mg_guess; ...
63      Q_guess; p_guess(k); rho_guess; beta_guess;TB_guess];
64  end
65
66  %% Run
67  T = 800; % [s] Total simulation time
68  N = T;   % [-] Number of steps
69
70  % Inital conditions
71  x = x_0;
72  z = z_0;
73
74  % Make result arrays
75  x_save = zeros(N+1, length(x_0));
76  z_save = zeros(N+1, length(z_0));
77  x_save(1, :) = x_0;
78  z_save(1, :) = z_0;
79
80  % Steps
81  % T_init_step = [(273.15+45)*ones(1,400), (273.15+35)*ones(1,400)];
82
83  time_save = [0];
84  running_time = 0;
85  for i = 1:N
86      % Apply steps
87      % constants(15) = p_init_step(i);
88
89      % Evaluate system
90      [xf_keys, zf_keys, result] = OTSGModel(n, x, z, constants);
91      x = result.xf;
92      z = result.zf;
93
94      % Save state
95      x_save(i+1, :) = full(x);
96      z_save(i+1, :) = full(z);
97      running_time   = running_time + T/N;
98      disp(running_time);
99      time_save = [time_save, running_time];
100 end
```

```
%% Make result dictionary
% 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);

dict = containers.Map();
for i=1:length(zf_keys)
    dict(zf_keys{i}) = z_save(:, i);
end

for i=1:length(xf_keys)
    dict(xf_keys{i}) = x_save(:, i);
end

keySet = keys(dict);

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

%% Store as CSV

if filepath

    valueSet = values(dict);
    % Initialize an empty cell array for storing data
    numKeys   = length(keySet);           % Total number of keys
    maxRows   = size(valueSet{1}, 1);     % Assuming same number of rows
    tableData = cell(maxRows + 1, numKeys);% +1 for the header (keys)

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

    % Store the values under each corresponding header
    for i = 1:numKeys
        dataArray = dict(keySet{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
```