Development of a Solution for Start-up Optimization of a Thermal Power Plant

Marcus Thelander Andrén

Christoffer Wedding

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
ISRN LUTFD2/TFRT--5972--SE
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2015

Abstract

This thesis covers optimizing the first phase of the start-up of a thermal power plant using Nonlinear Model Predictive Control (NMPC) and state estimation using an Unscented Kalman Filter (UKF). The start-up has been optimized in regards to time and fuel usage. The thesis is done as a joint project between Vattenfall and Modelon. Both NMPC and UKF are nonlinear methods and require a model of the power plant. The model used in this thesis has been developed in the language Modelica in a previous master thesis and has been extended and improved upon during this thesis. The optimization and simulation of the model required by the NMPC and UKF was done within the framework of JModelica.org. Another, more detailed, model of the power plant, developed by Vattenfall, was originally planned to be used as the process to be controlled.

State estimation using the UKF has been successful, with a maximum mean absolute error of 0.7 % when estimating the states of the detailed model in a reference start-up. When using the NMPC to control the optimization model itself, the simulated start-up time is 70 minutes faster compared to a reference start-up using the detailed model. This is more than half the time of the first phase of the start-up. The total firing power, which relates to the fuel amount, is also considerably less, with the optimized value being about 40 % of that in the reference soft start with the detailed model.

Due to difficulties in initializing the detailed model, it was not possible to run it online together with the NMPC and UKF. Running the NMPC and UKF together on the optimization model worked, but the NMPC failed to find an optimal trajectory 8 out of 10 iterations. The conclusion is that the start-up has potential for optimization, but requires more robust models to work with.

Keywords: NMPC, UKF, Thermal Power Plants, JModelica.org, Modelica

Acknowledgments

We would like to thank our supervisor at Modelon, Stéphane Velut, for his aid during the whole project and valuable insight in modelling and numerical optimization and simulation. Jonas Funkquist, our supervisor at Vattenfall, has our gratitude for his aid during the project, valuable discussions on thermal power plants and making us feel at home in Stockholm.

We would also like our supervisor at LTH, Pontus Giselsson, for his input on using MPC controllers and helping us with our report and Dr. Rolf Johansson for supporting us with the formalities of the master thesis work and his knowledge on UKF theory.

Other people who have been important to our work are Håkan Runvik who has helped us with the optimization model and Moritz Hübel and Sebastian Meinke who have modified the detailed plant model to make it compatible with our work and given us valuable insight into the power plant process.

Contents

List of Figures

List of Tables

Abbreviations and Terminology

Abbreviations

- EKF - Extended Kalman Filter
- FMU - Functional Mock-Up Unit
- HPP - High pressure preheater
- HP-T - High pressure turbine
- IP-T - Intermediate pressure turbine
- LPP - Low pressure preheater
- LS - Live steam
- L-T - Low pressure turbine
- MPC - Model Predictive Control
- NLP - Nonlinear programming
- NMPC - Nonlinear Model Predictive Control
- **O** - The optimal control problem that has to be solved each iteration for the MPC algorithm.
- RH - Reheater
- RS - Reheat steam
- SH - Superheater
- UKF - Unscented Kalman Filter
- UT - Unscented Transform

Terminology

- *Bypass valve* - A valve which bypasses the turbine inlet in the power plant.
- *Cold start* - A start-up scenario for the power plant, defined in Table 2.1.
- *Disturbance state* - A state in a model representing a disturbance.
- *Feasible solution* - A solution that satisfies the constraints in (2.2) and whose generated state sequence also satisfies (2.2).
- *Hot start* - A start-up scenario for the power plant, defined in Table 2.1.
- *Nominal system* - A system without any disturbances or model inaccuracies and whose states are fully known.
- *Observable* - A system where all states can be uniquely determined from a finite sequence of inputs and outputs.
- *Optimal control* - A method where a sequence of optimal control values are computed using a model of the system, a predefined cost function and some constraints on the states and control input.
- *Sigma points* - A set of points used in the unscented transform.
- *Soft start* - A start-up scenario for the power plant, defined in Table 2.1.
- *Target system* - The actual process being controlled or estimated.

1

Introduction

The requirement on more flexible thermal power plants is increasing as intermittent power production like solar power and wind power is increasing. In the future it is expected that the thermal power plants will have to shut down when the electricity demand is low and the production of sustainable energy is high enough to satisfy the demand. However, it is time consuming and expensive to start up a thermal power plant, and they are usually not designed for frequent start-ups. In particular they are not optimized for that type of operation. A problem is the wear on certain components due to the thermal stress they are exposed to during the start-up phase. A lot of time and resources could therefore potentially be saved if these start-ups can be executed as fast and smooth as possible. The environmental impact will also be reduced if less fuel is used for start-up of the plant and if the plant could be run for smaller periods of time. One way to achieve the goal of optimizing the start-up is to utilize model-based online optimization in the form of model predictive control (MPC). Vattenfall and Modelon have been collaborating on this for one of Vattenfall's thermal power plant as a part of the research project MODRIO [Modelica Association, 2015].

The two previous master theses [Andersson, 2013] and [Runvik, 2014] have been done at Modelon and Vattenfall with the aim of developing a simplified model suitable for online optimization and to set up an environment to run some typical optimization problems. In the master thesis [Axelsson, 2015] done at Modelon, a general MPC tool has been developed which is also compatible with the simplified model. In parallel, a detailed plant model has been developed in collaboration between Vattenfall and Rostock University. The detailed model is able to simulate the plant start-up in high detail and is intended to be utilized as the target plant in simulation studies. Building upon the results of the previous master theses and interfacing an MPC to the detailed model is a way to create a demonstrator from which the potential of optimizing the start-up can be analyzed.

1.1 Aim of Thesis

The objective of this master thesis is to research the possibility of using a nonlinear model predictive controller (NMPC) to optimize the start-up of a thermal power plant by minimizing the time and fuel consumption. This is to be done while regarding maximum allowable stress levels in critical components of the plant.

The objective will be achieved by deriving and implementing a state-estimator in the form of an unscented Kalman filter (UKF) including disturbance dynamics for the plant, which will be combined with an NMPC based on the modelling and optimization work done in [Runvik, 2014] and [Axelsson, 2015]. The controller and estimator is to be interfaced to a detailed simulation model, developed at Vattenfall, to form a demonstrator, which can be used to demonstrate and analyze start-up optimization scenarios for a thermal power plant.

1.2 Specifications

The following specifications were set from Vattenfall and Modelon:

- The UKF shall be compatible with JModelica.org.
- Optimization is to be done using JModelica.org.
- The UKF and NMPC shall use the model developed in [Runvik, 2014] as a basis for their internal models.
- The detailed model that has been developed by Vattenfall shall be used as the process model in the final demonstrator.
- Stress levels of the critical components in the plant should be small enough to ensure 2000 start-ups.

1.3 Research Questions

In order to achieve the aim of the thesis while conforming to the specifications, the following questions were investigated:

- How should the detailed model be interfaced to the JModelica.org framework in the implementation of a demonstrator?
- How can the start-up process be optimized, using NMPC, in regards to total time and fuel spent?
- How does the simplified optimization model need to be changed to make it as similar to the detailed plant model as possible?

- Are the current constraints appropriate for the actual plant or are there other sections that are more critical?
- How should the optimization model be modified to make it usable in NMPC optimization and UKF estimation?
- How should a general UKF class be implemented, and how should it interface with JModelica.org?
- How should the measurement signals from the plant be chosen to ensure observability?

1.4 Tasks

From the goal of the thesis and our research questions, the following tasks were defined:

- Add new critical components to the optimization model, and remove those deemed not critical.
- Tune the parameters of the optimization model to make it more similar to the detailed model during start-up.
- Implement and test a UKF within the JModelica.org framework.
- Make the optimization model usable with the NMPC and UKF.
- Make a simple observability analysis of the plant model.
- Make an analysis of the disturbance dynamics for the plant.
- Improve the formulation of the start-up optimization by testing different cost functions and in particular minimize fuel and time consumption.
- Find a solution to interface the detailed model to the JModelica.org framework.
- Integrate all components into a solution that can be used to demonstrate start-up optimization scenarios for a thermal power plant.

1.5 Structure of Thesis

In Chapter 2, background theory on the power plant start-up process is explained. Additionally, basic theory of NMPC, the UKF, disturbance modelling, nonlinear observability and numerical optimization is presented. The chapter is concluded by a brief description of the Modelica and Optimica languages used for modelling in this thesis.

In Chapter 3, the delimitations on the work are presented. The methods used in implementing the NMPC and UKF, modelling disturbances, observability analysis and validation of the optimization model are explained.

In Chapter 4, the setups of the experiments performed with the UKF and NMPC on the power plant models are presented. In chapter 5, the results from the experiments are presented. The full estimation results are found in appendix A.

The thesis is concluded with a discussion of the results, methodology and thoughts on improvements in Chapter 6. The conclusions and recommendations for future work are summarized in Chapter 7.

2

Background

2.1 The Jänschwalde Thermal Power Plant

The power plant which is simulated in this thesis is the Vattenfall owned power plant Jänschwalde in Germany. Built between 1976 and 1988, the plant is the third largest lignite fired power plant in Germany. It has a capacity of 3000 MW with an overall efficiency of about 36% [Runvik, 2014].

The Jänschwalde plant is divided into six blocks where each block contains two boilers. The two boilers of each block provide steam for a mutual turbine. An overview of a block of the Jänschwalde power plant is presented in Figure 2.1. The fundamental principle of a lignite-fired power plant is to convert the chemical energy stored in the lignite to electrical energy. The lignite is fed to the furnace in each boiler where it is mixed with air and combusted, producing hot flue gas. The energy emitted from the combustion and energy stored in the hot flue gas is transferred to the water in the evaporators of each boiler. The water evaporates, and pressure is built up. When the steam from the boilers is expanded in the turbine, mechanical force is exerted on the turbine blades, making them rotate. The turbines will power a generator, which outputs electrical energy.

Since the boilers in a block are started one at a time, we will, in this thesis, consider a simplified block with a single water steam cycle line and boiler connected to the turbine. All components common to both boilers can then be modelled as a symmetric part with half the size. An overview of the simplified block of the Jänschwalde power plant is presented in Figure 2.2. SH and RH denotes super-heaters and reheaters respectively while LPP and HPP denotes low pressure and high pressure preheaters. These are heat exchangers, consisting of bundles of tubes perpendicular to the flue gas flow in the flue gas channel. Between many of the heat exchangers there are spray attemperators that spray water into the hot steam to regulate the temperature. Eco denotes economizer, which also is a section of heat exchange. HP-T, IP-T and L-T represent the different turbine stages of the plant, and denote high, intermediate and low pressure turbine respectively. The steam entering the HP-T is

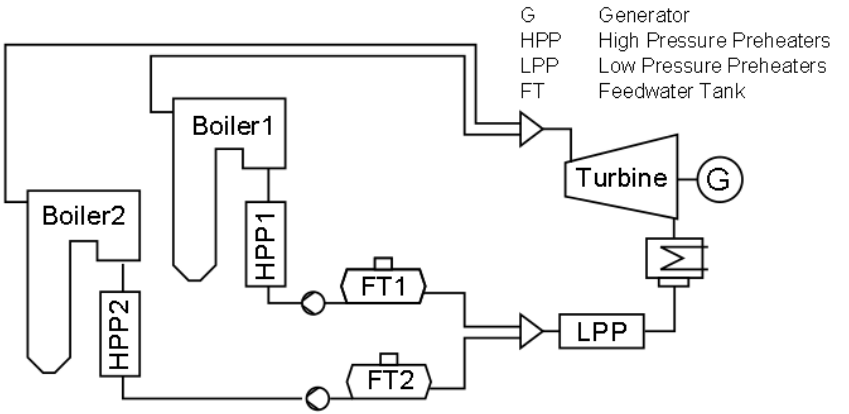

Figure 2.1 Overview of one of the blocks in the Jänschwalde power plant. The component seen between the turbine and the LPP is the condenser.

commonly known as live steam (LS). The steam entering the IP-T will in this thesis be referred to as the re-heat steam (RS). Between the outlet headers, which collect the steam from the heat exchangers, and the HP and IP turbines there are *bypass valves* that can be opened to let the steam continue the steam cycle without entering the turbine.

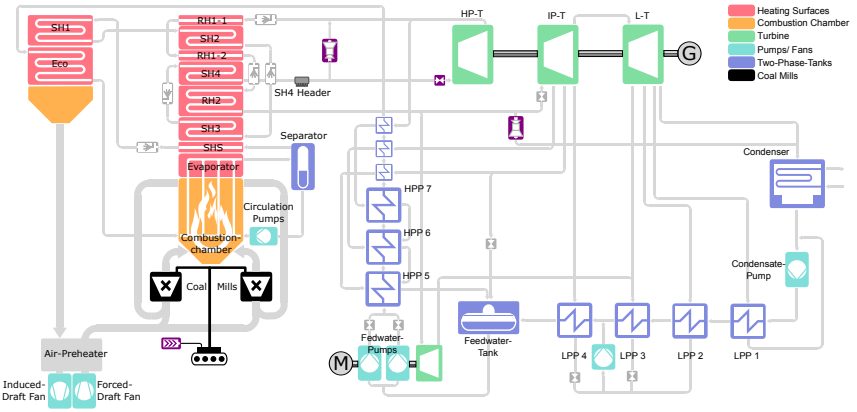

Figure 2.2 Overview of one boiler connected to the turbines in the Jänschwalde plant.

The steam cycle in each block is designed to maximize the efficiency of the process. This involves super-heating the steam before it enters the high pressure turbine, and

then re-heating the steam again before expanding it in the intermediate- and low pressure turbines. A detailed description concerning the thermodynamics of this procedure can be found in [Runvik, 2014].

2.1.1 The Start-up

The start-up of a block is divided into three different types depending on the stand-still time. The types are *cold*, *soft* and *hot start*. The three start-up types are defined in Table 2.1.

Start-up	Stand-still Time (T)	Approximate start-up time	Occurrences at Jänschwalde 2012
Cold	$T \geq 48h$	20 h	15
Soft	$12h < T < 48h$	9 h	6
Hot	$T \leq 12h$	3 h	4

Table 2.1 Definitions and occurrences of the different start-up types.

A typical start-up of a block is done step-wise and one boiler at a time. In Figure 2.3 a reference soft start of one boiler from the handbook of the plant [Vattenfall AB, 2012] is presented. The start-up is divided into three phases:

(i) Initially, oil burners are used to quickly start up the combustion process in the furnace. The oil burners are started one at a time. When the temperature in the furnace is sufficiently high, they are gradually switched out for the coal mills, which feed the furnace with lignite. The temperature and pressure of the LS and RS in the boiler are increased until they reach desired values. The turbine valves are closed during this phase while the bypass valves are open, meaning no steam enters the turbines. This phase corresponds to the time between 0 and 180 minutes in Figure 2.3.

(ii) The turbine valve of the intermediate pressure turbine is opened while the corresponding bypass valve starts to close, allowing steam to expand in the turbine. Later, the steam is allowed to enter the low and high pressure turbines as well. Turbine speed is increased step-wise until 3000 rpm is reached. At that time the bypass valves are fully closed. During this phase temperature and pressure of the steam are further increased. This phase corresponds to the time between 180 and 295 minutes in Figure 2.3.

(iii) The generator is connected to the electrical grid, and live steam pressure and temperature continues to increase until the desired power set-point of the plant is reached. A full load of 500 MW, when both boilers are online, corresponds to set-points of 535°C and 160 bar for the live steam (LS). The pressure set-point is 120 bar with one boiler online as seen in Figure 2.3.

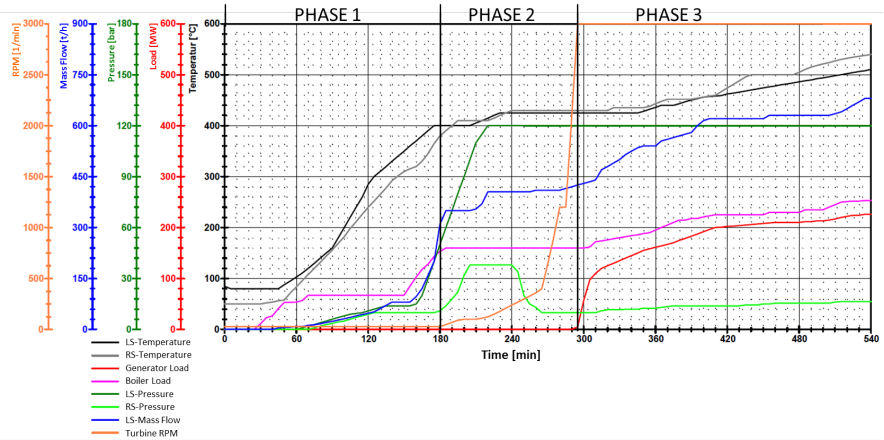

Figure 2.3 Signals for pressure, temperature and mass flow of LS, temperature and pressure of RS, generator load, boiler load and turbine speed during a reference soft start [Vattenfall AB, 2012]. The boiler load is the power of the fuel, divided by the efficiency of the power plant at full load. This means that at full load the boiler load and generator load should be equal.

The start-up time of one boiler for a soft start is approximately 9 hours. A more extreme case is a cold start for an entire block, which takes approximately 20 hours [Runvik, 2014].

2.1.2 Constraints on Start-up

The start-up is deliberately done in a slow and controlled manner in order to avoid too large temperature gradients in the thick-walled metal components of the plant. Steep temperature gradients give rise to large thermal stresses in the components and can reduce the life-time of the component significantly.

Components that are potentially critical to break due to stress in the start-up are the headers of the heat exchangers. The headers are cylindrical pipes connected with smaller pipes that collect the steam from all the tubes in the heat exchanger into one steam flow. The highest stress occurs on the edges of the branching smaller pipes, see Figure 2.4. It was previously assumed that the separator wall was also critical, but it was later deemed uncritical in [Runvik, 2014].

The total stress in a component is given as the sum of the thermal stress from temperature differences in the walls and the mechanical stress from the high pressure of the steam. The thermal stress during start-up will typically give rise to negative stress values while the mechanical stress will have positive values. Positive stress is defined as stress directed radially inwards in the cylindrical header, while negative

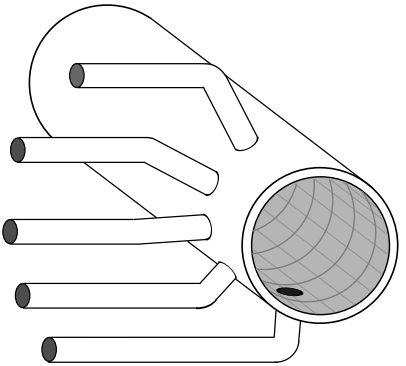

Figure 2.4 Design of header used in the plant.

stress is directed outwards. Formulas used for stress calculations of the headers can be found in [Runvik, 2014] and [Andersson, 2013].

Too large stress levels in components will cause life-time consumption. Vattenfall wish to optimize the start-up while still guaranteeing that the components will hold for 2000 cold start-ups.

2.1.3 The Detailed Plant Model

A detailed simulation model of the Jänschwalde power plant has been developed in Dymola using the programming language Modelica in a collaboration between Vattenfall and Rostock University. The top-layer of the model is presented in Figure 2.5. It models the control system and most of the components of the power plant. Because of the high complexity and the amount of time it takes to simulate, it can not be used in real-time for online optimization or as an observer model in a state estimator. Instead, the detailed model was planned to be used as the real process to be controlled and optimized. An overview of the model is presented in [Hübel et al., 2014b].

The available inputs to the plant model are set-points for pressures and temperatures of LS and RS and also a set-point for firing power. This input is a normalized firing power for the furnace in the range [0,1]. The control system then controls the flow of fuel in the form of oil or coal so that the correct firing power is met.

2.1.4 The Optimization Model

Since the detailed plant model is not suitable for optimization or state estimation in real-time, a simplified model of the plant is used instead. This model has been developed in Dymola in [Andersson, 2013] and [Runvik, 2014] using the language

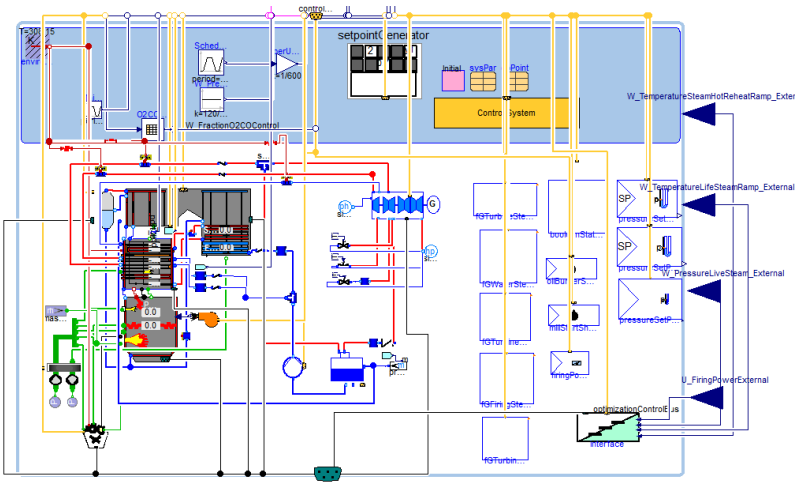

Figure 2.5 Top layer of the detailed model of the Jänschwalde power plant.

Modelica. In the latter thesis there is a description of the model that is used as a base for this project.

There are two versions of the model, one for the first and one for the second phase of the start-up described in section 2.1.1. In the model for phase 1, that is used as a base for the model in this thesis, the turbines and turbine valves are not present since they are not used in this phase. This model is presented in Figure 2.6.

The rate of change of the LS pressure set-point and the flue gas flow are used as inputs. An internal PI controller controls the opening degree of the HP-T bypass valve. The boiler itself is modelled from flue gas input through all the heat exchanger sections. It does not model the furnace and instead uses a controllable gas source. In addition the whole water steam cycle after RH2 including the intermediate and low-pressure turbines and the preheaters are exchanged for boundary conditions. The LS of the model is found in the SH4 header component, while the RS corresponds to the steam at the boundary condition after RH2, i.e the RS is considered to have constant pressure.

The dynamics of the model are on the steam side and in the walls of the heat exchangers, while the gas side is modelled with static relations. The dynamic states in each heat exchanger is the density and internal energy of the steam. The walls of each heat exchanger are modelled to have a uniform temperature, which is also a dynamic state. The SH4-header and the separator also have a separate wall-component with three nodal temperatures, which is used to calculate the thermal stress. The center-most nodal temperatures in these wall-components are also dynamic states in the model.

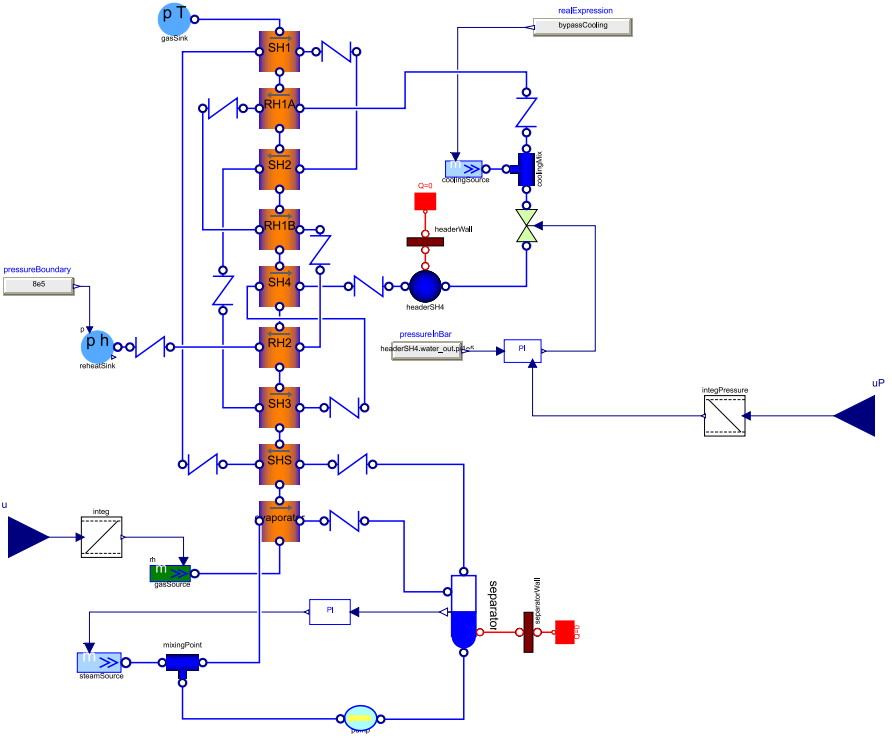

Figure 2.6 Top layer of the optimization model developed in [Runvik, 2014].

All the thermodynamic relations of the water in the model are approximated with polynomial functions, using the Modelon developed Modelica package *water_poly*. Having polynomial approximations rather than relations with discontinuities makes the model more optimization friendly. However, the approximations have a limited range of validity.

2.2 Nonlinear Model Predictive Control

The basic idea behind Model Predictive Control (MPC) is that a control law is derived by solving an *optimal control* problem using measurements from the *target system*. Optimal control is a method where a sequence of optimal control values are generated using a model of the system, a predefined cost function and some constraints on the states and control input. The problem is solved offline, and the generated control signal is applied to the system. Using MPC the first control action from the optimal control problem is applied to the system for one sampling instance and then a new optimal control problem is solved for the new point in the state space,

$\mathbb{X}$. A model of the dynamic behaviour of the system is used to model its behaviour over a specified time horizon, T_p, called the prediction horizon. The control action is then calculated for a time horizon, T_c, called the control horizon, where $T_c \leq T_p$ [Allgöwer et al., 2004]. One of the main advantages of MPC is that it can take constraints on the states or control signal into account explicitly [Grüne and Pannek, 2011]. In linear MPC the system model is linear while in nonlinear MPC (NMPC) the system model is nonlinear. The resulting closed loop system will be nonlinear in both cases however due to the presence of constraints [Allgöwer et al., 2004].

2.2.1 Basic Definition

The NMPC solves an optimal control problem for each iteration. With a sampling interval h and the discrete prediction horizon $N = T_p/h$, the optimization problem $\mathbf{O}$ that has to be solved is defined by

$$\min_{\{\mathbf{u}_i\}} V_N(\{\mathbf{x}_i\}, \{\mathbf{u}_i\}) \tag{2.1}$$

$$\text{subject to } \mathbf{u}_i \in \mathbb{U} \quad \mathbf{x}_i \in \mathbb{X} \quad \mathbf{x}_N \in \mathbb{X}_f \quad \forall i \in (0,\, N-1) \tag{2.2}$$

where $\{\mathbf{x}_i\}$ denotes the sequence of n number of states that is derived when using the sequence of m number of control input $\{\mathbf{u}_i\}$. V_N is the cost function. $\mathbb{U}$ and $\mathbb{X}$ are the constraint sets for $\{\mathbf{u}_i\}$ and $\{\mathbf{x}_i\}$ respectively and $\mathbb{X}_f$ is the terminal constraint set for $\{\mathbf{x}_i\}$.

The cost function, V_N, is defined by

$$V_N(\{\mathbf{x}_i\}, \{\mathbf{u}_i\}) = \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) \tag{2.3}$$

where the stage cost function, ℓ, is an arbitrary function If the control horizon is shorter than the prediction horizon the remaining values of the control sequence are chosen equal to the last value (i.e. a constant control signal) [Rawlings and Mayne, 2012].

Using the solution to $\mathbf{O}$, the control law can be defined as $\kappa_N(\mathbf{x}) := \{\mathbf{u}_0\}^o(\mathbf{x})$, where $\{\mathbf{u}_0\}^o(\mathbf{x})$ denotes the first element in the optimal control sequence starting at the point $\mathbf{x}$. This control sequence can generally not be calculated for all points, $\mathbf{x}$, and the optimization problem therefore has to be solved online when a new $\mathbf{x}$ is measured each sampling instance. The fact that the NMPC algorithm does not require the computation of $\kappa_N(\mathbf{x})\ \forall \mathbf{x} \in \mathbb{X}$ is one of the things which make NMPC so useful [Rawlings and Mayne, 2012].

It should be noted that with an infinite control horizon and a deterministic problem the control law, $\kappa_N(\{\mathbf{x}_i\})$, could be determined offline as $\kappa_N(\{\mathbf{x}_i\}) = \{\mathbf{u}_i\}$ where $\{\mathbf{u}_i\}$ is the solution to $\mathbf{O}$ at $\mathbf{x}_0$. When the control horizon is chosen as a finite value the control law will however differ from the open loop optimal control sequence. This is because new information is introduced at each step of the algorithm

as the control horizon is expanded [Allgöwer et al., 2004]. Real problems will also have disturbances, mismatches between model and target system and unmeasurable states. This means that an open loop solution in general will not work for a real system.

2.2.2 Stability and Convergence of Nominal System

The system with no disturbances, mismatches between model and target system or unmeasurable states is called the *nominal system*. Even if this is not true for a practical system the analysis of this system is still important since it makes it easier to analyse the system. Satisfactory behaviour of the nominal system is also necessary for our controller. If it cannot handle the simplified case it will never be able to cope with the actual system [Grüne and Pannek, 2011].

The convergence problem can be divided into the case with infinite control horizon and the one with a finite horizon. For the infinite horizon it is possible to calculate the optimal control sequence offline using the *dynamic programming principle* which means that the results of the NMPC algorithm can be verified. This is described by [Grüne and Pannek, 2011]. Since the closed and open loop solutions differ for finite control horizons the stability problem becomes much harder. In general the controlled system is not stable, but in [Rawlings and Mayne, 2012], [Grüne and Pannek, 2011] and [Ruan et al., 2014] a few ways to stabilize the system are described. These methods are based on modifying V_N to obtain a Lyapunov function in a neighbourhood of the reference point [Rawlings and Mayne, 2012].

It is also not necessary to find an optimal solution to $\mathbf{O}$ each iteration to ensure stability, which help with numerical calculations. As long as a *feasible solution* that decreases the value function compared to the last iteration can be found for each iteration the system will converge [Allgöwer et al., 2004].

2.2.3 Robustness

The robustness of a system is defined as how well it handles disturbances. Classic control theory states that you need a closed-loop system to handle disturbances. Since MPC samples the system each iteration it does use feedback which means that it has a certain built in robustness. However, the optimization problem itself is run open-loop, which might lead to unstable systems if constraints on the states are present. Therefore it would be good for the optimization problem to also be a closed loop problem, but this is very hard to implement practically [Allgöwer et al., 2004]. Different methods to provide robust NMPCs are described in [Rawlings and Mayne, 2012] and [Grüne and Pannek, 2011].

2.2.4 Using NMPC with an Observer

If all states of the system are not measurable they will need to be estimated by an observer. For linear systems the observer and controller can be decoupled and

designed separately without any loss in performance or stability. For nonlinear systems this is no longer the case, stability is only guaranteed for small observer errors and in general nothing can be said about the degree of smallness [Allgöwer et al., 2004]. Often a decoupled approach will still work for nonlinear systems and this is what will be tried in this thesis.

2.3 Numerical Optimization

The main focus of optimization theory is finding the minimum of some function under certain conditions and constraints. For general optimization problems with arbitrary functions and constraints, no algorithm exists that can solve all problems efficiently. Instead the problem needs to be restricted to a certain subgroup of problems that one particular method can handle. This is true for both symbolic and numerical optimization.

One important subset of problems for optimization are problems defined on convex sets or using convex functions. In convex optimization a local optimium is guaranteed to be a global minimizer. Unless the problem is strongly convex the solution is not guaranteed to be unique however. The proof of this and more information about optimization theory can be found in [Böiers, 2010]. This attribute makes convex problems much easier and faster to solve. Even if a whole problem is not convex, having convex constraints and using convex sets can make it possible to solve a problem much faster.

One of the subclasses of optimization problems is the nonlinear programming (NLP) problem described in [Wächter, 2009]. The problem is formulated as

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbb{R}^n} \; p(\mathbf{x}) \\ s.t.\; &b^L \leq b(\mathbf{x}) \leq b^U \\ &x^L \leq \mathbf{x} \leq x^U \end{aligned} \tag{2.4}$$

where $\mathbf{x}$ denotes the optimization variables, $p : R^n \to R$ and $b : R^n \to R^m$ are arbitrary discrete functions, and $b^L \in R^m$, $b^U \in R^m$, $x^L \in R^n$ and $x^U \in R^n$ are constant bounds. One method to solve this kind of problem is presented in [Wächter, 2009] and uses an interior point barrier method. The functions p and b are assumed to be at least once continuously differentiable and are not necessarily convex. More background on this method can be found in [Böiers, 2010].

Unless the problem is convex the solution that is found is not guaranteed to be a global minimizer. The algorithm has tools to avoid maximum and saddle points, but no guarantees are given [Wächter, 2009].

2.4 The Unscented Kalman Filter

State estimation is the concept of reconstructing the states of a system from the measured signals and control inputs. This is done by using an observer. The observer used in this thesis is an Unscented Kalman Filter (UKF), which is an observer for nonlinear systems. The UKF is a modified version of the Kalman filter, whose algorithm will be explained briefly before presenting the UKF framework.

2.4.1 The Kalman Filter

The Kalman Filter is a widely used algorithm for state estimation of non-measureable states or states subject to noise. It utilizes mean and covariance of the state to make an estimation of the true state value.

Consider the discrete system at time instant k, subject to process and measurement noise

$$\begin{aligned} \mathbf{x}_{k+1} &= f(\mathbf{x}_k, \mathbf{v}_k, \mathbf{u}_k) \\ \mathbf{y}_k &= h(\mathbf{x}_k, \mathbf{n}_k) \end{aligned} \tag{2.5}$$

where $\mathbf{x}_k$ are the states to be estimated, $\mathbf{u}_k$ are the input signals, $\mathbf{y}_k$ are the observed measurements, $\mathbf{v}_k$ is the process noise and $\mathbf{n}_k$ is the measurement noise. f and h are discrete functions that are assumed to be known. The algorithm of the Kalman filter is divided into two parts; prediction and correction. In the prediction step, the estimate $\hat{\mathbf{x}}$ of the states and its covariance $\mathbf{P}$ from the previous time instant are used to form the a priori estimate of the state $\hat{\mathbf{x}}^-$, measurement $\hat{\mathbf{y}}^-$ and covariance $\mathbf{P}^-$ of the state values at the next time instant

$$\hat{\mathbf{x}}_k^- = E[f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}_{k-1})] \tag{2.6}$$

$$\hat{\mathbf{y}}_k^- = E[h(\hat{\mathbf{x}}_k^-, \mathbf{n}_k)] \tag{2.7}$$

$$\mathbf{P}_k^- = E[\mathbf{e}_k^- \mathbf{e}_k^{-T}] \tag{2.8}$$

where $\mathbf{e}_k^- = \mathbf{x}_k - \hat{\mathbf{x}}_k^-$ is the a priori estimation error. When new measurements are sampled, the predicted estimates of the states and covariance are corrected in the correction step according to

$$\mathbf{P}_{x_k, y_k} = E[\mathbf{e}_k^- \mathbf{e}_{\mathbf{y}_k}^{-T}] \tag{2.9}$$

$$\mathbf{P}_{\bar{y}_k, \bar{y}_k} = E[\mathbf{e}_{\mathbf{y}_k}^- \mathbf{e}_{\mathbf{y}_k}^{-T}] \tag{2.10}$$

$$\mathbf{K} = \mathbf{P}_{x_k, y_k} \mathbf{P}_{\bar{y}_k, \bar{y}_k}^{-1} \tag{2.11}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}(\mathbf{y}_k - \hat{\mathbf{y}}_k^-) \tag{2.12}$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K} \mathbf{P}_{\bar{y}_k, \bar{y}_k} \mathbf{K}^T \tag{2.13}$$

where $\mathbf{e}_{\mathbf{y}_k}^- = \mathbf{y} - \hat{\mathbf{y}}^-$. The algorithm then iterates by using the current state and covariance estimate from the correction step to form the a priori estimates in the prediction step of the next time instant [Julier and Uhlmann, 2004].

2.4.2 The Unscented Transform

The Kalman filter can not be applied directly to a nonlinear system since (2.6) - (2.10) have no explicit expressions. Therefore the UKF uses an algorithm called unscented transform (UT) to approximate how the mean and covariance of the estimated states are transformed when f and h in (2.5) are nonlinear functions.

The general idea of the UT is to choose a set of points with a known mean and covariance and apply the nonlinear function to each of them, see Figure 2.7. These points are known as *sigma points* and the i:th sigma point is denoted $\mathbf{X}_i$. The statistical values of the transformed points are then calculated to form an approximation of the nonlinearly transformed mean and covariance [Julier and Uhlmann, 2004]. Consider the random variables $\mathbf{x}$ and $\mathbf{y}$ (both with dimension L) which are related

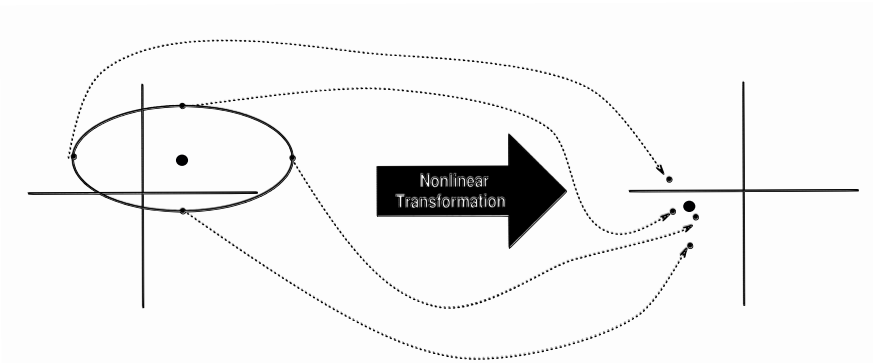

Figure 2.7 The principle of the UT. The sigma points in a symmetric set with known mean and covariance (left) are transformed nonlinearly and produce a new set of points (right). The statistics of the new set of points is used to approximate the transformed mean and covariance [Julier and Uhlmann, 2004].

through the nonlinear function $\mathbf{y} = g(\mathbf{x})$. Assume $\mathbf{x}$ has mean $\bar{\mathbf{x}}$ and covariance $\mathbf{P}_{\mathbf{x}}$. To calculate the statistical values of $\mathbf{y}$ we form $2L+1$ sigma points. The choice of sigma points comes in different variations. In this thesis they are chosen as described in [Wan and Van der Merwe, 2000]. The sigma points are chosen symmetrically

around the mean with corresponding weights W_i as

$$\mathbf{X}_i = \begin{cases} \bar{\mathbf{x}} & i = 0 \\ \bar{\mathbf{x}} + (\sqrt{(L+\lambda)\mathbf{P}_\mathbf{x}})_i & i = 1,...,L \\ \bar{\mathbf{x}} - (\sqrt{(L+\lambda)\mathbf{P}_\mathbf{x}})_{i-L} & i = L+1,...,2L \end{cases} \tag{2.14}$$

$$W_0^m = \frac{\lambda}{L+\lambda} \tag{2.15}$$

$$W_0^c = \frac{\lambda}{L+\lambda} + (1-\alpha^2+\beta) \tag{2.16}$$

$$W_i^m = W_i^c = \frac{1}{2(L+\lambda)} \qquad i = 1,...,2L \tag{2.17}$$

$$\lambda = \alpha^2(L+\kappa) - L \tag{2.18}$$

where the parameter α determines the spread of the sigma points around the mean, and is usually set to a small positive value [Wan and Van der Merwe, 2000]. The parameter β is used to include prior knowledge of the distribution of $\mathbf{x}$. $\beta = 2$ is for example optimal for Gaussian distributions, see [Julier and Uhlmann, 2004]). The parameter κ is a secondary, non-negative scaling parameter that is chosen to ensure the semi positive definiteness of the covariance matrix. The default choice is $\kappa = 0$ [Alkaya, 2014]. The expression $(\sqrt{(L+\lambda)\mathbf{P}_\mathbf{x}})_i$ corresponds to the ith row of the matrix square root.

Applying the nonlinear function to the sigma points yields

$$\mathbf{Y}_i = g(\mathbf{X}_i) \qquad i = 0,...,2L \tag{2.19}$$

where $\mathbf{Y}_i$ is the transformed sigma point. The mean $\bar{\mathbf{y}}$ and covariance $\mathbf{P}_\mathbf{y}$ of $\mathbf{y}$ are then approximated by the weighted mean and covariance of the transformed sigma points according to

$$\bar{\mathbf{y}} \approx \sum_{i=0}^{2L} W_i^m \mathbf{Y}_i \tag{2.20}$$

$$\mathbf{P}_\mathbf{y} \approx \sum_{i=0}^{2L} W_i^c (\mathbf{Y}_i - \bar{\mathbf{y}})(\mathbf{Y}_i - \bar{\mathbf{y}})^T \tag{2.21}$$

2.4.3 The UKF Algorithm

There are two common versions of the UKF algorithm; the augmented and the non-augmented version [Fuming et al., 2009]. The non-augmented one can be used when the process and measurement noise in (2.5) are considered additive, and has the benefit of using less sigma points in its estimation than the augmented version. In this thesis, the non-augmented version will be used. The algorithm presented below is based on the non-augmented UKF presented in [Fuming et al., 2009].

The UKF algorithm is initialized with

$$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0] \tag{2.22}$$

$$\mathbf{P}_0 = E[(\mathbf{x}_0 - \hat{\mathbf{x}}_0)(\mathbf{x}_0 - \hat{\mathbf{x}}_0)^T] \tag{2.23}$$

where $\hat{\mathbf{x}}_0$ is the initial estimate of $\mathbf{x}_k$ and $\mathbf{P}_0$ is the initial state covariance.

After initialization, the algorithm is divided into three steps for each time instant $k \in \mathbb{N}$. The first step is to calculate the sigma points. The estimate of the state vector, $\hat{\mathbf{x}}_{k-1}$, is used together with the covariance $\mathbf{P}_{k-1}$ to calculate the sigma points $\mathbf{X}_{i,k-1}$ according to (2.14). The corresponding weight to each sigma point is calculated according to (2.15) - (2.17).

The second step of the algorithm is the prediction step of the UKF. The sigma points are transformed by the nonlinear system. The resulting points are then used to obtain the a priori predicted mean $\hat{\mathbf{x}}_k^-$ and covariance $\mathbf{P}_k^-$ according to (2.20) and (2.21). The process noise is incorporated into the predicted state covariance by adding the process noise covariance matrix to the predicted state covariance.

The same procedure is done with the measurement function to obtain the a priori predicted mean $\hat{\mathbf{y}}_k^-$, covariance $\mathbf{P}_{\bar{y}_k,\bar{y}_k}$ and cross-covariance $\mathbf{P}_{x_k,y_k}$ of the measurement. The measurement noise is incorporated into the predicted measurement covariance by adding the measurement noise covariance matrix.

With the predicted covariance matrices, the Kalman filter gain $\mathbf{K}$ and the corrected state covariance $\mathbf{P}_k$ are calculated according to (2.11) and (2.13). The third step of the algorithm is performed when the measurement is available. Then the a priori estimates $\hat{\mathbf{x}}_k^-$ are corrected into $\hat{\mathbf{x}}_k$ as in (2.12). The three steps of the algorithm are then repeated to produce estimates for the next time instant.

2.5 Observability in the Nonlinear Case

An important concept in state estimation is the observability of a system. A system is defined in [Salau et al., 2014] as *observable* if for any initial state $\mathbf{x}_0$ and any final time $k_f > 0, k_f \in \mathbb{N}$, the initial state can be uniquely determined by knowledge of the sequence of inputs $\{\mathbf{u}_i\}$ and outputs$\{\mathbf{y}_i\}$ for all $k \in [0, k_f], k \in \mathbb{N}$.

Determining observability of a linear time-invariant system is straight-forward by checking the rank of its observability matrix. However, in the nonlinear case it is not as simple. Local observability for a nonlinear system can be proved formally by applying Lie derivatives to the system according as described in [Hendrick and Girard, 2010]. Proving observability using Lie derivatives can, however, easily become complex for large systems, as in the case of this master thesis.

Another method suggested in [Salau et al., 2014] is linearisation of the system at each discrete time instant and applying linear observability criterion. Observability

will then be guaranteed along a specific sampled trajectory as long as the linear observability criterion is satisfied in each time instant. The method is simple to implement and was therefore planned to be used for observability analysis in this thesis.

2.6 Disturbance Modelling

To cope with different disturbances to the system and model the kind of disturbance and its magnitude need to be analysed. Disturbances can enter the system in different ways. Three different kinds of disturbances are described in [Rawlings and Mayne, 2012]:

1. Measurement noise - Noise that enters the system through sensors and has the form $y = h(x) + v$, where v is the added noise. The noise is generally considered to be white noise with zero mean.
2. Process noise - Modelling uncertainties and noise inside the real process. The system function has to be changed like $x^+ = f(x,u,\theta)$ where θ represent parameters that are unknown.
3. Additive process noise - differences between the process and the model that can be described with an added noise variable, i.e. $x^+ = f(x,u) + w$ where w is the noise.

Another possibility is to have a load disturbance that enters on the input of the system. This can be defined as

4. Load disturbance - disturbance that enters on the input of the real system and affects the update equation as $x^+ = f(x,u+d)$.

In this thesis, (1), (3) and (4) are considered where (1) is analysed compared to measurements from the real plant and (3) and (4) are analysed compared to the detailed model.

2.7 The Modelica and Optimica Languages

Modelica is an open source, object based, modelling language [Aronsson et al., 2014]. It uses an hierarchical structure based on components which are defined using sub-components and symbolical equations. The equations are interpreted acausally which means that it makes no difference which order equations are defined. Instead Modelica connects different components using connection equations

and solve differential equations to calculate the dynamic behaviour of the model, based on some initial condition and input. The language uses both a graphical and a textual representation of the components which are linked so they always correspond to each other. More information about the language specification can be found in [Aronsson et al., 2014].

Modelica facilitates both building, simulation and transferring of models. Transferring is done using the Functional Mock-Up Interface (FMI) standard. This is a language independent standard that allows the exchange of equation based models between different tools and languages in the form of data packages called functional mock-up units (FMU) [Blochwitz et al., 2011]. An XML file is used to define all variables used in the model and then a file with C-code is used to define all the equations.

Standard Modelica does not support terminology for optimization problems. To facilitate this an extension to Modelica, called Optimica, has been introduced [Åkesson, 2008]. This extention introduces concepts such as cost functions, free variables and initial guesses and makes it possible to define large classes of optimization problems. A common concept is using previously designed Modelica models and extend them with Optimica code to add optimization. This extension is supported by the compiler JModelica.org developed at Modelon [Modelon AB, 2014].

In this project Modelica was used to create both a detailed model of the power plant and an optimization model while Optimica was used to create the optimization formulations in the optimization model.

3

Method

3.1 Delimitations

The optimization model of the start-up was initially divided into two separate models, corresponding to the first and second phase of the start-up described in section 2.1.1. Discussions with Vattenfall led to the decision that the model corresponding to the first phase was to be used. The motivation was that the first phase had the highest potential for optimization since it is longer than the second phase and because oil is partly used as fuel which is more expensive than lignite.

It was also decided that the soft start should be the scenario to investigate. The motivation was that it is a realistic scenario in the case of frequent shut-downs and start-ups. The current initial values of the detailed model were also close to those of a soft start.

The original plan was to have the detailed model working as the target system and interface with it through an FMU in JModelica.org. However, due to the complexity of the detailed model this was not possible. Attempting to instead interface with the detailed model in Dymola, using Python, it was discovered that it was not possible to simulate the detailed model iteratively. Hence it could not be used with the NMPC. The UKF could however still be used offline on a pre-simulated start-up trajectory of the detailed model.

3.2 Development Tools

The main tools that have been used are the open source program JModelica.org developed by Modelon [Modelon AB, 2014] and the programming language Python. JModelica.org has support for the Optimica extension of Modelica as well as standard Modelica and importing and exporting FMUs. It has solvers for both simulation and optimization and uses Python as a scripting language to interact with the user. As well as a scripting language for JModelica, Python was used to create the program structure and the UKF is implemented as a Python class. JModelica.org

was used for optimization in the NMPC algorithm and for simulation of the optimization model, imported as an FMU, in the UKF and NMPC.

JModelica.org uses the open source software IPOPT to solve optimization problems. IPOPT uses the algorithm described in Section 2.3 to solve the problem. To convert the problem into the correct form JModelica.org uses the CasADi interface [Andersson et al., 2011]. A method called direct collocation is used to convert the continuous Modelica model into a discrete NLP problem. Direct collocation is based around fitting a number of polynomials so that the function and derivative values match at a number of points, called collocation points, and using these polynomials to approximate the original system. The time space is divided into a certain number of elements and an equal number of collocation points are put inside each element. In the NMPC implementation the elements are chosen as one time step. More collocation points means that the original system will be more closely approximated, but result in a larger NLP problem.

Modelling was done using the commercial program Dymola from Dassault Systems [Dassault Systemes, 2015]. Dymola is a tool for creating and simulating Modelica code. It has both a graphical and a textual interface that are connected so they always match each other. In this project Dymolawas used to edit the optimization model to suit the needs of this project and to run tests and simulations on the detailed model. There is also a Python/Dymola interface that makes it possible to simulate a model from Python using Dymola.

3.3 Modelling

3.3.1 Critical Components and Stress Constraints

The optimization model developed in [Runvik, 2014] needed to be changed to match the detailed model better and to incorporate a new critical component. The previous model included the separator wall as a critical component but in [Runvik, 2014] this was considered not to be critical for the start up. The results in [Hübel et al., 2014a], suggests that the outlet header of the reheater 2 (RH2) component might be critical instead. Therefore the separator wall was removed from the model and a header component similar to the superheater 4 header (SH4) was added, together with a controllable valve that represent the IP-T bypass valve.

The specification from Vattenfall was that the critical components should be able to withstand stress from 2000 cold starts. In order to meet this demand, Vattenfall provided limits on admissible stress levels during start-up for the RH2 and SH4 headers, see Table 3.1. There are three limits for each header, where stresses above the most conservative value will ensure that there is no lifetime consumption. The middle value is the highest stress level on the components with the current start up procedure in the power plant. The critical limit is where the components will hold for 2000 start-ups.

Minimum allowed stress	SH4 header	RH2 header
No fatigue	-47.1 MPa	-44.1 MPa
Current level	-84 MPa	-96.3 MPa
2000 cold starts	-253.4 MPa	-259.5 MPa

Table 3.1 Stress level limits on SH4 and RH2 headers.

After adding the RH2 header, it was shown in early offline optimizations that the stress limit was active along the optimized trajectory, making it a well justified addition to the model.

3.3.2 Model Inputs

Both a version with a new input added, that controlled the IP-T bypass valve, and a version without this new input were tested. When the new input is not added the control of the IP-T bypass valve is handled internally. When the input is added, it can either control the IP-T bypass valve directly or control a set-point for a PI controller in the same way as the HP-T bypass valve is controlled in the SH section. The benefit of controlling a valve directly is that it removes some dynamics from the system and should make it easier to control, but the version with a built in PI controller is more similar to the control system in the real plant. For the finalized model it was chosen to control both the IP-T and the HP-T bypass valves directly.

To match the inputs of the real plant the input on the gas side needed to be changed. The input used to be flow of flue gas, but was changed to firing power. The mass flow of the flue gas is kept constant at a rate of 316 kg/s and the relation between firing power P_F and flue gas temperature T_{fg} is a simple quadratic function, $T_{fg} = 5431 \cdot {P_F}^2 - 1253 \cdot P_F + 621$. The quadratic relation was derived from a least square fitting to the detailed model, performed by Moritz Hübel. The comparison between the derived functions and trajectories from the detailed model is shown in Figures 3.1 and 3.2. The derived function for flue gas temperature has its minimum in 0.115 which means that for very low firing powers the function will have unphysical behaviour and fit the detailed model poorly. Therefore a minimum constraint of 0.12 was introduced on the firing power to avoid approaching the minimum value.

The finalized optimization model with added RH section, direct valve control and firing power input can be seen in Figure 3.3.

3.3.3 Model Validation and Fitting

The parameters of the updated optimization model were tuned to the detailed model. The difference between the models was investigated using a reference soft start of the detailed model. This was done by simulating the optimization model with the input trajectories from the detailed model as inputs. The main differences were the energy balance and pressure drops. The valve characteristics were also very

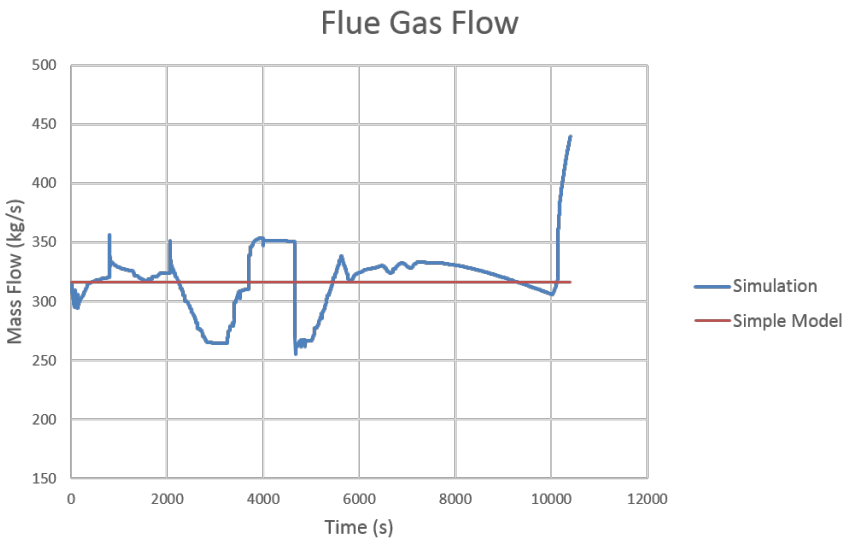

Figure 3.1 A comparison between the flue gas flow of the detailed model and the approximation used in the optimization model.

different in the two models and the separator level in the simulation of the detailed model was around 15.5 m while the set-point in the optimization model was 14 m.

There seemed to be more energy in total in the optimization model compared to the detailed model. The reason for this was the simplified firing power conversion in the optimization model and the fact that the spray attemperators are not modelled in the optimization model. This was assumed to be handled by adding a *disturbance state* to the firing power input.

In addition to the higher total energy, the distribution of energy in the boiler was different. There was too much energy taken from the flue gas in the SH section and too little in the evaporator leading to very high temperatures in the SH section. To compensate for this in a simple way, the value of the coefficient of heat transfer between the flue gas and the evaporator wall, $alpha\_gas\_wall$, was increased in the optimization model. The result of this can be seen in Figures 3.4 and 3.5. Since the mass flow is much higher in the evaporator the enthalpy difference, and therefore temperature difference, is negligible there while the difference in the live steam temperature is significant. The value of the heat transfer coefficient was chosen so that the live steam temperatures between the models would fit as well as possible resulting in increasing $alpha\_gas\_wall$ from 195 to 600 W/m^{2}K.

The pressure drops were also tuned in the optimization model. The difference in

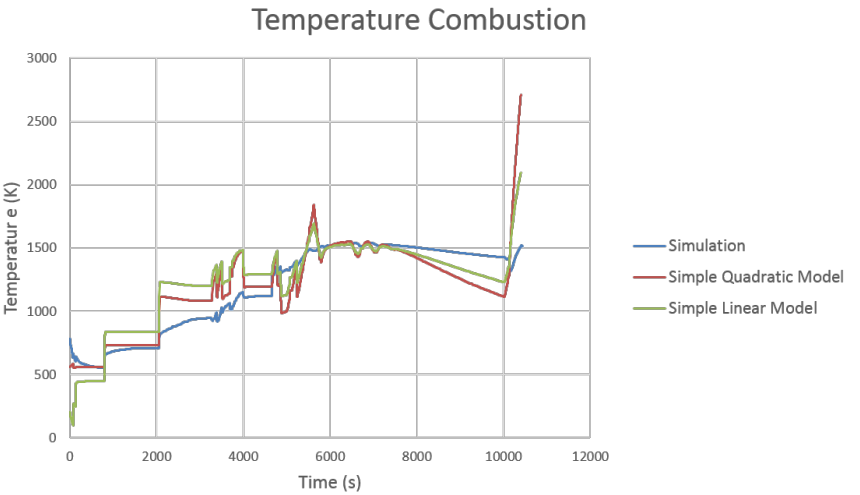

Figure 3.2 A comparison between the flue gas temperature of the detailed model, the quadratic relation used in the linear model and a linear relation that was also considered.

pressure drops between components was analyzed by holding the LS and RS pressure the same in both models. The results showed large differences in the pressure drops and pressure to mass flow ratio, see Table 3.2. Since the pressure drop is flow dependent in both models, the flow coefficients of the pressure drop components in the optimization model where modified so that the quotient between pressure drop and mass flow was the same as in the detailed model. As in [Runvik, 2014] the pressure drop is evenly distributed over all components in the same section. When measuring the pressures using the new flow coefficients in the optimization model, the pressures in the models matched almost perfectly over the whole soft start reference simulation.

The difference in bypass valve dynamics was not possible to solve without making larger modifications. In contrast to the optimization model, the detailed model has density dependent valves which are controlled by an elaborate control system. Similarly to the firing power, the valve differences were represented with disturbance states. To make sure both models had the same initial RS and LS pressures and temperatures with identical initial inputs, the flow coefficients of the bypass valves were tuned. The flow coefficient was changed to $3 \cdot 10^{-4}$ kg/sPa for the HP-T bypass valve and $6 \cdot 10^{-4}$ kg/sPa for the IP-T bypass valve.

During the start-up of the power plant, both models have a separator level set-point at 14 m. Despite the set-point, the detailed model kept a level of 15.5 m during start-up. The reason is that there is an overflow valve in the detailed model that keeps the

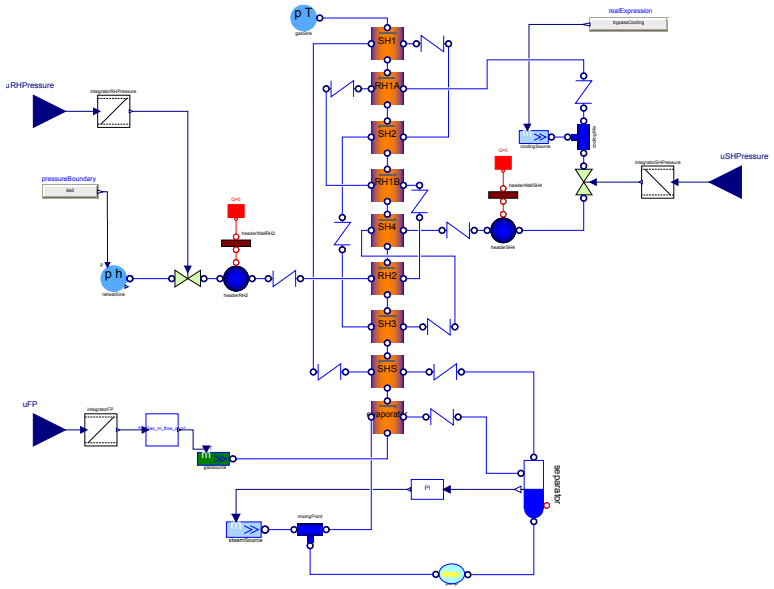

Figure 3.3 The optimization model used by the NMPC. The inputs (large blue arrows) are rate of change of firing power and openings of the HP-T and IP-T bypass valves.

maximum level at 15.5 m. During the start-up the evaporator does not evaporate as much steam as during regular operation which results in separator overflow. The simplest way to represent this was to change the set-point in the optimization model to 15.5 m which showed reasonable results. Not representing the overflow might contribute to the higher energy content in the optimization model though. When the overflow valve in the detailed model is activated the water will flow back to the feed water tank which represents an energy loss in the separator.

3.3.4 Modifications for Optimization and UKF

Changes had to be made to keep the optimization model simple to optimize and suitable to use with the UKF. These changes resulted in two different versions of the model, one for the UKF and one for the NMPC.

For the UKF the thermodynamic states for the steam was changed back to enthalpy and pressure as in [Andersson, 2013]. The reason was that all functions in the *water_poly* package were explicit in pressure and enthalpy, and using density and internal energy resulted in the need to invert the functions. Inverting functions made the initialization of the model difficult. Since each sigma point in the UKF requires

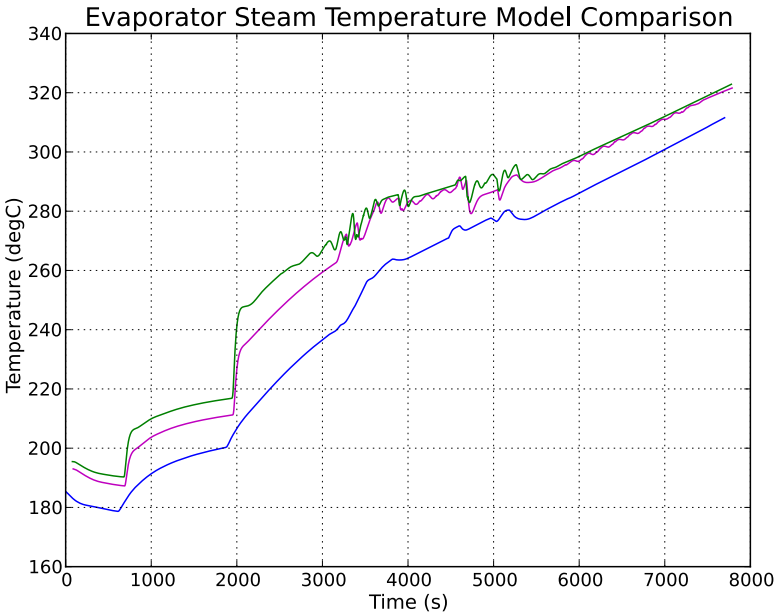

Figure 3.4 Comparison of the evaporator steam temperature between the detailed model (blue line) and the optimization model with *alpha_gas_wall* = 195 (magenta line) and 600 W/m^{2}K (green line).

an initialization, the model worked better with pressure and enthalpy as states.

To make the model more optimization friendly the fast dynamics in the system needed to be removed. The reason is that the collocation will become a very stiff problem if fast dynamics are present. The fast dynamics were the PI controllers for the bypass valves and the dynamics of the steam, which are considerably faster than the temperature dynamics of the thick-walled metal components. Therefore the PI controllers for the bypass valves were removed and the steam dynamics were exchanged for static relations in the optimization model used by the NMPC. However, the steam dynamics were not exchanged in the UKF version of the optimization model. The reason was that the static relations resulted in an equation system that proved too difficult to solve in many sigma point initializations.

The models with and without dynamic steam were very similar. In steady state the models were equal, but the static version had faster responses to changes in inputs and also some fast dynamics in the dynamic version of the model were not present in the static version. The results of the comparison can be seen in Figure 3.6. The maximum difference between the models was 0.41% for the temperature and 4.13%

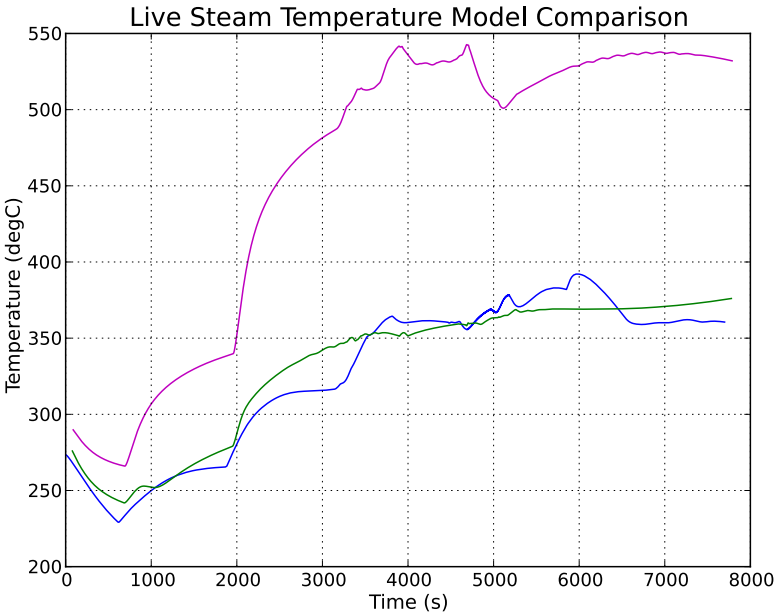

Figure 3.5 Comparison of the LS temperature between the detailed model (blue line) and the optimization model with *alpha_gas_wall* = 195 (magenta line) and 600 W/m^{2}K (green line).

for the pressure and the mean difference was 0.15% and 1.93% respectively.

Like the PI controllers for the bypass valves, the separator level control represents another fast dynamic in the system. Ideally, the separator should be replaced by an ideal component that separates the liquid and steam water and have an ideal controller that controls the mass flow to keep the level constant. This was tested in the end of the thesis work with a separator component developed by our supervisor at Modelon, Stephane Velut. The mass flow required to keep the separator at a constant level was derived explicitly and sent through a first order filter that removed any fast dynamics before sending it as input to the feed water component. This way the PI controller for the separator level could be removed. Due to the time constraints on the project this modified version of the optimization model was not validated against the detailed model and only limited experiments were run this setup.

A few other, minor, changes were made to the optimization model to make it more optimization friendly. All differential equations were scaled so that both sides had a magnitude close to one, making them numerically easier to solve. The cost function was also divided into parts for each cost term that were then added together. A

Section	Model	Pressure Drop	Mass Flow	Quotient (dP/ϕ)
Evaporator	Detailed	$2.37 \cdot 10^6$ *Pa*	888.0 kg/s	$2.67 \cdot 10^3$
	Optimization	$1.47 \cdot 10^6$ *Pa*	883.0 kg/s	$1.66 \cdot 10^3$
Superheater	Detailed	$1.64 \cdot 10^4$ *Pa*	59.0 kg/s	$2.78 \cdot 10^2$
	Optimization	$1.11 \cdot 10^6$ *Pa*	111.9 kg/s	$9.87 \cdot 10^3$
Reheater	Detailed	$9.00 \cdot 10^3$ *Pa*	61.7 kg/s	$1.46 \cdot 10^2$
	Optimization	$2.39 \cdot 10^5$ *Pa*	116.7 kg/s	$2.05 \cdot 10^3$

Table 3.2 Comparison between pressure drops and mass flows in the detailed and optimization model. The values are extracted at the time where the mass flow is equal to the mean mass flow in the detailed model during phase 1 of the start-up and all measurements for each section are taken at the same time instance.

nominal value was then added to each part of the cost function so that they could be scaled properly by the optimizer. Also, minimal and maximal values of -0.1 and 10 were added to the the valve openings. This was well outside the normal range of operation so as not to change the solution, but still meant that the optimizer had a smaller set of feasible points to search.

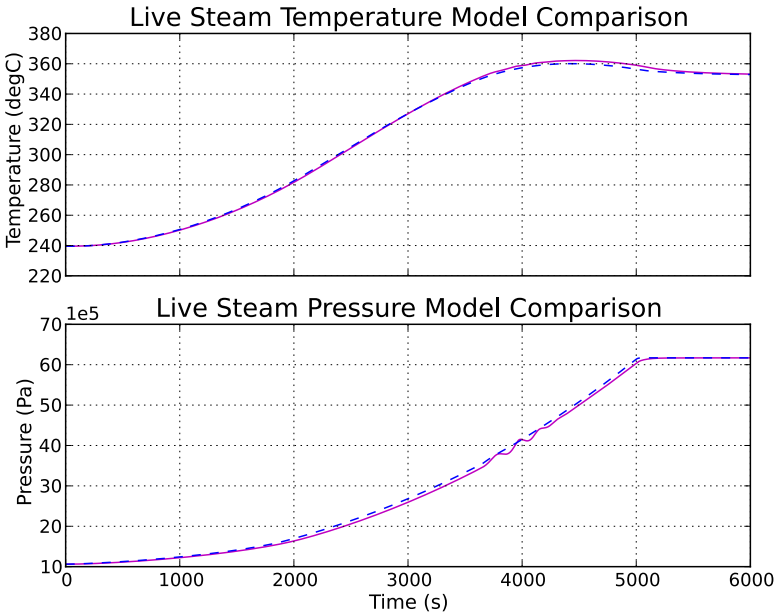

Figure 3.6 Comparison of the LS temperature and pressure between the optimization model with dynamic steam state (whole, magenta, line) and the one with static steam states (dashed, blue, line). The input is a constant firing power ramp with a slope of $7 \cdot 10^{-5}$ /s from 0 to 5000 s and a constant firing power after that. The valve openings are kept constant.

3.4 Implementation of the NMPC

The NMPC controller was implemented using Python, JModelica.org and a general MPC class presented in [Axelsson, 2015]. It uses the optimization model with static steam relations presented in Section 3.3 as its internal model.

The NMPC setup was tested on the nominal system to tune parameters and see if the approach was feasible for the model. This was done by using the optimization model as the target system. Later, measurement noise were added to the nominal system to test the robustness of the controller.

The original plan was to run the NMPC with the detailed model as the target system but as is outlined in Section 3.1 this was not possible.

3.4.1 The MPC Class

The MPC class that has been used in this project was developed in Python using JModelica.org to solve the optimal control problem. It was developed in [Axelsson,

2015]. The class provides methods to update the state of the internal optimization model and perform one MPC step. It uses the built in IPOPT solver to solve the optimization problem but provides features such as warm start of the optimization each iteration. The warm start is performed by shifting the result array one step to the left and setting the new last value equal to the current last value. This array is used to initialize the next iteration of the MPC. It also provides support for softening constraints, setting the prediction horizon and extracting states from simulation results.

The implementation is based on an initialization step and a loop where the MPC algorithm is applied iteratively by first setting the states of the optimization model, then doing one MPC calculation and sending the result into the system. The methods for this and how they are to be used are described in [Axelsson, 2015].

3.4.2 Optimization Parameters

In offline optimization tests the updated optimization model proved very hard to optimize. The offline optimizations were run for 5000 s and the optimizer was only able to find a solution when using a prediction interval that was equal to, or almost equal to, the whole runtime of the simulation. With this large prediction horizon the number of degrees of freedom for the optimization problem needed to be minimized in order for it to be solvable. This was done partly by choosing only ten elements for the optimization making the element length 500 s. The number of collocation points in each element were chosen to be three which is the least amount of points that still provide an acceptable discretization. The minimum sample time of the NMPC is the length of one element which means that a sample time of 500 s had to be chosen.

Only piecewise constant inputs were considered. This means that the inputs are held constant over each sample interval which simplifies the optimization problem significantly. A control horizon different from the prediction horizon was implemented by adding blocking factors to the optimization problem. Blocking factors tell the optimizer that it should keep the input constant over a specified number of elements. Therefore, to have a control horizon of n elements a list that has the length n, with the first $n-1$ elements being ones and the last element being the prediction horizon minus n, should be used as blocking factors for the optimization problem. In our experiments a slightly different list of [1,1,1,2,5] was used providing a slightly smoother transition between the controlled elements and the rest of the prediction horizon. The reason for using a control horizon of only 5 elements was to reduce the degrees of freedom in the optimization problem further and that in an NMPC application only the control signal for the first element will be used anyway.

3.4.3 Cost Function Selection

To make the optimization problem possible to solve some parts of the standard problem needed to be simplified. In this thesis only quadratic cost functions were

considered. This means that the stage cost function has the form

$$\ell(\mathbf{x}_i, \mathbf{u}_i) = (\mathbf{x}_i - \mathbf{x}_{ref})^T A(\mathbf{x}_i - \mathbf{x}_{ref}) + \mathbf{u}_i{}^T B\mathbf{u}_i \tag{3.1}$$

where A and B are symmetric matrices and $\mathbf{x}_{ref}$ are the reference values for the set-points that are used.

The A and B matrices were both chosen to be diagonal, with non-zero weights only for the states that had a corresponding set-point. One exception to this was the firing power where a weight was added to the integrated firing power input. This was added to penalize large firing power values making the start-up more cost efficient. Also, since the state is always positive the value was not squared since this distorts the cost between low and high values.

The weight for each set-point were initially the same as in [Runvik, 2014], with the addition of a weight for RS pressure because of the added IP-T bypass valve input to the model. Offline optimizations with this setup resulted in two different optimal solutions with different state values at the final time. Both solutions reached the set-points, but one had about half the firing power and significantly smaller mass flows. The state values were compared to the detailed model and the result with higher firing power had state values that matched the detailed model better. To push the optimizer into choosing this end result, the RS temperature was added as a new set-point. The reason for choosing this value as set-point was that it is one of the states that is controlled in the start-up of the real plant. Its value was also very different in the two optimization results making it a good candidate to make sure that the correct end result was reached. With the RS temperature there were four set-points in the optimization problem, with only 3 degrees of freedom. Therefore all states for which there are set-points cannot be controlled independently. The RS temperature was therefore set to be the end value the optimizer reached previously so that all set-points could still be reached.

3.4.4 Constraint Implementation

The values of the constraints were set to make sure that the stress constraints on the real systems were not violated. Initial tests showed that the execution time was much shorter using soft constraints rather than hard constraints. Soft constraints mean that a large cost is added to the cost function when the solution to the optimization problem violates the constraints. Soft constraints can be violated a bit but the optimizer will push the system back into the feasible set.

During test runs with the optimization model as target system the solution violated the constraints even using hard constraints. Therefore it was decided to set the constraints as soft constraints but keeping them very tight, knowing that they will likely be violated slightly. In addition, model differences mean that the stresses might be larger in the real process than in the optimization model.

The chosen constraints were the no fatigue constraints in Table 3.1 which left a good safety margin to the critical constraints for the power plant. The input constraints were kept as hard constraints as that does not make the problem much harder to solve.

3.4.5 Cost Function Evaluation

With the setup defined in Sections 3.4.2, 3.4.3 and 3.4.4 it was possible to solve the optimization problem offline and for most of the NMPC iterations. The problem was, however, sensitive to parameter changes. To assess the impact of different weights in the cost function a number of offline optimizations were performed.

The results were mostly consistent with what would be expected from theory. It is possible to push the optimized trajectory in different ways by manipulating the weight on the appropriate set-point. A quantitative result can be found in Table 3.3. The experiment was done before the additional set-point on RS temperature was added and the optimizations where the lower firing power solution was chosen can be easily seen as the firing power integral is almost half that of the other experiments, i.e., all with a firing power integral below 1000.

The cost terms show what impact the weights actually have and should be viewed in comparison with each other and with other experiments. Since these terms are the weight multiplied with the deviation from the set-points it shows how the optimality of the corresponding trajectory changes between experiments. It can be used to get a quantitative figure of the impact of different weights on the solution. If for example the corresponding weight stays the same and the cost term increases the new solution is less optimal in that its mean value is further away from the set-points. If the weight is increased however, and the cost term is increased by a smaller factor, the new solution is more optimal.

There were big differences in the time it took to reach the set-points between the different setups. Among the setups that reached the higher firing power the difference between the longest and the shortest one was about 29%. The difference in firing power for these setups was about 15%. In addition to this experiment a weight of 10^8 was tried on the inputs instead of 10^6, but this had almost no affect on the trajectory.

Weights				Mean Costs					Set-points Reached
FP	LST	LSP	RSP	FP	LST	LSP	RSP	FP integral	At Time
1	0.01	0.01	0.1	0.3	29.6	10.0	1.8	1739	4000 s
1	0.01	0.01	10	0.3	30.9	10.2	155.3	1671	4250 s
1	0.01	1	0.1	0.3	30.8	971.1	1.8	1666	4500 s
1	0.01	1	10	0.3	31.5	974.9	167.7	1592	5000 s
1	0.1	0.01	0.1	0.4	283.7	10.7	3.8	1838	3500 s
1	0.1	0.01	10	0.3	288.2	12.7	174.9	1758	4500 s
1	0.1	1	0.1	-	-	-	-	-	Diverged
1	0.1	1	10	0.3	303.1	974.4	173.5	1689	4000 s
100	0.01	0.01	0.1	19.8	29.2	9.7	1.8	994	2500 s
100	0.01	0.01	10	19.8	30.3	9.8	154.6	993	2500 s
100	0.01	1	0.1	19.8	29.6	970.1	2.0	992	3500 s
100	0.01	1	10	-	-	-	-	-	Diverged
100	0.1	0.01	0.1	35.8	284.0	10.7	3.5	1809	3500 s
100	0.1	0.01	10	33.4	291.2	13.1	171.9	1684	3750 s
100	0.1	1	0.1	33.3	301.2	973.1	1.9	1683	3750 s
100	0.1	1	10	32.9	303.9	974.4	172.9	1664	4000 s
0	0.01	0.01	0.1	0	29.0	10.5	1.9	1763	3750 s
1	1	0.01	0.1	0.4	2798.5	18.1	16.5	1787	3500 s

Table 3.3 Offline optimization of the optimization model. All input weights are kept constant at 10^6. FP, LST, LSP and RSP are firing power, LS temperature, LS pressure and RS pressure. Weights stand for weights in the cost function. The mean costs are the mean value of the corresponding term in ℓ over all elements. The FP integral is the integral of the firing power over the interval [0, 5000].

3.4.6 Initiation

In the same way as in [Runvik, 2014] the target system was simulated once to get rid of transients and then the internal model of the MPC class was initialized using the state of the simulated model. However, instead of using a simulation with ramped input signals as initial guess to the NMPC, the result of an offline optimization was used. This was done because the optimization problem was much harder to solve with the new constraints. When using an initial trajectory close to the correct optimal solution the optimization was considerably easier.

3.5 Implementation of the UKF

As JModelica.org is interfaced through a Python shell, and the MPC class was implemented in Python, it was decided to implement the UKF as a Python class. The algorithm used by the UKF class is described in section 2.4.3. The nonlinear model the UKF uses for predictions is represented as an FMU, where a discrete time step corresponds to simulating the FMU for one sample interval. Each sigma point corresponds to one set of values for the state variables in the model. These are set as initial values in the FMU before each simulation. The FMU is then reset to initial conditions before setting a new sigma point.

The noise in the UKF implementation was assumed to be additive, corresponding to the third noise representation in Section 2.6, and also zero-mean. This is a simple but often found assumption in UKF implementations [Yuanxin et al., 2005]. The estimated covariance of the noise is set through input matrices.

The implementation consists of three classes. The class `UKF` is the main class used for producing state estimates, with methods for prediction and measurement update as described in Section 2.4.3. The class `UKFOptions` is a class containing design parameter values for the UKF. Lastly, the class `ScaledVariable` is a class representing a variable, which contains information such as name, current value and nominal value.

3.5.1 The UKF Class

A `UKF` object is created by calling the constructor:

```
#Create a UKF-object:
ukf = UKF(model, x_0, measurements, h, options)
```

The input arguments to the constructor are:

- `model`: A reference to an `FMUModel`-object, representing the observer model.
- `x_0`: A dictionary with strings of the state names as keys and initial values of each state as values.
- `measurements`: A list of strings of the names of the variables in the model that are considered measurable.
- `h`: The sampling interval in seconds.
- `options`: A reference to an object of the `UKFOptions`-class.

The main public methods of the `UKF`-class are `predict` and `update`. The method `predict` corresponds to the prediction step described in Section 2.4.3, and is called on the `UKF`-object `ukf` as:

```
#Make a prediction:
ukf.predict(u, known_values)
```

The input arguments to `predict` are:

- `u`: The input object to the simulation, which should follow the structure defined in the FMI-standard for simulation of FMU's. This should be a tuple where the first element is a list of strings of the input names, and the second

element is a matrix with input values. The first column is the time vector, and the following columns are corresponding input values at each time instant. It is also possible to send in the input function generated by the MPC class from [Axelsson, 2015].

- `known_values`: A dictionary whose keys are strings of the names of states which are considered known (i.e are not estimated by the UKF), and whose values are the value of each known state.

When measurements from the process are available, the method `update` is called to produce the corrected state estimates according to (2.12):

```
#Make a correction and retrieve the state estimates:
x = ukf.update(y)
```

The input argument and output from `update` are;

- `y`: A dictionary whose keys are strings of the measurement names and whose values are the measurement values.
- `x`: A dictionary whose keys are strings of the state names and whose values are the current estimations.

In addition to `predict` and `update` there are the public methods `get_options` and `update_options` to retrive a copy of and set new parameter values for the internal `UKFOptions` object respectively.

In addition to its public methods, the `UKF`-class has the following two private methods:

- `_calc_weights`: Calculates the sigma point weights according to (2.15) - (2.17).
- `_calc_sigma`: Calculates the sigma points around the current state estimate according to (2.14).

3.5.2 The UKFOptions Class

The `UKFOptions`-class extends the JModelica.org class `OptionBase` [*Python API Docs, Common Folder*] and is used for storage of the sigma weight parameters α, β and κ and the covariance matrices $\mathbf{P}_0, \mathbf{P}_v$ and $\mathbf{P}_n$ described in Section 2.4.3. It stores these values in a dictionary, and to set and retrieve these parameters the following string keys are referred to:

- 'alpha': Key for a float value corresponding to α.
- 'beta': Key for a float value corresponding to β.
- 'kappa': Key for a float value corresponding to κ.
- 'P_0': Key for a dictionary corresponding to $\mathbf{P}_0$.
- 'P_v': Key for a dictionary corresponding to $\mathbf{P}_v$.
- 'P_n': Key for a dictionary corresponding to $\mathbf{P}_n$.

The covariance matrices are represented as dictionaries, where the keys are strings of the state or measurement names, whose corresponding values are the covariance values. The representation assumes that the covariance matrices are diagonal. For instance, in the case of two states called x1 and x2 subject to process noise with variance 1 and 10 respectively, $\mathbf{P}_v$ would be represented as:

```
#Create a process noise covariance matrix
   represented by a dictionary
P_v = {'x1':1, 'x2':10}
```

3.5.3 The ScaledVariable Class

The range and magnitude of the variable values in the power plant varies a lot. Handling values of very different scale can be numerically unsafe and lead to poor results. One example is the calculation of the matrix square root in (2.14), which in the implementation is done by a Cholesky decomposition of the covariance matrix. Having values that differ in several orders of magnitude will make the matrix ill-conditioned, which risks making the Cholesky decompositon break down [Yanagisawa et al., 2014]. This motivated the implementation of the class ScaledVariable.

The class ScaledVariable represents a state or measured variable, and holds a string containing the name of the variable, the actual value of the variable and the nominal value of the variable. The class is used internally in the UKF class to use scaled values in all computations. The values are scaled back to their original value when simulating in JModelica.org and when returning values to the user.

The following public methods are available in the ScaledVariable class:

- get_name: Returns a string with the name of the variable.
- set_actual_value: Set the value of the variable using an unscaled value.
- set_scaled_value: Set the value of the variable using a scaled value.

- `get_scaled_value`: Returns the scaled value of the variable.
- `get_actual_value`: Returns the unscaled value of the variable.
- `get_nominal_value`: Returns the nominal value of the variable.

3.5.4 Running the UKF in JModelica.org

In this section an example of using the `UKF`-class for state estimation with models represented as `FMUModel`-object is presented. The model used by the UKF in this example is assumed to have two states called `x1` and `x2`, which are to be estimated by measuring `x1`. The model also has the input `u`.

First the Modelica-models corresponding to the actual process and the observer model are compiled and loaded as `FMUModel`-objects using the JModelica.org functions `compile_fmu` and `load_fmu`;

```
from pymodelica import compile_fmu
from pyfmi import load_fmu

#Compile process and observer model as FMU's
process_fmu = compile_fmu('Process', 'process.mo',
    separate_process = True)
obsModel_fmu = compile_fmu('ObserverModel',
     'observerModel.mo', separate_process = True)

#Load process and observer model as FMUModel-objects
process = load_fmu(process)
obsModel = load_fmu(observer_fmu)
```

Then the parameters for the UKF are defined, and a `UKF`-object is created:

```
#Create dictionary for initial state estimate
x_0 = {'x1':0.0, 'x2': 0.5}

#Create a list of which variables are measured
measurements = ['x1']

#Define the sampling interval (in seconds)
h = 0.1

#Create a UKFOptions-object, and assign parameters
opt = UKFOptions()
opt['P_0'] = {'x1': 1.0, 'x2': 1.0}
opt['P_v'] = {'x1': 1e-3, 'x2': 1e-3}
```

```
opt['P_n'] = {'x1': 1e-1}
opt['alpha'] = 1e-3
opt['beta'] = 2.0
opt['kappa'] = 0.0

#Create a UKF-object with the defined parameters
ukf = UKF(obsModel, x_0, measurements, h, opt)
```

With the `UKF`-object available, it can be used together with the FMU representing the process to estimate the states x1 and x2. First a prediction for the next sample instant is performed:

```
#Define input to the system.
import numpy as N
t = N.array([0.0, 0.005])
u = N.array([1.0, 1.1])
u_traj = N.transpose(N.vstack((t,u))
input_object = ('u', u_traj)

#No states are considered known, so send an empty
   dictionary
known_values = {}

#Make a prediction
ukf.predict(input_object, known_values)
```

Then the process model is simulated and measurements are extracted. The measurements are then used to perform a measurement update, and the state estimates are retrieved:

```
#Simulate the process FMU
result = process.simulate(start_time = 0.0,
   final_time = h, input = input_object)

#Retrieve the last value (index -1 in Python) of the
    measured variable from the simulation result.
y = {}
for meas in measurements:
    y[meas] = result[meas][-1]

#Send the dictionary containing the measurement to
   the ukf to retrieve the estimation.
x = ukf.update(y)
```

This framework can then be iterated to simulate the sampling of a real process, where the UKF is producing state estimates at each sample.

3.5.5 Validation of the UKF Implementation

The correctness of the implementation of the UKF class was verified by using it to reproduce the results of the UKF state estimation of the Van-Der-Pol-oscillator (VDP) scenarios described in [Rambabu et al., 2008]. In [Rambabu et al., 2008] the performance of an extended Kalman filter (EKF) and UKF are compared when estimating the states of the normal and reverse-time VDP, which are nonlinear processes with a stable and an unstable limit cycle respectively. The scenarios were chosen for comparison because the VDP-model is easy to overview with its two states, simplifying debugging of the UKF implementation.

The normal VDP-model is defined as

$$\dot{x_1} = x_2 \tag{3.2}$$
$$\dot{x_2} = \mu(1 - {x_1}^2)x_2 - x_1 \tag{3.3}$$
$$y = [x_1, x_2]^T \tag{3.4}$$

and in reverse-time as

$$\dot{x_1} = -x_2 \tag{3.5}$$
$$\dot{x_2} = -\mu(1 - {x_1}^2)x_2 + x_1 \tag{3.6}$$
$$y = [x_1, x_2]^T \tag{3.7}$$

where x_1 and x_2 are the states, μ is a constant parameter and y is the measurement signal. Both states are directly measured in both models.

The tests were conducted with the VDP-models and compared to the corresponding results in [Rambabu et al., 2008]. The processes were subject to additive Gaussian zero-mean uncorrelated process and measurement noise, all with variance 10^{-3}. The sampling interval was 0.1 s in both tests. The scenario with the normal VDP model is testing the performance of the UKF when there is a model difference between the process and the model used by the UKF, in this case a change in the parameter μ. The setup of the test scenario with the normal VDP is presented in Table 3.4.

The result of the state estimation of the normal VDP model from [Rambabu et al., 2008] and the corresponding result with the UKF-implementation from this thesis is presented in Figure 3.7. A comparison of the errors in amplitude and phase are presented in Figure 3.8. The performance of the implemented UKF is very similar to the results from [Rambabu et al., 2008]. Since the seed for the random generator used in [Rambabu et al., 2008] was not available, it is not possible to reproduce exactly the same results, which explains the small differences that can be seen.

Normal VDP test	
Initial state values	$[x_1, x_2] = [0.5, 0]$
Initial state estimates	$[\hat{x}_1, \hat{x}_2] = [5, -1]$
UKF model parameter	$\mu = 0.5$
Process model parameter	$\mu = 0.2$
Initial estimated state covariance	$P_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
Processs noise covariance	$P_v = \begin{pmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{pmatrix}$
Measurement noise covariance	$P_n = \begin{pmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{pmatrix}$

Table 3.4 Test setup for the first VDP test.

Reverse-time VDP test	
Initial state values	$[x_1, x_2] = [1.4, 0]$
Initial state estimates	$[\hat{x}_1, \hat{x}_2] = [0, 5]$
UKF model parameter	$\mu = 0.2$
Process model parameter	$\mu = 0.2$
Initial estimated state covariance	$P_0 = \begin{pmatrix} 10^{-2} & 0 \\ 0 & 10^{-2} \end{pmatrix}$
Processs noise covariance	$P_v = \begin{pmatrix} 10^{-3} & 0 \\ 0 & 10^{-3} \end{pmatrix}$
Measurement noise covariance	$P_n = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Table 3.5 Test setup for the second VDP test.

The second comparison was done with the reverse-time VDP, in a scenario where the initial state estimates and covariance matrices are chosen poorly. The setup of the test scenario with the reverse-time VDP is presented in Table 3.5.

The comparison of the reverse-time VDP scenario is presented in Figure 3.9. It can be seen that the UKF estimates in both results are quite similar. Also here, the difference in generated noise is the probable explanation to the differences that can be seen. Additionally, since the process is in an unstable limit cycle it is very sensitive to perturbations, which explains why there are differences in the true state values of the process in the results.

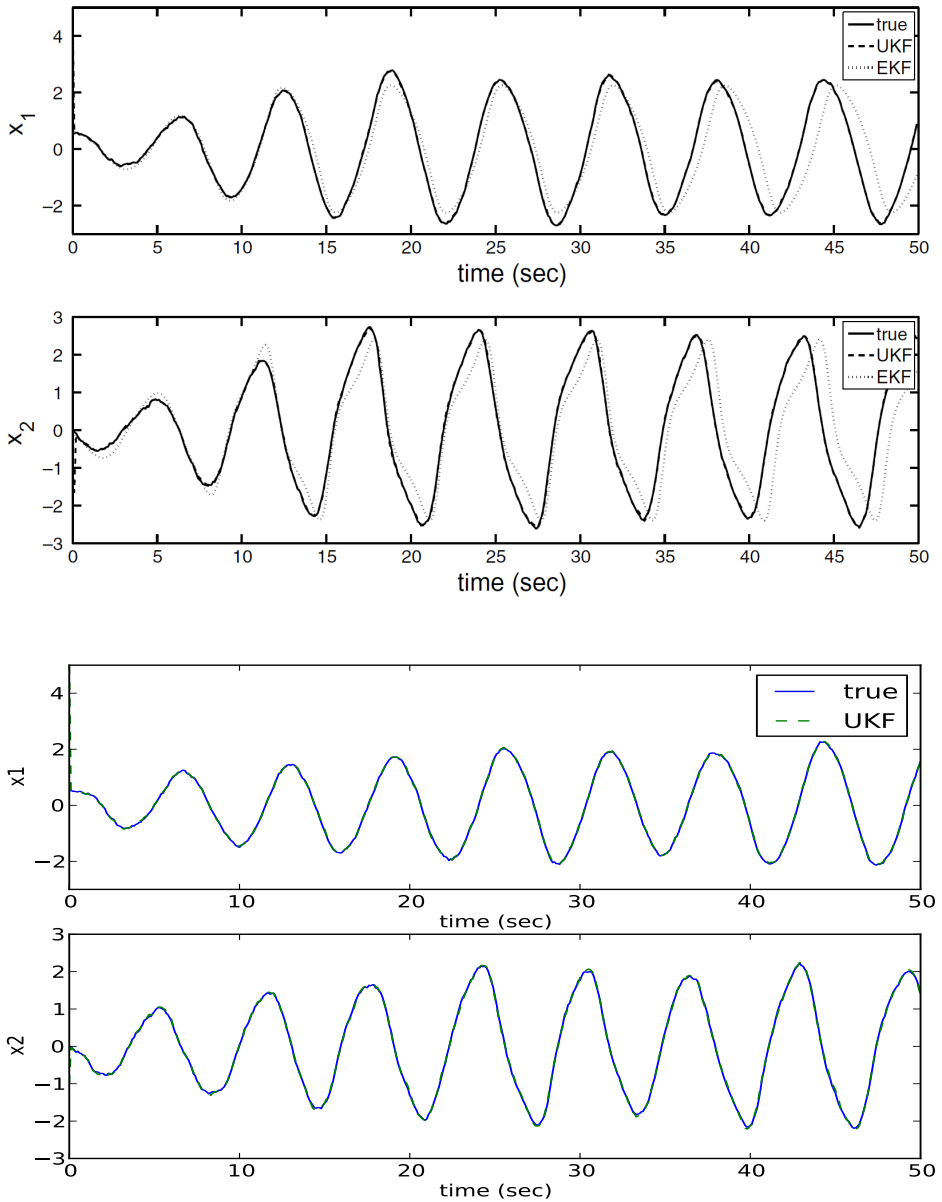

Figure 3.7 Comparison of the estimation results for the normal VDP model obtained in [Rambabu et al., 2008] (top two plots) and with the UKF-implementation in this thesis (bottom two plots).

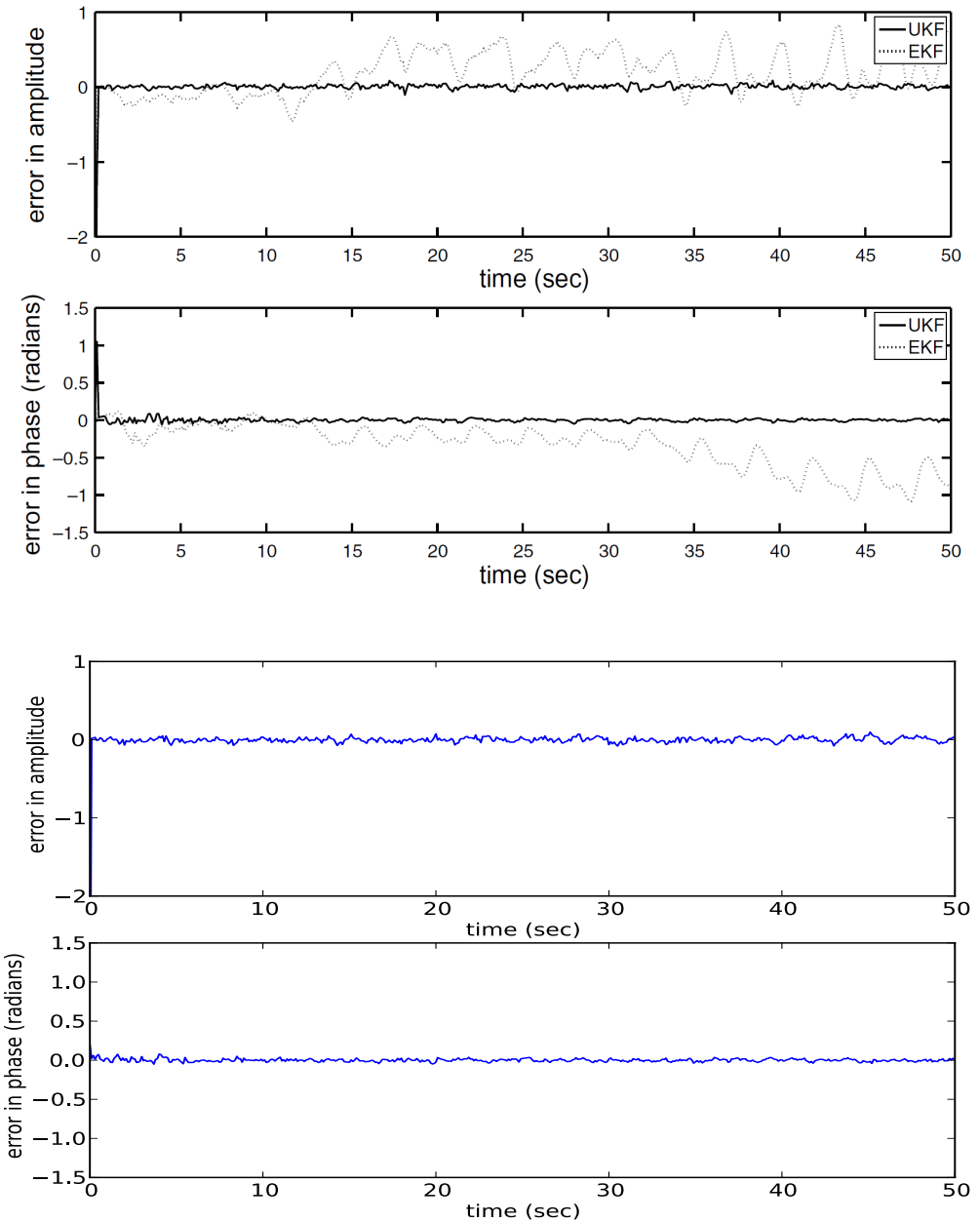

Figure 3.8 Comparison of the estimation errors in polar coordinates for the normal VDP model obtained in [Rambabu et al., 2008] (top two plots) and with the UKF-implementation in this thesis (bottom two plots).

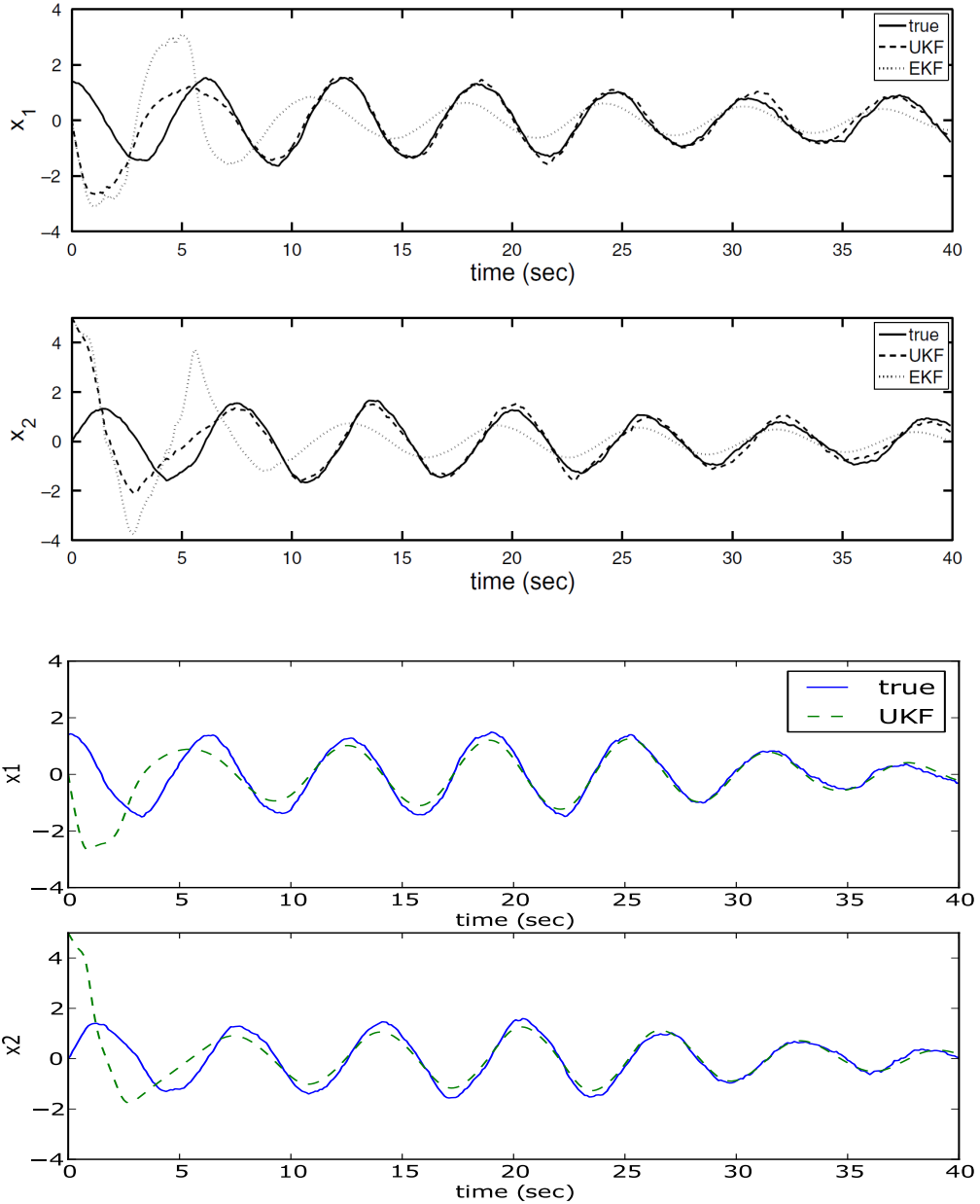

Figure 3.9 Comparison of the estimation results for the reverse-time VDP model obtained in [Rambabu et al., 2008] (top two plots) and with the UKF-implementation in this thesis (bottom two plots).

3.6 Disturbance Estimation

3.6.1 Disturbance States

Process disturbances represents the deviations between the optimization model and the detailed model. Therefore, the disturbance modelling was done in parallel to the model validation. In the model validation in Section 3.3.3 it was concluded that the valve dynamics were different in the models. It was also concluded that the overall energy in the optimization model was generally higher. A way to model these differences in the models is to assume load disturbances on the inputs to the model, i.e, the fourth noise representation in Section 2.6.

The load disturbances chosen for the inputs of the optimization model were of the form

$$u_d = u_{in} + d \tag{3.8}$$

$$\dot{d} = 0 \tag{3.9}$$

where u_d is the disturbed input signal, u_{in} is the undisturbed signal and d is a constant additive disturbance. The disturbances become additional states that the UKF estimates in order to compensate for model differences.

In addition to the inputs there was an uncertainty in the value of the heat transfer coefficient on the flue gas side of the evaporator, $alpha\_gas\_wall$. Therefore a constant disturbance state was added to the parameter in the model for some of the experiments.

3.6.2 Process Noise

The UKF-implementation assumes process noise on each state that is additive, uncorrelated and of zero-mean. The UKF needs an estimate of the variance of that process noise. This is regarded as one of the design parameters for the UKF, and will affect both the spread of the sigma points and also the gain of the measurement update for each state.

Because of the complexity of the model, suitable process noise variance values were determined systematically. In order to get values that would be in the proper magnitude, each state trajectory was compared between the detailed model and the optimization model during a reference soft start. The mean-square-error (MSE) of each compared state trajectory was used as the value for the standard deviation of the assumed process noise. The three input disturbance states and the integral error of the PI controller controlling the separator level had no equivalents in the detailed model, and so these values were chosen by trial and error. The MSE value used as estimate for the standard deviation of the process noise for each state is presented in Table 3.6

State	Estimated Process Noise Standard Deviation
RH1A Wall Temperature	37 K
RH1A Enthalpy	0.081 MJ/Kg
RH1A Pressure	0.094 bar
RH1B Wall Temperature	47 K
RH1B Enthalpy	0.11 MJ/Kg
RH1B Pressure	0.069 bar
RH2 Wall Temperature	33 K
RH2 Enthalpy	0.074 MJ/Kg
RH2 Pressure	0.048 bar
SHS Wall Temperature	27 K
SHS Enthalpy	0.13 MJ/Kg
SHS Pressure	0.20 bar
SH1 Wall Temperature	25 K
SH1 Enthalpy	0.12 MJ/Kg
SH1 Pressure	0.19 bar
SH2 Wall Temperature	36 K
SH2 Enthalpy	0.10 MJ/Kg
SH2 Pressure	0.16 bar
SH3 Wall Temperature	25 K
SH3 Enthalpy	0.068 MJ/Kg
SH3 Pressure	0.15 bar
SH4 Wall Temperature	13 K
SH4 Enthalpy	0.035 MJ/Kg
SH4 Pressure	0.14 bar
Header SH4 Wall Temperature (Center)	15 K
Header SH4 Enthalpy	0.035 MJ/Kg
Header SH4 Pressure	0.14 bar
Header RH2 Wall Temperature (Center)	32 K
Header RH2 Enthalpy	0.077 MJ/Kg
Header RH2 Pressure	0.034 bar
Evaporator Wall Temperature	22 K
Evaporator Enthalpy	0.085 MJ/Kg
Evaporator Pressure	2.0 bar
Level-Controller Integral Error	10
Separator Mixture Enthalpy	0.015 MJ/Kg
Separator Mixture Pressure	0.20 bar
Firing Power Disturbance	0.01
HP-Valve Opening Disturbance	0.01
IP-Valve Opening Disturbance	0.01

Table 3.6 The MSE of each state in a comparison between the optimization model and the detailed model during a reference soft start. The MSE is used as an estimate of the standard deviation of the process noise. The values for the level-controller integral error and the disturbance states could not be compared between the models, and were chosen through trial and error.

3.6.3 Measurement noise

There is no measurement noise in the signals retrieved from the detailed model. To make the setup more realistic and to make it more likely to work on the real plant, artificial measurement noise can be added. Since there is no information on which kind of noise there is in the real plant, zero mean, Gaussian, noise was assumed. To determine the standard deviation of the noise, real measurements from the plant during a start-up were studied. This approach was problematic since the data from the plant uses a dead-band that holds the output constant as long as it stays inside a predefined interval. This means that standard techniques using frequency analysis cannot be used since the shape of the signal destroys the frequency of the original measurement. The studied signals were however constant for large periods of time which means that all high frequency white noise must have a peak to peak amplitude smaller than the dead-band. The assumption is therefore that the standard deviation of the measurement noise is roughly one fourth of the dead-band, meaning that about 96 % of the noise values will be inside the dead-band. A typical measurement is provided in Figure 3.10 to show how the data looks. The dead bands are clearly visible, note the time scale.

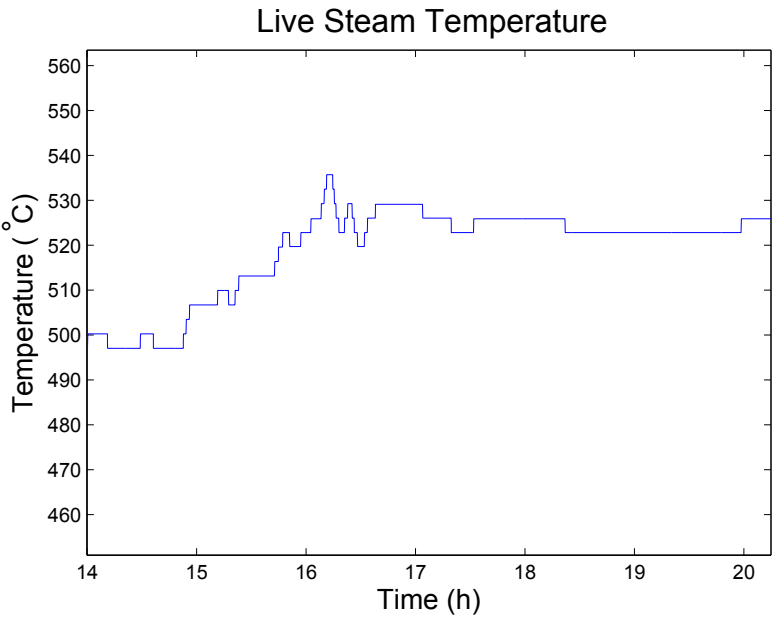

Figure 3.10 A measurement signal from the power plant.

The data that Vattenfall provided did not have measurements for all states that are assumed measureable in the detailed model. This meant that some noise levels have

Measurement Signal	Dead Band Amplitude	Part of mean
Live Steam Temperature	3.2 °C	0.81 %
Separator Steam Temperature	2.2 °C	0.75 %
RH1 Outflow Steam Temperature	2.1 °C	0.54 %
SH1 Outflow Steam Temperature	2.1 °C	0.63 %
SH2 Outflow Steam Temperature	2.1 °C	0.57 %
Live Steam Pressure	0.04 bar	0.04 %
Separator Steam Pressure	0.04 bar	0.03 %
HP Safety Valve Wall Temperature	3.2 °C	0.85 %

Table 3.7 Dead band peak to peak amplitudes for measurement signals from the real plant. Also includes values for how many percent of the mean value the dead band is for each signal.

to be assumed from similar measurements. The measurements that have been analysed and their dead band amplitudes are presented in Table 3.7. All the noise values in the table have values that are smaller than 1 % of the mean value of the measurement. Temperatures have values between 0.5 and 1 % while pressures are more precise with values below 0.05 %.

3.7 Observability analysis

3.7.1 Linearization Method

In order to choose suitable measurement signals from the power plant model, an observability analysis had to be done. The analysis was to be done according to the method for observability analysis described in Section 2.5. Therefore a script for linearizing the power plant model using JModelica.org and performing a linear observability analysis at each sample instant was developed.

The linearization was done with the functions `linearize_dae_with_simresult` and `linear_dae_to_ode`, which are part of the JModelica.org Model Interface (JMI) library [Modelon AB, 2014]. `linearize_dae_with_simresult` takes an object representing the model and linearizes it around a certain time instant using simulation results. Since the model is usually represented as a system of nonlinear differential algebraic equations (DAE), the function returns matrices corresponding to a linearizied DAE. Using the function `linear_dae_to_ode`, the linear DAE can be transformed to a linear system of ordinary differential equations (ODE), which is the desired form for performing a linear observability analysis.

Initial testing of the observability script on the VDP model (see (3.2)-(3.3)) looked promising, and the script flagged observability or non-observability in the correct cases. However, when testing the script on the optimization model the obtained observability matrix was very ill-conditioned, leading to large numerical errors and

incorrect rank computations. Since correct rank computations are essential to get a correct observability analysis, the method was not reliable.

3.7.2 Steady-State Estimation Test

Another test was developed to analyze observability, where the UKF was used to estimate all states of the model in steady-state, when using a given set of measurement signals. The reasoning was that when using the same model both as process and observer model, but with an initial steady-state offset, that if the UKF could manage to make all estimates converge using the given measurements, then this would indicate observability.

Measurement signals were chosen from a set of variables that were deemed realistic measurements by Vattenfall. The states to be estimated in the optimization model used by the UKF are pressures, enthalpies, mean wall temperatures of the heat exchangers and the integral error of the PI controller controlling the separator water level. Additionally, the center-most nodal temperature in the walls of the headers are also considered states in the model. Pressures were considered realistic measurements, and so these states could be measured directly. Since the thermal stress in the headers are monitored in the real power plant, the wall temperatures of the headers were also considered directly measurable. Steam temperatures were also realistic measurements, and these measurements were chosen since they would relate strongly to the steam enthalpy and also the wall temperature of the heat exchangers. The water level in the separator was chosen as a measurement to relate to the integral error in the PI controller. The final list of chosen measurements is presented in Table 3.8.

The measurement signals listed in Table 3.8 were used in an observability test, where both the UKF and process model were simulated to steady-state before starting the estimation. The initial offset between the states in the models was due to setting different values of the constant input disturbance states in the process. The firing power had an added offset of 0.01, and the valves had an added offset of 0.05 each. The estimation then proceeded with the same constant input values for firing power (0.15), HP-valve opening (0.1) and IP-valve opening (0.2). Measure-

Final Measurements:

Component	Measured Signals
Evaporator	Steam pressure and temperature
Separator	Steam/water-mix pressure and temperature, liquid water level
SHS, SH1-SH4	Steam pressure and temperature
RH1A, RH1B, RH2	Steam pressure and temperature
SH4 Header	Steam pressure and temperature, wall temperature at center
RH2 Header	Steam pressure and temperature, wall temperature at center

Table 3.8 The measurement signals chosen from the observability analysis.

ment noise was added to the measured signals, and the sampling interval was 100 s. The results for all state estimates are presented in Table A.1 in the appendix. The estimation errors are presented relative to the true state value according to:

$$\frac{\text{True value} - \text{Estimation}}{\text{True value}} \tag{3.10}$$

When comparing the initial and final percentage errors in Table A.1 it is apparent that all errors have declined drastically during the estimation. The results from the steady-state test indicated that the chosen measurement signals would provide observability for the optimization model, and so the measurement set in Table 3.8 was used in the experiments.

3.8 Demonstrator Implementation

The aim of the thesis is to run the NMPC and UKF together in a demonstrator. The flow of data in the demonstrator and in which way the models are used is described in Figure 3.11. One instance of the optimization model is used by the UKF to estimate the states with measurements taken from the target system. The estimated states are used to initialize the model the optimizer in the NMPC uses to solve the optimization problem, $\mathbf{O}$. As is described in Section 2.2.1 the first value in $\{\mathbf{u}_i\}^o$ is then sent into the target system. It is also sent to the UKF class for use in its prediction. The NMPC and UKF does not have to have the same sampling period. It is possible for the UKF to sample the target system more frequently than the NMPC solves $\mathbf{O}$. The NMPC sampling period does however need to be an even multiple of the UKF sampling time.

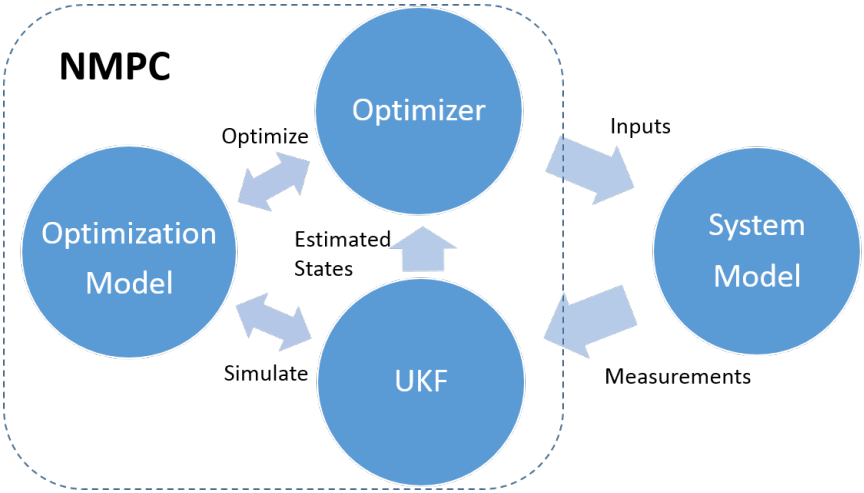

Figure 3.11 Flowchart of the data flow in the demonstrator.

3.8.1 Python Implementation

Firstly the UKF is instantiated according to Section 2.4.3 and the NMPC is instantiated in the same way as in [Axelsson, 2015]. Both models are initialized by simulating them to a steady state and the inital trajectory of the NMPC is set to the result of an offline optimization. The sampling time of the NMPC, `h_MPC`, and of the UKF, `h_UKF`, are defined separately.

The main control loop of the demonstrator is presented below. Since the NMPC and UKF uses two different version of the optimization model with the UKF having more states than the NMPC they have different state vectors, `x_MPC` and `x_UKF`. Before the loop, `x_MPC` is initialized to have the same values as the optimization model. The states in `x_MPC` have names that start with `_start_`, which comes from using the compilation option `state_initial_equations` when the optimization problem object is created. This means that when getting the actual state names the first seven letters have to be removed, which in Python code translates to: `state = _start_state[7:]`.

```
#Begin the main loop of the demonstrator
for i in range(number_samp_tot):
    # Update the state and compute the optimal input
    # for the next sample period
    MPC_object.update_state(x_MPC)
    u_k = MPC_object.sample()

    for j in range(h_MPC/h_UKF):
        ukf.predict(u_k, known_values)

        # Simulate the model with the optimized
        # control inputs
        tStart = i*h_MPC + j*h_UKF

        sim_res = process.simulate(start_time =
            tStart, final_time = tStart + h_UKF,
            options = sim_opt, input = u_k)

        # Extract measurements
        for meas in measurements:
            y[meas] = sim_res[meas][-1] +
                Random.gauss(0, N.sqrt(P_n[meas]))

        # Update with measurements and retrieve
        # estimates
        x_UKF = ukf.update(y)

        for name in known_values:
            known_values[name] = sim_res[name][-1]

    # After UKF loop, update NMPC state start values
    for state in x_MPC:
        if state[7:] in known_values:
            x_MPC[state] = known_values[state[7:]]
        else:
            x_MPC[state] = x_UKF[state[7:]]
```

4

Experiment Setups

The experiments were chosen to test the performance of the NMPC and UKF both individually and also together as a demonstrator. Since the detailed model was not available for control purposes, the demonstrator was instead tested on the optimization model. Four main categories of experiments were defined:

- UKF estimation of the optimization model.
- UKF estimation of the detailed model.
- NMPC control of the optimization model.
- Demonstrator run on the optimization model.

The set of additive process noise assumed by the UKF in experiments with the detailed model was the set that was presented in Table 3.6. The corresponding set used for the optimization model experiments is presented in Table 4.1. The values in the experiments with the optimization model were chosen generally lower because of the greater similarity between the process and the model used by the UKF in those scenarios.

The UKF assumed the same variance of the measurement noise as the actual noise that was applied to the measurement signals. From the measurement noise analysis in Section 3.6.3, a set of zero-mean Gaussian noise distributions was defined for each type of measurement signal. The measurement noise characteristics are presented in Table 4.2.

The set of parameters for the UKF used in the experiments is presented in Table 4.3. The values for β and κ were chosen using the default values defined in Section 2.4.2 and α was chosen to get a suitable spread of the sigma points. The set of parameters used by the NMPC is presented in Table 4.4. The reasons for choosing these values for the parameters are described in Section 3.4.2.

State Type	Process Noise Standard Deviation
Wall Temperatures	1K
SH/Evaporator/Separator Enthalpies	0.001 MJ/Kg
SH/Evaporator/Separator Pressures	0.01 bar
RH Enthalpies	0.01 MJ/Kg
RH Pressures	1.0 bar
Disturbance States	0. 001
Level-Controller Integral Error	10

Table 4.1 Standard deviations of the assumed process noise used in estimation experiments with the optimization model.

Measurement Type	Measurement Noise Standard Deviation
Temperatures	0.75 K
Pressures	0.01 bar
Separator Water Level	5 cm

Table 4.2 Measurement noise covariances used in all estimations.

Parameter	Value
α	0.1
β	2
κ	0

Table 4.3 Standard set of parameters used by the UKF during experiments.

The set-points for the LS and RS pressure and temperature were chosen from a reference soft start done with the detailed model. The set-points represent the end of the first phase of the start-up. The initial values were chosen as the steady-state values of the optimization model with the firing power 0.15, the HP-valve opening 0.1 and the IP-valve opening 0.2, which are the same initial inputs as in the detailed model. The set-points and initial values are presented in Table 4.5 and were used in all experiments with the NMPC.

Parameter	Value
Sampling Interval	500 s
Number of Collocation Points per Sample	3
Prediction Horizon	5000 s
Blocking Factors	[1,1,1,2,5]
Constraint Violation Cost	10^4
Noise Seed	1

Table 4.4 Standard set of parameters used by the NMPC during experiments.

Variable	Initial Value	Set-point
LS Pressure	10.6 bar	93 bar
LS Temperature	512 K	633 K
RS Pressure	2.1 bar	15 bar
RS Temperature	517 K	693 K

Table 4.5 Set-points used by the NMPC during experiments.

4.1 UKF Estimation of Optimization Model

In these experiments the performance of the UKF estimation is tested when the optimization model is used both as UKF-model and as target system. Both models have dynamic steam states. To test how well the UKF can handle un-modelled differences, an offset is set in the heat transfer coefficient (the parameter *alpha_gas_wall* in Section 3.3.3) on the flue gas side in the evaporator of the model used by the UKF. The value used by the UKF-model is set to 400 W/m^{2}K, while the process has the value 600 W/m^{2}K.

The three experiments are:

A.1 Estimation when the process has a ramp input in firing power and valve openings. Firing power is increased at the rate $7 \cdot 10^{-5}$ /s, HP-T bypass valve closes with the rate $-1 \cdot 10^{-5}$ /s and the IP-T bypass valve with $-2 \cdot 10^{-5}$ /s.

A.2 Estimation when the process has a ramp input as in A.1 and the UKF-model includes a disturbance state for the parameter difference.

A.3 The same test as in A.1, but when the process has an offline-optimized input trajectory.

In all three experiments the UKF has a disturbance model with constant disturbance states added to each input. A constant added offset of 0.01 to firing power and 0.05 to both valve openings are present in the process. In the second experiment, the

UKF also has an additional constant disturbance state added to the parameter that differs from the process.

Measurement noise according to Table 4.2 is added to each measurement signal during the experiments. The sampling interval of the UKF is set to 100 s, and the total simulation time for each experiment is 5000 s.

4.2 UKF Estimation of Detailed Model

In these experiments the performance of the UKF estimation was tested on a pre-simulated reference soft start with the detailed model. The task proved difficult for the UKF, and so a model where the pressure at the IP-T bypass valve was set identical to the detailed model was used to simplify the estimation. No disturbance estimation for the IP-T bypass valve was therefore performed in these experiments. The disturbance states at the other inputs were still present. In total, three experiments were performed. The first two were performed using the optimization model with dynamic steam in the UKF:

B.1 Estimation with no measurement noise, with 10 s sample interval.

B.2 Estimation with no measurement noise, with 100 s sample interval.

In the third experiment the model used by the UKF was changed to have static steam dynamics and an ideal separator as described in Section 3.3.4. Minimum and maximum values were also set on the variables in the model to keep them within the valid range of the *water_poly* functions:

B.3 Estimation with measurement noise, with 10 s sample interval. UKF uses optimization model with static steam dynamics, ideal separator and limits on variables.

Measurement noise according to Table 4.2 was added to each measurement signal during the third experiment. The total time of the first phase of the reference soft start in the detailed model is 7700s.

4.3 NMPC Control of Optimization Model

In this experiment the NMPC was used to control the optimization model. The model used by the NMPC and the target system are the same, without any parameter changes, and uses static relations for the steam. All the states were considered measurable and Gaussian zero-mean noise was added to each measurement signal.

Name of set	Input	FP	LST	RST	LSP	RSP
C1	10^6	1	0.1	0.001	0.01	0.1
C2	10^6	100	0.1	0.001	1	0.1

Table 4.6 Two sets of cost function weights for inputs and states, used in the NMPC experiments. The parameter names are defined in the same way as in Table 3.3.

The total simulation time was 5000 s. Two different cost functions, C1 and C2, were tested. The weights in the cost functions are defined in Table 4.6. C1 correspond to the fastest, most fuel consuming, of the setups in Table 3.3, while C2 corresponds to the one with least fuel consumption. An offline optimization using the C1 cost function was also performed for comparison to the NMPC results.

4.4 Demonstrator Run

In this experiment the NMPC and UKF were run together. The model used by the UKF was the optimization model with dynamic steam, while the NMPC and target system used the optimization model with static steam relations. The target system had constant load disturbances added on firing power (0.001) and on the bypass valve openings (0.01). Measurement noise was added according to Table 4.2. The UKF used a sampling interval of 100 s, while the NMPC used 500 s. The total simulation time was 5000 s. The NMPC used the cost function weights defined in Table 4.7.

Input	FP	LST	RST	LSP	RSP
10^6	10	0.01	0.001	0.01	0.1

Table 4.7 Cost function weights for inputs and states, used in the demonstrator experiment. The parameter names are defined in the same way as in Table 3.3.

5

Results

In this chapter the results from the experiments with the UKF, NMPC and the demonstrator are presented. The experiment setups are defined in Chapter 4.

5.1 UKF Estimation of Optimization Model

The mean absolute errors for the state estimates in experiment A.1 (ramp input and no model of the evaporator disturbance) was in the range 0.1-9.0 %. The corresponding range in experiment A.2 (ramp input and including a model of the evaporator disturbance) was 0.02-1.0 %. The estimation results for some of the enthalpies, wall temperatures and disturbance states from experiments A.1 and A.2 are presented in Figures 5.1-5.3, where the actual and estimated values are compared. A comparison is also shown for the estimation of the level controller's integral error in Figure 5.4 which diverged in experiment A.1. The estimation in A.1 ran for 4700 s until the UKF produced an estimate that proved infeasible when trying to simulate. Each new estimation took approximately 40 s to 30 min in real-time to produce. The estimation in A.2 ran for the full 5000 s, where each new estimation took approximately 40 s up to 4 min in real-time to produce.

The mean absolute errors for the state estimates in experiment A.3 (optimized input) was 0.1-54.3 %. The estimation results for some of the enthalpies, wall temperatures and disturbance states from experiment A.3 are presented in Figures 5.5-5.7. The diverging estimates of the level controller's integral error and the separator mixture enthalpy is presented in Figure 5.8. The estimation ran for 4800 s until the UKF produced an estimate that proved infeasible when trying to simulate. Each new estimation took approximately 40 s to 30 min in real-time to produce.

The estimation errors for all states from experiments A.1, A.2 and A.3 are presented in the appendix in Tables A.2, A.3 and A.4 respectively.

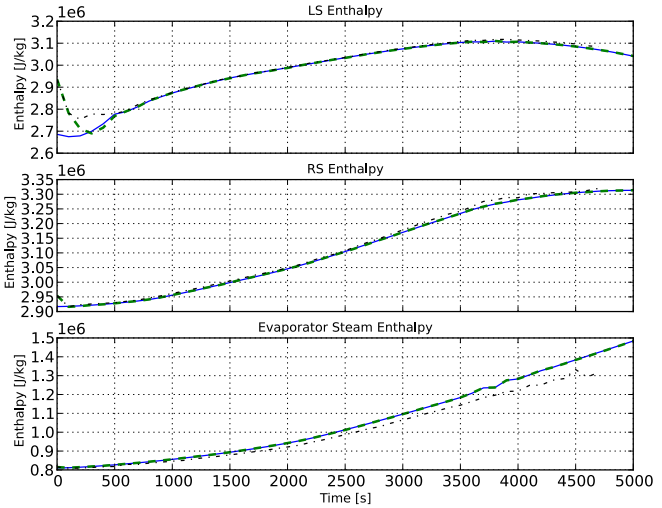

Figure 5.1 Enthalpy estimation results for LS (SH4 header), RS (RH2 header) and evaporator from experiment A.1 and A.2 (ramp inputs). The solid line is the true state value, while the dash-dotted and dashed lines corresponds to experiment A.1 and A.2 respectively.

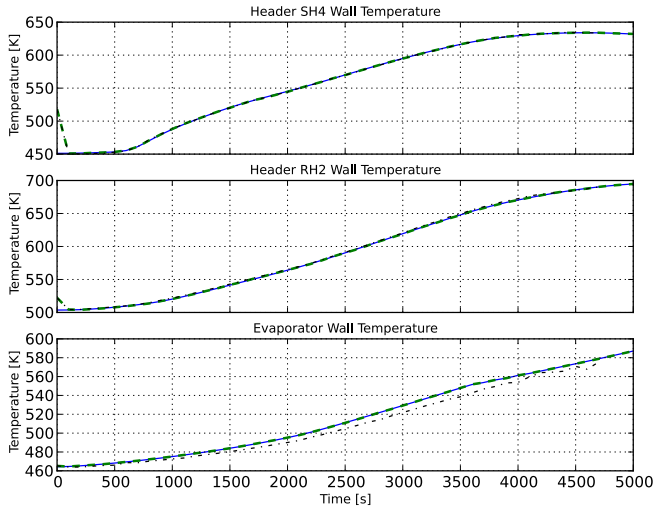

Figure 5.2 Wall Temperature estimation results for the SH4 header, the RH2 header and the evaporator from experiment A.1 and A.2 (ramp inputs). The solid line is the true state value, while the dash-dotted and dashed lines corresponds to experiment A.1 and A.2 respectively.

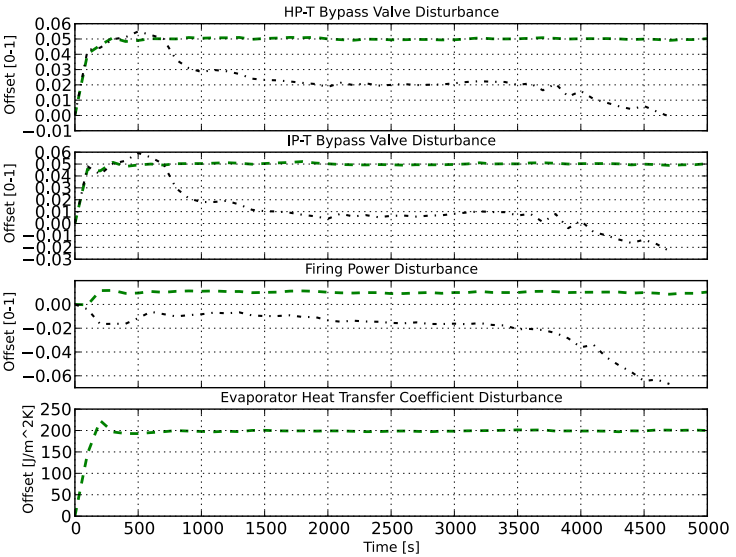

Figure 5.3 Disturbance state estimation results from experiment A.1 and A.2 (ramp inputs). The dash-dotted and dashed lines corresponds to estimates in experiment A.1 and A.2 respectively. Note that the heat transfer coefficient disturbance is only part of the disturbance model in experiment A.2.

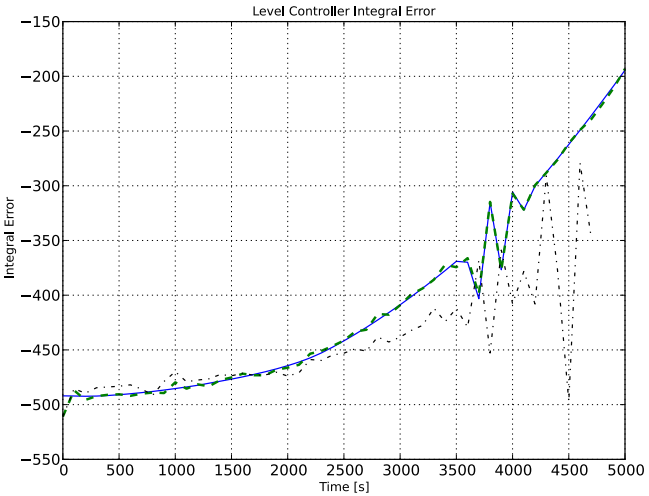

Figure 5.4 Level controller integral error estimation results from experiment A.1 and A.2 (ramp inputs). The solid line is the true state value, while the dash-dotted and dashed lines corresponds to experiment A.1 and A.2 respectively.

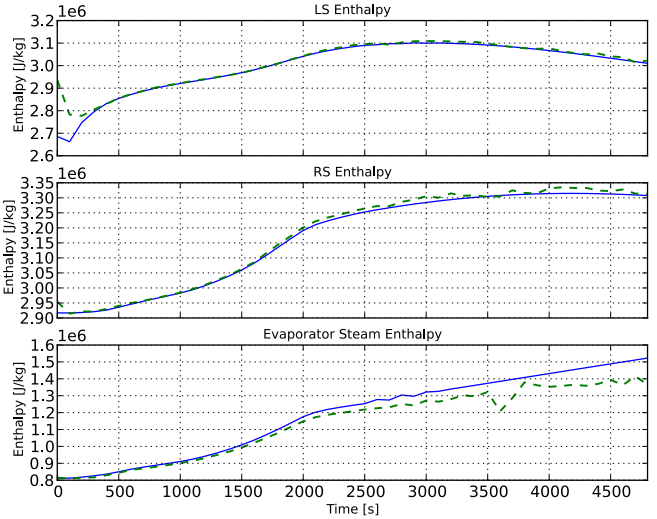

Figure 5.5 Enthalpy estimation results for LS (SH4 header), RS (RH2 header) and evaporator from experiment A.3 (optimized inputs). The solid line is the true state value, while the dashed line is the estimate.

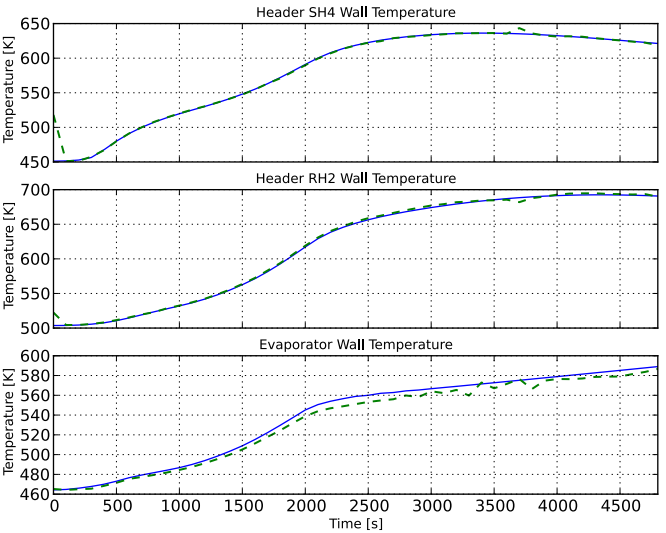

Figure 5.6 Wall Temperature estimation results for the SH4 header, the RH2 header and the evaporator from experiment A.3 (optimized inputs). The solid line is the true state value, while the dashed line is the estimate.

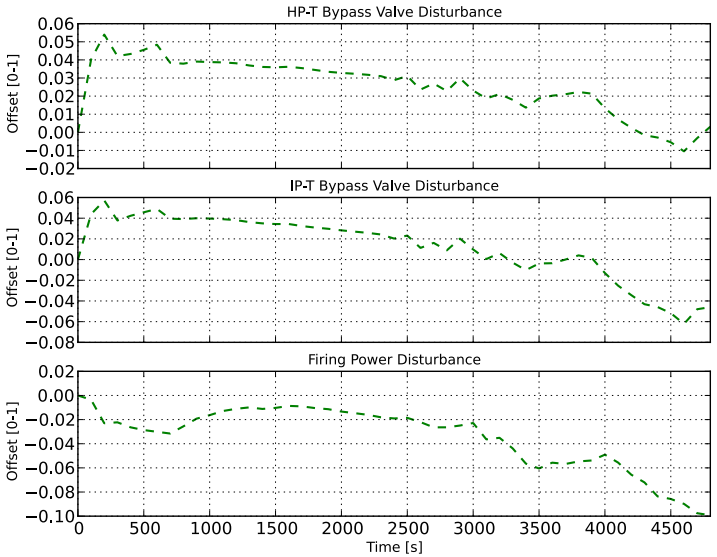

Figure 5.7 Disturbance state estimation results from experiment A.3 (optimized inputs).

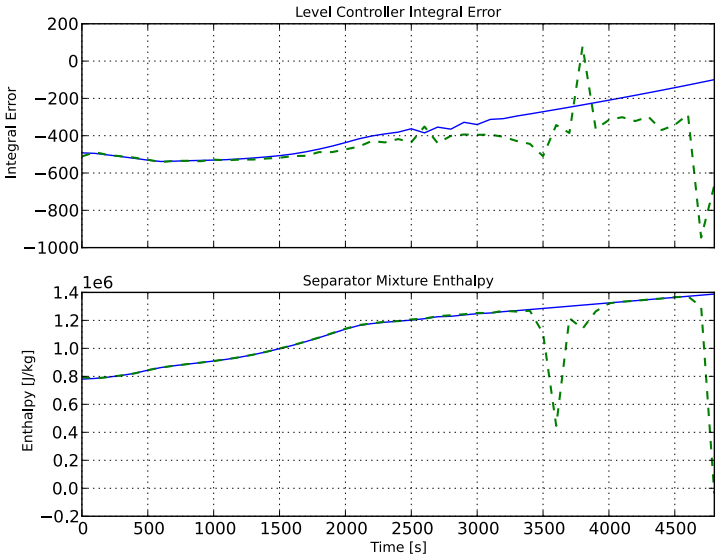

Figure 5.8 Level controller integral error and separator mixture enthalpy estimation results from experiment A.3. The solid line is the true state value, while the dashed line is the estimate.

5.2 UKF Estimation of Detailed Model

The mean absolute errors of the state estimates in experiment B.1 (no measurement noise and 10 s sampling interval) were in the range 0.09-52.3 %. The estimation results for some of the pressures, enthalpies and wall temperatures from experiment B.1 are presented in Figures 5.9-5.11. The diverging estimates of the steam enthalpies in SHS, SH1 and SH2 are presented in Figure 5.12. The estimation ran for 280 s until the UKF produced an estimate that proved infeasible when trying to simulate. Each new estimation took approximately 40 s up to 30 min in real-time to produce.

The estimation in experiment B.2 (no measurement noise and 100 s sampling interval) resulted in divergence for the majority of the states after two iterations. The estimation ran for 700 s, i.e, 7 iterations, until the UKF produced an estimate that proved infeasible to simulate.

The mean absolute errors of the state estimates in B.3 (fast dynamics removed, measurement noise present and 10 s sampling interval) were in the range 0.09-0.7 %. The estimation results for some of the wall temperatures and disturbance states from experiment B.3 are presented in Figures 5.13, 5.14 and 5.15. Since the steam dynamics were changed to static relations, steam enthalpy and pressure were no longer states in the model and therefore not estimated. The estimation ran for the full 7700 s. Each new estimation took approximately 5 s in real-time to produce.

The full results from experiment B.1 and B.3 are presented in the appendix in Tables A.5 and A.6 respectively.

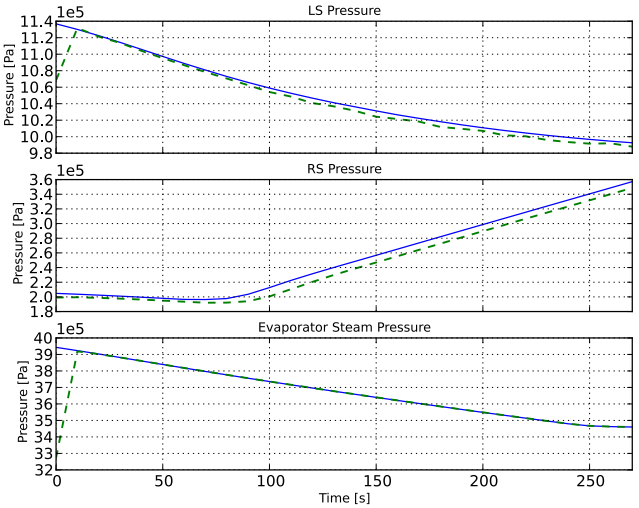

Figure 5.9 Pressure estimation results for LS (SH4 header), RS (RH2 header) and evaporator from experiment B.1 (10 s sample interval). The solid line is the true state value, while the dashed line is the estimate.

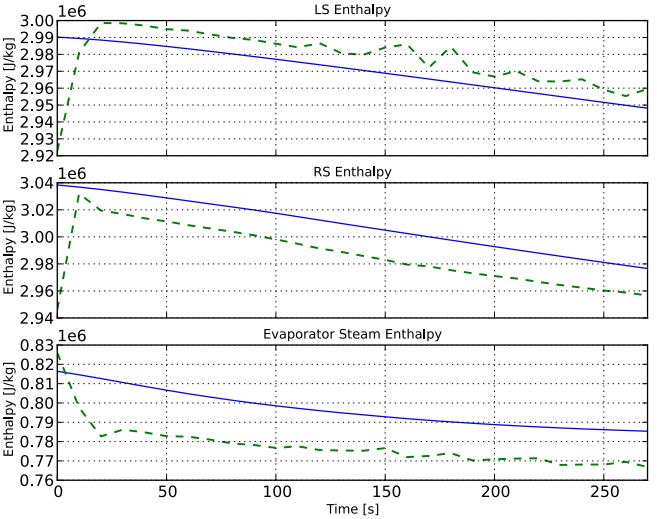

Figure 5.10 Enthalpy estimation results for LS (SH4 header), RS (RH2 header) and evaporator from experiment B.1 (10 s sample interval). The solid line is the true state value, while the dashed line is the estimate.

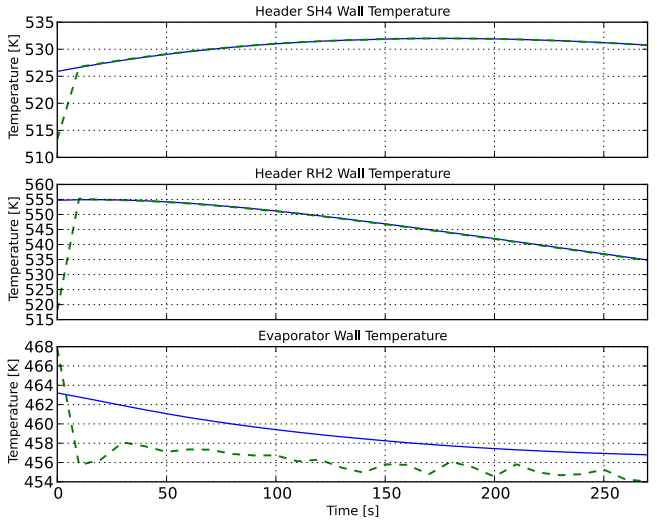

Figure 5.11 Wall Temperature estimation results for the SH4 header, the RH2 header and the evaporator from experiment B.1 (10 s sample interval). The solid line is the true state value, while the dashed line is the estimate.

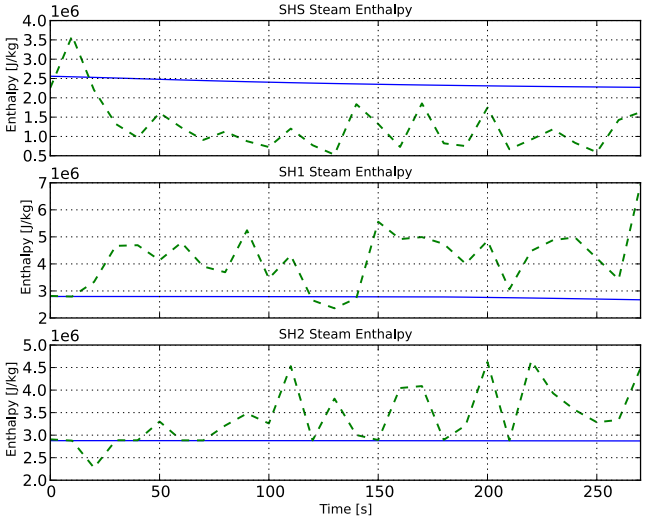

Figure 5.12 Diverging enthalpy estimations from experiment B.1 (10 s sample interval). The solid line is the true state value, while the dashed line is the estimate.

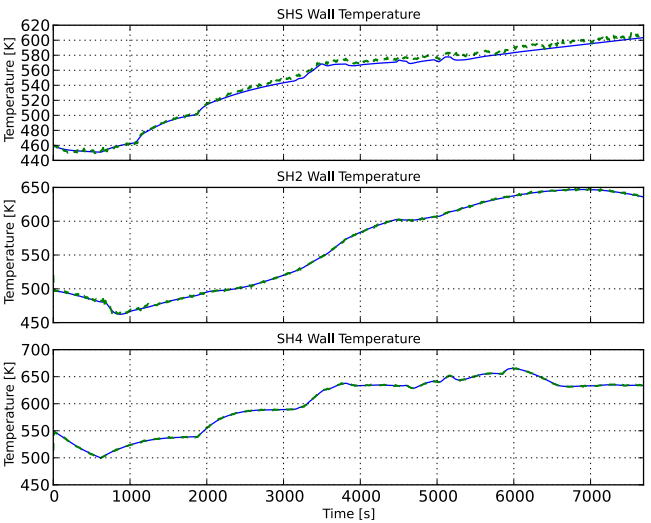

Figure 5.13 Wall temperature estimation results for SHS, SH2 and SH4 from experiment B.3 (modified optimization model and 10 s sampling interval). The solid line is the true state value, while the dashed line is the estimate.

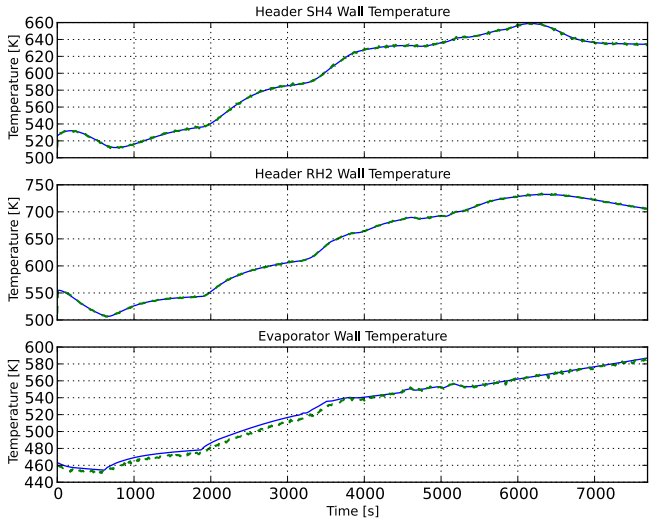

Figure 5.14 Wall temperature estimation results for the SH4 header, the RH2 header and the evaporator from experiment B.3 (modified optimization model and 10 s sampling interval). The solid line is the true state value, while the dashed line is the estimate.

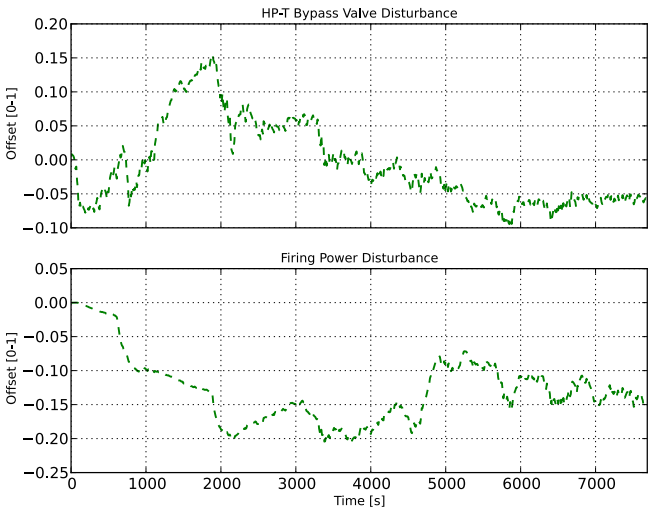

Figure 5.15 Estimated disturbance states from experiment B.3 (modified optimization model and 10 s sampling interval).

5.3 NMPC Control of Optimization Model

The optimized trajectories for each NMPC iteration with C1 from Table 4.6 as cost function are presented in Figure 5.16. The trajectories using C2 from Table 4.6 are presented in Figure 5.17. Actual simulation results of the target model using the NMPC inputs together with simulation results using an offline optimization as comparison are presented in Figures 5.18 - 5.21.

The integral of the firing power, which relates to the total amount of fuel used, was 1176 after 3500 s and 1890 after the full simulation at 5000 s when using C1. The corresponding values for C2 were 1178 and 1885. Each NMPC iteration took roughly 1-30 seconds depending on how many iterations the solver needed to perform.

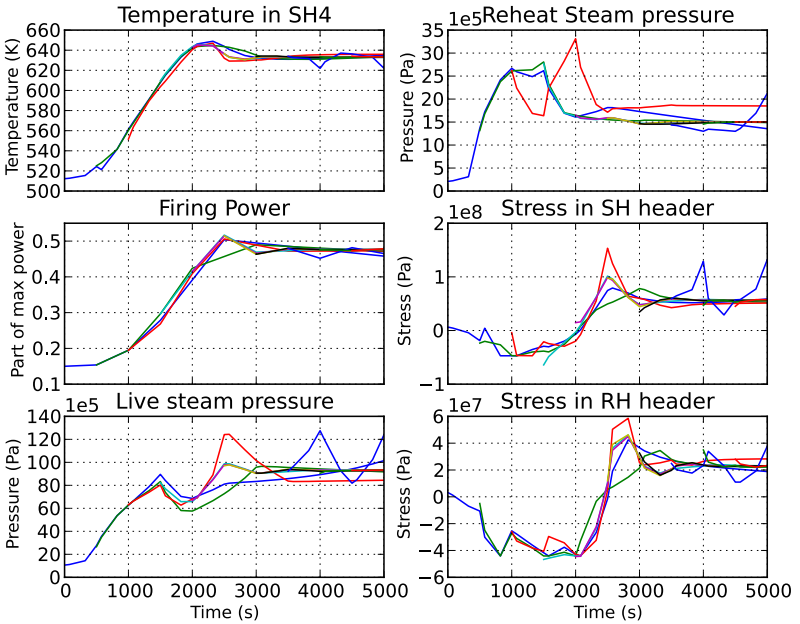

Figure 5.16 The solution to **O** (see Section 2.2.1) at each iteration of the NMPC using the C1 cost function weights. Each curve corresponds to the predicted trajectory at each sampling instant by the NMPC. The optimizer fails to find an optimal solution at iteration 3 and 8.

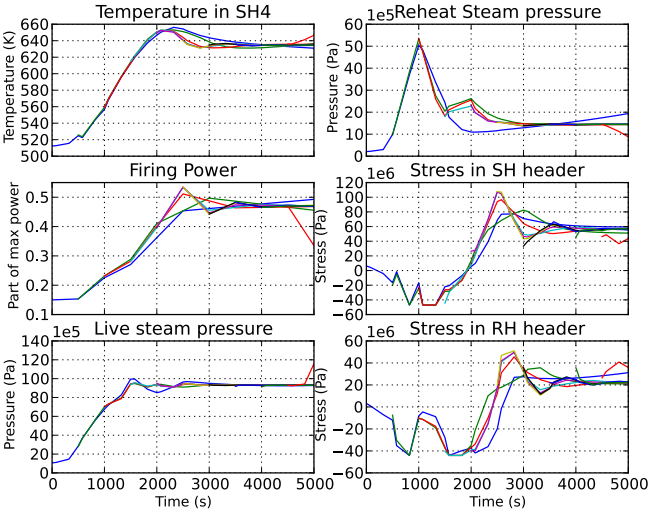

Figure 5.17 The solution to **O** (see Section 2.2.1) at each iteration of the NMPC using the C2 cost function weights. Each curve corresponds to the predicted trajectory at each sampling instant by the NMPC. The optimizer fails to find an optimal solution at iteration 10.

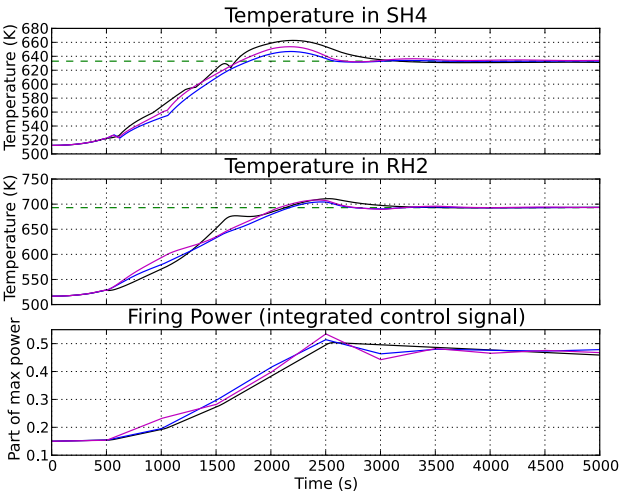

Figure 5.18 The LS and RS temperature of the plant together with the firing power. The blue line is the simulation result of the target system controlled by the NMPC using the C1 cost function, the magenta line is the value for the C2 cost function and the black line is the simulation result using the offline optimization with the C1 cost function. The dashed, green, line is the temperature set-point.

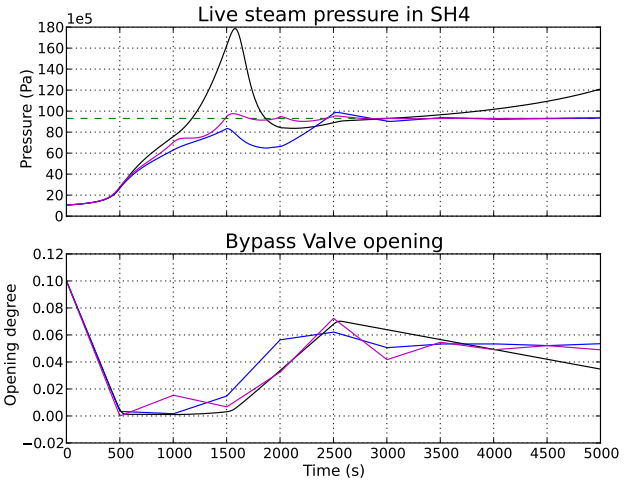

Figure 5.19 The LS pressure of the plant together with the valve opening of the HP-T bypass valve. The blue line is the simulation result of the target system controlled by the NMPC using the C1 cost function, the magenta line is the value for the C2 cost function and the black line is the simulation result using the offline optimization with the C1 cost function. The dashed, green, line is the pressure set-point.

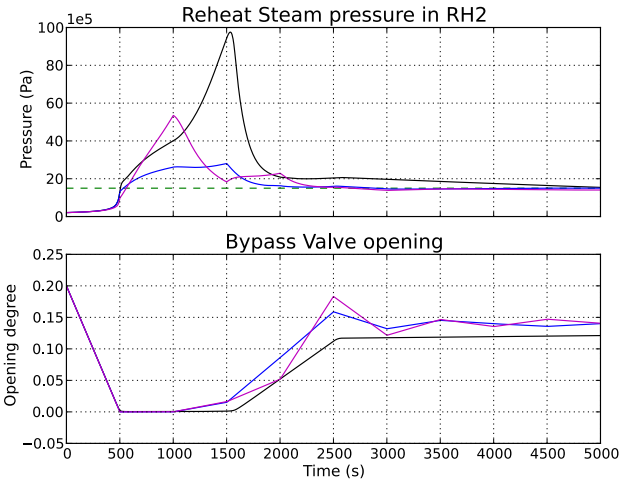

Figure 5.20 The RS pressure of the plant together with the valve opening of the IP-T bypass valve. The blue line is the simulation result of the target system controlled by the NMPC using the C1 cost function, the magenta line is the value for the C2 cost function and the black line is the simulation result using the offline optimization with the C1 cost function. The dashed, green, line is the pressure set-point.

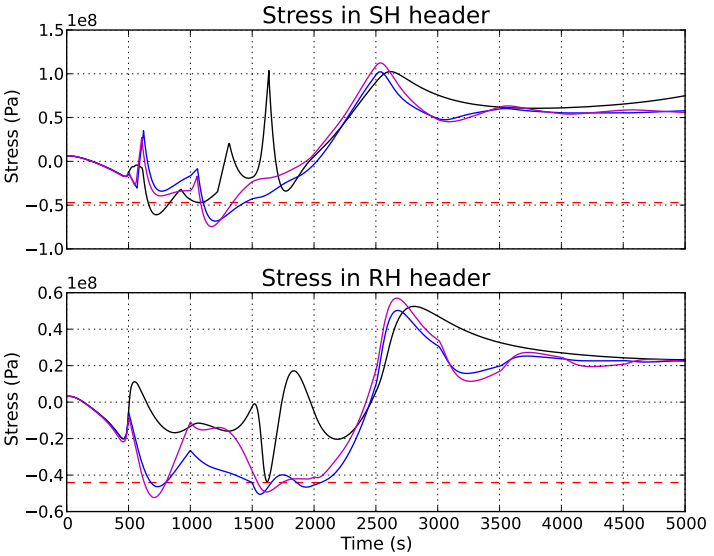

Figure 5.21 The stresses in the SH4 and RH2 headers of the plant. The blue line is the simulation result of the target system controlled by the NMPC using the C1 cost function, the magenta line is the value for the C2 cost function and the black line is the simulation result using the offline optimization with the C1 cost function. The dashed red line is the stress constraint used by the NMPC.

5.4 Demonstrator Run

The demonstrator was able to run successfully for the full 5000 s. State estimates for pressures, enthalpies and wall temperatures had a mean absolute error in the range 0.009-0.09 %, while the level controller integral error had 0.5 % and the disturbance states 4.1-55.6%. The optimizer in the NMPC only managed to find a solution to **O** for two iterations 1 and 7, meaning that most of the time the target system was controlled using an offline simulation.

The simulation results of the target system are presented in Figures 5.22 - 5.25. The full results from the UKF estimation is presented in table A.7 in the appendix.

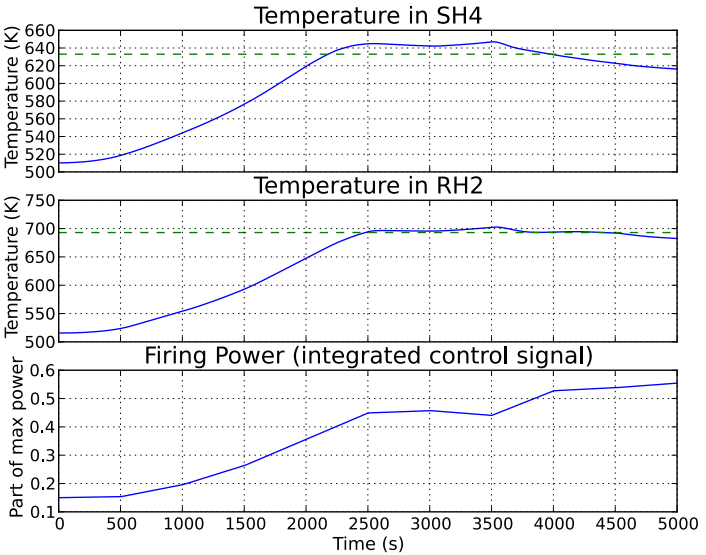

Figure 5.22 The LS and RS temperature of the plant together with the firing power for the demonstrator run. The dashed, green, line is the temperature set-point.

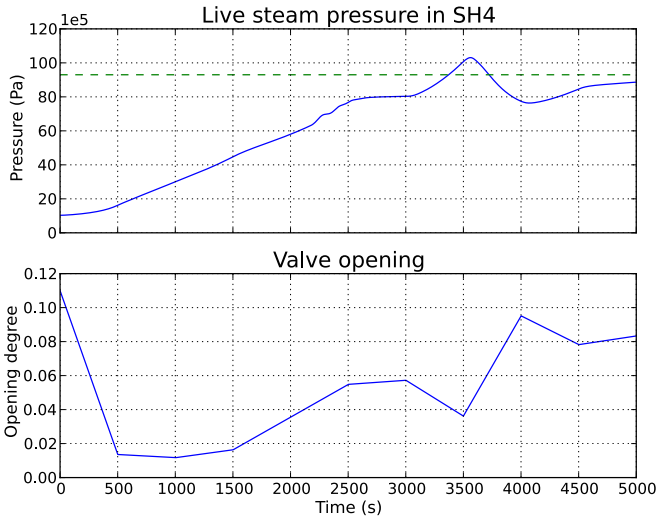

Figure 5.23 The LS pressure of the plant together with the valve opening of the HP-T bypass valve for the demonstrator run. The dashed, green, line is the pressure set-point.

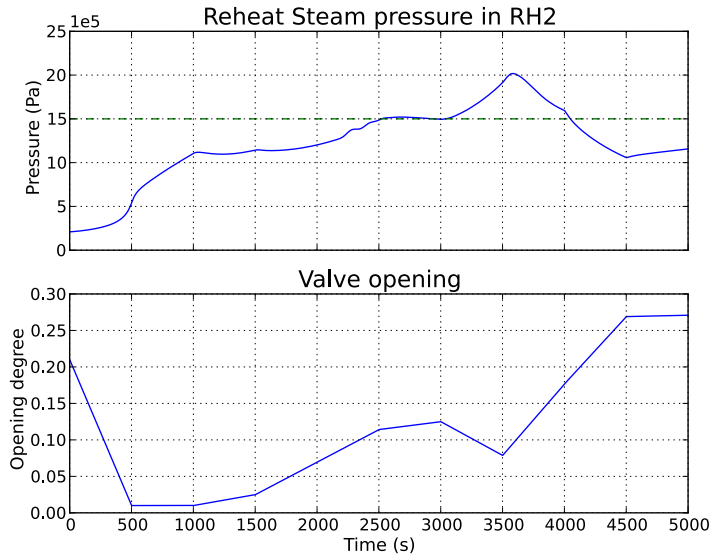

Figure 5.24 The RS pressure of the plant together with the valve opening of the IP-T bypass valve for the demonstrator run. The dashed, green, line is the pressure set-point.

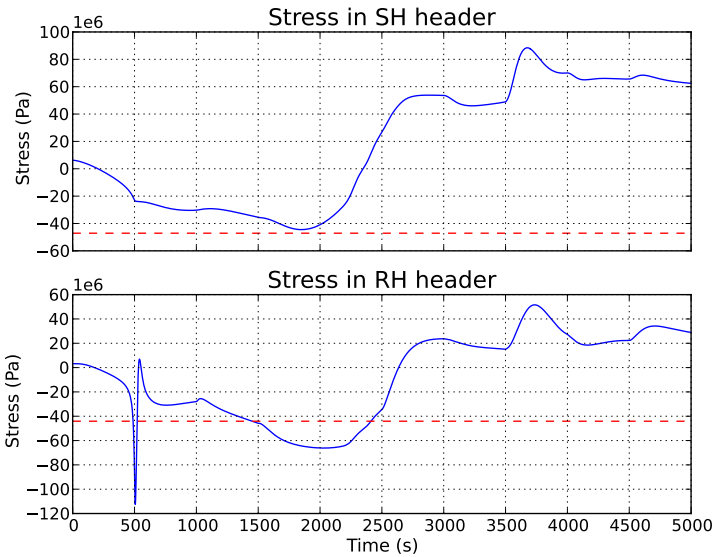

Figure 5.25 The stresses in the SH4 and RH2 headers of the plant for the demonstrator run. The dashed, red, line is the stress constraint used by the NMPC.

6

Discussion

6.1 State Estimation using UKF

The UKF robustness to parameter choices in the model and input disturbances is tested in experiments A.1-A.3 (defined in Section 4.1). The results in A.1 and A.3 show that the estimations of pressures and header wall temperatures are robust to the model changes, with mean absolute errors in the range 0.1-0.4 %. The robustness is expected since these states are directly measurable. However, the parameter change makes the estimation of enthalpies and wall temperatures difficult when there is no disturbance model for the change (see Figures 5.1, 5.2, 5.5 and 5.6), with mean absolute errors 0.2-4.5 %. The parameter change has a large impact on the predicted steam enthalpy which in turn affects the water level of the separator and the control action to keep it at 15.5 m. The fast dynamics of the level controller makes the predicted integral error deviate much from the true value before the measurement update is done. This eventually makes the estimate diverge, see Figures 5.4 and 5.8. This issue is solved in experiment A.2 where the UKF successfully estimates the offset in the heat transfer coefficient and the model and process essentially become identical, see Figure 5.3. The experiments A.1 - A.3 indicates that the dynamics of the separator level controller is too fast for a sampling interval of 100 s, and that good disturbance modelling is essential for a robust UKF performance.

The main objective of the UKF is to produce estimates of the states in the detailed model. This is done in experiments B.1 - B.3, where the choice of sampling interval is analyzed in B.1 and B.2. In B.2 the majority of the estimates diverges which indicates that 100 s sample interval is too long. However, the results in B.1 indicate that also 10 s sample interval might be too long when using the current optimization model. As in A.1 - A.3, the UKF in B.1 provides the most accurate estimates in percentage for the directly measurable pressures and header wall temperatures (see Figures 5.9 and 5.11), with mean absolute errors in the range 0.09-3.0 %. The enthalpy estimations are difficult for the UKF and are diverging in SHS, SH1 and SH2 (see Figure 5.12). The results show that even with 10 s sampling interval the measurement updates are not enough to keep the estimates close to the true values.

The robustness and performance of the UKF is improved significantly with the modified optimization model (fast dynamics removed and using an ideal separator) in experiment B.3 (see Figures 5.13 and 5.14), where the UKF manages to get all mean absolute estimation errors within 0.7 % for the entire first phase of the reference soft start. This confirms that removing the separator level controller component from the optimization model is beneficial, and that setting limits on the variables helps the robustness. The experiment B.3 also shows that the measurement variables were chosen adequately, and that the noise assumptions made for the UKF implementation are not too simple. However, further testing should be done with the UKF where the IP-T bypass valve is also included in the model.

The UKF performs well in the demonstrator run with the NMPC, mainly because the only not modelled difference between the UKF model and the process is that the steam dynamics are static in the process. All state estimations except for the level controller integral error and disturbance states have a mean absolute error in the range 0.009-0.09 %, which is better results than in any other experiment. The NMPC still fails to find an optimal solution in most iterations in the demonstrator run, but that seems to be a robustness issue with the optimization model rather than poor estimates from the UKF.

Overall, the robustness and performance of the UKF implementation with interface to JModelica.org depend heavily on the model used. If the model is sensitive to the choice of initial state values, then the UKF risks failing several sigma point simulations which leads to poor performance or total break down. Another factor that the UKF robustness depends upon is the choice of the parameter α. With an unstable model, a large spread of sigma points (i.e α close to 1) will risk having a number of sigma point simulations diverging. The diverging sigma points will have a large impact on the weighted mean calculations and produce unreasonable predictions.

The model used in B.3 addresses the issues of model robustness and therefore improves the performance of the UKF. With the results from experiment B.3 the UKF proves to be a viable solution for state estimation during the start-up of the power plant.

6.2 Disturbance Modelling

In the UKF experiments A.1, A.3 and B.3 (defined in Sections 4.1 and 4.2) the input disturbance states are used by the UKF to compensate for model mismatches. Experiment A.1 and A.3 showed that the lack of disturbance model for the heat transfer coefficient makes the UKF underestimate the amount of transferred heat in the evaporator while overestimating the transferred heat in the other heat exchangers, similar to what was seen in Figures 3.4 and 3.5. To compensate for this the UKF estimates a negative offset on the firing power (see Figures 5.3 and 5.7) and

underestimates the offsets on the valve openings to keep the steam pressure up. The lowered firing power makes the UKF underestimate the heat transferred to the evaporator even more, and essentially trade better enthalpy and wall temperature estimates for all the heat exchangers at the price of poorer estimates of the evaporator states.

The results from B.3 show that a negative firing power offset is estimated and the evaporator wall temperature is underestimated, see Figures 5.14 and 5.15. The behavior is similar to the results in A.1 and A.3, which indicates that the heat transfer coefficient of the evaporator in the optimization model differs to the one in the detailed model. Adding a disturbance state as in A.2 to the evaporator in the optimization model used in B.3 could prove to be a method to improve the performance. Also, results from A.2 indicates that the disturbance state is observable with the current set of measurements which would make the addition straight-forward to test.

The heat transfer coefficients of all the heat exchangers are dependent on the flow of flue gas and steam on each side of the wall. However, with the approximation made in Section 3.3.2 the flue gas flow will always be constant in the optimization model, which in turn makes all the heat transfer coefficient on the gas side constant. After evaluating the addition of the disturbance state to the evaporator, the addition of similar disturbance states to the rest of the heat exchangers could be tested to see if improved performance can be gained.

Another not modelled effect in the optimization model that could possibly be compensated for by disturbance modelling is the spray attemperators in the power plant. The spray attemperators are present in the detailed model, and cool down the steam in the heat exchangers. The cooling leads to an enthalpy decrease that is not present in the optimization model. A suggestion is to add constant disturbance states to the energy balance equations of each heat exchanger where spray attemperators are present. Note that these disturbances should only be added to the heat exchangers where spray attemperators are actually present and their estimated values should be limited to represent energy loss. Otherwise they could risk representing unphysical losses or even gains of energy to the power plant.

All disturbance states used in this thesis are assumed to be constant. This assumption works when the disturbances can be regarded as approximately constant on the current time scale. In the case of A.2, it works well since the disturbances really are constant, and in B.3 it works at the relatively short time scale of 10 s. However, if the sampling interval is increased the assumption could prove insufficient.

6.3 Control using NMPC

When controlling the optimization model the NMPC seems to be working well. All set-points are reached and even without any integral action there are no stationary

errors. This shows that no terminal constraints or costs are needed to reach the set-points. Reaching all set-points exactly is also not critical for the plant and if small stationary errors arise from model disparities it should cause no problems for the overall control of the start-up. As there is measurement noise present in these experiments, the NMPC seems to be robust to the level of noise that is expected from the real plant. The stress constraints put on the NMPC are violated slightly and the most negative stress value found was around -75 MPa in the SH header and -50 MPa in the RH header, see Figure 5.21. However, both of these are within the stress levels for the currently used start-up scheme found in Table 3.1 and well below the requirement that the plant should manage 2000 cold starts. In the demonstrator experiment the stress violations were larger and there was a sharp pressure dip in the RH header stress. This is probably due to the load disturbance put on the system which was not yet estimated by the UKF at the first NMPC iteration.

Comparing offline and NMPC results it is clear that the NMPC provides better results even when no process disturbances are present. The NMPC reaches its set-point which the offline optimization does not, and the offline optimization has huge overshoots in both pressure signals in Figures 5.19 and 5.20. The reason for the better performance is the feedback incorporated in the MPC algorithm. Due to the coarse discretization in the optimization problem the simulation results will differ from what the NMPC expects. Here feedback helps as the states in the NMPC are updated each iteration with the correct values in the same way as if process noise was present. Another benefit is that the NMPC allows more changes in the input. Both NMPC and offline optimization have a control horizon of five samples to make the optimization converge. With the NMPC a new input can still be calculated for each sampling interval.

The different cost functions did not seem to have a big effect on the end result. The set-points were reached in a similar time even if the trajectories have different profiles. Setting a larger weight is effective if the goal is to reach one specific set-point earlier. It shows, both quantitatively in Table 3.3 and in the plots that increasing the weight on a state made that trajectory more optimal. The difference in total firing power used during the start up, that could be seen in the offline optimizations in Table 3.3, was not present in the NMPC results, however.

The coarse discretization is one of the largest approximations made by the NMPC. The long sample time will also lead to errors if there are differences between the optimization model and the target system. With shorter sampling time the feedback would be more frequent letting the NMPC react more to process noise. In order to have a shorter sample time the prediction horizon will need to be shorter however so that the degrees of freedom do not increase too much. There is probably a limit on how small the prediction horizon can be made though. An educated guess is that it needs to be at least 2000-2500 s so that the initial temperature and pressure ramp, stress dip and pressure overshoot can be predicted in the first samples.

The robustness problems and failed optimizations were the biggest problems for the NMPC used in this thesis. Even a small deviation in start values from the optimal trajectory can lead to a diverging optimization. A few failed iterations does not have a great impact on the end result assuming the optimization model is a good enough approximation of the target system thanks to the error handling implemented in [Axelsson, 2015]. It indicates, however, that the model is not robust enough to be used in a real application. When running the NMPC against the detailed model or the real plant the deviations from the preferred trajectories will also be much larger. The largest problem seems to be the fast dynamics in the optimization model creating a stiff problem. Early tests using a model with an ideal separator, as was used in experiment B.3, seems promising and resulted in a more robust NMPC.

6.4 Optimization Model

The largest problems with using the optimization model were that some dynamics were much faster than the sampling time of the system which leads to stiff problems. Removing the dynamic steam states improved the model significantly and made it more optimization friendly compared to the model used in [Runvik, 2014]. The only fast dynamic left in the model was the separator level control. The first test with an ideal separator, with these dynamics removed, showed much promise. In experiment B.3 the UKF convergence is shown to improve greatly and the initialization problems present with the earlier model with a static steam model also seem to have been fixed. The steady state values of this new model does differ from those of the old model however, so before it can be used in a real application it needs to be calibrated and validated against the detailed model in the same way as the current model.

The robustness issue with the optimization model is due to a limited feasibility region. One limitation is that the pressures need to decrease for each heat exchanger in the chain to avoid back flows (which are unphysical). Another limitation is that the *water_poly* functions have a limited range on their inputs where they are valid. The first problem is not easy to handle since the values that lead to a feasible problem all depend on each other, but there does not seem to have been any issues with this during the experiments. It is still something that should be monitored when evaluating the model however. A test to handle the second issue was performed in experiment B.3 where maximum and minimum values were added to these signals and that seems to be working. No sigma points failed during this test and the states converged.

When comparing the values of the optimization model to the detailed model they seem to be matching fairly well, see the MSE for each state in Table 3.6. The largest differences are the energy balances in the system, causing the relatively large temperature devitations, and the valve characteristics. The valve characteristic differ-

ences are not that easy to solve but it is possible to circumvent that by sending calculated pressure values as set-points to the detailed models instead of directly using valve positions.

6.5 Real-Time Aspects

The real-time aspects of the UKF and NMPC must be considered if the demonstrator is to be applied to the real power plant. In the UKF experiments the sampling deadlines were only met in B.3, where the UKF completed each iteration in about 5 s with a sampling interval of 10 s. In the case of the NMPC the sampling deadlines were met in all experiments, where each iteration was completed in about 1-30 s with a sampling interval of 500 s. Preferably the sampling interval of the NMPC should be shorter, and will then require faster optimization.

For the UKF, there are two ways of improving the computational speed. The first is to minimize the number of sigma points used in each prediction. Since the number of sigma points is directly related to the number of states (see Section 2.4.2), this means that simplifying the model by assuming static relations will increase the speed of the UKF. The second way is to make the integrator able to simulate the model with larger time steps. This can be done by ensuring that the model is non-stiff. Both suggested methods were applied to the optimization model in experiment B.3, where the UKF went from a worst case of about 30 min per iteration to 5 s per iteration.

For the NMPC, the ways of improving the computational speed are similar to those for the UKF. An initial test with the model from B.3 shows a promising increase in speed also for the NMPC that should be further investigated. Another way of increasing the speed could be to linearize the optimization problem on a shorter time horizon and use a linear MPC. A convex optimization problem can then be solved, which not only simplifies the optimization but also guarantees a global optimal solution, see Section 2.3.

Simulations and optimization in JModelica.org are not done with respect to real-time constraints, and for the Jänschwalde plant this will require the application to run in a real-time operating system. Especially the UKF could benefit from this since several sigma points could be simulated in parallel.

Even if all deadlines are met, a long optimization time can lead to delays. The UKF prediction step will not cause this problem as it is run after the NMPC has sent the inputs to the target system. The UKF update and NMPC optimization steps will however need to be performed between the sampling of the system and sending a control input. This is no problem in simulation, but when running against the real plant significant delays, compared to the sampling period, will lead to disturbances.

One way to make the NMPC more robust to delays is to let the UKF predict the states at some point in the future and let the NMPC optimize using these states. This value is then sent as input to the system when the predicted time point has arrived. The time point needs to be further into the future than the optimization time. If the UKF predictions are unreliable this might lead to larger disturbances than the delay though so this must be inspected carefully if it is to be implemented.

6.6 Start-up Optimization Potential

The specified constraints on stress in critical components are fulfilled and as discussed in Section 6.3 the stress levels during simulations are far from the limits for 2000 cold start-ups, found in Table 3.1. There is therefore potential of ramping up the power plant even faster, but because of the model disparities between the optimization model and the detailed model it is probably good to leave a large margin to the constraints.

The total time for the first phase of the soft start is 7700 s in the reference start-up with the detailed model. The reference soft start in Figure 2.3 starts with lower pressure and temperature values than in the detailed model, and the initial values of the detailed model corresponds to roughly the time instant 90 min in Figure 2.3. The corresponding start-up time in the hand book is then 5400 s. In the NMPC experiment the time to reach and stabilize the set-points is below 3500 s with both C1 and C2 cost functions. This indicates a time saving of 1900-4200 s or roughly 30-70 min. In the case of the demonstrator the set-points are not entirely met after the time frame of 5000 s and time saving is therefore not estimated.

Saving time during start-up increases income as the power plant is online and sells electricity for a longer time. A time saving of 30-70 min for a block of two boilers with a generator output of 500 MW means an additional energy output of about 250-583 MWh to the electricity grid. Assuming an energy price of € 33,25 per MWh [EEX, 2015], that means an additional income of roughly € 8000-19000 per block for one soft start.

The fuel consumption of the start-up is related to the integral of the firing power. The integral of the firing power is similar for both cost functions C1 and C2 in the NMPC experiment, with values 1176 and 1178 respectively after 3500 s. The corresponding value for the reference soft start with the detailed model is 3047 after 7700 s. This also indicates that fuel could be saved when optimizing the start-up. However, a more accurate calculation of possible savings has to be done when using the NMPC on the detailed model.

Saving fuel during start-up is desirable both from an environmental and a monetary viewpoint. Saving lignite and oil reduces the negative impact on global warming by reducing the amount of CO_2 emissions during start-up. From a monetary viewpoint

it is desirable to reduce the total amount of fuel used to reduce fuel costs. It has however not been possible to convert the firing power to amount of coal and oil used, so no figure on the amount of money or CO_2 saved can be calculated at this point. There is no explicit relation between this firing power signal and mass of fuel. The only way to get these values is to simulate the detailed model with the correct firing power and let it calculate the fuel flow. This was not possible in the current study since the oil burners and coal mills are run with a fixed time schedule, restricting the firing power.

7

Conclusions

In this thesis a solution for start-up optimization of a thermal power plant has been developed. An NMPC (Nonlinear Model Predictive Controller) solution based on the class developed in [Axelsson, 2015] has been implemented to optimize the input trajectory to the power plant while keeping stress levels in the plant within specified constraints. To estimate the states of the plant, a UKF (Unscented Kalman Filter) class compatible with the framework JModelica.org has been implemented. The optimization model used in the NMPC and UKF implementations has been previously developed in [Runvik, 2014], but was modified to increase performance and robustness. The NMPC and UKF implementations were tested separately and in an integrated solution against the optimization model acting as the real plant. Additionally, the UKF was also tested offline on a more detailed model of the power plant.

The aim was to achieve an integrated solution by packaging the different models as FMUs (Functional Mock-up Units). However, the detailed model could not be compiled into an FMU and used with the JModelica.org framework, nor could it be simulated iteratively through the Python/Dymola interface. Because of this, the specification on using the detailed model as target system could not be met. To include the detailed model, further work has to be performed to simplify the initialization at each sample instant.

The performance of the UKF is highly dependent on the model itself, and requires a model which is robust to initialization with various initial state values. The results from the experiments with the UKF show a robustness issue with the optimization model, where estimates of states with fast dynamics diverged when the differences between process and model were too large. Using a modified optimization model which addresses these issues, the performance of the UKF is drastically improved in terms of robustness and computation time. Using the modified model the UKF manages to estimate the states during a soft start with the detailed model with a maximum mean absolute error of 0.7 % on the estimates.

To choose suitable measurement signals from the plant, an observability analysis was performed. Only signals that are considered measurable in the real plant were considered. The initial method was to linearize the plant model at each sampling instant and apply linear observability criterion. The method proved numerically unreliable. Therefore a simpler test, estimating the states in a steady-state simulation, was derived and performed. The final set of measurement signals derived from the observability analysis includes steam pressures and temperatures from all heat exchanger, and is sufficient for the UKF to reduce the estimation error of all states.

The optimization model's input dynamics differ from the detailed model, and the two models respond differently, especially for different valve inputs. Using disturbance states on the inputs as a solution is shown to be beneficial for the estimation performance. Experiments where a constant disturbance is put on the evaporator heat transfer coefficient in the optimization model show similar behaviour as in the estimation of the detailed model. This indicates that the estimation of the detailed model could be improved by modelling an added disturbance to the heat transfer coefficient of the evaporator.

The results from the optimized start-ups using NMPC show potential for saving time and fuel, but this has still to be verified against the detailed model. In all optimized start-up experiments the stress levels of the critical components stay within the limits of the 2000 cold start specification. The largest stress value encountered does not reach 50 % of the critical limit, which gives a large margin for model disparities and also indicates a potential for optimizing the start-up time even further. All set-points were reached in 3500 s, which is considerably faster than the reference soft starts in both the detailed model and the power plant hand book. The NMPC also seems to be robust to measurement noise of the level expected in the plant. In the same experiments, the total energy input from fuel is less than 40 % of the corresponding total in the reference soft start with the detailed model.

Simulation with the detailed model indicated that the stress constraint in the RH2 header component is active during start-up. Therefore the RH2 header was added to the optimization model. The addition proved critical as the stress constraint was active during the optimized start-up of the optimization model. However, the addition also made the optimization problem more difficult to solve.

The optimization model proved to be stiff due to the different speed of its dynamics. The dynamics of the wall temperatures were slower than the steam and separator level control dynamics. The stiffness issue was solved by using static steam relations and replacing the level control with an ideal separator. The robustness was improved by limiting the variables within the valid range of the polynomial functions that approximate the thermodynamic relations in the model. The modifications produce significantly better results in terms of robustness for the UKF, and initial testing suggests that the NMPC robustness is also improved.

With an average computation time of 5 s and a sampling interval of 10 s the UKF should be feasible in a real-time application. The NMPC managed to complete each iteration with a computation time of 1-30 s, with a sampling interval of 500 s. Hence, the NMPC met its real-time sampling deadline. These results indicate that an integrated solution could be feasible in a real time application. However, more testing is still needed to verify this, and the impact of real time issues such as computational delays must be investigated.

7.1 Future Work

Recommendations for future work are:

- Calibrate and validate the modified optimization model with the ideal separator, and use it for future experiments.
- Evaluate the addition of disturbance states on the heat transfer coefficients of the heat exchangers when testing against the detailed model.
- Evaluate the addition of disturbance states to the energy balances of the heat exchangers that have spray attemperators in the real plant.
- Simplify the detailed model so that it can be initialized at any time instant in a robust manner.
- Find a solution to shorten the sampling time of the NMPC.
- Improve the optimization so that it converges for all NMPC iterations.
- Test the demonstrator using the detailed model as target system.
- Calculate monetary and environmental savings using the optimized start-up.

Bibliography

Åkesson, J. (2008). "Optimica — An Extension of Modelica Supporting Dynamic Optimization". In: *6th Modelica Conference March 3rd - 4th*. URL: https://www.modelica.org/events/modelica2008/Proceedings/proceedings/volume_1.pdf (Accessed: 2015-06-18). Bielefeld, Germany, pp. 57–66.

Alkaya, A. (2014). "Unscented Kalman filter performance for closed-loop nonlinear state estimation: a simulation case study". *Electrical Engineering* **96**:4, pp. 299–308. DOI: 10.1007/s00202-014-0298-x.

Allgöwer, F., R. Findeisen, and Z. Nagy (2004). "Nonlinear Model Predictive Control: From Theory to Application". *J. Chin. Inst. Chem. Engrs.* **35**:3, pp. 299–315. ISSN: 0368-1653.

Andersson, E. (2013). *Development of a dynamical model for start-up optimization of coal-fired power plant*. MS thesis 2013:010. Department of Electrical Engineering Royal Institute of Technology, Stockholm, Sweden.

Andersson, J., J. Åkesson, F. Casella, and M. Diehl (2011). "Integration of CasADi and JModelica.org". In: *8th Modelica Conference March 20th - 22nd*. Dresden, Germany, pp. 218–231. DOI: 10.3384/ecp11063218.

Aronsson, P., P. Fritzson, H. Elmqvist, C. Höger, G. Kurzbach, J. Mattsson, H. Olsson, A. Pop, E. Shmoylova, M. Sjölund, and S. Vorkoetter (2014). *Modelica® - A Unified Object-Oriented Language for Systems Modeling, Language Specification Version 3.3 Revision 1*. Ed. by O. H. URL: https://www.modelica.org/documents/ModelicaSpec33Revision1.pdf (Accessed: 2015-06-18).

Axelsson, M. (2015). *Nonlinear Model Predictive Control in JModelica.org*. MS thesis ISSN: 0280–5316. Department of Automatic Control Lund University, Lund, Sweden.

Böiers, L. (2010). *Mathematical Methods of Optimization*. Studentlitteratur, Lund, Sweden. ISBN: 9789144070759.

Blochwitz, T., M. Otter, M. Arnold, C. Bausch, C. Clauß, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, T. Neidhold, D. Neumerkel, H. Olsson, J.-V. Peetz, and S. Wolf (2011). "The Functional Mockup Interface for Tool independent Exchange of Simulation Models". In: *8th Modelica Conference March 20th - 22nd*. Dresden, Germany, pp. 105–114. DOI: 10.3384/ecp11063105.

Dassault Systemes (2015). *Dymola*. URL: http://www.3ds.com/products-services/catia/products/dymola (Accessed: 2015-06-18).

EEX (2015). *EPEX SPOT, Phelix Day Base Price*. URL: http://www.eex.com/en#/en (Accessed: 2015-06-22).

Fuming, S., L. Guanglin, and W. Jingli (2009). "Unscented Kalman Filter using Augmented State in the Presence of Additive Noise". In: *2009 IITA International Conference on Control, Automation and Systems Engineering 11th - 12th July*. Zhangjiajie, China, pp. 379–382. DOI: 10.1109/CASE.2009.51.

Grüne, L. and J. Pannek (2011). *Nonlinear Model Predictive Control - Theory and Algorithms*. Springer-Verlag, London, England, pp. 43–46, 88–101, 113–125, 211–225. ISBN: 978-0-85729-501-9.

Hübel, M., C. Ziems, A. Berndt, M. Richter, C. Gierow, J. Nocke, E. Hassel, and H Weber (2014a). "Effects of integrating large amounts of wind and solar energy on conventional power plants and optimisation strategies for this new challenge". In: *13th Wind Integration Workshop November 11th - 13th*. Berlin, Germany, pp. 14–19. ISBN: 978-3-98 13870-9-4.

Hübel, M., A. Berndt, S. Meinke, M. Richter, P. Mutschler, E. Hassel, H. Weber, M. Sander, and J. Funquist (2014b). "Modelling a Lignite Power Plant in Modelica to Evaluate the Effects of Dynamic Operation and Offering Grid Services". In: *10th Modelica Conference March 10th - 12th*. Lund, Sweden, pp. 1037–1046. DOI: 10.3384/ECP140961037.

Hendrick, J. and A. Girard (2010). *Control of Nonlinear Dynamic Systems: Theory and Applications*. Class notes for course ME 237, Control of Nonlinear Dynamic Systems, Deptartment of Mechanical Engineering, University of California, Berkeley, CA. URL: http://www.me.berkeley.edu/ME237/ControlOfNonlinearDynamicSystems.pdf (Accessed: 2015-06-18).

Julier, S. and J. Uhlmann (2004). "Unscented Filtering and Nonlinear Estimation". *Proceedings of the IEEE* **92**, pp. 401–422. DOI: 10.1109/JPROC.2003.823141.

Modelica Association (2015). *MODRIO - Model Driven Physical Systems Operation*. URL: https://modelica.org/external-projects/modrio (Accessed: 2015-06-03).

Modelon AB. *Python API Docs, Common Folder*. URL: http://www.jmodelica.org/api-docs/pyjmi/pyjmi.common.html (Accessed: 2015-06-18).

— (2014). *JModelica.org User Guide - Version 1.15*. URL: http://www.jmodelica.org/page/25809 (Accessed: 2015-06-17).

Rambabu, K., B. Foss, and L. Imsland (2008). "Applying the unscented Kalman filter for nonlinear state estimation". *Journal of Process Control* **18**, 753–768. DOI: 10.1016/j.jprocont.2007.11.004.

Rawlings, J. and D. Mayne (2012). *Model Predictive Control: Theory and Design*. Nob Hill Publishing, Madison, Wisconsin, pp. 89–99, 112–124, 147–153, 187–203, 218–237, 371–375, 399–403. ISBN: 978-0-9759377-0-9.

Ruan, X., X. Hou, and H. Ma (2014). "Stability analysis of constrained MPC with CLF applied to discrete-time nonlinear system". *Science China, Information Sciences* **57**, pp. 1–9. DOI: 10.1007/s11432-014-5111-y.

Runvik, H. (2014). *Modelling and start-up optimization of a coal-fired power plant*. MS thesis ISSN: 0280–5316. Department of Automatic Control Lund University, Lund, Sweden.

Salau, N., J. Trierweiler, and A. Secchi (2014). "Observability analysis and model formulation for nonlinear state estimation". *Applied Mathematical Modelling* **38**, 5407–5420. DOI: 10.1016/j.apm.2014.03.053.

Vattenfall AB (2012). *Kraftwerk Jänschwalde, BV 500 Block A-F, Band 1*. (Internal document).

Wächter, A. (2009). "Short Tutorial: Getting Started With Ipopt in 90 Minutes". In: *Combinatorial Scientific Computing 1st June*. Dagstuhl Seminar Proceedings. URN: nbn:de:0030-drops-20890. Wadern, Germany, pp. 1–17.

Wan, E. and R. Van der Merwe (2000). "The unscented Kalman filter for nonlinear estimation". In: *Adaptive Systems for Signal Processing, Communications, and Control Symposium*. Lake Louise, Alberta, Canada, pp. 153–158. DOI: 10.1109/ASSPCC.2000.882463.

Yanagisawa, Y., T. Ogita, and S. Oishi (2014). "Convergence analysis of an algorithm for accurate inverse Cholesky factorization". *Japan Journal of Industrial and Applied Mathematics* **31**, pp. 461–482. DOI: 10.1007/s13160-014-0154-4.

Yuanxin, W., H. Dewen, W. Meiping, and H. Xiaoping (2005). "Unscented Kalman Filtering for Additive Noise Case: Augmented vs. Non-augmented". *IEEE Signal Processing Letters* **12**, pp. 357–360. DOI: 10.1109/LSP.2005.845592.

A

Estimation Results

State	Initial Error	Final Error
RH1A Wall Temperature	-11.2 %	0.1 %
RH1A Enthalpy	-3.6 %	0.05 %
RH1A Pressure	1.2 %	-0.07 %
RH1B Wall Temperature	-5.5 %	0.001 %
RH1B Enthalpy	-1.9 %	0.001 %
RH1B Pressure	1.1 %	-0.07 %
RH2 Wall Temperature	-2.5 %	-0.02 %
RH2 Enthalpy	-0.9 %	-0.009 %
RH2 Pressure	1.0 %	-0.07 %
SHS Wall Temperature	-0.9 %	0.007 %
SHS Enthalpy	-8.8 %	0.2 %
SHS Pressure	-10.3 %	0.009 %
SH1 Wall Temperature	-4.1 %	0.009 %
SH1 Enthalpy	-23.2 %	1.0 %
SH1 Pressure	-10.4 %	0.009 %
SH2 Wall Temperature	-12.0 %	0.002 %
SH2 Enthalpy	-18.3 %	1.0 %
SH2 Pressure	-10.5 %	0.01 %
SH3 Wall Temperature	-10.7 %	0.002 %
SH3 Enthalpy	-13.6 %	1.0 %
SH3 Pressure	-10.5 %	0.01 %
SH4 Wall Temperature	-13.4 %	0.002%
SH4 Enthalpy	-8.7 %	1.0 %
SH4 Pressure	-10.6 %	0.01%
Header SH4 Wall Temperature	-13.5 %	-0.001%
Header SH4 Enthalpy	-8.7 %	1.0 %
Header SH4 Pressure	-10.6 %	0.01%
Header RH2 Wall Temperature	-2.6 %	0.01 %
Header RH2 Enthalpy	-0.9 %	-0.008 %
Header RH2 Pressure	0.8 %	-0.07%
Evaporator Wall Temperature	-0.6 %	-0.02%
Evaporator Enthalpy	-1.6 %	-0.04%
Evaporator Pressure	-2.6 %	-0.01%
Level-Controller Integral Error	-3.4 %	0.3 %
Separator Mixture Enthalpy	-2.7 %	0.008 %
Separator Mixture Pressure	-10.3 %	0.008%
Firing Power Disturbance	100.0%	-10.1 %
HP-T Bypass Valve Disturbance	100.0 %	-0.9 %
IP-T Bypass Valve Disturbance	100.0 %	-0.2 %

Table A.1 Procentual estimation errors for each state in the observability analysis.

State	Initial Error	Final Error	Mean Error	Mean Absolute Error
RH1A Wall Temperature	-12.6 %	-1.3 %	-0.6 %	0.6 %
RH1A Enthalpy	-4.0 %	-0.7 %	-0.2 %	0.3 %
RH1A Pressure	4.2 %	0.4 %	0.2 %	0.3 %
RH1B Wall Temperature	-6.7 %	-1.8 %	-0.7 %	0.7 %
RH1B Enthalpy	-2.3 %	-0.9 %	-0.3 %	0.3 %
RH1B Pressure	4.0 %	0.2 %	0.1 %	0.2 %
RH2 Wall Temperature	-3.6 %	-0.3 %	-0.5 %	0.5 %
RH2 Enthalpy	-1.3 %	-0.3 %	-0.2 %	0.2 %
RH2 Pressure	4.0 %	-0.03 %	0.02 %	0.2 %
SHS Wall Temperature	-0.7 %	-0.3 %	-0.2 %	0.2 %
SHS Enthalpy	-6.2 %	-0.3 %	-0.2 %	0.5 %
SHS Pressure	-7.0 %	0.09 %	-0.1 %	0.2 %
SH1 Wall Temperature	-4.5 %	0.6 %	0.01 %	0.3 %
SH1 Enthalpy	-23.5 %	0.3 %	-1.9 %	2.2 %
SH1 Pressure	-7.0 %	0.04 %	-0.1 %	0.2 %
SH2 Wall Temperature	-13.1 %	-0.8 %	-0.6 %	0.6 %
SH2 Enthalpy	-18.8 %	-0.5 %	-1.6 %	1.7 %
SH2 Pressure	-7.0 %	0.00005 %	-0.2 %	0.2 %
SH3 Wall Temperature	-11.8 %	-0.6 %	-0.4 %	0.5 %
SH3 Enthalpy	-14.2 %	-0.4 %	-1.0 %	1.5 %
SH3 Pressure	-7.1 %	-0.05 %	-0.2 %	0.2 %
SH4 Wall Temperature	-14.7 %	-0.7 %	-0.5 %	0.6 %
SH4 Enthalpy	-9.3 %	-0.5 %	-0.5 %	0.5 %
SH4 Pressure	-7.1 %	-0.09 %	-0.2 %	0.2 %
Header SH4 Wall Temperature	-14.8 %	0.02 %	-0.3 %	0.4 %
Header SH4 Enthalpy	-9.3 %	-0.5 %	-0.6 %	0.6 %
Header SH4 Pressure	-7.2 %	-0.1 %	-0.2 %	0.2 %
Header RH2 Wall Temperature	-3.7 %	-0.006 %	-0.3 %	0.3 %
Header RH2 Enthalpy	-1.3 %	-0.3 %	-0.2 %	0.2 %
Header RH2 Pressure	3.8 %	-0.2 %	-0.09 %	0.3 %
Evaporator Wall Temperature	-0.2 %	0.6 %	0.9 %	0.9 %
Evaporator Enthalpy	-0.6 %	7.4 %	2.6 %	2.6 %
Evaporator Pressure	-1.5 %	0.03 %	0.06 %	0.1 %
Level-Controller Integral Error	-3.9 %	-47.4 %	-8.1 %	9.0 %
Separator Mixture Enthalpy	-1.8 %	0.3 %	-0.08 %	0.2 %
Separator Mixture Pressure	-6.9 %	0.1 %	-0.1 %	0.2 %

Table A.2 Results from the UKF estimation in experiment A.1 (ramp input).

State	Initial Error	Final Error	Mean Error	Mean Absolute Error
RH1A Wall Temperature	-12.6 %	-0.02 %	-0.3 %	0.3 %
RH1A Enthalpy	-4.0 %	-0.01 %	-0.08 %	0.1 %
RH1A Pressure	4.2 %	-0.008 %	0.07 %	0.2 %
RH1B Wall Temperature	-6.7 %	-0.02 %	-0.1 %	0.2 %
RH1B Enthalpy	-2.3 %	-0.008 %	-0.05 %	0.06 %
RH1B Pressure	4.0 %	-0.008 %	0.07 %	0.2 %
RH2 Wall Temperature	-3.6 %	-0.02 %	-0.09 %	0.1 %
RH2 Enthalpy	-1.3 %	-0.01 %	-0.03 %	0.04 %
RH2 Pressure	4.0 %	-0.007 %	0.06 %	0.2 %
SHS Wall Temperature	-0.7 %	0.002 %	-0.01 %	0.02 %
SHS Enthalpy	-6.2 %	0.0008 %	-0.1 %	0.2 %
SHS Pressure	-7.0 %	0.002 %	-0.1 %	0.1 %
SH1 Wall Temperature	-4.5 %	-0.02 %	-0.06 %	0.1 %
SH1 Enthalpy	-23.5 %	-0.01 %	-0.9 %	1.0 %
SH1 Pressure	-7.0 %	0.002 %	-0.1 %	0.1 %
SH2 Wall Temperature	-13.1 %	-0.03 %	-0.3 %	0.3 %
SH2 Enthalpy	-18.8 %	-0.02 %	-0.7 %	0.7 %
SH2 Pressure	-7.0 %	0.002 %	-0.1 %	0.2 %
SH3 Wall Temperature	-11.8 %	-0.04 %	-0.2 %	0.3 %
SH3 Enthalpy	-14.2 %	-0.02 %	-0.5 %	0.5 %
SH3 Pressure	-7.1 %	0.002 %	-0.1 %	0.2 %
SH4 Wall Temperature	-14.7 %	-0.04 %	-0.3 %	0.3 %
SH4 Enthalpy	-9.3 %	-0.03 %	-0.3 %	0.3 %
SH4 Pressure	-7.1 %	0.002 %	-0.1 %	0.2 %
Header SH4 Wall Temperature	-14.8 %	-0.03 %	-0.3 %	0.3 %
Header SH4 Enthalpy	-9.3 %	-0.03 %	-0.3 %	0.3 %
Header SH4 Pressure	-7.2 %	0.003 %	-0.1 %	0.2 %
Header RH2 Wall Temperature	-3.7 %	0.009 %	-0.08 %	0.1 %
Header RH2 Enthalpy	-1.3 %	-0.01 %	-0.03 %	0.04 %
Header RH2 Pressure	3.8 %	-0.006 %	0.06 %	0.2 %
Evaporator Wall Temperature	-0.2 %	-0.003 %	-0.006 %	0.02 %
Evaporator Enthalpy	-0.6 %	-0.03 %	-0.01 %	0.05 %
Evaporator Pressure	-1.5 %	-0.003 %	-0.03 %	0.05 %
Level-Controller Integral Error	-3.9 %	0.6 %	-0.07 %	0.6 %
Separator Mixture Enthalpy	-1.8 %	0.02 %	-0.04 %	0.05 %
Separator Mixture Pressure	-6.9 %	0.001 %	-0.1 %	0.1 %

Table A.3 Results from the UKF estimation in experiment A.2 (ramp input and added disturbance state in UKF model)

State	Initial Error	Final Error	Mean Error	Mean Absolute Error
RH1A Wall Temperature	-12.6 %	-1.2 %	-0.7 %	0.7 %
RH1A Enthalpy	-4.0 %	-0.7 %	-0.3 %	0.3 %
RH1A Pressure	4.2 %	0.3 %	0.2 %	0.3 %
RH1B Wall Temperature	-6.7 %	-1.6 %	-0.9 %	0.9 %
RH1B Enthalpy	-2.3 %	-0.9 %	-0.4 %	0.4 %
RH1B Pressure	4.0 %	0.07 %	0.1 %	0.2 %
RH2 Wall Temperature	-3.6 %	0.3 %	-0.5 %	0.5 %
RH2 Enthalpy	-1.3 %	-0.1 %	-0.3 %	0.3 %
RH2 Pressure	4.0 %	-0.2 %	0.03 %	0.1 %
SHS Wall Temperature	-0.7 %	1.0 %	-0.4 %	0.4 %
SHS Enthalpy	-6.2 %	0.7 %	-0.6 %	0.8 %
SHS Pressure	-7.0 %	0.05 %	-0.1 %	0.2 %
SH1 Wall Temperature	-4.5 %	0.4 %	0.1 %	0.3 %
SH1 Enthalpy	-23.5 %	-0.03 %	-1.2 %	1.4 %
SH1 Pressure	-7.0 %	0.06 %	-0.1 %	0.2 %
SH2 Wall Temperature	-13.1 %	-0.5 %	-0.5 %	0.6 %
SH2 Enthalpy	-18.8 %	-0.6 %	-1.7 %	1.8 %
SH2 Pressure	-7.0 %	0.03 %	-0.2 %	0.2 %
SH3 Wall Temperature	-11.8 %	-0.3 %	-0.4 %	0.5 %
SH3 Enthalpy	-14.2 %	-0.4 %	-0.9 %	0.9 %
SH3 Pressure	-7.1 %	-0.006 %	-0.2 %	0.2 %
SH4 Wall Temperature	-14.7 %	-0.4 %	-0.5 %	0.6 %
SH4 Enthalpy	-9.3 %	-0.4 %	-0.5 %	0.5 %
SH4 Pressure	-7.1 %	-0.05 %	-0.2 %	0.2 %
Header SH4 Wall Temperature	-14.8 %	0.4 %	-0.3 %	0.4 %
Header SH4 Enthalpy	-9.3 %	-0.4 %	-0.4 %	0.4 %
Header SH4 Pressure	-7.2 %	-0.09 %	-0.2 %	0.2 %
Header RH2 Wall Temperature	-3.7 %	0.1 %	-0.2 %	0.3 %
Header RH2 Enthalpy	-1.3 %	-0.2 %	-0.3 %	0.3 %
Header RH2 Pressure	3.8 %	-0.5 %	-0.08 %	0.2 %
Evaporator Wall Temperature	-0.2 %	0.4 %	0.7 %	0.8 %
Evaporator Enthalpy	-0.6 %	10.3 %	3.4 %	3.5 %
Evaporator Pressure	-1.5 %	0.009 %	0.03 %	0.1 %
Level-Controller Integral Error	-3.9 %	-568 %	-48.4 %	54.3 %
Separator Mixture Enthalpy	-1.8 %	103 %	4.2 %	4.5 %
Separator Mixture Pressure	-6.9 %	0.04 %	-0.09 %	0.2 %

Table A.4 Results from the UKF estimation in experiment A.3 (optimized input).

State	Initial Error	Final Error	Mean Error	Mean Absolute Error
RH1A Wall Temperature	-6.3 %	-0.03 %	-0.2 %	0.2 %
RH1A Enthalpy	-1.6 %	0.4 %	0.4 %	0.5 %
RH1A Pressure	3.4 %	0.5 %	1.3 %	1.3 %
RH1B Wall Temperature	-5.4 %	-0.09 %	-0.3 %	0.3 %
RH1B Enthalpy	-1.3 %	0.5 %	0.4 %	0.5 %
RH1B Pressure	3.2 %	1.2 %	1.9 %	1.9 %
RH2 Wall Temperature	7.1 %	-0.8 %	-0.4 %	0.9 %
RH2 Enthalpy	3.1 %	0.3 %	0.5 %	0.5 %
RH2 Pressure	2.9 %	1.7 %	2.4 %	2.4 %
SHS Wall Temperature	0.5 %	-0.2 %	-0.4 %	0.7 %
SHS Enthalpy	11.8 %	28.3 %	47.0 %	50.0 %
SHS Pressure	6.2 %	-0.1 %	0.1 %	0.3 %
SH1 Wall Temperature	-1.6 %	-4.8 %	-2.5 %	2.5 %
SH1 Enthalpy	-0.7 %	-159 %	-50.7 %	52.3 %
SH1 Pressure	6.2 %	-0.1 %	0.1 %	0.3 %
SH2 Wall Temperature	-2.0 %	3.0 %	0.7 %	1.4 %
SH2 Enthalpy	-1.0 %	-56.2 %	-18.0 %	19.4 %
SH2 Pressure	6.1 %	-0.2 %	0.09 %	0.3 %
SH3 Wall Temperature	12.0 %	1.8 %	1.0 %	1.1 %
SH3 Enthalpy	4.8 %	-0.5 %	2.3 %	3.0 %
SH3 Pressure	6.1 %	0.06 %	0.2 %	0.3 %
SH4 Wall Temperature	6.5 %	-0.07 %	0.3 %	0.4 %
SH4 Enthalpy	2.4 %	-0.5 %	-0.2 %	0.4 %
SH4 Pressure	6.0 %	0.3 %	0.4 %	0.4 %
Header SH4 Wall Temperature	2.4 %	0.003 %	0.08 %	0.09 %
Header SH4 Enthalpy	2.3 %	-0.4 %	-0.2 %	0.4 %
Header SH4 Pressure	6.0 %	0.5 %	0.6 %	0.6 %
Header RH2 Wall Temperature	6.6 %	0.03 %	0.3 %	0.3 %
Header RH2 Enthalpy	3.0 %	0.7 %	0.7 %	0.7 %
Header RH2 Pressure	2.8 %	2.4 %	3.1 %	3.0 %
Evaporator Wall Temperature	-1.0 %	0.6 %	0.6 %	0.7 %
Evaporator Enthalpy	-1.2 %	2.4 %	2.4 %	2.5 %
Evaporator Pressure	17.0 %	-0.0007 %	0.6 %	0.6 %
Separator Mixture Enthalpy	1.6 %	0.2 %	0.7 %	1.3 %
Separator Mixture Pressure	6.0 %	-0.4 %	0.02 %	0.4 %

Table A.5 Results from the UKF estimation in experiment B.1 (10 s sampling interval).

State	Initial Error	Final Error	Mean Error	Mean Absolute Error
RH1A Wall Temperature	-4.6 %	0.03 %	-0.09 %	0.2 %
RH1B Wall Temperature	-3.8 %	0.02 %	-0.08 %	0.1 %
RH2 Wall Temperature	8.3 %	-0.03 %	-0.1 %	0.1 %
SHS Wall Temperature	-0.3 %	-0.5 %	-0.6 %	0.7 %
SH1 Wall Temperature	-11.9 %	-0.006 %	-0.01 %	0.1 %
SH2 Wall Temperature	-4.8 %	-0.04 %	-0.04 %	0.1 %
SH3 Wall Temperature	13.2 %	-0.02 %	-0.03 %	0.1 %
SH4 Wall Temperature	6.7 %	-0.02 %	0.004 %	0.09 %
Header SH4 Wall Temperature	2.6 %	0.005 %	-0.001 %	0.1 %
Header RH2 Wall Temperature	7.8 %	0.05 %	0.02 %	0.1 %
Evaporator Wall Temperature	1.1 %	0.4 %	0.5 %	0.5 %

Table A.6 Results from the UKF estimation in experiment B.3 (modified optimization model and 10 s sampling interval). Since the steam relations are static in B.3 only wall temperatures are estimated in the UKF.

State	Initial Error	Final Error	Mean Error	Mean Absolute Error
RH1A Wall Temperature	-0.6 %	0.1 %	-0.008 %	0.05 %
RH1A Enthalpy	-0.2 %	0.05 %	-0.003 %	0.02 %
RH1A Pressure	0.4 %	-0.03 %	0.02 %	0.05 %
RH1B Wall Temperature	-0.3 %	0.09 %	0.001 %	0.03 %
RH1B Enthalpy	-0.1 %	0.04 %	0.0009 %	0.01 %
RH1B Pressure	0.4 %	-0.02 %	0.01 %	0.05 %
RH2 Wall Temperature	-0.2 %	0.04 %	0.008 %	0.03 %
RH2 Enthalpy	-0.07 %	0.02 %	0.004 %	0.01 %
RH2 Pressure	0.4 %	-0.02 %	0.01 %	0.05 %
SHS Wall Temperature	-0.2 %	-0.0003 %	-0.007 %	0.009 %
SHS Enthalpy	-2.2 %	0.002 %	-0.05 %	0.07 %
SHS Pressure	-2.6 %	-0.002 %	-0.05 %	0.06 %
SH1 Wall Temperature	-1.9 %	0.04 %	-0.04 %	0.09 %
SH1 Enthalpy	-0.7 %	0.04 %	-0.02 %	0.05 %
SH1 Pressure	-2.6 %	-0.001 %	-0.05 %	0.06 %
SH2 Wall Temperature	-1.0 %	0.05 %	-0.02 %	0.06 %
SH2 Enthalpy	-0.4 %	0.05 %	-0.004 %	0.03 %
SH2 Pressure	-2.7 %	-0.0009 %	-0.05 %	0.06 %
SH3 Wall Temperature	-0.5 %	0.05 %	-0.008 %	0.05 %
SH3 Enthalpy	-0.2 %	0.05 %	-0.001 %	0.03 %
SH3 Pressure	-2.7 %	-0.0006 %	-0.05 %	0.06 %
SH4 Wall Temperature	-0.4 %	0.05 %	-0.005 %	0.04 %
SH4 Enthalpy	-0.1 %	0.04 %	0.00006 %	0.02 %
SH4 Pressure	-2.7 %	-0.0002 %	-0.05 %	0.06 %
Header SH4 Wall Temperature	-0.4 %	0.02 %	-0.006 %	0.04 %
Header SH4 Enthalpy	-0.1 %	0.04 %	0.005 %	0.02 %
Header SH4 Pressure	-2.7 %	0.0001 %	-0.05 %	0.06 %
Header RH2 Wall Temperature	-0.2 %	-0.002 %	0.004 %	0.04 %
Header RH2 Enthalpy	-0.07 %	0.02 %	0.006 %	0.02 %
Header RH2 Pressure	0.3 %	-0.02 %	0.01 %	0.05 %
Evaporator Wall Temperature	-0.2 %	-0.008 %	-0.001 %	0.01 %
Evaporator Enthalpy	-0.5 %	-0.06 %	-0.001 %	0.04 %
Evaporator Pressure	-0.7 %	-0.01 %	-0.02 %	0.02 %
Level Controller Integral Error	-0.7 %	3.4 %	0.1 %	0.5 %
Separator Mixture Enthalpy	-0.7 %	0.02 %	-0.01 %	0.03 %
Separator Mixture Pressure	2.6 %	-0.002 %	-0.05 %	0.06 %
HP-T Bypass Valve Disturbance	100 %	-3.5 %	2.9 %	4.1 %
IP-T Bypass Valve Disturbance	100 %	-9.8 %	3.1 %	5.4 %
Firing Power Disturbance	100 %	-73.9 %	27.0 %	55.6 %

Table A.7 Results from the UKF estimation in the demonstrator run.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER´S THESIS

Date of issue
June 2015

Document Number
ISRN LUTFD2/TFRT--5972--SE

Author(s)
Marcus Thelander Andrén
Christoffer Wedding

Supervisor
Jonas Funkquist, Vattenfall R&D
Stéphane Velut, Modelon
Pontus Giselsson, Dept. of Automatic Control, Lund University, Sweden
Rolf Johansson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle

Development of a Solution for Start-up Optimization of a Thermal Power Plant

Abstract

This thesis covers optimizing the first phase of the start-up of a thermal power plant using Nonlinear Model Predictive Control (NMPC) and state estimation using an Unscented Kalman Filter (UKF). The start-up has been optimized in regards to time and fuel usage. The thesis is done as a joint project between Vattenfall and Modelon. Both NMPC and UKF are nonlinear methods and require a model of the power plant. The model used in this thesis has been developed in the language Modelica in a previous master thesis and has been extended and improved upon during this thesis. The optimization and simulation of the model required by the NMPC and UKF was done within the framework of JModelica.org. Another, more detailed, model of the power plant, developed by Vattenfall, was originally planned to be used as the process to be controlled.

State estimation using the UKF has been successful, with a maximum mean absolute error of 0.7 % when estimating the states of the detailed model in a reference startup. When using the NMPC to control the optimization model itself, the simulated start-up time is 70 minutes faster compared to a reference start-up using the detailed model. This is more than half the time of the first phase of the start-up. The total firing power, which relates to the fuel amount, is also considerably less, with the optimized value being about 40 % of that in the reference soft start with the detailed model.

Due to difficulties in initializing the detailed model, it was not possible to run it online together with the NMPC and UKF. Running the NMPC and UKF together on the optimization model worked, but the NMPC failed to find an optimal trajectory 8 out of 10 iterations. The conclusion is that the start-up has potential for optimization, but requires more robust models to work with.

Keywords
NMPC, UKF, Thermal Power Plants, JModelica.org, Modelica

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-110

Recipient's notes

Security classification