Industrial Model Predictive Control

Emil Schultz Christensen

Kongens Lyngby 2013
DTU Compute-M.Sc.-2013-49

Technical University of Denmark
DTU Compute
Matematiktovet, Building 303B, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@compute.dtu.dk
www.compute.dtu.dk DTU Compute-M.Sc.-2013-49

Summary

This thesis presents a procedure for controlling an industrial process using Model Predictive Control (MPC).

The first part of the thesis introduces the basic ideas of model predictive control and the mathematical theory on which the procedure is based. In particular, it is investigated how a linear model of the process to be controlled can be identified from input-output data of the process. Furthermore, it is discussed how a simulation model of the industrial process can be modeled.

Disturbance rejection and offset free control are important concepts in industrial control. To achieve offset free control in the face of unknown disturbances and/or plant-model mismatch, integrators are added to the identified linear model. Three different approaches to adding these integrators are presented.

Based on the identified linear model extended with integrators an unconstrained MPC is formulated and subsequently transformed into a convex quadratic optimization problem. This optimization problem can be solved explicitly and the resulting optimal control law is linear.

The linear controller is combined with the linear process model forming a closed-loop state-space model. For the purpose of tuning the developed MPC, an optimization based tuning approach was studied. To set up this optimization problem different performance measures for the closed-loop control system have been analyzed. One of the key elements of the optimization is the addition of a non-linear constraint, which is used to ensure robustness of the resulting controller.

In the second part of the thesis a case study has been conducted for a modified 4-tank process, this process has been used as a representative of an industrial process. The process exhibits some of the typical behavior of an industrial process such as strong interaction and non-minimum phase behavior.

The first part of the case study identifies a linear model of the modified 4-tank system from input-output data. Since no real industrial process has been available for this project, the input-output data were obtained by simulation using a first-principles non-linear model of the process. Secondly, tuning parameters were obtained from the optimization based tuning approach.

Finally, closed-loop simulations have been carried out using the tuning parameters obtained by the optimization problem. In these simulations the first-principles non-linear model for the modified 4-tank process was used as the plant.

In general it was seen that the optimization approach produced some reasonably good tuning parameters for the modified 4-tank process. Furthermore, in closed-loop simulation it was illustrated that the closed-loop performance obtained was satisfactory with respect to both tracking and disturbance rejection even under a high level of noise in the system.

Preface

This thesis was prepared at DTU Compute, department of Applied Mathematics and Computer Science at the Technical University of Denmark in fulfillment with the requirements of acquiring an M.Sc. in mathematical modeling and computations. The thesis was carried out in the period February 4th to July 5th of 2013.
The thesis has been conducted under the supervision of Associate Professor John B. Jørgensen, DTU Compute.

I would like to thank my advisor for his guidance, our discussions and the feedback throughout the project period.
I would also like to thank Emil K. Nielsen, Emil B. Kœrgaard, Kristian R. Jensen and Søren S. Hansen for their help, fruitful discussions and general support. Finally, I would like to stress that this thesis would not been possible without the support and encouragement from my family and my girlfriend Sanita Dhaubanjar.

Lyngby, 5-july-2013

Emil Schultz Christensen

Contents

Part I

Theory

Chapter 1

Introduction

This thesis considers the topic of industrial Model Predictive Control (MPC). The term industrial model predictive control refers to the procedure of controlling an industrial process using MPC.

Model predictive control, also known as receding horizon control, refers to control strategies in which a model of the process to be controlled (the plant) is used to predict the future dynamic behavior of the plant. Based on these predictions an optimal control problem is set up, in order to optimize the future dynamic output of the plant.

MPC is one of the most advanced control approaches and has had a substantial impact during the past 30 years on the control industry, especially for processes with constraints. One of the main reasons for this, is that the MPC strategy allows for explicit inclusion of constraints in its formulation. This allows the controlled output of the process to be moved closer to constraints and hereby obtaining more desirable outputs and as a consequence, possibly higher profits [Mac02].

The strategy was originally developed to be used in the control of power production and in the petroleum industry [QB03]. However, today it has emerged to be successfully used and implemented in a wide range of other applications, including processes in the automotive industry and medical applications such as an artificial pancreas.

The processes encountered in different industrial applications usually have very different characteristics. The main characteristics are the time scales on which processes take place and the dimension of the systems describing the processes. The processes encountered in the petroleum industry are usually slow processes with time constants ranging from minutes to hours [Zhu06]. However, the dimensions of these processes are typically large, i.e. a large number of both manipulated and controlled variables. When a process is slow, it is only necessary to sample the process infrequently to capture the dynamic behavior. This implies that the time available to solve the optimal control problem is large with respect to the numerical optimization solvers used today.

In the automotive industry the opposite situation is usually the case. The time constants are very small, in the range of milliseconds [Was10]. However, here the dimensions of the processes are typically small.

In this thesis mainly the slower processes are in focus, and it is therefore assumed that the time available to solve the optimal control problem is sufficient.

Another reason for the popularity of MPC is its intuitive way of addressing the control problem and the resulting structure. The MPC is built up from a number of building blocks. The main building blocks are

- A model of the process.
- An objective function describing the desired performance of the controller. The typical objective function includes the reference tracking error and the control action.
- A set of constraints, which the controller needs to work within.
- An optimization algorithm, used to compute the optimal control inputs that minimize the objective function subject to the given constraints.
- A state estimator, used for state feedback.
- A disturbance model, to obtain offset free control.

The basic idea of MPC is to compute an optimal control sequence such that the controlled outputs of the plant follow a predefined reference trajectory.
At each sampling time k, a measurement $\boldsymbol{y}_k$ of the output of the plant is obtained. From this measurement, the current state $\boldsymbol{x}_k$ of the plant is estimated.

Then a finite horizon optimal control problem is solved over a prediction horizon N, using the current estimated state $\hat{\boldsymbol{x}}_{k|k}$ of the process as the initial state. The solution of the optimal control problem yields a sequence of optimal control inputs $\{\boldsymbol{u}^*_{k+j|k}\}_{j=0}^{N-1}$ over the prediction horizon. The first control input in the optimal control sequence is then applied to the plant at sample instant k. At the next sample $k+1$ the prediction horizon is shifted one sample and the optimization procedure is repeated, with a new plant measurement. Figure 1.1 illustrates the basic principle of model predictive control.

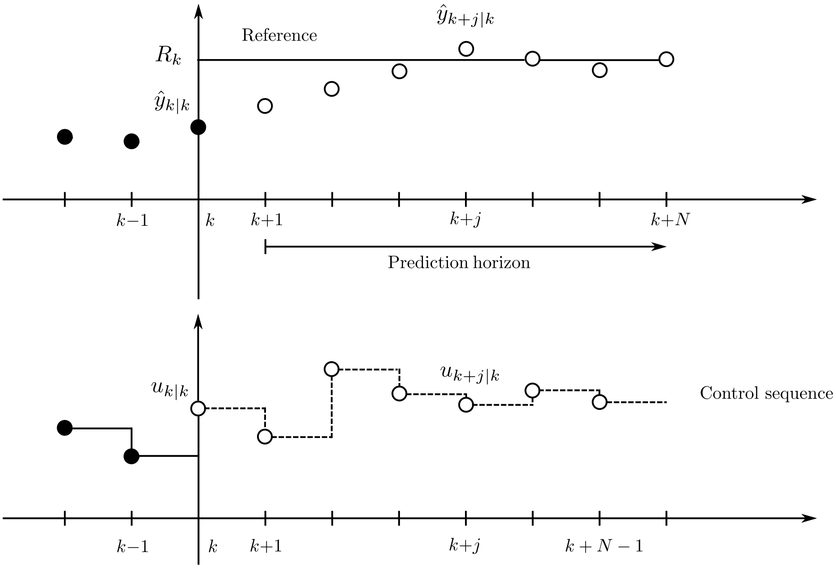

Figure 1.1: Basic principle of model predictive control.

1.1 Thesis Objectives

The aim of this thesis is to develop a procedure for control of industrial processes using linear MPC. As outlined in the introduction, the MPC is built up from a number of building blocks. The main objectives of the thesis is to analyze, develop and implement these building blocks and furthermore to combine the building blocks into a complete control methodology.

In order to achieve the main objectives of the thesis several topics in control theory need to be studied and sub-objectives need to be accomplished.

First of all a linear model of the industrial process to be controlled has to be obtained, for this reason different system identification methods will be considered. Secondly, different disturbance models should be investigated in order to obtain offset free control. Based on the identified linear model combined with a disturbance model, a state estimator should be employed and the MPC designed.

A closed-loop description for the unconstrained MPC should be derived and its properties analyzed. For the purpose of tuning the designed MPC different performance measures for the closed-loop will be investigated. Subsequently these measures are used to formulate an optimization based tuning approach.

To verify, test and illustrate the developed procedure a case study will be conducted. All the building blocks will be implemented in Matlab.

1.2 Thesis Structure and Overview

This section gives an overview of the content of the thesis and presents its structure. The thesis is organized as follows:

Chapter 2: Simulation Models and System Identification. This chapter presents some approaches to formulating simulation models for industrial processes and considers how to obtain linear models by system identification for the use in linear MPC.

Chapter 3: Disturbance Modeling. Here different disturbance models to ensure offset free control for the MPC are presented and analyzed.

Chapter 4: State Estimation. Introduces the concept of state estimation. A stationary Kalman filter is presented, for the estimation of the state of the linear model used by the MPC.

Chapter 5: Model Predictive Control. An MPC is set up based on the Kalman filter model and the MPC regulation problem is transformed into a convex quadratic optimization problem. For the unconstrained MPC the convex quadratic optimization problem is solved explicitly and an optimal linear control law is derived. For constrained MPC different cases are stated.

Chapter 6: Closed-loop Analysis. In this chapter a state-space model for the closed-loop is derived. Furthermore, the closed-loop is analyzed with respect to stochastic and sensitivity properties.

Chapter 7: Tuning. This chapter presents different performance measures for the closed-loop control system. The measures are divided into three categories and subsequently analyzed and discussed. In this chapter an optimization based tuning approach is also formulated.

Chapter 8-11: Case Study. In these chapters a case study is presented. The case study illustrates how the developed procedure can be used to control an industrial process.

Chapter 12-13: Conclusion and Future Work. In these chapter the overall conclusions of the project are made and key findings are summarized. Lastly, some considerations regrading future work are given.

Chapter 2

Simulation Models and System Identification

The purpose of this chapter is to present how industrial processes can be modeled mathematically in order to obtain accurate models for simulation of the processes. Secondly, the chapter will consider how linear models of industrial processes can be obtained by system identification, for the use in MPC.

The chapter is organized as follows. In the first section non-linear process models are considered. The second section addresses how linear models can be obtained by linearization of non-linear process models. Finally, it is considered how a linear model can be identified from plant input-output data.

2.1 Introduction

Many industrial processes behave, in reality, in some complicated non-linear fashion. This implies that the task of obtaining models for such processes can be difficult, very time consuming and not economically feasible.

However, it is in general of interest to have accurate models for simulation of industrial processes. The main reason for this, in connection with process control, is that accurate simulation models can be used as representatives of real life industrial processes. Furthermore, the simulation models can be used to test the performance of the designed model predictive controllers.

One of the main building blocks of a model predictive controller is a model of the process to be controlled. According to [Zhu06], between $70 - 80\%$ of the time used when implementing an MPC system in the industry, is spent on model development. This underlines the importance of the modeling aspect in connection with MPC.

The purpose of controlling an industrial process is typically to keep the controlled variables of the process within some desired operation range. Despite the fact that the industrial process behaves non-linear in the entire range of the process, the behavior within the desired operation range can often be accurately described by a linear model. This is what the MPC takes advantage of.

2.2 Non-Linear Process Models (Simulation Models)

For some processes, a non-linear simulation model of the plant can be derived by *first-principles*. This means that the equations governing the process are obtained from the underlying physical laws of the process. However, for very complicated processes this approach might be infeasible due to mathematical difficulties but also economical considerations. The processes that can be modeled in this way can often be described by implicit equations of the form [Mac02]

$$F(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)) = \mathbf{0} \tag{2.1}$$

$$\boldsymbol{y}(t) = g(\boldsymbol{x}(t)) \tag{2.2}$$

$$\boldsymbol{z}(t) = h(\boldsymbol{x}(t)) \tag{2.3}$$

where $\boldsymbol{x}(t)$ is the state of the process, $\dot{\boldsymbol{x}}(t)$ is the derivative of the state, that is the time evolution of the state. $\boldsymbol{u}(t)$ and $\boldsymbol{d}(t)$ denote the control input and the disturbance to the process, respectively. Here $F(\cdot)$ denotes some general implicit functional relation. Furthermore, $\boldsymbol{y}(t)$ and $\boldsymbol{z}(t)$ denote the measurement and output of the process, respectively.

Some of these models allow for an explicit description. The resulting model can often be described by a set of first-order non-linear differential equations as

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)) \tag{2.4}$$
$$\boldsymbol{y}(t) = g(\boldsymbol{x}(t)) \tag{2.5}$$
$$\boldsymbol{z}(t) = h(\boldsymbol{x}(t)) \tag{2.6}$$

where $f(\cdot)$ is some non-linear (vector) function describing the evolution of the process state and $\boldsymbol{y}(t)$ and $\boldsymbol{z}(t)$ denote the measurement and output of the process, respectively. Furthermore $g(\cdot)$ and $h(\cdot)$ are functions that relate the process state to the measurement and the output. It is assumed that these functions only depend on the process state.

Both the description in (2.1)-(2.3) and the model (2.4)-(2.6) are deterministic systems. To represent more realistic models, also the situation where an uncertain initial state, process noise, and measurement noise are present, can be considered. In this case (2.4) is reformulated as a Stochastic Differential Equation (SDE) by

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))dt + \sigma(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))d\boldsymbol{w}(t) \tag{2.7}$$

where the diffusion term is a standard Wiener process (Brownian motion)

$$d\boldsymbol{w}(t) \sim \mathcal{N}_{iid}(\boldsymbol{0}, \boldsymbol{I}dt) \tag{2.8}$$

The initial state is assumed to be normally distributed

$$\boldsymbol{x}(t_0) \sim \mathcal{N}(\bar{\boldsymbol{x}}_0, \boldsymbol{P}_0) \tag{2.9}$$

where $\bar{\boldsymbol{x}}_0$ is the expected initial state and $\boldsymbol{P}_0$ is the associated initial covariance. When the process is modeled as an SDE, both the state and output of the system are sequences of random variables (stochastic processes). The output of the system is, in the stochastic case, also expressed as a function relation of the state

$$\boldsymbol{z}(t) = h(\boldsymbol{x}(t)) \tag{2.10}$$

The measurement of the system is modeled as the output corrupted by measurement noise

$$\boldsymbol{y}(t_k) = \boldsymbol{z}(t_k) + \boldsymbol{v}(t_k) \tag{2.11}$$

where the measurement noise is assumed to be Gaussian white noise $\boldsymbol{v}(t_k) \sim \mathcal{N}_{iid}(\mathbf{0}, \boldsymbol{R})$. It should be emphasized that the measurement is obtained in discrete-time, indicated by t_k. The sample instances are given by

$$t_k = t_0 + kT_s, \quad k = 0, 1, 2, \ldots \tag{2.12}$$

where T_s is the sample time.
The theory of stochastic differential equations is outside the scope of this thesis and only the basic formulation will be considered. The reader is referred to [Øks00] for further details.

For this thesis, no real life industrial process is available. Therefore the real processes will be represented by a modified version of (2.4)-(2.6). The modification will be considered in the case study.

Remark 1 *It should be emphasized that no matter how the real process is modeled, the model will always only provide an approximation to the real process.*

2.3 Linear Process Models (Controller Models)

In order to control an industrial process by linear MPC, a linear model of the process is needed. This model can be obtained mainly by two general approaches. The approaches are

- Linearization of a non-linear process model
- System identification from plant input-output data

It should be stressed that linearization of a non-linear process model requires that the parameters of the model have been identified (in addition to the fact that a non-linear process model has been derived). This is in general not a trivial task to conduct and this is also one of the reasons why linearization is rarely used in practice.

2.3.1 Linearization of a non-linear process model

In the situation where a non-linear process model is available, a linear model can be obtained by linearization of the non-linear model, around a steady state. This approach can be applied to both (2.4)-(2.6) and (2.7)-(2.11). In the following, the linearization approach will be illustrated for (2.4)-(2.6).

Given $\boldsymbol{u}(t) = \boldsymbol{u}_s$ and $\boldsymbol{d}(t) = \boldsymbol{d}_s$, the steady state $\boldsymbol{x}_s$ of (2.4) can be determined by solving the equation

$$f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s) = \boldsymbol{0} \tag{2.13}$$

From the steady state, the steady state measurement and output can be computed as

$$\boldsymbol{y}_s = g(\boldsymbol{x}_s) \tag{2.14}$$

$$\boldsymbol{z}_s = h(\boldsymbol{x}_s) \tag{2.15}$$

Next a first-order Taylor expansion is performed on f, around the point $(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s)$

$$\begin{aligned} f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)) \approx f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s) &+ \left(\frac{\partial f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s)}{\partial \boldsymbol{x}}\right)(\boldsymbol{x}(t) - \boldsymbol{x}_s) \\ &+ \left(\frac{\partial f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s)}{\partial \boldsymbol{u}}\right)(\boldsymbol{u}(t) - \boldsymbol{u}_s) \\ &+ \left(\frac{\partial f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s)}{\partial \boldsymbol{d}}\right)(\boldsymbol{d}(t) - \boldsymbol{d}_s) \end{aligned} \tag{2.16}$$

Denoting the deviation variables by

$$\boldsymbol{X}(t) = \boldsymbol{x}(t) - \boldsymbol{x}_s \tag{2.17}$$

$$\boldsymbol{U}(t) = \boldsymbol{u}(t) - \boldsymbol{u}_s \tag{2.18}$$

$$\boldsymbol{D}(t) = \boldsymbol{d}(t) - \boldsymbol{d}_s \tag{2.19}$$

and the Jacobian matrices by

$$\boldsymbol{A}_c = \frac{\partial f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s)}{\partial \boldsymbol{x}} \tag{2.20}$$

$$\boldsymbol{B}_c = \frac{\partial f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s)}{\partial \boldsymbol{u}} \tag{2.21}$$

$$\boldsymbol{E}_c = \frac{\partial f(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{d}_s)}{\partial \boldsymbol{d}} \tag{2.22}$$

the first-order Taylor expansion can be expressed as

$$f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)) \approx \boldsymbol{A}_c \boldsymbol{X}(t) + \boldsymbol{B}_c \boldsymbol{U}(t) + \boldsymbol{E}_c \boldsymbol{D}(t) \tag{2.23}$$

The time derivative of the deviation state $\boldsymbol{X}(t)$ is

$$\dot{\boldsymbol{X}}(t) = \frac{d}{dt}\boldsymbol{X}(t) = \frac{d}{dt}(\boldsymbol{x}(t) - \boldsymbol{x}_s) = \frac{d\boldsymbol{x}(t)}{dt} - \frac{d\boldsymbol{x}_s}{dt} = \frac{d\boldsymbol{x}(t)}{dt} - \boldsymbol{0} = \dot{\boldsymbol{x}}(t) \tag{2.24}$$

The same steps can be conducted for the measurement and the output. Denoting the deviation variables by

$$\boldsymbol{Y}(t) = \boldsymbol{y}(t) - \boldsymbol{y}_s \tag{2.25}$$
$$\boldsymbol{Z}(t) = \boldsymbol{z}(t) - \boldsymbol{z}_s \tag{2.26}$$

the measurement and the output can be expressed as

$$\begin{aligned} \boldsymbol{y}(t) = g(\boldsymbol{x}(t)) &\approx g(\boldsymbol{x}_s) + \left(\frac{\partial g(\boldsymbol{x}_s)}{\partial \boldsymbol{x}}\right)(\boldsymbol{x}(t) - \boldsymbol{x}_s) \\ &= \boldsymbol{y}_s + \boldsymbol{C}_c \boldsymbol{X}(t) \end{aligned} \tag{2.27}$$

and

$$\begin{aligned} \boldsymbol{z}(t) = h(\boldsymbol{x}(t)) &\approx h(\boldsymbol{x}_s) + \left(\frac{\partial h(\boldsymbol{x}_s)}{\partial \boldsymbol{x}}\right)(\boldsymbol{x}(t) - \boldsymbol{x}_s) \\ &= \boldsymbol{z}_s + \boldsymbol{C}_{cz} \boldsymbol{X}(t) \end{aligned} \tag{2.28}$$

The continuous-time linearized model can finally be expressed as

$$\dot{\boldsymbol{X}}(t) = \boldsymbol{A}_c \boldsymbol{X}(t) + \boldsymbol{B}_c \boldsymbol{U}(t) + \boldsymbol{E}_c \boldsymbol{D}(t) \tag{2.29}$$
$$\boldsymbol{Y}(t) = \boldsymbol{C}_c \boldsymbol{X}(t) \tag{2.30}$$
$$\boldsymbol{Z}(t) = \boldsymbol{C}_{cz} \boldsymbol{X}(t) \tag{2.31}$$

The non-linear model is linearized at a steady state. At the steady state the linearized model is an approximation of the non-linear model in the neighborhood of the steady state. The linearized model expresses this neighborhood in terms of the deviation variables.

2.3.2 System Identification from Plant Input-Output Data

In many applications a non-linear model of the process is not available and the linearization approach can therefore not be used.

The second approach is concerned with system identification from observed data. The general system identification procedure mainly consists of the following building blocks [Lju99]

- Experimental design.
- A set of input-output data.
- A set of models, defining the model structure.
- A method for estimating the parameters of the model, based on the data.
- Validation of the model.

In this approach, tests are conducted directly on the actual process. In these tests, known input signals are used to excite the process and the resulting outputs of the process are measured and recorded. This results in a set of input-output data of the *true* process. The idea is then that a linear model of the process can be obtained using different techniques from the field of system identification, on the input-output data. There exist many different techniques and there is extensive literature on the topic of system identification. The reader is referred to [Lju99] for further details.

Some commonly used input signals [Mac02], [Pou07] are

- Step signals of different magnitudes.
- Sine waves of different amplitudes and frequencies.
- Pseudo-Random or Pseudo-Random Binary Signals (PRBS).

In connection with MPC especially input-output model structures such as auto regressive models with exogenous inputs (ARX) and auto regressive moving average models with exogenous inputs (ARMAX) are of interest. One of the reasons for this is that these input-output models may be realized as state-space models in innovation form [JHR11], which fit the framework of MPC well.

The state-space model in innovation form is given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{K}\boldsymbol{\varepsilon}_k \tag{2.32}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{\varepsilon}_k \tag{2.33}$$

where $\boldsymbol{\varepsilon}_k \sim \mathcal{N}_{iid}(\boldsymbol{0}, \boldsymbol{R}_\varepsilon)$.

2.3.2.1 System Identification using Step Tests

An identification method commonly used to obtain a linear model of a process, is the step test [Zhu01]. In this method, a step change is introduced in the manipulated variable (MV) of the real process and the response of the controlled variable (CV) is recorded and visualized.

For MIMO systems, a step change is introduced in one of the manipulated variables while the rest are kept constant and the responses of all the controlled variables are recorded. The procedure is conducted systematically for all the manipulated variables. The reason for only changing one MV at that time is, to ensure that the results are uncorrelated.

The step test should ideally be started from some steady state of the process, representing the desired operation point (range) for the process. The step should furthermore be applied for a sufficiently long time, until some clear step responses are seen and the process has reached a new steady state.

Since the behavior of the real process is generally non-linear, the size of the step should be chosen carefully in order to both observe a clear response and not violate any constraints of the process and/or not upset the process operation excessively.

The input-output data obtained from the step tests are then fitted to an appropriate model structure. The standard model structures used are either first-order plus dead time (FOPDT) models given by

$$G_{ij}(s) = \frac{K_{ij}}{\tau_{ij}s + 1}e^{-\theta_{ij}s} \tag{2.34}$$

or second-order plus dead time (SOPDT) models given by

$$G_{ij}(s) = \frac{K_{ij}}{(\tau_{ij}^{(1)}s+1)(\tau_{ij}^{(2)}s+1)}e^{-\theta_{ij}s} \tag{2.35}$$

where $i = 1, 2, \dots, n_y$ and $j = 1, 2, \dots, n_u$ denotes the number of MVs and CVs, respectively. For both types of models K_{ij} is the gain, τ_{ij} is the time constant and θ_{ij} is the delay.

The resulting input-output model can then be expressed as

$$\boldsymbol{Y}(s) = \boldsymbol{G}(s)\boldsymbol{U}(s) \tag{2.36}$$

where $\boldsymbol{Y}(s) = [Y_1(s)\ \ Y_2(s)\ \ \dots\ \ Y_{n_y}(s)]^T$, $\boldsymbol{U}(s) = [U_1(s)\ \ U_2(s)\ \ \dots\ \ U_{n_u}(s)]^T$ and

$$\boldsymbol{G}(s) = \begin{bmatrix} G_{11}(s) & G_{12}(s) & \dots & G_{1n_u}(s) \\ G_{21}(s) & G_{22}(s) & \dots & G_{2n_u}(s) \\ \vdots & \vdots & \ddots & \vdots \\ G_{n_y1}(s) & G_{n_y2}(s) & \dots & G_{n_yn_u}(s) \end{bmatrix} \tag{2.37}$$

The input-output model may subsequently be realized as a discrete-time deterministic state-space model given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k \tag{2.38}$$
$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}\boldsymbol{u}_k \tag{2.39}$$

This is in practice done using the MATLAB function `mimoctf2dss` [Jør04], which provides a minimal realization of (2.36). The realization is conducted by computing the impulse response of the transfer function and doing a balanced realization from the Hankel matrix of the impulse response matrices.

2.4 Summary

In this chapter different models for the simulation of industrial processes have been presented. Both a purely deterministic and a stochastic formulation was considered. The chapter has also illustrated how linear models of industrial processes can be obtained by system identification for the use in MPC. In particular, it was presented how step tests can be conducted and used for system identification of first and second-order input-output models.

Chapter 3

Disturbance Modeling

The main purpose of this chapter is to describe different disturbance models in order to achieve offset free control for model predictive controllers.

The situation where an unmeasured step disturbance is present in the plant, will result in an offset when the process is controlled by an MPC which is based on a model where the disturbances are not included [PR03]. Therefore the model used by the controller is augmented with a disturbance model.

The chapter is organized as follows. In the first part of the chapter the settings and assumptions for the disturbance modeling set up are introduced. The following parts of the chapter describe the different disturbance models.

3.1 Introduction

It is assumed that the model of the plant used by the controller can be described by the discrete-time linear stochastic state-space model given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{G}\boldsymbol{w}_k \tag{3.1}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k \tag{3.2}$$

$$\boldsymbol{z}_k = \boldsymbol{C}_z\boldsymbol{x}_k \tag{3.3}$$

where $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ is the state vector, $\boldsymbol{u}_k \in \mathbb{R}^{n_u}$ is the control input, $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$ is the measurement, $\boldsymbol{z}_k \in \mathbb{R}^{n_z}$ is the output and $\boldsymbol{w}_k \in \mathbb{R}^{n_w}$, $\boldsymbol{v}_k \in \mathbb{R}^{n_y}$ are process and measurement noise, respectively.

The dimensions of the corresponding matrices are $\boldsymbol{A} \in \mathbb{R}^{n_x \times n_x}$, $\boldsymbol{B} \in \mathbb{R}^{n_x \times n_u}$, $\boldsymbol{G} \in \mathbb{R}^{n_x \times n_w}$, $\boldsymbol{C} \in \mathbb{R}^{n_y \times n_x}$ and $\boldsymbol{C}_z \in \mathbb{R}^{n_z \times n_x}$. Here the pair $(\boldsymbol{A}, \boldsymbol{C})$ is assumed to be detectable and the pair $(\boldsymbol{A}, \boldsymbol{B})$ assumed to be stabilizable.

The process noise and measurement noise are assumed to be identically independently normally distributed (*iid*) as

$$\begin{bmatrix} \boldsymbol{w}_k \\ \boldsymbol{v}_k \end{bmatrix} \sim \mathcal{N}_{iid}\left(\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}, \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{S} \\ \boldsymbol{S}^T & \boldsymbol{R} \end{bmatrix} \right) \tag{3.4}$$

where $\boldsymbol{Q}$ is the covariance matrix of $\boldsymbol{w}_k$ and $\boldsymbol{R}$ is the covariance matrix of $\boldsymbol{v}_k$. Furthermore, $\boldsymbol{S}$ is the cross-covariance between the process noise $\boldsymbol{w}_k$ and the measurement noise $\boldsymbol{v}_k$.

It is assumed that the number of measured outputs n_y is equal to the number of controlled outputs n_z and furthermore that the set of measured outputs are the same as the controlled outputs ($\boldsymbol{C} = \boldsymbol{C}_z$).

3.2 Deterministic-Stochastic Model

The modeling of an unmeasured disturbance using a deterministic-stochastic model is done, by first setting up a deterministic model describing the dynamic behavior of the system to be controlled. Secondly a stochastic model describing the unmeasured disturbance is constructed and then these are combined.

The deterministic model is, in general, given by

$$\boldsymbol{x}^d_{k+1} = \boldsymbol{A}_d\boldsymbol{x}^d_k + \boldsymbol{B}_d\boldsymbol{u}_k \tag{3.5}$$

$$\boldsymbol{y}^d_k = \boldsymbol{C}_d\boldsymbol{x}^d_k \tag{3.6}$$

where $\boldsymbol{A}_d$, $\boldsymbol{B}_d$ and $\boldsymbol{C}_d$ are assumed to be known matrices (obtained by linearization of a non-linear plant, estimated using identification procedures or as a realization of a transfer function). Here the d indicates deterministic.

In this setup each disturbance is modeled as a first-order ARMA-process

$$\varepsilon_{i,k} = \frac{1-\alpha_i q^{-1}}{1-q^{-1}} e_{i,k}, \quad i = 1, 2, \dots, n_y \tag{3.7}$$

where $e_{i,k} \sim \mathcal{N}_{iid}(0, \sigma_e^2)$. This corresponds to a combination of white and integrated white noise [HPJJ12]. Here n_y refers to the number of controlled measurements, for which offset free control is desired.

An ARMA-process can be realized as a discrete-time state-space, in observer canonical form, as

$$\boldsymbol{x}_{k+1}^s = \boldsymbol{x}_k^s + \boldsymbol{K}_s \boldsymbol{e}_k \tag{3.8}$$

$$\boldsymbol{y}_k^s = \boldsymbol{x}_k^s + \boldsymbol{e}_k \tag{3.9}$$

where

$$\boldsymbol{K}_s = \begin{bmatrix} 1-\alpha_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & 1-\alpha_{n_y} \end{bmatrix} \tag{3.10}$$

Here the s indicates stochastic. Now, since both the deterministic and the stochastic models are linear, they can be combined into a single model as follows. Let the state vector of the combined model be defined as

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{x}_k^d \\ \boldsymbol{x}_k^s \end{bmatrix} \tag{3.11}$$

and the measurement of the combined model be

$$\boldsymbol{y}_k = \boldsymbol{y}_k^d + \boldsymbol{y}_k^s \tag{3.12}$$

The combined model can be expressed as

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{K}\boldsymbol{e}_k \tag{3.13}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{e}_k \tag{3.14}$$

where

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_d & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_d \\ \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{K} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{K}_s \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} \boldsymbol{C}_d & \boldsymbol{I} \end{bmatrix} \tag{3.15}$$

and $\boldsymbol{e}_k \sim \mathcal{N}_{iid}(\boldsymbol{0}, \boldsymbol{Q}_e)$. It is noted that the combined state-space model in (3.13)-(3.14) is in innovation form [HPJJ12].

Using this model, the state and disturbance are then estimated from the plant measurement $\boldsymbol{y}_k$ by means of a steady-state Kalman filter, which takes a particularly simple form for a state-space model in innovation form [Pou07].

3.3 MISO ARX Model

A MIMO (multiple-input multiple-output) model can be represented as n_y (number of outputs) MISO (multiple-input single-output) models. Each MISO system relates all the inputs of the MIMO model to a single output.

The idea is to first model each output as a MISO ARX model

$$A_i(q^{-1})y_{i,k} = \sum_{j=1}^{n_u} B_{ij}(q^{-1})u_{i,k} + \varepsilon_{i,k}, \quad i = 1, 2, \dots, n_y \tag{3.16}$$

where

$$A_i(q^{-1}) = 1 + a_{i,1}q^{-1} + a_{i,2}q^{-2} + \dots + a_{i,n}q^{-n} \tag{3.17}$$
$$B_{ij}(q^{-1}) = b_{ij,1}q^{-1} + b_{ij,2}q^{-2} + \dots + b_{ij,n}q^{-n} \tag{3.18}$$

are polynomials in the backshift operator of order n and $\varepsilon_{i,k} \sim \mathcal{N}_{iid}(0, \sigma_\varepsilon^2)$. Secondly, the noise model is extended such that offset free control is achieved, in the case where an unmeasured step disturbance enters the system.

In this setup (like in the deterministic-stochastic model setup) the disturbance is modeled as a first-order ARMA-process

$$\varepsilon_{i,k} = \frac{1 - \alpha_i q^{-1}}{1 - q^{-1}} e_{i,k}, \quad i = 1, 2, \dots, n_y \tag{3.19}$$

where α_i is a tuning parameter of the disturbance model and is in the interval $\alpha_i \in [0; 1]$.

When $\alpha_i = 0$, (3.19) becomes

$$\varepsilon_{i,k} = \frac{1}{1 - q^{-1}} e_{i,k} \tag{3.20}$$

This corresponds to modeling the noise as integrated white noise. Furthermore this case tries to approximate the disturbance in one step that is as fast as possible.

When $\alpha_i = 1$, (3.19) becomes

$$\varepsilon_{i,k} = \frac{1-q^{-1}}{1-q^{-1}} e_{i,k} = e_{i,k} \tag{3.21}$$

which corresponds to no extension of the disturbance model and results in the original MISO ARX model. In this case it is not possible to achieve offset free control, since this description cannot reject a non-zero constant [HPJJ12].

The above cases constitute the limits of the disturbance model, which means that either the disturbance is not rejected at all or it is rejected as fast as possible at the expense of increased variance of the approximation.

The extended noise model (3.19) is then substituted into the MISO ARX description in (3.16)

$$\begin{aligned} A_i(q^{-1}) y_{i,k} &= \sum_{j=1}^{n_u} B_{ij}(q^{-1}) u_{i,k} + \varepsilon_{i,k} && (3.22) \\ &= \sum_{j=1}^{n_u} B_{ij}(q^{-1}) u_{i,k} + \frac{1-\alpha_i q^{-1}}{1-q^{-1}} e_{i,k} && (3.23) \end{aligned}$$

This model can be rewritten as

$$\bar{A}_i(q^{-1}) y_{i,k} = \sum_{j=1}^{n_u} \bar{B}_{ij}(q^{-1}) u_{i,k} + \bar{C}_i(q^{-1}) e_{i,k} \tag{3.24}$$

where the polynomials $\bar{A}_i(q^{-1})$, $\bar{B}_{ij}(q^{-1})$ and $\bar{C}_i(q^{-1})$ are

$$\begin{aligned} \bar{A}_i(q^{-1}) &= (1-q^{-1}) A_i(q^{-1}) && (3.25) \\ \bar{B}_{ij}(q^{-1}) &= (1-q^{-1}) B_{ij}(q^{-1}) && (3.26) \\ \bar{C}_i(q^{-1}) &= 1 - \alpha_i q^{-1} && (3.27) \end{aligned}$$

and $i = 1, 2, \dots, n_y$.

The model in (3.24) can be realized as the state-space model

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{K} e_k && (3.28) \\ y_k &= \boldsymbol{C}\boldsymbol{x}_k + e_k && (3.29) \end{aligned}$$

in observer canonical form with

$$\boldsymbol{A} = \begin{bmatrix} -\bar{a}_{i,1} & 1 & 0 & \dots & 0 \\ -\bar{a}_{i,2} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ -\bar{a}_{i,n-1} & 0 & 0 & \dots & 1 \\ -\bar{a}_{i,n} & 0 & 0 & \dots & 0 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \bar{B}_{i1}^T & \bar{B}_{i2}^T & \dots & \bar{B}_{in_u}^T \end{bmatrix} \tag{3.30}$$

$$\boldsymbol{K} = \begin{bmatrix} \alpha_i - \bar{a}_{i,1} \\ -\bar{a}_{i,2} \\ \vdots \\ -\bar{a}_{i,n} \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix} \tag{3.31}$$

and $e_k \sim \mathcal{N}_{iid}(0, \sigma_e^2)$. Each output $y_{i,k}$ will result in a state-space model description (3.28)-(3.29). A state-space model describing all the outputs $i = 1, 2, \dots, n_y$, can be obtained by combining the individual models.

3.4 Unstructured Disturbance Model

In the unstructured disturbance model, the unmeasured disturbance is modeled as integrated white noise, that is

$$\boldsymbol{d}_{k+1} = \boldsymbol{d}_k + \boldsymbol{\xi}_k \tag{3.32}$$

where $\boldsymbol{d}_k \in \mathbb{R}^{n_d}$ and $\boldsymbol{\xi}_k \sim \mathcal{N}_{iid}(\boldsymbol{0}, \boldsymbol{Q}_\xi)$.

The model of the disturbance is then combined with the original model (3.1)-(3.3) with additional matrices, describing how the disturbance enters the state and the measurement. Let $\boldsymbol{B}_d \in \mathbb{R}^{n_x \times n_d}$ denote the matrix describing how the disturbance enters the state and $\boldsymbol{C}_d \in \mathbb{R}^{n_x \times n_d}$ through the measurement.

Combining the model in (3.1)-(3.2) and the disturbance description in (3.32), an augmented model is obtained. The augmented model is given by [RRQ09]

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{d}_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B}_d \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{d}_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{u}_k + \begin{bmatrix} \boldsymbol{G} & 0 \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_k \\ \boldsymbol{\xi}_k \end{bmatrix} \tag{3.33}$$

$$\boldsymbol{y}_k = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{C}_d \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{d}_k \end{bmatrix} + \boldsymbol{v}_k \tag{3.34}$$

where the noise now can be described by

$$\begin{bmatrix} \boldsymbol{w}_k \\ \boldsymbol{v}_k \\ \boldsymbol{\xi}_k \end{bmatrix} \sim \mathcal{N}_{iid}\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{S}^T & \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{Q}_\xi \end{bmatrix} \right) \tag{3.35}$$

In order to ensure offset free control using the augmented model, the matrices $\boldsymbol{B}_d$ and $\boldsymbol{C}_d$ have to satisfy some conditions related to detectability, the conditions are given in [PR03]. The number of disturbances n_d has to be chosen as

$n_d = n_y$ to ensure zero offset in all the controlled outputs [PR03].

In general the structure of the matrices $\boldsymbol{B}_d$ and $\boldsymbol{C}_d$ is unknown, since the disturbance is unknown. However, a common choice is to assume some simple structure of the disturbance [RRQ09].

Let

$$\boldsymbol{B}_d = \boldsymbol{B}, \quad \boldsymbol{C}_d = \boldsymbol{0} \tag{3.36}$$

then the disturbance is modeled as an input disturbance. While letting

$$\boldsymbol{B}_d = \boldsymbol{0}, \quad \boldsymbol{C}_d = \boldsymbol{I} \tag{3.37}$$

models the disturbance as being an output disturbance. The matrices can also be chosen such that the disturbance $\boldsymbol{d}_k$ is modeled as a combination of input and output disturbances.

Using the augmented model, the state and the disturbance are then estimated from the plant measurements $\boldsymbol{y}_k$ by means of a steady-state Kalman filter. The steady-state Kalman filter is described in Chapter 4.

3.5 Summary

In this chapter the concept of disturbance modeling and rejection has been introduced. To reject the disturbance and obtain offset free control, three classic disturbance models have been presented. Each of the models have been described and analyzed.

Chapter 4

State Estimation

The purpose of this chapter is to introduce the concept of state estimation and furthermore to develop the filter and predictor for the model used by the MPC. State estimation is central to MPC, since the estimator incorporates feedback into the model predictive controller.
The chapter is organized as follows. The first section introduces the subject of state estimation. In the second and third sections the Kalman filter and the stationary Kalman filter for stochastic LTI state-space models will be considered.

4.1 Introduction

For most industrial processes and their corresponding linear models, the process state $\boldsymbol{x}_k$ is unknown and cannot be measured directly. Therefore the process state needs to be estimated, this is done from process measurements which are somehow related to the state. It is important to have an estimate of the process state as it is used in the MPC to predict the future dynamic behavior of the system.

It is assumed that the model of the plant used by the controller can be described by a discrete-time linear stochastic state-space model given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{G}\boldsymbol{w}_k \tag{4.1}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k \tag{4.2}$$

where $\boldsymbol{x}_k$ is the state, $\boldsymbol{u}_k$ is the control input and $\boldsymbol{y}_k$ is the measurement of the system. Furthermore, $\boldsymbol{w}_k$ is process noise and $\boldsymbol{v}_k$ is measurement noise.

The process and measurement noise are assumed to be identically independently normally distributed (*iid*) as

$$\begin{bmatrix} \boldsymbol{w}_k \\ \boldsymbol{v}_k \end{bmatrix} \sim \mathcal{N}_{iid}\left(\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}, \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{S} \\ \boldsymbol{S}^T & \boldsymbol{R} \end{bmatrix} \right) \tag{4.3}$$

where $\boldsymbol{Q}$ is the covariance matrix of $\boldsymbol{w}_k$ and $\boldsymbol{R}$ is the covariance matrix of $\boldsymbol{v}_k$. Furthermore, $\boldsymbol{S}$ is the cross-covariance between the process noise $\boldsymbol{w}_k$ and the measurement noise $\boldsymbol{v}_k$.

The covariance matrices $\boldsymbol{Q}$, $\boldsymbol{R}$, and the cross-covariance matrix $\boldsymbol{S}$ are positive definite and symmetric.
In the case where the process noise and the measurement noise are uncorrelated, the cross-covariance is $\boldsymbol{S} = \boldsymbol{S}^T = \boldsymbol{0}$. This is the most common situation, however, there exist systems where this is not true. This is the case for state-space models in innovation form, for these models there is a perfect correlation between the process and measurement noise.

The initial state of the system is unknown, but it is assumed that its distribution is known and given by

$$\boldsymbol{x}_0 \sim \mathcal{N}(\bar{\boldsymbol{x}}_0, \boldsymbol{P}_0) \tag{4.4}$$

and is independent of process and measurement noise.

It is in this chapter assumed that the control inputs $\boldsymbol{u}_k$, $k = 1, 2, \ldots$ are deterministic and known.

4.2 Kalman Filtering

In the discrete-time interval $[0, N]$ the measurements $\boldsymbol{y}_k,\ k = 0, 1, \ldots, N$ are recorded. Let $\mathcal{Y}_N$ denote the set of measurements from discrete-time $k = 0$ to discrete-time $k = N$

$$\mathcal{Y}_N = \{\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_N\} \tag{4.5}$$

The state estimation problem is concerned with obtaining an estimate $\hat{\boldsymbol{x}}_{k|k}$ of the state $\boldsymbol{x}_k$, based on the set of measurement data $\mathcal{Y}_k$ available at time k.

To obtain the state estimate $\hat{\boldsymbol{x}}_{k|k}$, the Kalman filter is used. The Kalman filter is a recursive approach to state estimation due to R.E Kalman [Kal60]. The Kalman filter seeks to minimize the sum of squared errors between the *true* state $\boldsymbol{x}_k$ and estimated state $\hat{\boldsymbol{x}}_{k|k}$. The filter is said to be optimal under the following assumptions: the estimation model is identical to the true system, the process and measurement noise is white and the covariances of the noise sources are known exactly.

The Kalman filter for the discrete-time stochastic state-space is given by two sets of recursions, a time update (also referred to as the one-step prediction) and a data update (the filter).

The prediction recursion corresponds to the computation of conditional expectations of the state $\boldsymbol{x}_{k+1}$ and associated covariances, given the data set $\mathcal{Y}_k$

$$\hat{\boldsymbol{x}}_{k+1|k} = E\{\boldsymbol{x}_{k+1}|\mathcal{Y}_k\} \tag{4.6}$$

$$\boldsymbol{P}_{k+1|k} = Var\{\boldsymbol{x}_{k+1}|\mathcal{Y}_k\} \tag{4.7}$$

The filter recursion corresponds to the computation of conditional expectations [JHR11, Pou07] of the state $\boldsymbol{x}_k$ and the process-noise $\boldsymbol{w}_k$, given the data set $\mathcal{Y}_k$

$$\hat{\boldsymbol{x}}_{k|k} = E\{\boldsymbol{x}_k|\mathcal{Y}_k\} \tag{4.8}$$

$$\hat{\boldsymbol{w}}_{k|k} = E\{\boldsymbol{w}_k|\mathcal{Y}_k\} \tag{4.9}$$

and associated covariances of the state estimate

$$\boldsymbol{P}_{k|k} = Var\{\boldsymbol{x}_k|\mathcal{Y}_k\} \tag{4.10}$$

4.3 The Discrete-Time Kalman Filter

The recursions of the Kalman filter for the discrete-time stochastic state-space (4.1)-(4.2), are given by [Jør04]

Filter

$$\boldsymbol{R}_{e,k} = \boldsymbol{C}\boldsymbol{P}_{k|k-1}\boldsymbol{C}^T + \boldsymbol{R} \tag{4.11}$$
$$\boldsymbol{K}_{fx,k} = \boldsymbol{P}_{k|k-1}\boldsymbol{C}^T\boldsymbol{R}_{e,k}^{-1} \tag{4.12}$$
$$\boldsymbol{K}_{fw,k} = \boldsymbol{S}\boldsymbol{R}_{e,k}^{-1} \tag{4.13}$$

$$\boldsymbol{e}_k = \boldsymbol{y}_k - \boldsymbol{C}\hat{\boldsymbol{x}}_{k|k-1} \tag{4.14}$$
$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx,k}\boldsymbol{e}_k \tag{4.15}$$
$$\hat{\boldsymbol{w}}_{k|k} = \boldsymbol{K}_{fw,k}\boldsymbol{e}_k \tag{4.16}$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_{fx,k}\boldsymbol{R}_{e,k}\boldsymbol{K}_{fx,k}^T \tag{4.17}$$
$$\boldsymbol{Q}_{k|k} = \boldsymbol{Q} - \boldsymbol{K}_{fw,k}\boldsymbol{R}_{e,k}\boldsymbol{K}_{fw,k}^T \tag{4.18}$$

One-Step Prediction (Time Update)

$$\hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} \tag{4.19}$$

$$\boldsymbol{P}_{k+1|k} = \boldsymbol{A}\boldsymbol{P}_{k|k}\boldsymbol{A}^T + \boldsymbol{G}\boldsymbol{Q}_{k|k}\boldsymbol{G}^T - \boldsymbol{A}\boldsymbol{K}_{fx,k}\boldsymbol{S}^T\boldsymbol{G}^T - \boldsymbol{G}\boldsymbol{S}\boldsymbol{K}_{fx,k}^T\boldsymbol{A}^T \tag{4.20}$$

The recursions are started at sample $k = 0$, where the initial covariance and state estimate are given by

$$\boldsymbol{P}_{0|-1} = \boldsymbol{P}_0 \tag{4.21}$$
$$\hat{\boldsymbol{x}}_{0|-1} = \bar{\boldsymbol{x}}_0 \tag{4.22}$$

By substitution of the filtered state covariance (4.17) and filtered process noise covariance (4.18) into (4.20), the recursion for the one-step predicted state covariance can be formulated as

$$\begin{aligned}\boldsymbol{P}_{k+1|k} = \boldsymbol{A}\left[\boldsymbol{P}_{k|k-1} - \boldsymbol{K}_{fx,k}\boldsymbol{R}_{e,k}\boldsymbol{K}_{fx,k}^T\right]\boldsymbol{A}^T + \boldsymbol{G}\left[\boldsymbol{Q} - \boldsymbol{K}_{fw,k}\boldsymbol{R}_{e,k}\boldsymbol{K}_{fw,k}^T\right]\boldsymbol{G}^T \\ - \boldsymbol{A}\boldsymbol{K}_{fx,k}\boldsymbol{S}^T\boldsymbol{G}^T - \boldsymbol{G}\boldsymbol{S}\boldsymbol{K}_{fx,k}^T\boldsymbol{A}^T\end{aligned}$$

4.4 The Stationary Kalman Filter

The stationary Kalman filter is obtained in the case where the covariance matrix $\boldsymbol{P}_{k|k-1}$ associated with the one-step state prediction becomes stationary. This can mathematically be formulated as

$$\boldsymbol{P} = \boldsymbol{P}_{k+1|k} = \boldsymbol{P}_{k|k-1}, \quad k \to \infty \tag{4.23}$$

The conditions under which the covariance matrix becomes stationary are given in [Jør04].

In this case the recursion for the covariance matrix becomes

$$\boldsymbol{P} = \boldsymbol{A}\boldsymbol{P}\boldsymbol{A}^T + \boldsymbol{G}\boldsymbol{Q}\boldsymbol{G}^T - (\boldsymbol{A}\boldsymbol{P}\boldsymbol{C}^T + \boldsymbol{G}\boldsymbol{S})(\boldsymbol{C}\boldsymbol{P}\boldsymbol{C}^T + \boldsymbol{R})^{-1}(\boldsymbol{A}\boldsymbol{P}\boldsymbol{C}^T + \boldsymbol{G}\boldsymbol{S})^T \tag{4.24}$$

which is known as the Discrete Algebraic Riccati Equation (DARE). The equation can be solved using the built-in MATLAB function `dare`.

In the stationary case, the filter gains for the state and process-noise are given by

$$\boldsymbol{R}_e = \boldsymbol{C}\boldsymbol{P}\boldsymbol{C}^T + \boldsymbol{R} \tag{4.25}$$

$$\boldsymbol{K}_{fx} = \boldsymbol{P}\boldsymbol{C}^T\boldsymbol{R}_e^{-1} \tag{4.26}$$

$$\boldsymbol{K}_{fw} = \boldsymbol{S}\boldsymbol{R}_e^{-1} \tag{4.27}$$

with the associated filtered covariance matrices

$$\boldsymbol{P}_f = \boldsymbol{P} - \boldsymbol{K}_{fx}\boldsymbol{R}_e\boldsymbol{K}_{fx}^T \tag{4.28}$$

$$\boldsymbol{Q}_f = \boldsymbol{Q} - \boldsymbol{K}_{fw}\boldsymbol{R}_e\boldsymbol{K}_{fw}^T \tag{4.29}$$

Since the filter gains are stationary, they can be computed offline reducing the computational complexity of estimating the current state $\boldsymbol{x}_k$.

The filter and time update for the stationary Kalman filter are given by

$$\boldsymbol{e}_k = \boldsymbol{y}_k - \boldsymbol{C}\hat{\boldsymbol{x}}_{k|k-1} \tag{4.30}$$

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx}\boldsymbol{e}_k \tag{4.31}$$

$$\hat{\boldsymbol{w}}_{k|k} = \boldsymbol{K}_{fw}\boldsymbol{e}_k \tag{4.32}$$

and

$$\hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} \tag{4.33}$$

It should be noted that the stationary covariance matrix $\boldsymbol{P}$ obtained by solving the Riccati equation in (4.24), depends on the *true* noise covariance matrices $\boldsymbol{Q}$, $\boldsymbol{R}$ and $\boldsymbol{S}$. However, it is not always the case that these are actually known.

Furthermore, it should also be noted that the Kalman filter depends on the matrices of the system in (4.1)-(4.2). However, since the system might need to be augmented in order to achieve offset free control, the model matrices are allowed to be different. The model matrices for this case are denoted $(\hat{\boldsymbol{A}}, \hat{\boldsymbol{B}}, \hat{\boldsymbol{G}}, \hat{\boldsymbol{C}})$.

4.4.1 Predictions Using The Stationary Kalman Filter

In order to predict the future dynamic behavior of the controlled process, the filtered state estimate $\hat{\boldsymbol{x}}_{k|k}$ is used as initial point for the future predictions. The one-step prediction is given by (4.33), where the filtered process-noise is included. To predict the state of the entire prediction horizon, it is first noticed that the process-noise estimate $\hat{\boldsymbol{w}}_{k+j|k} = \boldsymbol{0}$ for $j > 0$. This is the case due to the fact that $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are only correlated for the current k. Consequently the $(j+1)$-step state prediction may be computed by

$$\hat{\boldsymbol{x}}_{k+1+j|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k+j|k} + \boldsymbol{B}\boldsymbol{u}_{k+j|k}, \quad j = 1, 2, \ldots \tag{4.34}$$

4.4.2 Work Flow of the Stationary Kalman Filter

The recursions of the stationary Kalman filter are started at sample $k = 0$, where the initial state estimate and covariance are given by

$$\hat{\boldsymbol{x}}_{0|-1} = \bar{\boldsymbol{x}}_0 \tag{4.35}$$

$$\boldsymbol{P} = \boldsymbol{P}_0 \tag{4.36}$$

At each sample $k = 0, 1, 2, \ldots$ a new measurement $\boldsymbol{y}_k$ of the system becomes available to the state estimator, from this the innovation $\boldsymbol{e}_k$ is computed. The innovation is the error between the actual measurement $\boldsymbol{y}_k$ and the, by the state estimator, predicted measurement $\hat{\boldsymbol{y}}_{k|k-1}$

$$\boldsymbol{e}_k = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_{k|k-1}, \quad k = 0, 1, \ldots \tag{4.37}$$

The predicted measurement $\hat{\boldsymbol{y}}_{k|k-1}$, is a prediction of the actual measurement $\boldsymbol{y}_k$ at sample k, computed at sample $k-1$ and based on the measurement data set $\mathcal{Y}_{k-1}$. The predicted measurement is given by

$$\hat{\boldsymbol{y}}_{k|k-1} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k|k-1}, \quad k = 0, 1, \ldots \tag{4.38}$$

The predicted measurement $\hat{\boldsymbol{y}}_{k|k-1}$ is based on the one-step state prediction $\hat{\boldsymbol{x}}_{k|k-1}$. The one-step state prediction $\hat{\boldsymbol{x}}_{k|k-1}$, is a prediction of the (true) state $\boldsymbol{x}_k$ at sample k, computed at sample $k-1$ and based on the measurement data set $\mathcal{Y}_{k-1}$.

Using the innovation $\boldsymbol{e}_k$ and one-step state prediction $\hat{\boldsymbol{x}}_{k|k-1}$, the filtered state estimate is computed by

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx}\boldsymbol{e}_k \tag{4.39}$$

and the filtered process-noise estimate by

$$\hat{\boldsymbol{w}}_{k|k} = \boldsymbol{K}_{fw}\boldsymbol{e}_k \tag{4.40}$$

The Kalman filter is then prepared for the next process measurement by computing one-step prediction of the state. The one-step state prediction is given by

$$\hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} \tag{4.41}$$

4.5 Summary

In this chapter the basic idea of state estimation has been presented. It has been shown how an estimate of the process state can be obtained by use of the stationary Kalman filter. Also the assumption on the formulation of the filter with regard to optimality has been considered. Furthermore, it has been shown how the state estimate can be used to predict the future states of the process.

Chapter 5

Model Predictive Control

The main purpose of this chapter is to introduce the basic theory and formulation of model predictive control.

The chapter is organized as follows. The first part of the chapter introduces the state-space model that the MPC will be based on. The second part of the chapter deals with the formulation of the unconstrained MPC based on the state-space model, and illustrates how the MPC regulation problem can be formulated as a quadratic optimization problem.
For the unconstrained MPC the associated quadratic optimization problem can be solved explicitly. The third part derives a state-space model for the unconstrained controller.
The last part of the chapter considers the formulation of different constrained model predictive controllers and the quadratic optimization problems resulting from the constrained MPC regulation problems.

5.1 Introduction

The model of the plant used to predict the future dynamic evolution of the controlled system, is in this chapter assumed to be described by

$$\begin{aligned}
\hat{\boldsymbol{x}}_{k+1|k} &= \boldsymbol{A}\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{B}\boldsymbol{u}_{k|k} + \boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} && (5.1)\\
\hat{\boldsymbol{y}}_{k+1|k} &= \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1|k} && (5.2)\\
\hat{\boldsymbol{x}}_{k+1+j|k} &= \boldsymbol{A}\hat{\boldsymbol{x}}_{k+j|k} + \boldsymbol{B}\boldsymbol{u}_{k+j|k}, \qquad j = 1, 2, \ldots, N-1 && (5.3)\\
\hat{\boldsymbol{y}}_{k+1+j|k} &= \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1+j|k}, \qquad j = 1, 2, \ldots, N-1 && (5.4)
\end{aligned}$$

which is the Kalman filter model for the system model in (4.1)-(4.2), along with the corresponding $(j+1)$-step predictor, of the states $\hat{\boldsymbol{x}}_{k+1+j|k}$ and measurements $\hat{\boldsymbol{y}}_{k+1+j|k}$.

In the following sections, the model in (5.1)-(5.4) is used to develop unconstrained and constrained receding horizon optimal controllers.

5.2 Unconstrained MPC for State-Space Models

In this section, the basic setup of the unconstrained MPC, based on the state-space model (5.1)-(5.4) is treated.

First the MPC objective function is defined. This can be done in different ways depending on the purpose of controlling the process. A common requirement in the industrial process industry, is to have the controlled output of the process track a reference specified by the control operator. This is described by the tracking error, which is the difference between the predicted controlled output $\hat{\boldsymbol{y}}_{k+1+j|k}$ and the reference $\boldsymbol{r}_{k+1+j|k}$. Furthermore, in order not to use excessive control action a regularization term is also commonly added to the objective function. This is also the objective considered in this thesis.

Many processes are MIMO systems which means that the predicted measurement and control input usually are vectors, and therefore some norm is used to quantify the tracking error and the control action. Using the squared norm indicates that positive and negative deviation from the reference are equally undesirable and furthermore by weighting the norm, the individual elements can be treated differently. The objective function is defined as

$$\phi = \frac{1}{2}\sum_{j=0}^{N-1} \left\| \hat{\boldsymbol{y}}_{k+1+j|k} - \boldsymbol{r}_{k+1+j|k} \right\|_{\boldsymbol{Q}_y}^2 + \left\| \Delta \boldsymbol{u}_{k+j|k} \right\|_{\boldsymbol{S}_u}^2 \tag{5.5}$$

where the weighted norm is defined by

$$\|\boldsymbol{x}\|_{\boldsymbol{Q}}^2 = \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} \tag{5.6}$$

The objective function expresses the sum of the weighted squared norm of difference between the measurement prediction $\hat{\boldsymbol{y}}_{k+1+j|k}$ and the corresponding setpoint $\boldsymbol{r}_{k+1+j|k}$ and the weighted squared norm of the rate movement $\Delta\boldsymbol{u}_{k+j|k}$ for $j = 0, 1, \dots, N-1$. The objective function is subjected to the model dynamics described by (5.1)-(5.4).

The MPC problem can be expressed as

$$\begin{aligned}
\min_{\{\boldsymbol{u}_{k+j|k}\}_{j=0}^{N-1}} \quad & \phi = \frac{1}{2}\sum_{j=0}^{N-1}\left\|\hat{\boldsymbol{y}}_{k+1+j|k} - \boldsymbol{r}_{k+1+j|k}\right\|_{\boldsymbol{Q}_y}^2 + \left\|\Delta\boldsymbol{u}_{k+j|k}\right\|_{\boldsymbol{S}_u}^2 && (5.7)\\
s.t. \quad & \hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{B}\boldsymbol{u}_{k|k} + \boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} && (5.8)\\
& \hat{\boldsymbol{y}}_{k+1|k} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1|k} && (5.9)\\
& \hat{\boldsymbol{x}}_{k+1+j|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k+j|k} + \boldsymbol{B}\boldsymbol{u}_{k+j|k}, \qquad j = 1, 2, \dots, N-1 && (5.10)\\
& \hat{\boldsymbol{y}}_{k+1+j|k} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1+j|k}, \qquad j = 1, 2, \dots, N-1 && (5.11)
\end{aligned}$$

where N is the prediction horizon and $\boldsymbol{Q}_y$, $\boldsymbol{S}_u$ are weight matrices associated with the reference tracking and the input rate movement, respectively. These matrices are user-specified and can be regarded as tuning parameters for the MPC. How these weight matrices should be chosen is not obvious and different choices lead to different closed-loop performances. The tuning of the MPC will be considered in Chapter 7.

The structure of the weight matrices is of importance. From a theoretical point of view, the MPC objective function constitutes a weighted least-squares problem. This can give some indications on how the structure of the weight matrices should be chosen. In this thesis the weight matrices are assumed to be positive definite diagonal matrices.

In general the sum of the two terms in the objective function could be different. In this case the length of the reference tracking is referred to as the prediction horizon and the one on the control as the control horizon. In this thesis, this case is not considered and the two horizons are equal.

The MPC regulation problem can be transformed into a convex Quadratic Program (QP) by performing state elimination in (5.8)-(5.11). The states can be expressed as

$$\hat{\boldsymbol{x}}_{k+1+j|k} = \boldsymbol{A}^{j+1}\hat{\boldsymbol{x}}_{k|k} + \sum_{i=0}^{j} \boldsymbol{A}^{j-i}\boldsymbol{B}\boldsymbol{u}_{k+i|k} + \boldsymbol{A}^{j}\boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} \tag{5.12}$$

by successive substitution in (5.8) and (5.10). Under the assumption that the predicted measurements $\hat{\boldsymbol{y}}_{k+1+j|k}$ are computed by $\hat{\boldsymbol{y}}_{k+1+j|k} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1+j|k}$, it follows from (5.12) that $\hat{\boldsymbol{y}}_{k+1+j|k}$ can be written as

$$\begin{aligned}
\hat{\boldsymbol{y}}_{k+1+j|k} &= \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1+j|k} \\
&= \boldsymbol{C}\Big(\boldsymbol{A}^{j+1}\hat{\boldsymbol{x}}_{k|k} + \sum_{i=0}^{j} \boldsymbol{A}^{j-i}\boldsymbol{B}\boldsymbol{u}_{k+i|k} + \boldsymbol{A}^{j}\boldsymbol{G}\hat{\boldsymbol{w}}_{k|k}\Big) \\
&= \boldsymbol{C}\boldsymbol{A}^{j+1}\hat{\boldsymbol{x}}_{k|k} + \sum_{i=0}^{j} \boldsymbol{C}\boldsymbol{A}^{j-i}\boldsymbol{B}\boldsymbol{u}_{k+i|k} + \boldsymbol{C}\boldsymbol{A}^{j}\boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} \qquad (5.13)
\end{aligned}$$

Using vector notation (5.13) can be expressed (for $j = 0, 1, \dots, N-1$) as

$$\begin{bmatrix} \hat{\boldsymbol{y}}_{k+1|k} \\ \hat{\boldsymbol{y}}_{k+2|k} \\ \hat{\boldsymbol{y}}_{k+3|k} \\ \vdots \\ \hat{\boldsymbol{y}}_{k+N|k} \end{bmatrix} = \boldsymbol{\Phi}_x\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{\Phi}_w\hat{\boldsymbol{w}}_{k|k} + \boldsymbol{\Gamma} \begin{bmatrix} \boldsymbol{u}_{k|k} \\ \boldsymbol{u}_{k+1|k} \\ \boldsymbol{u}_{k+2|k} \\ \vdots \\ \boldsymbol{u}_{k+N-1|k} \end{bmatrix} \tag{5.14}$$

where $\boldsymbol{\Phi}_x$, $\boldsymbol{\Phi}_w$, $\boldsymbol{\Gamma}$ are given by

$$\boldsymbol{\Phi}_x = \begin{bmatrix} \boldsymbol{C}\boldsymbol{A} \\ \boldsymbol{C}\boldsymbol{A}^2 \\ \boldsymbol{C}\boldsymbol{A}^3 \\ \vdots \\ \boldsymbol{C}\boldsymbol{A}^N \end{bmatrix}, \quad \boldsymbol{\Phi}_w = \begin{bmatrix} \boldsymbol{C}\boldsymbol{G} \\ \boldsymbol{C}\boldsymbol{A}\boldsymbol{G} \\ \boldsymbol{C}\boldsymbol{A}^2\boldsymbol{G} \\ \vdots \\ \boldsymbol{C}\boldsymbol{A}^{N-1}\boldsymbol{G} \end{bmatrix}, \quad \boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{H}_1 & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{H}_2 & \boldsymbol{H}_1 & \dots & \boldsymbol{0} \\ \boldsymbol{H}_3 & \boldsymbol{H}_2 & \dots & \boldsymbol{0} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{H}_N & \boldsymbol{H}_{N-1} & \dots & \boldsymbol{H}_1 \end{bmatrix}$$

The elements $\boldsymbol{H}_i$ of the matrix $\boldsymbol{\Gamma}$ are given by

$$\boldsymbol{H}_i = \boldsymbol{C}\boldsymbol{A}^{i-1}\boldsymbol{B}, \quad i = 1, 2, \dots, N. \tag{5.15}$$

These are known as the impulse response coefficients (Markov parameters) of the system.

Defining

$$\boldsymbol{Y}_k = \begin{bmatrix} \hat{\boldsymbol{y}}_{k+1|k} \\ \hat{\boldsymbol{y}}_{k+2|k} \\ \hat{\boldsymbol{y}}_{k+3|k} \\ \vdots \\ \hat{\boldsymbol{y}}_{k+N|k} \end{bmatrix}, \quad \boldsymbol{R}_k = \begin{bmatrix} \boldsymbol{r}_{k+1|k} \\ \boldsymbol{r}_{k+2|k} \\ \boldsymbol{r}_{k+3|k} \\ \vdots \\ \boldsymbol{r}_{k+N|k} \end{bmatrix}, \quad \boldsymbol{U}_k = \begin{bmatrix} \boldsymbol{u}_{k|k} \\ \boldsymbol{u}_{k+1|k} \\ \boldsymbol{u}_{k+2|k} \\ \vdots \\ \boldsymbol{u}_{k+N-1|k} \end{bmatrix} \tag{5.16}$$

and

$$\Delta\boldsymbol{U}_k = \begin{bmatrix} \Delta\boldsymbol{u}_{k|k} \\ \Delta\boldsymbol{u}_{k+1|k} \\ \Delta\boldsymbol{u}_{k+2|k} \\ \vdots \\ \Delta\boldsymbol{u}_{k+N-1|k} \end{bmatrix} \tag{5.17}$$

the measurement predictions can be expressed as

$$\boldsymbol{Y}_k = \boldsymbol{b}_k + \boldsymbol{\Gamma}\boldsymbol{U}_k \tag{5.18}$$

where $\boldsymbol{b}_k$ is

$$\boldsymbol{b}_k = \boldsymbol{\Phi}_x\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{\Phi}_w\hat{\boldsymbol{w}}_{k|k} \tag{5.19}$$

The input movement rate can be expressed as

$$\Delta\boldsymbol{U}_k = \begin{bmatrix} \Delta\boldsymbol{u}_{k|k} \\ \Delta\boldsymbol{u}_{k+1|k} \\ \Delta\boldsymbol{u}_{k+2|k} \\ \vdots \\ \Delta\boldsymbol{u}_{k+N-1|k} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_{k|k} - \boldsymbol{u}_{k-1|k} \\ \boldsymbol{u}_{k+1|k} - \boldsymbol{u}_{k|k} \\ \boldsymbol{u}_{k+2|k} - \boldsymbol{u}_{k+1|k} \\ \vdots \\ \boldsymbol{u}_{k+N-1|k} - \boldsymbol{u}_{k+N-2|k} \end{bmatrix} \tag{5.20}$$

using that $\Delta\boldsymbol{u}_{k+j|k} = \boldsymbol{u}_{k+j|k} - \boldsymbol{u}_{k+j-1|k}$. The expression for the input movement rate (5.20) can be reformulated as

$$\Delta\boldsymbol{U}_k = \boldsymbol{\Psi}\boldsymbol{U}_k - \boldsymbol{I}_0\boldsymbol{u}_{k-1|k} \tag{5.21}$$

where $\boldsymbol{\Psi}$ and $\boldsymbol{I}_0$ are

$$\boldsymbol{\Psi} = \begin{bmatrix} \boldsymbol{I} & & & & \\ -\boldsymbol{I} & \boldsymbol{I} & & & \\ & -\boldsymbol{I} & \boldsymbol{I} & & \\ & & \ddots & \ddots & \\ & & & -\boldsymbol{I} & \boldsymbol{I} \end{bmatrix}, \quad \boldsymbol{I}_0 = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \tag{5.22}$$

In (5.21), $\boldsymbol{u}_{k-1|k}$ is the control input associated with sample $k-1$ considered at the current sample k. This corresponds to the control input determined at sample $k-1$ and is therefore known at the current sample k. In order to simplify notation, this will be denoted by $\boldsymbol{u}_{k-1}$.

Let

$$\mathcal{Q} = \begin{bmatrix} \boldsymbol{Q}_y & & & \\ & \boldsymbol{Q}_y & & \\ & & \ddots & \\ & & & \boldsymbol{Q}_y \end{bmatrix}, \quad \mathcal{S} = \begin{bmatrix} \boldsymbol{S}_u & & & \\ & \boldsymbol{S}_u & & \\ & & \ddots & \\ & & & \boldsymbol{S}_u \end{bmatrix} \tag{5.23}$$

be the weight matrices associated with the prediction horizon N, corresponding to the dimensions of (5.18) and (5.21), respectively.

The objective function (5.7) can be expressed as

$$\begin{aligned}
\phi &= \frac{1}{2}\,\|\boldsymbol{Y}_k - \boldsymbol{R}_k\|^2_{\mathcal{Q}} + \frac{1}{2}\,\|\Delta \boldsymbol{U}_k\|^2_{\mathcal{S}} \\
&= \frac{1}{2}\,\|\boldsymbol{b}_k + \boldsymbol{\Gamma}\boldsymbol{U}_k - \boldsymbol{R}_k\|^2_{\mathcal{Q}} + \frac{1}{2}\,\|\boldsymbol{\Psi}\boldsymbol{U}_k - \boldsymbol{I}_0\boldsymbol{u}_{k-1}\|^2_{\mathcal{S}} \\
&= \frac{1}{2}[\boldsymbol{\Gamma}\boldsymbol{U}_k + \boldsymbol{b}_k - \boldsymbol{R}_k]^T\mathcal{Q}[\boldsymbol{\Gamma}\boldsymbol{U}_k + \boldsymbol{b}_k - \boldsymbol{R}_k] \\
&\quad + \frac{1}{2}[\boldsymbol{\Psi}\boldsymbol{U}_k - \boldsymbol{I}_0\boldsymbol{u}_{k-1}]^T\mathcal{S}[\boldsymbol{\Psi}\boldsymbol{U}_k - \boldsymbol{I}_0\boldsymbol{u}_{k-1}] \\
&= \frac{1}{2}[\boldsymbol{U}_k^T\boldsymbol{\Gamma}^T + \boldsymbol{b}_k^T - \boldsymbol{R}_k^T]\mathcal{Q}[\boldsymbol{\Gamma}\boldsymbol{U}_k + \boldsymbol{b}_k - \boldsymbol{R}_k] \\
&\quad + \frac{1}{2}[\boldsymbol{U}_k^T\boldsymbol{\Psi}^T - \boldsymbol{u}_{k-1}^T\boldsymbol{I}_0^T]\mathcal{S}[\boldsymbol{\Psi}\boldsymbol{U}_k - \boldsymbol{I}_0\boldsymbol{u}_{k-1}] \\
&= \frac{1}{2}\boldsymbol{U}_k^T\boldsymbol{H}\boldsymbol{U}_k + \boldsymbol{g}_k^T\boldsymbol{U}_k + \boldsymbol{\rho}_k
\end{aligned} \tag{5.24}$$

where $\boldsymbol{H}$, $\boldsymbol{g}_k$, $\boldsymbol{\rho}_k$ are

$$\boldsymbol{H} = \boldsymbol{\Gamma}^T\mathcal{Q}\boldsymbol{\Gamma} + \boldsymbol{\Psi}^T\mathcal{S}\boldsymbol{\Psi} \tag{5.25}$$

$$\boldsymbol{g}_k = \boldsymbol{\Gamma}^T\mathcal{Q}(\boldsymbol{b}_k - \boldsymbol{R}_k) - \boldsymbol{\Psi}^T\mathcal{S}\boldsymbol{I}_0\boldsymbol{u}_{k-1} \tag{5.26}$$

$$\boldsymbol{\rho}_k = \frac{1}{2}\,\|\boldsymbol{b}_k - \boldsymbol{R}_k\|^2_{\mathcal{Q}} + \frac{1}{2}\,\|\boldsymbol{I}_0\boldsymbol{u}_{k-1}\|^2_{\mathcal{S}} \tag{5.27}$$

Consequently, by state elimination the MPC regulation problem can be expressed as the following unconstrained convex QP

$$\min_{\boldsymbol{U}_k} \quad \phi = \frac{1}{2}\boldsymbol{U}_k^T\boldsymbol{H}\boldsymbol{U}_k + \boldsymbol{g}_k^T\boldsymbol{U}_k + \boldsymbol{\rho}_k \tag{5.28}$$

Provided the weights matrices $\boldsymbol{Q}_y$ and $\boldsymbol{S}_u$ in (5.7) are chosen such that $\boldsymbol{H}$ is positive definite, the QP is convex and has a unique solution, which is given by [JHR11]

$$\boldsymbol{U}_k = -\boldsymbol{H}^{-1}\boldsymbol{g}_k = \bar{\boldsymbol{L}}_x\hat{\boldsymbol{x}}_{k|k} + \bar{\boldsymbol{L}}_w\hat{\boldsymbol{w}}_{k|k} + \bar{\boldsymbol{L}}_R\boldsymbol{R}_k + \bar{\boldsymbol{L}}_u\boldsymbol{u}_{k-1} \tag{5.29}$$

where

$$\bar{\boldsymbol{L}}_x = -\boldsymbol{H}^{-1}\boldsymbol{\Gamma}^T\mathcal{Q}\boldsymbol{\Phi}_x \tag{5.30}$$

$$\bar{\boldsymbol{L}}_w = -\boldsymbol{H}^{-1}\boldsymbol{\Gamma}^T\mathcal{Q}\boldsymbol{\Phi}_w \tag{5.31}$$

$$\bar{\boldsymbol{L}}_R = \boldsymbol{H}^{-1}\boldsymbol{\Gamma}^T\mathcal{Q} \tag{5.32}$$

$$\bar{\boldsymbol{L}}_u = \boldsymbol{H}^{-1}\boldsymbol{\Psi}^T\mathcal{S}\boldsymbol{I}_0 \tag{5.33}$$

The requirement on $\boldsymbol{H}$ ensures convexity of the QP and is guaranteed if the weight matrices $\boldsymbol{Q}_y$ and $\boldsymbol{S}_u$ are positive definite.

In the receding horizon MPC approach, only the first control input $\boldsymbol{u}_{k|k}$ is implemented on the plant in each sample. The first control input $\boldsymbol{u}_{k|k}$ can be expressed as

$$\boldsymbol{u}_k = \boldsymbol{u}_{k|k} = \boldsymbol{I}_0^T\boldsymbol{U}_k = \boldsymbol{L}_x\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{L}_w\hat{\boldsymbol{w}}_{k|k} + \boldsymbol{L}_R\boldsymbol{R}_k + \boldsymbol{L}_u\boldsymbol{u}_{k-1} \tag{5.34}$$

where

$$\boldsymbol{L}_x = \boldsymbol{I}_0^T\bar{\boldsymbol{L}}_x = -\boldsymbol{I}_0^T\boldsymbol{H}^{-1}\boldsymbol{\Gamma}^T\mathcal{Q}\boldsymbol{\Phi}_x \tag{5.35}$$

$$\boldsymbol{L}_w = \boldsymbol{I}_0^T\bar{\boldsymbol{L}}_w = -\boldsymbol{I}_0^T\boldsymbol{H}^{-1}\boldsymbol{\Gamma}^T\mathcal{Q}\boldsymbol{\Phi}_w \tag{5.36}$$

$$\boldsymbol{L}_R = \boldsymbol{I}_0^T\bar{\boldsymbol{L}}_R = \boldsymbol{I}_0^T\boldsymbol{H}^{-1}\boldsymbol{\Gamma}^T\mathcal{Q} \tag{5.37}$$

$$\boldsymbol{L}_u = \boldsymbol{I}_0^T\bar{\boldsymbol{L}}_u = \boldsymbol{I}_0^T\boldsymbol{H}^{-1}\boldsymbol{\Psi}^T\mathcal{S}\boldsymbol{I}_0 \tag{5.38}$$

In practice $(\boldsymbol{L}_x, \boldsymbol{L}_w, \boldsymbol{L}_R, \boldsymbol{L}_u)$ are not computed explicitly by inversion of $\boldsymbol{H}$, but by a Cholesky factorization.

Remark 2 *It should be noted that the state elimination approach used in this section, is best suited for stable systems. That is, for systems where the eigenvalues of the system matrix $\boldsymbol{A}$ is strictly inside the unit disc, eig$(\boldsymbol{A}) < 1$. This is the case due to the structure of the matrices in (5.14), which can result in very small and large elements and hereby cause numerical instability [Mac02].*

Remark 3 *For most industrial processes it is reasonable to assume that the processes to be controlled are stable. For the case where the process is unstable, it is assumed that the process has been stabilized by some appropriate controller prior to the implementation of MPC. This could for example be done by designing a stabilizing state-feedback (LQG) controller $\boldsymbol{u}_k = \boldsymbol{L}\hat{\boldsymbol{x}}_{k|k}$ [Mac02].*

5.2.1 Controller State-Space for the Unconstrained MPC

In this section the expressions for the optimal control input and the stationary Kalman filter for the controller model, are combined and formulated in state-space form resulting in a controller state-space.

The optimal control input for the unconstrained MPC is given by

$$\boldsymbol{u}_k = \boldsymbol{u}_{k|k} = \boldsymbol{L}_x \hat{\boldsymbol{x}}_{k|k} + \boldsymbol{L}_w \hat{\boldsymbol{w}}_{k|k} + \boldsymbol{L}_R \boldsymbol{R}_k + \boldsymbol{L}_u \boldsymbol{u}_{k-1} \tag{5.39}$$

and the stationary Kalman filter for the controller model is given by

$$\boldsymbol{e}_k = \boldsymbol{y}_k - \hat{\boldsymbol{C}} \hat{\boldsymbol{x}}_{k|k-1} \tag{5.40}$$
$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx} \boldsymbol{e}_k \tag{5.41}$$
$$\hat{\boldsymbol{w}}_{k|k} = \boldsymbol{K}_{fw} \boldsymbol{e}_k \tag{5.42}$$

and

$$\hat{\boldsymbol{x}}_{k+1|k} = \hat{\boldsymbol{A}} \hat{\boldsymbol{x}}_{k|k} + \hat{\boldsymbol{B}} \boldsymbol{u}_k + \hat{\boldsymbol{G}} \hat{\boldsymbol{w}}_{k|k} \tag{5.43}$$

where the filter gains $\boldsymbol{K}_{fx}$ and $\boldsymbol{K}_{fw}$ are given in Chapter 4.

First the optimal control input (5.39) is substituted into the one-step prediction of the Kalman filter (5.43)

$$\begin{aligned}
\hat{\boldsymbol{x}}_{k+1|k} &= \hat{\boldsymbol{A}} \hat{\boldsymbol{x}}_{k|k} + \hat{\boldsymbol{B}} \boldsymbol{u}_k + \hat{\boldsymbol{G}} \hat{\boldsymbol{w}}_{k|k} \\
&= \hat{\boldsymbol{A}} \hat{\boldsymbol{x}}_{k|k} + \hat{\boldsymbol{B}} \Big[\boldsymbol{L}_x \hat{\boldsymbol{x}}_{k|k} + \boldsymbol{L}_w \hat{\boldsymbol{w}}_{k|k} + \boldsymbol{L}_R \boldsymbol{R}_k + \boldsymbol{L}_u \boldsymbol{u}_{k-1} \Big] + \hat{\boldsymbol{G}} \hat{\boldsymbol{w}}_{k|k}
\end{aligned} \tag{5.44}$$

Next the filtered state (5.41) and the filtered process-noise (5.42) are substituted into the above expression, giving

$$\begin{aligned}
\hat{\boldsymbol{x}}_{k+1|k} = \hat{\boldsymbol{A}} \Big[\hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx} \boldsymbol{e}_k \Big] + \hat{\boldsymbol{G}} \boldsymbol{K}_{fw} \boldsymbol{e}_k \\
+ \hat{\boldsymbol{B}} \Big[\boldsymbol{L}_x \Big[\hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx} \boldsymbol{e}_k \Big] + \boldsymbol{L}_w \boldsymbol{K}_{fw} \boldsymbol{e}_k + \boldsymbol{L}_R \boldsymbol{R}_k + \boldsymbol{L}_u \boldsymbol{u}_{k-1} \Big]
\end{aligned} \tag{5.45}$$

Now the innovation (5.40) is inserted

$$\begin{aligned}
\hat{\boldsymbol{x}}_{k+1|k} = \hat{\boldsymbol{A}} \Big[\hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx} (\boldsymbol{y}_k - \hat{\boldsymbol{C}} \hat{\boldsymbol{x}}_{k|k-1}) \Big] \qquad (5.46) \\
+ \hat{\boldsymbol{B}} \boldsymbol{L}_x \Big[\hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx} (\boldsymbol{y}_k - \hat{\boldsymbol{C}} \hat{\boldsymbol{x}}_{k|k-1}) \Big] \\
+ \hat{\boldsymbol{B}} \boldsymbol{L}_w \boldsymbol{K}_{fw} (\boldsymbol{y}_k - \hat{\boldsymbol{C}} \hat{\boldsymbol{x}}_{k|k-1}) + \hat{\boldsymbol{B}} \boldsymbol{L}_R \boldsymbol{R}_k + \hat{\boldsymbol{B}} \boldsymbol{L}_u \boldsymbol{u}_{k-1} \\
+ \hat{\boldsymbol{G}} \boldsymbol{K}_{fw} (\boldsymbol{y}_k - \hat{\boldsymbol{C}} \hat{\boldsymbol{x}}_{k|k-1})
\end{aligned}$$

By expanding terms in (5.46) the following is obtained

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1|k} = {} & \hat{\boldsymbol{A}}\hat{\boldsymbol{x}}_{k|k-1} + \hat{\boldsymbol{A}}\boldsymbol{K}_{fx}\boldsymbol{y}_k - \hat{\boldsymbol{A}}\boldsymbol{K}_{fx}\hat{\boldsymbol{C}}\hat{\boldsymbol{x}}_{k|k-1} + \hat{\boldsymbol{B}}\boldsymbol{L}_x\hat{\boldsymbol{x}}_{k|k-1} \\ & + \hat{\boldsymbol{B}}\boldsymbol{L}_x\boldsymbol{K}_{fx}\boldsymbol{y}_k - \hat{\boldsymbol{B}}\boldsymbol{L}_x\boldsymbol{K}_{fx}\hat{\boldsymbol{C}}\hat{\boldsymbol{x}}_{k|k-1} + \hat{\boldsymbol{B}}\boldsymbol{L}_w\boldsymbol{K}_{fw}\boldsymbol{y}_k \\ & - \hat{\boldsymbol{B}}\boldsymbol{L}_w\boldsymbol{K}_{fw}\hat{\boldsymbol{C}}\hat{\boldsymbol{x}}_{k|k-1} + \hat{\boldsymbol{B}}\boldsymbol{L}_R\boldsymbol{R}_k + \hat{\boldsymbol{B}}\boldsymbol{L}_u\boldsymbol{u}_{k-1} + \hat{\boldsymbol{G}}\boldsymbol{K}_{fw}\boldsymbol{y}_k \\ & - \hat{\boldsymbol{G}}\boldsymbol{K}_{fw}\hat{\boldsymbol{C}}\hat{\boldsymbol{x}}_{k|k-1}\end{aligned} \tag{5.47}$$

Defining

$$\boldsymbol{\Lambda} = \boldsymbol{L}_x\boldsymbol{K}_{fx} + \boldsymbol{L}_w\boldsymbol{K}_{fw} \tag{5.48}$$

$$\hat{\boldsymbol{\Lambda}} = \hat{\boldsymbol{A}}\boldsymbol{K}_{fx} + \hat{\boldsymbol{G}}\boldsymbol{K}_{fw} + \hat{\boldsymbol{B}}\boldsymbol{\Lambda} \tag{5.49}$$

the expression in (5.47) can be simplified and written as

$$\hat{\boldsymbol{x}}_{k+1|k} = \begin{bmatrix}\hat{\boldsymbol{A}} + \hat{\boldsymbol{B}}\boldsymbol{L}_x - \hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{C}} & \hat{\boldsymbol{B}}\boldsymbol{L}_u\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{x}}_{k|k-1} \\ \boldsymbol{u}_{k-1}\end{bmatrix} + \hat{\boldsymbol{\Lambda}}\boldsymbol{y}_k + \hat{\boldsymbol{B}}\boldsymbol{L}_R\boldsymbol{R}_k \tag{5.50}$$

The optimal control input for the unconstrained MPC, can also be written in terms of the one-step prediction and the previous optimal control input. This is done by conducting the equivalent substitutions as done above. The expression for the optimal control input is given by

$$\begin{aligned}\boldsymbol{u}_k &= \boldsymbol{L}_x\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{L}_w\hat{\boldsymbol{w}}_{k|k} + \boldsymbol{L}_R\boldsymbol{R}_k + \boldsymbol{L}_u\boldsymbol{u}_{k-1} \\ &= \boldsymbol{L}_x[\hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx}\boldsymbol{e}_k] + \boldsymbol{L}_w\boldsymbol{K}_{fw}\boldsymbol{e}_k + \boldsymbol{L}_R\boldsymbol{R}_k + \boldsymbol{L}_u\boldsymbol{u}_{k-1} \\ &= \boldsymbol{L}_x[\hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_{fx}(\boldsymbol{y}_k - \hat{\boldsymbol{C}}\hat{\boldsymbol{x}}_{k|k-1})] + \boldsymbol{L}_w\boldsymbol{K}_{fw}(\boldsymbol{y}_k - \hat{\boldsymbol{C}}\hat{\boldsymbol{x}}_{k|k-1}) \\ &\quad + \boldsymbol{L}_R\boldsymbol{R}_k + \boldsymbol{L}_u\boldsymbol{u}_{k-1} \\ &= \boldsymbol{L}_x\hat{\boldsymbol{x}}_{k|k-1} + [\boldsymbol{L}_x\boldsymbol{K}_{fx} + \boldsymbol{L}_w\boldsymbol{K}_{fw}](\boldsymbol{y}_k - \hat{\boldsymbol{C}}\hat{\boldsymbol{x}}_{k|k-1}) \\ &\quad + \boldsymbol{L}_R\boldsymbol{R}_k + \boldsymbol{L}_u\boldsymbol{u}_{k-1}\end{aligned} \tag{5.51}$$

Using equations (5.48) and (5.49) the expression for the optimal control input in (5.51) can be written as

$$\boldsymbol{u}_k = \begin{bmatrix}\boldsymbol{L}_x - \boldsymbol{\Lambda}\hat{\boldsymbol{C}} & \boldsymbol{L}_u\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{x}}_{k|k-1} \\ \boldsymbol{u}_{k-1}\end{bmatrix} + \boldsymbol{\Lambda}\boldsymbol{y}_k + \boldsymbol{L}_R\boldsymbol{R}_k \tag{5.52}$$

Now (5.50) and (5.52) can be combined to one equation describing the evolution of the one-step prediction and optimal control input. The combined equation is given by

$$\begin{bmatrix}\hat{\boldsymbol{x}}_{k+1|k} \\ \boldsymbol{u}_k\end{bmatrix} = \begin{bmatrix}\hat{\boldsymbol{A}} + \hat{\boldsymbol{B}}\boldsymbol{L}_x - \hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{C}} & \hat{\boldsymbol{B}}\boldsymbol{L}_u \\ \boldsymbol{L}_x - \boldsymbol{\Lambda}\hat{\boldsymbol{C}} & \boldsymbol{L}_u\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{x}}_{k|k-1} \\ \boldsymbol{u}_{k-1}\end{bmatrix} + \begin{bmatrix}\hat{\boldsymbol{\Lambda}} \\ \boldsymbol{\Lambda}\end{bmatrix}\boldsymbol{y}_k + \begin{bmatrix}\hat{\boldsymbol{B}}\boldsymbol{L}_R \\ \boldsymbol{L}_R\end{bmatrix}\boldsymbol{R}_k \tag{5.53}$$

Define the controller state as

$$\boldsymbol{x}_k^c = \begin{bmatrix} \hat{\boldsymbol{x}}_{k|k-1} \\ \boldsymbol{u}_{k-1} \end{bmatrix} \tag{5.54}$$

then the evolution of the controller model can be written in state-space form as

$$\boldsymbol{x}_{k+1}^c = \boldsymbol{A}_c \boldsymbol{x}_k^c + \boldsymbol{B}_{cy} \boldsymbol{y}_k + \boldsymbol{B}_{cr} \boldsymbol{R}_k \tag{5.55}$$

$$\boldsymbol{u}_k = \boldsymbol{C}_c \boldsymbol{x}_k^c + \boldsymbol{D}_{cy} \boldsymbol{y}_k + \boldsymbol{D}_{cr} \boldsymbol{R}_k \tag{5.56}$$

where

$$\boldsymbol{A}_c = \begin{bmatrix} \hat{\boldsymbol{A}} + \hat{\boldsymbol{B}}\boldsymbol{L}_x - \hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{C}} & \hat{\boldsymbol{B}}\boldsymbol{L}_u \\ \boldsymbol{L}_x - \boldsymbol{\Lambda}\hat{\boldsymbol{C}} & \boldsymbol{L}_u \end{bmatrix}, \quad \boldsymbol{B}_{cy} = \begin{bmatrix} \hat{\boldsymbol{\Lambda}} \\ \boldsymbol{\Lambda} \end{bmatrix}, \quad \boldsymbol{B}_{cr} = \begin{bmatrix} \hat{\boldsymbol{B}}\boldsymbol{L}_R \\ \boldsymbol{L}_R \end{bmatrix} \tag{5.57}$$

and

$$\boldsymbol{C}_c = \begin{bmatrix} \boldsymbol{L}_x - \boldsymbol{\Lambda}\hat{\boldsymbol{C}} & \boldsymbol{L}_u \end{bmatrix}, \quad \boldsymbol{D}_{cy} = \boldsymbol{\Lambda}, \quad \boldsymbol{D}_{cr} = \boldsymbol{L}_R \tag{5.58}$$

5.3 Unconstrained MPC for State-Space Models in Innovation Form

The formulation of the unconstrained MPC controller and the controller state-space considered in the previous section, can also be formulated for state-space models in innovation form

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{K}\boldsymbol{\varepsilon}_k \tag{5.59}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{\varepsilon}_k \tag{5.60}$$

where $\boldsymbol{\varepsilon}_k \sim \mathcal{N}_{iid}(\boldsymbol{0}, \boldsymbol{R}_\varepsilon)$.

Remark 4 *The model in innovation form (5.59)-(5.60) is a special case of the linear model (4.1)-(4.2). (5.59)-(5.60) may be expressed in the form (4.1)-(4.2) with $\boldsymbol{G} = \boldsymbol{K}$, $\boldsymbol{w}_k = \boldsymbol{v}_k = \boldsymbol{\varepsilon}_k$ [JHR11].*

The results for the model in innovation form, are derived in the same way as done in the previous section. Furthermore, the results are very similar and are therefore not derived. The results for the model in innovation form can be found in [JHR11].

5.4 Constrained MPC

In many cases the system to be controlled is not unconstrained. It might be the case that the control input of the system is limited to some interval and/or the rate of which the control input can be changed is bounded. There exist different constraints, they are usually classified as input, output and state constraints. In this section some constrained MPC formulations are considered.

When constraints are present, the optimization problem becomes more complicated and implies that there exists no closed form expression for the optimal control input.

5.4.1 MPC with Bound Constraints

When control input constraints are present, the MPC regulation problem can be formulated as

$$\begin{aligned}
\min_{\{\boldsymbol{u}_{k+j|k}\}_{j=0}^{N-1}} \quad & \phi = \frac{1}{2}\sum_{j=0}^{N-1} \left\| \hat{\boldsymbol{y}}_{k+1+j|k} - \boldsymbol{r}_{k+1+j|k} \right\|_{\boldsymbol{Q}_y}^2 + \left\| \Delta \boldsymbol{u}_{k+j|k} \right\|_{\boldsymbol{S}_u}^2 && (5.61)\\
s.t. \quad & \hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{B}\boldsymbol{u}_{k|k} + \boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} && (5.62)\\
& \hat{\boldsymbol{y}}_{k+1|k} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1|k} && (5.63)\\
& \hat{\boldsymbol{x}}_{k+1+j|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k+j|k} + \boldsymbol{B}\boldsymbol{u}_{k+j|k}, \qquad j = 1,2,\dots,N-1 && (5.64)\\
& \hat{\boldsymbol{y}}_{k+1+j|k} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1+j|k}, \qquad j = 1,2,\dots,N-1 && (5.65)\\
& \boldsymbol{u}_{min} \leq \boldsymbol{u}_{k+j|k} \leq \boldsymbol{u}_{max}, \qquad j = 0,1,\dots,N-1 && (5.66)
\end{aligned}$$

This formulation is almost identical with the unconstrained case. Since there are no modifications to the objective function and the equality constraints, the MPC regulation problem can again be expressed as a convex QP. But due to the control input constraints this cannot be solved explicitly.

The constraints on the control inputs

$$\boldsymbol{u}_{min} \leq \boldsymbol{u}_{k+j|k} \leq \boldsymbol{u}_{max}, \quad j = 0,1,\dots,N-1 \tag{5.67}$$

can be written as

$$\begin{bmatrix} \boldsymbol{u}_{min} \\ \boldsymbol{u}_{min} \\ \vdots \\ \boldsymbol{u}_{min} \end{bmatrix} \leq \begin{bmatrix} \boldsymbol{u}_{k|k} \\ \boldsymbol{u}_{k+1|k} \\ \vdots \\ \boldsymbol{u}_{k+N-1|k} \end{bmatrix} \leq \begin{bmatrix} \boldsymbol{u}_{max} \\ \boldsymbol{u}_{max} \\ \vdots \\ \boldsymbol{u}_{max} \end{bmatrix} \tag{5.68}$$

and furthermore as

$$\boldsymbol{U}_{min} \leq \boldsymbol{U}_k \leq \boldsymbol{U}_{max} \tag{5.69}$$

The MPC regulation problem may be expressed as the convex QP

$$\min_{\boldsymbol{U}_k} \quad \phi = \frac{1}{2}\boldsymbol{U}_k^T \boldsymbol{H} \boldsymbol{U}_k + \boldsymbol{g}_k^T \boldsymbol{U}_k + \boldsymbol{\rho}_k \tag{5.70}$$

$$\text{s.t.} \quad \boldsymbol{U}_{min} \leq \boldsymbol{U}_k \leq \boldsymbol{U}_{max} \tag{5.71}$$

In a quadratic program the constant $\boldsymbol{\rho}_k$ has no influence on the minimum, so it can be left out when solving the QP.

5.4.2 MPC with Bound and Input Rate Movement Constraints

In the MPC formulation also input rate movement bounds can be added. This leads to the following formulation of the MPC regulation problem

$$\min_{\{\boldsymbol{u}_{k+j|k}\}_{j=0}^{N-1}} \quad \phi = \frac{1}{2}\sum_{j=0}^{N-1} \left\|\hat{\boldsymbol{y}}_{k+1+j|k} - \boldsymbol{r}_{k+1+j|k}\right\|_{\boldsymbol{Q}_y}^2 + \left\|\Delta \boldsymbol{u}_{k+j|k}\right\|_{\boldsymbol{S}_u}^2 \tag{5.72}$$

$$s.t. \quad \hat{\boldsymbol{x}}_{k+1|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k|k} + \boldsymbol{B}\boldsymbol{u}_{k|k} + \boldsymbol{G}\hat{\boldsymbol{w}}_{k|k} \tag{5.73}$$

$$\hat{\boldsymbol{y}}_{k+1|k} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1|k} \tag{5.74}$$

$$\hat{\boldsymbol{x}}_{k+1+j|k} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k+j|k} + \boldsymbol{B}\boldsymbol{u}_{k+j|k}, \qquad j = 1, 2, \ldots, N-1 \tag{5.75}$$

$$\hat{\boldsymbol{y}}_{k+1+j|k} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1+j|k}, \qquad j = 1, 2, \ldots, N-1 \tag{5.76}$$

$$\boldsymbol{u}_{min} \leq \boldsymbol{u}_{k+j|k} \leq \boldsymbol{u}_{max}, \qquad j = 0, 1, \ldots, N-1 \tag{5.77}$$

$$\Delta\boldsymbol{u}_{min} \leq \Delta\boldsymbol{u}_{k+j|k} \leq \Delta\boldsymbol{u}_{max}, \qquad j = 0, 1, \ldots, N-1 \tag{5.78}$$

The constraints on the input rate movement

$$\Delta\boldsymbol{u}_{min} \leq \Delta\boldsymbol{u}_{k+j|k} \leq \Delta\boldsymbol{u}_{max}, \quad j = 0, 1, \ldots, N-1 \tag{5.79}$$

can be written as

$$\begin{bmatrix}\Delta \boldsymbol{u}_{min}\\ \Delta \boldsymbol{u}_{min}\\ \vdots \\ \Delta \boldsymbol{u}_{min}\end{bmatrix} \leq \begin{bmatrix}\Delta \boldsymbol{u}_{k|k}\\ \Delta \boldsymbol{u}_{k+1|k}\\ \vdots \\ \Delta \boldsymbol{u}_{k+N-1|k}\end{bmatrix} \leq \begin{bmatrix}\Delta \boldsymbol{u}_{max}\\ \Delta \boldsymbol{u}_{max}\\ \vdots \\ \Delta \boldsymbol{u}_{max}\end{bmatrix} \tag{5.80}$$

and furthermore

$$\Delta \boldsymbol{U}_{min} \leq \Delta \boldsymbol{U}_k \leq \Delta \boldsymbol{U}_{max}$$

Inserting the expression for $\Delta \boldsymbol{U}_k$ given in (5.21) gives

$$\Delta \boldsymbol{U}_{min} \leq \Delta \boldsymbol{U}_k \leq \Delta \boldsymbol{U}_{max} \iff \tag{5.81}$$

$$\Delta \boldsymbol{U}_{min} + \boldsymbol{I}_0 \boldsymbol{u}_{k-1} \leq \boldsymbol{\Psi} \boldsymbol{U}_k \leq \Delta \boldsymbol{U}_{max} + \boldsymbol{I}_0 \boldsymbol{u}_{k-1} \tag{5.82}$$

Denote these bounds by

$$\boldsymbol{b}_{l,k} = \Delta \boldsymbol{U}_{min} + \boldsymbol{I}_0 \boldsymbol{u}_{k-1}, \quad \boldsymbol{b}_{u,k} = \Delta \boldsymbol{U}_{max} + \boldsymbol{I}_0 \boldsymbol{u}_{k-1} \tag{5.83}$$

The MPC regulation problem may now be expressed as the convex QP

$$\min_{\boldsymbol{U}_k} \quad \phi = \frac{1}{2} \boldsymbol{U}_k^T \boldsymbol{H} \boldsymbol{U}_k + \boldsymbol{g}_k^T \boldsymbol{U}_k + \boldsymbol{\rho}_k \tag{5.84}$$

$$\text{s.t.} \quad \boldsymbol{U}_{min} \leq \boldsymbol{U}_k \leq \boldsymbol{U}_{max} \tag{5.85}$$

$$\boldsymbol{b}_{l,k} \leq \boldsymbol{\Psi} \boldsymbol{U}_k \leq \boldsymbol{b}_{u,k} \tag{5.86}$$

Remark 5 *It should be noted that if all constraints remain inactive, the optimal solution $\boldsymbol{U}_k^*$ for the constrained MPC will be equal to the optimal solution for unconstrained MPC.*

5.5 Summary

In this chapter the basic MPC formulation has been presented and the main components, such as prediction horizon, control horizon and weight matrices, of the setup have been discussed. Furthermore, the optimal control law for the unconstrained MPC was derived explicitly and combined with the Kalman filter to form a controller state-space.

It was also considered how to formulate MPC controllers with constraints on the control input and on the input rate movement.

Chapter 6

Closed-Loop Analysis

The purpose of this chapter is to derive a closed-loop description for the control system, i.e. the process and the controller, for the unconstrained MPC derived in Chapter 5. Subsequently the properties of the closed-loop system are analyzed.

The chapter is organized as follows. The first part of the chapter derives the closed-loop system. In the second part of the chapter the covariance matrices associated with the closed-loop are derived. The third part deals with the transfer function describing the process model and the controller model. Finally the general notion of sensitivity is introduced and the sensitivity functions of the closed-loop MPC system are derived.

6.1 Closed-Loop State-Space Description

In this section the controller state-space derived in Section 5.2.1 is combined with a linear process model of system to be controlled. For this purpose the considered plant is assumed to be a stochastic linear time invariant state-space system given by

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{G}\boldsymbol{w}_k && (6.1)\\ \boldsymbol{z}_k &= \boldsymbol{C}\boldsymbol{x}_k && (6.2)\\ \boldsymbol{y}_k &= \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k && (6.3)\end{aligned}$$

Recall that the controller state-space is given by

$$\begin{aligned}\boldsymbol{x}^c_{k+1} &= \boldsymbol{A}_c\boldsymbol{x}^c_k + \boldsymbol{B}_{cy}\boldsymbol{y}_k + \boldsymbol{B}_{cr}\boldsymbol{R}_k && (6.4)\\ \boldsymbol{u}_k &= \boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{D}_{cy}\boldsymbol{y}_k + \boldsymbol{D}_{cr}\boldsymbol{R}_k && (6.5)\end{aligned}$$

Since both the controller state-space and the process model are linear, they can be combined and expressed as a closed-loop linear state-space. In the following the closed-loop description will be derived.

The control input (6.5) is first substituted into the process state equation (6.1)

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{G}\boldsymbol{w}_k\\ &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\Big[\boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{D}_{cy}\boldsymbol{y}_k + \boldsymbol{D}_{cr}\boldsymbol{R}_k\Big] + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{G}\boldsymbol{w}_k && (6.6)\end{aligned}$$

Insert the plant measurement (6.3) to obtain

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\Big[\boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{D}_{cy}(\boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k) + \boldsymbol{D}_{cr}\boldsymbol{R}_k\Big] + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{G}\boldsymbol{w}_k\\ &= (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{D}_{cy}\boldsymbol{C})\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{B}\boldsymbol{D}_{cy}\boldsymbol{v}_k + \boldsymbol{B}\boldsymbol{D}_{cr}\boldsymbol{R}_k + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{G}\boldsymbol{w}_k && (6.7)\end{aligned}$$

The process state equation is hereby described in terms of the controller states. A similar expression can be obtained for the controller state, as follows. The controller state is given by

$$\begin{aligned}\boldsymbol{x}^c_{k+1} &= \boldsymbol{A}_c\boldsymbol{x}^c_k + \boldsymbol{B}_{cy}\boldsymbol{y}_k + \boldsymbol{B}_{cr}\boldsymbol{R}_k\\ &= \boldsymbol{A}_c\boldsymbol{x}^c_k + \boldsymbol{B}_{cy}(\boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k) + \boldsymbol{B}_{cr}\boldsymbol{R}_k\\ &= \boldsymbol{A}_c\boldsymbol{x}^c_k + \boldsymbol{B}_{cy}\boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{B}_{cy}\boldsymbol{v}_k + \boldsymbol{B}_{cr}\boldsymbol{R}_k && (6.8)\end{aligned}$$

Insert the plant measurement (6.3) into the expression for the control input (6.5) results in

$$\begin{aligned}\boldsymbol{u}_k &= \boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{D}_{cy}\boldsymbol{y}_k + \boldsymbol{D}_{cr}\boldsymbol{R}_k\\ &= \boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{D}_{cy}(\boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k) + \boldsymbol{D}_{cr}\boldsymbol{R}_k\\ &= \boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{D}_{cy}\boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}_{cy}\boldsymbol{v}_k + \boldsymbol{D}_{cr}\boldsymbol{R}_k && (6.9)\end{aligned}$$

By defining the closed-loop state as

$$\boldsymbol{x}_k^{cl} = \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{x}_k^c \end{bmatrix} \tag{6.10}$$

a closed-loop state-space can now be formulated.

The resulting closed-loop state-space description for the combined system is given by

$$\begin{aligned} \boldsymbol{x}_{k+1}^{cl} &= \boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl} + \boldsymbol{B}_{wcl}\boldsymbol{w}_k + \boldsymbol{B}_{vcl}\boldsymbol{v}_k + \boldsymbol{B}_{rcl}\boldsymbol{R}_k + \boldsymbol{B}_{dcl}\boldsymbol{d}_k && (6.11) \\ \boldsymbol{z}_k &= \boldsymbol{C}_{zcl}\boldsymbol{x}_k^{cl} && (6.12) \\ \boldsymbol{y}_k &= \boldsymbol{C}_{ycl}\boldsymbol{x}_k^{cl} + \boldsymbol{v}_k && (6.13) \\ \boldsymbol{u}_k &= \boldsymbol{C}_{ucl}\boldsymbol{x}_k^{cl} + \boldsymbol{D}_{cy}\boldsymbol{v}_k + \boldsymbol{D}_{cr}\boldsymbol{R}_k && (6.14) \end{aligned}$$

where

$$\boldsymbol{A}_{cl} = \begin{bmatrix} \boldsymbol{A} + \boldsymbol{B}\boldsymbol{D}_{cy}\boldsymbol{C} & \boldsymbol{B}\boldsymbol{C}_c \\ \boldsymbol{B}_{cy}\boldsymbol{C} & \boldsymbol{A}_c \end{bmatrix}, \quad \boldsymbol{B}_{wcl} = \begin{bmatrix} \boldsymbol{G} \\ \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{B}_{vcl} = \begin{bmatrix} \boldsymbol{B}\boldsymbol{D}_{cy} \\ \boldsymbol{B}_{cy} \end{bmatrix} \tag{6.15}$$

$$\boldsymbol{B}_{rcl} = \begin{bmatrix} \boldsymbol{B}\boldsymbol{D}_{cr} \\ \boldsymbol{B}_{cr} \end{bmatrix}, \quad \boldsymbol{B}_{dcl} = \begin{bmatrix} \boldsymbol{E} \\ \boldsymbol{0} \end{bmatrix} \tag{6.16}$$

and

$$\boldsymbol{C}_{ycl} = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{C}_{zcl} = \begin{bmatrix} \boldsymbol{C} & \boldsymbol{0} \end{bmatrix}, \quad \boldsymbol{C}_{ucl} = \begin{bmatrix} \boldsymbol{D}_{cy}\boldsymbol{C} & \boldsymbol{C}_c \end{bmatrix} \tag{6.17}$$

6.2 Covariance

The closed-loop description of the unconstrained MPC, can be used to determine the closed-loop covariance matrices for the input, measurement and the output. To this end, first the closed-loop state covariance matrix needs to be computed.

Let $\boldsymbol{R}_{x,k}$ denote the covariance matrix of the closed-loop state $\boldsymbol{x}_k^{cl}$. Now the variance of the closed-loop state equation in (6.11) is computed.

$$\begin{aligned} \boldsymbol{R}_{x,k+1} &= Var\{\boldsymbol{x}_{k+1}^{cl}\} \\ &= Var\{\boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl} + \boldsymbol{B}_{wcl}\boldsymbol{w}_k + \boldsymbol{B}_{vcl}\boldsymbol{v}_k + \boldsymbol{B}_{rcl}\boldsymbol{R}_k + \boldsymbol{B}_{dcl}\boldsymbol{d}_k\} \\ &= Var\{\boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl} + \boldsymbol{B}_{wcl}\boldsymbol{w}_k + \boldsymbol{B}_{vcl}\boldsymbol{v}_k\} && (6.18) \end{aligned}$$

Since the reference $\boldsymbol{R}_k$ and the disturbance $\boldsymbol{d}_k$ are deterministic quantities, these do not have influence on the variance and can be left out.

$$\begin{aligned}
&Var\{\boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl} + \boldsymbol{B}_{wcl}\boldsymbol{w}_k + \boldsymbol{B}_{vcl}\boldsymbol{v}_k\}\\
&= Cov\{\boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl} + \boldsymbol{B}_{wcl}\boldsymbol{w}_k + \boldsymbol{B}_{vcl}\boldsymbol{v}_k, \boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl} + \boldsymbol{B}_{wcl}\boldsymbol{w}_k + \boldsymbol{B}_{vcl}\boldsymbol{v}_k\}\\
&= Cov\{\boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl}, \boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl}\} + Cov\{\boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl}, \boldsymbol{B}_{wcl}\boldsymbol{w}_k\} + Cov\{\boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl}, \boldsymbol{B}_{vcl}\boldsymbol{v}_k\}\\
&\quad + Cov\{\boldsymbol{B}_{wcl}\boldsymbol{w}_k, \boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl}\} + Cov\{\boldsymbol{B}_{wcl}\boldsymbol{w}_k, \boldsymbol{B}_{wcl}\boldsymbol{w}_k\} + Cov\{\boldsymbol{B}_{wcl}\boldsymbol{w}_k, \boldsymbol{B}_{vcl}\boldsymbol{v}_k\}\\
&\quad + Cov\{\boldsymbol{B}_{vcl}\boldsymbol{v}_k, \boldsymbol{A}_{cl}\boldsymbol{x}_k^{cl}\} + Cov\{\boldsymbol{B}_{vcl}\boldsymbol{v}_k, \boldsymbol{B}_{wcl}\boldsymbol{w}_k\} + Cov\{\boldsymbol{B}_{vcl}\boldsymbol{v}_k, \boldsymbol{B}_{vcl}\boldsymbol{v}_k\}\\
&= \boldsymbol{A}_{cl}Cov\{\boldsymbol{x}_k^{cl}, \boldsymbol{x}_k^{cl}\}\boldsymbol{A}_{cl}^T + \boldsymbol{A}_{cl}Cov\{\boldsymbol{x}_k^{cl}, \boldsymbol{w}_k\}\boldsymbol{B}_{wcl}^T + \boldsymbol{A}_{cl}Cov\{\boldsymbol{x}_k^{cl}, \boldsymbol{v}_k\}\boldsymbol{B}_{vcl}^T\\
&\quad + \boldsymbol{B}_{wcl}Cov\{\boldsymbol{w}_k, \boldsymbol{x}_k^{cl}\}\boldsymbol{A}_{cl}^T + \boldsymbol{B}_{wcl}Cov\{\boldsymbol{w}_k, \boldsymbol{w}_k\}\boldsymbol{B}_{wcl}^T + \boldsymbol{B}_{wcl}Cov\{\boldsymbol{w}_k, \boldsymbol{v}_k\}\boldsymbol{B}_{vcl}^T\\
&\quad + \boldsymbol{B}_{vcl}Cov\{\boldsymbol{v}_k, \boldsymbol{x}_k^{cl}\}\boldsymbol{A}_{cl}^T + \boldsymbol{B}_{vcl}Cov\{\boldsymbol{v}_k, \boldsymbol{w}_k\}\boldsymbol{B}_{wcl}^T + \boldsymbol{B}_{vcl}Cov\{\boldsymbol{v}_k, \boldsymbol{v}_k\}\boldsymbol{B}_{vcl}^T
\end{aligned}$$

It is assumed that the closed-loop state $\boldsymbol{x}_k^{cl}$ and the process-noise $\boldsymbol{w}_k$ are uncorrelated and the same for $\boldsymbol{x}_k^{cl}$ and the measurement noise $\boldsymbol{v}_k$. Due to the possible correlation of the process-noise $\boldsymbol{w}_k$ and the measurement noise $\boldsymbol{v}_k$, no assumptions are imposed on these.

The equation can now be expressed as

$$\begin{aligned}\boldsymbol{R}_{x,k+1} = {} & \boldsymbol{A}_{cl}\boldsymbol{R}_{x,k}\boldsymbol{A}_{cl}^T + \boldsymbol{B}_{wcl}\boldsymbol{Q}\boldsymbol{B}_{wcl}^T + \boldsymbol{B}_{vcl}\boldsymbol{R}\boldsymbol{B}_{vcl}^T + \boldsymbol{B}_{wcl}\boldsymbol{S}\boldsymbol{B}_{vcl}^T\\ & + \boldsymbol{B}_{vcl}\boldsymbol{S}^T\boldsymbol{B}_{wcl}^T\end{aligned} \tag{6.19}$$

For the case where the process-noise $\boldsymbol{w}_k$ and the measurement noise $\boldsymbol{v}_k$ are uncorrelated, the equation simplifies to

$$\boldsymbol{R}_{x,k+1} = \boldsymbol{A}_{cl}\boldsymbol{R}_{x,k}\boldsymbol{A}_{cl}^T + \boldsymbol{B}_{wcl}\boldsymbol{Q}\boldsymbol{B}_{wcl}^T + \boldsymbol{B}_{vcl}\boldsymbol{R}\boldsymbol{B}_{vcl}^T \tag{6.20}$$

Now provided that all eigenvalues of $\boldsymbol{A}_{cl}$ are strictly inside the unit disk, the closed-loop system is asymptotically stable. This implies that the covariance matrix of the state $\boldsymbol{R}_{x,k}$ becomes stationary as $k \rightarrow \infty$ with limit $\boldsymbol{R}_x$ [Pou07]. The equation for the covariance of the closed-loop state is given by

$$\boldsymbol{R}_x = \boldsymbol{A}_{cl}\boldsymbol{R}_x\boldsymbol{A}_{cl}^T + \boldsymbol{B}_{wcl}\boldsymbol{Q}\boldsymbol{B}_{wcl}^T + \boldsymbol{B}_{vcl}\boldsymbol{R}\boldsymbol{B}_{vcl}^T \tag{6.21}$$

This equation is known as the discrete-time Lyapunov equation. Note that when $\boldsymbol{B}_{wcl}$, $\boldsymbol{B}_{vcl}$, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are known matrices, this is a matrix equation in the closed-loop steady state covariance $\boldsymbol{R}_x$. The equation can be solved using the built-in MATLAB function `dlyap`.

Next the closed-loop steady state covariance matrix of the measurement $\boldsymbol{y}_k$, can be computed as

$$\begin{aligned}\boldsymbol{R}_y &= Var\{\boldsymbol{y}_k\}\\ &= Var\{\boldsymbol{C}_{ycl}\boldsymbol{x}_k^{cl} + \boldsymbol{v}_k\}\\ &= Var\{\boldsymbol{C}_{ycl}\boldsymbol{x}_k^{cl}\} + Var\{\boldsymbol{v}_k\} + Cov\{\boldsymbol{C}_{ycl}\boldsymbol{x}_k^{cl}, \boldsymbol{v}_k\} + Cov\{\boldsymbol{v}_k, \boldsymbol{C}_{ycl}\boldsymbol{x}_k^{cl}\}\\ &= \boldsymbol{C}_{ycl}Var\{\boldsymbol{x}_k^{cl}\}\boldsymbol{C}_{ycl}^T + Var\{\boldsymbol{v}_k\} + \boldsymbol{C}_{ycl}Cov\{\boldsymbol{x}_k^{cl}, \boldsymbol{v}_k\} + Cov\{\boldsymbol{v}_k, \boldsymbol{x}_k^{cl}\}\boldsymbol{C}_{ycl}^T\\ &= \boldsymbol{C}_{ycl}\boldsymbol{R}_x\boldsymbol{C}_{ycl}^T + \boldsymbol{R} \qquad (6.22)\end{aligned}$$

The closed-loop steady state covariance matrix of the control input $\boldsymbol{u}_k$ is

$$\begin{aligned}\boldsymbol{R}_u &= Var\{\boldsymbol{u}_k\} \qquad (6.23)\\ &= Var\{\boldsymbol{C}_{ucl}\boldsymbol{x}_k^{cl} + \boldsymbol{D}_{cy}\boldsymbol{v}_k + \boldsymbol{D}_{cr}\boldsymbol{R}_k\}\\ &= Var\{\boldsymbol{C}_{ucl}\boldsymbol{x}_k^{cl} + \boldsymbol{D}_{cy}\boldsymbol{v}_k\}\\ &= \boldsymbol{C}_{ucl}Var\{\boldsymbol{x}_k^{cl}\}\boldsymbol{C}_{ucl}^T + \boldsymbol{D}_{cy}Var\{\boldsymbol{v}_k\}\boldsymbol{D}_{cy}^T + Cov\{\boldsymbol{C}_{ucl}\boldsymbol{x}_k^{cl}, \boldsymbol{D}_{cy}\boldsymbol{v}_k\}\\ &\quad + Cov\{\boldsymbol{C}_{ucl}\boldsymbol{x}_k^{cl}, \boldsymbol{D}_{cy}\boldsymbol{v}_k\}^T\\ &= \boldsymbol{C}_{ucl}\boldsymbol{R}_x\boldsymbol{C}_{ucl}^T + \boldsymbol{D}_{cy}\boldsymbol{R}\boldsymbol{D}_{cy}^T \qquad (6.24)\end{aligned}$$

Last, the closed-loop steady state covariance matrix of the output $\boldsymbol{z}_k$ is computed. The covariance matrix of the output is

$$\begin{aligned}\boldsymbol{R}_z &= Var\{\boldsymbol{z}_k\} = Var\{\boldsymbol{C}_{zcl}\boldsymbol{x}_k^{cl}\} \qquad (6.25)\\ &= \boldsymbol{C}_{zcl}\boldsymbol{R}_x\boldsymbol{C}_{zcl}^T \qquad (6.26)\end{aligned}$$

6.3 Transfer Functions

In this section the linear process model, the controller model and the closed-loop system are described as input-output models in the z domain using discrete-time transfer functions.

This is done by transforming the equations using the (unilateral) Z-transform defined by

$$\boldsymbol{F}(z) = \mathcal{Z}\{\boldsymbol{f}_k\} := \sum_{k=0}^{\infty} z^{-k}\boldsymbol{f}_k \tag{6.27}$$

where $\boldsymbol{f}_k$ denotes the discrete-time domain function. The Z-transform is a linear operator and has many interesting properties. The reader is referred to [Pou07] for more details.

6.3.1 Transfer function of the Process Model

The input-output function for the linear process model in (6.1)-(6.3), is computed as follows. Let the Z-transform of the current state, control input, disturbance and process-noise be given by

$$\mathcal{Z}\{\boldsymbol{x}_k\} = \boldsymbol{X}(z) \tag{6.28}$$
$$\mathcal{Z}\{\boldsymbol{u}_k\} = \boldsymbol{U}(z) \tag{6.29}$$
$$\mathcal{Z}\{\boldsymbol{d}_k\} = \boldsymbol{D}(z) \tag{6.30}$$
$$\mathcal{Z}\{\boldsymbol{w}_k\} = \boldsymbol{W}(z) \tag{6.31}$$

and of the next state

$$\mathcal{Z}\{\boldsymbol{x}_{k+1}\} = z\boldsymbol{X}(z) - z\boldsymbol{x}_0 \tag{6.32}$$

The state equation of the process model can now be written as

$$\begin{aligned}
\mathcal{Z}\{\boldsymbol{x}_{k+1}\} &= \mathcal{Z}\{\boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{G}\boldsymbol{w}_k\} \\
z\boldsymbol{X}(z) - z\boldsymbol{x}_0 &= \boldsymbol{A}\boldsymbol{X}(z) + \boldsymbol{B}\boldsymbol{U}(z) + \boldsymbol{E}\boldsymbol{D}(z) + \boldsymbol{G}\boldsymbol{W}(z) \\
z\boldsymbol{X}(z) - \boldsymbol{A}\boldsymbol{X}(z) &= z\boldsymbol{x}_0 + \boldsymbol{B}\boldsymbol{U}(z) + \boldsymbol{E}\boldsymbol{D}(z) + \boldsymbol{G}\boldsymbol{W}(z) \\
(z\boldsymbol{I} - \boldsymbol{A})\boldsymbol{X}(z) &= z\boldsymbol{x}_0 + \boldsymbol{B}\boldsymbol{U}(z) + \boldsymbol{E}\boldsymbol{D}(z) + \boldsymbol{G}\boldsymbol{W}(z) \qquad (6.33)
\end{aligned}$$

Isolating $\boldsymbol{X}(z)$ in (6.33) gives

$$\boldsymbol{X}(z) = (z\boldsymbol{I} - \boldsymbol{A})^{-1}[z\boldsymbol{x}_0 + \boldsymbol{B}\boldsymbol{U}(z) + \boldsymbol{E}\boldsymbol{D}(z) + \boldsymbol{G}\boldsymbol{W}(z)] \tag{6.34}$$

Next the Z-transform of the measurement equation is given by

$$\boldsymbol{Y}(z) = \boldsymbol{C}\boldsymbol{X}(z) + \boldsymbol{V}(z) \tag{6.35}$$

Inserting the expression for $\boldsymbol{X}(z)$ gives

$$\begin{aligned}\boldsymbol{Y}(z) &= \boldsymbol{C}\boldsymbol{X}(z) + \boldsymbol{V}(z) \\ &= \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}[z\boldsymbol{x}_0 + \boldsymbol{B}\boldsymbol{U}(z) + \boldsymbol{E}\boldsymbol{D}(z) + \boldsymbol{G}\boldsymbol{W}(z)] + \boldsymbol{V}(z) \qquad (6.36)\end{aligned}$$

Assuming that the initial state is $\boldsymbol{x}_0 = \boldsymbol{0}$, the input-output function for the process model measurement can be written as

$$\boldsymbol{Y}(z) = \boldsymbol{G}_{yu}(z)\boldsymbol{U}(z) + \boldsymbol{G}_{yd}(z)\boldsymbol{D}(z) + \boldsymbol{G}_{yw}(z)\boldsymbol{W}(z) + \boldsymbol{V}(z) \tag{6.37}$$

with discrete-time transfer functions

$$\boldsymbol{G}_{yu}(z) = \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{B} \tag{6.38}$$

$$\boldsymbol{G}_{yd}(z) = \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{E} \tag{6.39}$$

$$\boldsymbol{G}_{yw}(z) = \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{G} \tag{6.40}$$

Last the Z-transform of the output equation is given by

$$\boldsymbol{Z}(z) = \boldsymbol{C}\boldsymbol{X}(z) \tag{6.41}$$

Inserting the expression for $\boldsymbol{X}(z)$ gives

$$\boldsymbol{Z}(z) = \boldsymbol{C}\boldsymbol{X}(z) \tag{6.42}$$

$$= \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}[z\boldsymbol{x}_0 + \boldsymbol{B}\boldsymbol{U}(z) + \boldsymbol{E}\boldsymbol{D}(z) + \boldsymbol{G}\boldsymbol{W}(z)] \tag{6.43}$$

Now the the input-output function for the process model output can be written as

$$\boldsymbol{Z}(z) = \boldsymbol{G}_{zu}(z)\boldsymbol{U}(z) + \boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) + \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) \tag{6.44}$$

with discrete-time transfer functions

$$\boldsymbol{G}_{zu}(z) = \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{B} \tag{6.45}$$

$$\boldsymbol{G}_{zd}(z) = \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{E} \tag{6.46}$$

$$\boldsymbol{G}_{zw}(z) = \boldsymbol{C}(z\boldsymbol{I} - \boldsymbol{A})^{-1}\boldsymbol{G} \tag{6.47}$$

6.3.2 Transfer Function of the Controller Model

The state-space description of the controller is given by

$$\boldsymbol{x}^c_{k+1} = \boldsymbol{A}_c\boldsymbol{x}^c_k + \boldsymbol{B}_{cy}\boldsymbol{y}_k + \boldsymbol{B}_{cr}\boldsymbol{R}_k \tag{6.48}$$

$$\boldsymbol{u}_k = \boldsymbol{C}_c\boldsymbol{x}^c_k + \boldsymbol{D}_{cy}\boldsymbol{y}_k + \boldsymbol{D}_{cr}\boldsymbol{R}_k \tag{6.49}$$

The Z-transformation of the controller state equation (6.48) is

$$\begin{aligned} z\boldsymbol{X}_c(z) - \boldsymbol{A}_c\boldsymbol{X}_c(z) &= z\boldsymbol{x}^c_0 + \boldsymbol{B}_{cy}\boldsymbol{Y}(z) + \boldsymbol{B}_{cr}\boldsymbol{R}(z) \\ (z\boldsymbol{I} - \boldsymbol{A}_c)\boldsymbol{X}_c(z) &= z\boldsymbol{x}^c_0 + \boldsymbol{B}_{cy}\boldsymbol{Y}(z) + \boldsymbol{B}_{cr}\boldsymbol{R}(z) \\ \boldsymbol{X}_c(z) &= (z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}[z\boldsymbol{x}^c_0 + \boldsymbol{B}_{cy}\boldsymbol{Y}(z) + \boldsymbol{B}_{cr}\boldsymbol{R}(z)] \end{aligned} \tag{6.50}$$

Transformation of the (6.49) and inserting $\boldsymbol{X}_c(z)$ gives

$$\begin{aligned} \boldsymbol{U}(z) &= \boldsymbol{C}_c\boldsymbol{X}_c(z) + \boldsymbol{D}_{cy}\boldsymbol{Y}(z) + \boldsymbol{D}_{cr}\boldsymbol{R}(z) \\ &= \boldsymbol{C}_c(z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}[\boldsymbol{B}_{cy}\boldsymbol{Y}(z) + \boldsymbol{B}_{cr}\boldsymbol{R}(z)] + \boldsymbol{D}_{cy}\boldsymbol{Y}(z) + \boldsymbol{D}_{cr}\boldsymbol{R}(z) \\ &= \boldsymbol{C}_c(z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}\boldsymbol{B}_{cy}\boldsymbol{Y}(z) + \boldsymbol{C}_c(z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}\boldsymbol{B}_{cr}\boldsymbol{R}(z) \\ &\quad + \boldsymbol{D}_{cy}\boldsymbol{Y}(z) + \boldsymbol{D}_{cr}\boldsymbol{R}(z) \end{aligned} \tag{6.51}$$

The controller can now (in the z domain) be described by

$$\boldsymbol{U}(z) = \boldsymbol{C}_{uy}(z)\boldsymbol{Y}(z) + \boldsymbol{C}_{ur}(z)\boldsymbol{R}(z) \tag{6.52}$$

with discrete-time transfer functions

$$\boldsymbol{C}_{uy}(z) = \boldsymbol{C}_c(z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}\boldsymbol{B}_{cy} + \boldsymbol{D}_{cy} \tag{6.53}$$

$$\boldsymbol{C}_{ur}(z) = \boldsymbol{C}_c(z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}\boldsymbol{B}_{cr} + \boldsymbol{D}_{cr} \tag{6.54}$$

6.3.3 Transfer Function of the Closed-Loop

The input-output function for the closed-loop system of the unconstrained MPC, can in a similar manner be derived. Z-transforming the closed-loop state equation gives

$$\begin{aligned} \boldsymbol{X}_{cl}(z) &= (z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}z\boldsymbol{x}^{cl}_0 + (z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{wcl}\boldsymbol{W}(z) + (z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{vcl}\boldsymbol{V}(z) \\ &\quad + (z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{rcl}\boldsymbol{R}(z) + (z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{dcl}\boldsymbol{D}(z) \end{aligned} \tag{6.55}$$

Assuming that the closed-loop initial state is $\boldsymbol{x}_0^{cl} = \mathbf{0}$ and inserting $\boldsymbol{X}_{cl}(z)$ in the Z-transformed measurement equation

$$\begin{aligned}
\boldsymbol{Y}(z) &= \boldsymbol{C}_{ycl}\boldsymbol{X}_{cl}(z) + \boldsymbol{V}(z) \\
&= \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{wcl}\boldsymbol{W}(z) + [\boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{vcl} + \boldsymbol{I}]\boldsymbol{V}(z) \\
&\quad + \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{rcl}\boldsymbol{R}(z) + \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{dcl}\boldsymbol{D}(z) \qquad (6.56)
\end{aligned}$$

The measured output of the closed-loop system can (in the z domain) be described by

$$\boldsymbol{Y}(z) = \boldsymbol{G}_{yw}(z)\boldsymbol{W}(z) + \boldsymbol{G}_{yv}(z)\boldsymbol{V}(z) + \boldsymbol{G}_{yr}(z)\boldsymbol{R}(z) + \boldsymbol{G}_{yd}(z)\boldsymbol{D}(z) \qquad (6.57)$$

where

$$\begin{aligned}
\boldsymbol{G}_{yw}(z) &= \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{wcl} && (6.58)\\
\boldsymbol{G}_{yv}(z) &= \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{vcl} + \boldsymbol{I} && (6.59)\\
\boldsymbol{G}_{yr}(z) &= \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{rcl} && (6.60)\\
\boldsymbol{G}_{yd}(z) &= \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{dcl} && (6.61)
\end{aligned}$$

6.4 Sensitivity

In this section the sensitivity functions for the closed-loop unconstrained MPC are derived. The sensitivity function and the complementary sensitivity function in general describe the influence of external inputs on the the closed-loop behavior.

In order to introduce the concept of sensitivity, a classic one degree-of-freedom feedback SISO control system is first considered [SP01]. The system is illustrated in figure 6.1.

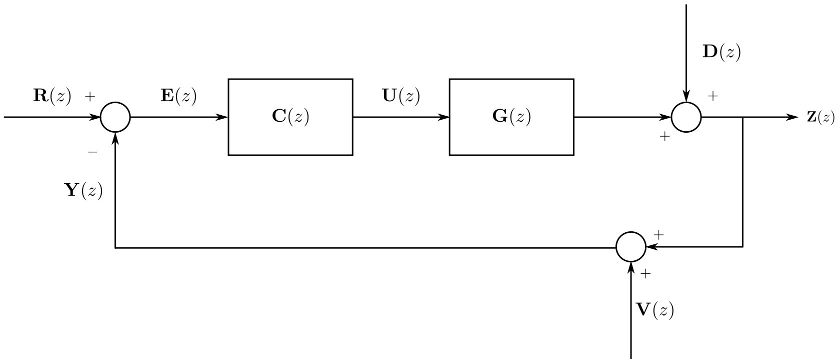

Figure 6.1: One degree-of-freedom feedback control system. The controller is described by $C(z)$ and the process by $G(z)$.

Figure 6.1 shows that the output of the process is influenced by an output disturbance $D(z)$ and furthermore that the measured output $Y(z)$ is corrupted by measurement noise $V(z)$. The controlled output of the process is given by

$$\begin{aligned} Z(z) &= G(z)U(z) + D(z) \\ &= G(z)C(z)E(z) + D(z) \\ &= G(z)C(z)[R(z) - Y(z)] + D(z) \\ &= G(z)C(z)[R(z) - (Z(z) + V(z))] + D(z) \\ &= G(z)C(z)R(z) - G(z)C(z)Z(z) - G(z)C(z)V(z) + D(z) \end{aligned} \tag{6.62}$$

Isolating $Z(z)$ gives

$$\begin{aligned} Z(z) &= \frac{G(z)C(z)}{(1+G(z)C(z))}R(z) - \frac{G(z)C(z)}{(1+G(z)C(z))}V(z) + \frac{1}{(1+G(z)C(z))}D(z) \\ &= T(z)R(z) - T(z)V(z) + S(z)D(z) \end{aligned} \tag{6.63}$$

It is seen that the controlled output of the system can be written in terms of the external inputs (the reference $R(z)$, measurement noise $V(z)$ and the output disturbance $D(z)$) and the transfer functions

$$S(z) = \frac{1}{1+G(z)C(z)} \tag{6.64}$$

and

$$T(z) = \frac{G(z)C(z)}{1+G(z)C(z)} \tag{6.65}$$

$S(z)$ is referred to as the sensitivity function and is the closed-loop transfer function from the output disturbances to the controlled output. Furthermore, $T(z)$ is referred to as the complimentary sensitivity function and is the closed-loop transfer function from the reference (and measurement noise) to the controlled output.

For the MIMO system case, the sensitivity functions generalize to

$$\boldsymbol{S}(z) = [\boldsymbol{I} + \boldsymbol{G}(z)\boldsymbol{C}(z)]^{-1} \tag{6.66}$$

and

$$\boldsymbol{T}(z) = [\boldsymbol{I} + \boldsymbol{G}(z)\boldsymbol{C}(z)]^{-1}\boldsymbol{G}(z)\boldsymbol{C}(z) \tag{6.67}$$

where the $\boldsymbol{S}(z)$ and $\boldsymbol{T}(z)$ are now matrices.

6.4.1 Sensitivity of the Unconstrained MPC

The above sensitivity functions are not valid for an MPC controller formulation, since the MPC is a two degree-of-freedom controller. The MPC has an unsymmetrical treatment of the reference signal $\boldsymbol{R}(z)$ and the measured output $\boldsymbol{Y}(z)$.

Recall from Section 6.3.2, that the unconstrained MPC control law can by expressed as

$$\boldsymbol{U}(z) = \boldsymbol{C}_{uy}(z)\boldsymbol{Y}(z) + \boldsymbol{C}_{ur}(z)\boldsymbol{R}(z) \tag{6.68}$$

with discrete-time transfer functions

$$\boldsymbol{C}_{uy}(z) = \boldsymbol{C}_c(z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}\boldsymbol{B}_{cy} + \boldsymbol{D}_{cy} \tag{6.69}$$
$$\boldsymbol{C}_{ur}(z) = \boldsymbol{C}_c(z\boldsymbol{I} - \boldsymbol{A}_c)^{-1}\boldsymbol{B}_{cr} + \boldsymbol{D}_{cr} \tag{6.70}$$

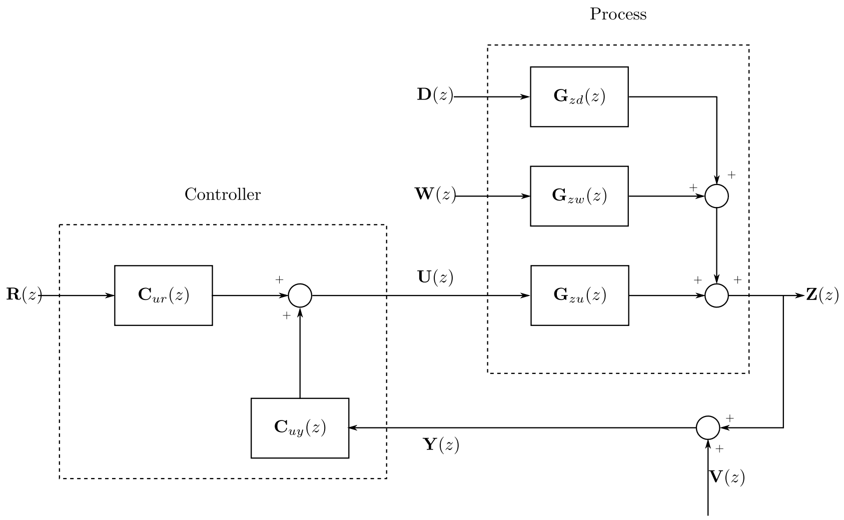

Figure 6.2: Two degree-of-freedom feedback control system representing the closed-loop unconstrained MPC.

Furthermore, the controlled output of the linear process model can be expressed as

$$\boldsymbol{Z}(z) = \boldsymbol{G}_{zu}(z)\boldsymbol{U}(z) + \boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) + \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) \tag{6.71}$$

Figure 6.2 illustrates the feedback loop for the unconstrained MPC control law in closed-loop with the linear process model. The process is influenced by an output disturbance $\boldsymbol{D}(z)$, which enters through the transfer function $\boldsymbol{G}_{zd}(z)$. Furthermore, the process is affected by process noise $\boldsymbol{W}(z)$, which is propagated to the output by the transfer function $\boldsymbol{G}_{zw}(z)$. Finally, the measured output $Y(z)$ is corrupted by measurement noise $V(z)$.

Since the measured output $\boldsymbol{Y}(z)$ is used by the controller, it is of particular interest to have the sensitivity functions that relate the external inputs to the measured output. The measured output can be expressed from the controlled output by

$$\begin{aligned}\boldsymbol{Y}(z) &= \boldsymbol{Z}(z) + \boldsymbol{V}(z)\\ &= \boldsymbol{G}_{zu}(z)\boldsymbol{U}(z) + \boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) + \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) + \boldsymbol{V}(z)\end{aligned} \tag{6.72}$$

Inserting the expression for the controller (6.68) into (6.72) gives

$$\begin{aligned}\boldsymbol{Y}(z) &= \boldsymbol{G}_{zu}(z)\boldsymbol{U}(z) + \boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) + \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) + \boldsymbol{V}(z)\\ &= \boldsymbol{G}_{zu}(z)[\boldsymbol{C}_{uy}(z)\boldsymbol{Y}(z) + \boldsymbol{C}_{ur}(z)\boldsymbol{R}(z)] + \boldsymbol{G}_{zd}(z)\boldsymbol{D}(z)\\ &\quad + \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) + \boldsymbol{V}(z)\\ &= \boldsymbol{G}_{zu}(z)\boldsymbol{C}_{uy}(z)\boldsymbol{Y}(z) + \boldsymbol{G}_{zu}(z)\boldsymbol{C}_{ur}(z)\boldsymbol{R}(z) + \boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) \qquad (6.73)\\ &\quad + \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) + \boldsymbol{V}(z)\end{aligned}$$

Isolating $\boldsymbol{Y}(z)$ results in the following expression for the measured output

$$\begin{aligned}\boldsymbol{Y}(z) = [\boldsymbol{I} - \boldsymbol{G}_{zu}(z)\boldsymbol{C}_{uy}(z)]^{-1}[&\boldsymbol{G}_{zu}(z)\boldsymbol{C}_{ur}(z)\boldsymbol{R}(z) + \boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) \qquad (6.74)\\ &+ \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) + \boldsymbol{V}(z)]\end{aligned}$$

The sensitivity functions for the closed-loop unconstrained MPC can be identified as

$$\boldsymbol{S}(z) = [\boldsymbol{I} - \boldsymbol{G}_{zu}(z)\boldsymbol{C}_{uy}(z)]^{-1} \qquad (6.75)$$

and

$$\begin{aligned}\boldsymbol{T}(z) &= [\boldsymbol{I} - \boldsymbol{G}_{zu}(z)\boldsymbol{C}_{uy}(z)]^{-1}\boldsymbol{G}_{zu}(z)\boldsymbol{G}_{ur}(z) \qquad (6.76)\\ &= \boldsymbol{S}(z)\boldsymbol{G}_{zu}(z)\boldsymbol{G}_{ur}(z) \qquad (6.77)\end{aligned}$$

The measured output can be expressed in terms of the sensitivity functions $\boldsymbol{T}(z)$ and $\boldsymbol{S}(z)$ as

$$\begin{aligned}\boldsymbol{Y}(z) &= \boldsymbol{T}(z)\boldsymbol{R}(z) + \boldsymbol{S}(z)\boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) + \boldsymbol{S}(z)\boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) + \boldsymbol{S}(z)\boldsymbol{V}(z)\\ &= \boldsymbol{T}(z)\boldsymbol{R}(z) + \boldsymbol{S}(z)[\boldsymbol{G}_{zd}(z)\boldsymbol{D}(z) + \boldsymbol{G}_{zw}(z)\boldsymbol{W}(z) + \boldsymbol{V}(z)] \qquad (6.78)\end{aligned}$$

It is seen that the sensitivity function $\boldsymbol{S}(z)$ is the closed-loop transfer function from both the output disturbances, the process noise and measurement noise to the measured output.

The closed-loop MPC is expressed in terms of the associated state-space formulation for the controller and the linear process model. It is of interest to also express the sensitivity function in terms of the closed-loop state-space description. This can be done by comparing the expressions obtained above to the expression obtained directly from the closed-loop description done in Section 6.3.3.

The sensitivity functions can be expressed as

$$\boldsymbol{S}(z) = \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{vcl} + \boldsymbol{I} \qquad (6.79)$$

and

$$\boldsymbol{T}(z) = \boldsymbol{C}_{ycl}(z\boldsymbol{I} - \boldsymbol{A}_{cl})^{-1}\boldsymbol{B}_{rcl} \qquad (6.80)$$

6.5 Summary

In this chapter a closed-loop description of the unconstrained MPC for the linear process model has been derived. From the closed-loop description the closed-loop steady-state covariance matrices for the state, control input, measurement and controlled output were computed. For the linear process model, the controller state-space and the closed-loop state-space model, $z-$transformed expressions have been computed.

Finally, a sensitivity analysis of the closed-loop description has been conducted and the sensitivity functions have been expressed in terms of the closed-loop state-space matrices.

Chapter 7

Tuning

The purpose of this chapter is to investigate tuning of model predictive controllers.

The chapter is organized as follows. The first part of the chapter considers different measures for assessing performance of the closed-loop control system. The performance measures are related to the deterministic, stochastic and sensitivity properties of the control system. The second part of the chapter considers the formulation of an optimization based tuning approach.

7.1 Introduction

One of the main purposes of controlling an industrial process is to make the controlled output of the process follow some predefined reference trajectory. In any industrial application this will always be done under the presence of model uncertainty, different noise sources and disturbances, which complicate the objective.

In order to obtain the best possible control of the process, the controller needs to be tuned. This means to determine the most suitable set of tuning parameters for the control system. However, what the best possible control is, can depend on what type of process is controlled and is therefore somewhat subjective.

A strategy to quantified the quality of a controller is to setup different performance measures for the control system and subsequently evaluate these performance measures for different scenarios according to the tasks of the control system.

The general considerations to take into account when tuning a controller, are listed below.

- Disturbance rejection
- Reference tracking
  - Step change in reference
- Robust performance
  - Sensitivity to disturbances
  - Sensitivity to process and measurement noise
  - Sensitivity to plant-model mismatch

7.2 Performance Measures

In this section different performance measures for the closed-loop system are considered. The closed-loop performance measures are divided into three categories: Deterministic measures, stochastic measures and sensitivity/robustness measures.

7.2.1 Deterministic Measures

Two classic ways to assess the performance of a controller are the controllers ability to track changes in the reference and to reject disturbances on the controlled outputs. There are different ways of quantifying the controllers ability to do this.

One such measure is the settling time for the controlled output, which describes the time from the disturbance (or the change in reference) enters the system, until the time when the controlled output is back at the reference level (or has reached the new reference level). This measure does, however, not provide any information about the deviation between the controlled output and the reference.

Another class of measures is integrated error measures. In these measures the error between the reference $\boldsymbol{r}_k$ and the controlled output $\boldsymbol{y}_k$ is computed in some form. The most common measures are the integrated absolute error (IAE) and the integrated squared error (ISE). These measures do on the other hand, not include any information of the duration time of a reference step or a disturbance.

A measure that combines information of the error and duration time, is the time-weighted integrated (absolute or squared) error. In this measure the individual errors are weighted with the current time. In effect this penalizes deviations for long time periods heavily and will, when the measure is minimized, result in controllers that more rapidly eliminate disturbances.

In this project only the integrated error measures are considered.

The integrated absolute error (IAE) is given by

$$J_i = \sum_{k=0}^{n_f-1} |y_{i,k} - r_{i,k}| \tag{7.1}$$

where $i = 1, 2, \ldots, n_y$ and n_f is the number of samples in the simulation.

The IAE measure computes the absolute difference between measurement $y_{i,k}$ and the corresponding reference $r_{i,k}$ at samples $k = 0, 1, \ldots, n_f$ under the presence of either a step disturbance or a step change in the reference. The IAE corresponds to the numerical (Euler) integral of the error.

The integrated squared error (ISE) is given by

$$J_i = \sum_{k=0}^{n_f-1} (y_{i,k} - r_{i,k})^2 \tag{7.2}$$

In the ISE measure the error between measurement $y_{i,k}$ and the corresponding reference $r_{i,k}$ is squared. For the point of view of minimization of the measure this implies that large deviations are penalized more than smaller ones.

The measures are evaluated by simulation of the closed-loop system (6.11)-(6.14) for specified reference and disturbance scenarios, $\{\boldsymbol{r}_k\}_{k=0}^{n_f-1}$ and $\{\boldsymbol{d}_k\}_{k=0}^{n_f-1}$. The measures are only evaluated for the deterministic part of the closed-loop system, that is for the case where $\boldsymbol{w}_k = 0$ and $\boldsymbol{v}_k = 0$. This is done to be able to asses the performance of the controller only due to deterministic changes and such that it does not depend on a specific realization of noise. Furthermore, the system is started in steady state $\boldsymbol{x}_0^{cl} = \mathbf{0}$, such that only deviation due to disturbances or reference changes are considered.

For MIMO systems, a step disturbance or a reference change on one controlled output generally affects all the other controlled outputs of the system. This does, however, depend to some extent on the specific system and the controller. Based on this behavior it is therefore necessary to compute the desired measure for all combinations of measurements $y_{i,k}$ and step disturbances $d_{j,k}$ and all combinations of measurements $y_{i,k}$ and reference steps $r_{j,k}$ individually. Hereby obtaining the effect of a specific disturbance or reference step on all the controlled outputs.

The components of the measure can subsequently be organized in a matrix or all the individual terms can be summed.

7.2.2 Stochastic Measures

For the closed-loop unconstrained MPC description derived in Chapter 6, the closed-loop covariance matrices can be regarded as stochastic measures. The matrices quantify the steady-state variance and covariance of the measured outputs, controlled output and control input due to the exogenous stochastic noise signals $\boldsymbol{v}_k$ and $\boldsymbol{w}_k$.

The closed-loop covariance matrices associated with the measured outputs, controlled outputs and control inputs can be used to asses the controllers performance under the presence of noise. For a specified set of tuning parameters an associated set of closed-loop covariance matrices is obtained.

For a SISO system these are all scalar quantities and are easy to compare individually and furthermore for different choices of tuning parameters. However, for the MIMO system case, these are matrices and can therefore not directly be compared. To be able to compare the covariance matrices, different scalar measures of the size of a matrix are presented.

The measures used in this project comes from the field of optimal design of experiments [MH97]. The measures are referred to as the

- A-criterion

- D-criterion
- E-criterion

and are described in the following.

The A-criterion is given by the average trace of the covariance matrix

$$\frac{1}{n} tr(\boldsymbol{R}) \tag{7.3}$$

where tr denotes the trace, defined by the sum of the diagonal elements of the matrix and n is the dimension of the covariance matrix $\boldsymbol{R}$. From an optimization point of view, this criterion seeks to minimize the average variance of the covariance matrix and does not take the covariance terms into consideration.

The D-criterion is given by the determinant of the covariance matrix

$$Det(\boldsymbol{R}) \tag{7.4}$$

where Det denotes the determinant. From an optimization point of view, the D-criterion can be interpreted as the volume of the confidence ellipsoid.

The E-criterion is given by the maximum eigenvalue of the covariance matrix

$$\lambda_{max}(\boldsymbol{R}) \tag{7.5}$$

where λ_{max} denotes the maximum eigenvalue. This criterion corresponds to reducing the major axis of the confidence ellipsoid of the covariance matrix $\boldsymbol{R}$ and hereby minimizing the largest variance of the system [MH97].

In general, from a performance/tuning point of view, both the variance of the control input and the outputs should be as low as possible. However, it is well known from the classical *Minimal variance controller* [Pou07] that reducing the variance of the controlled output to the minimal, comes at the expense of increased control input variance. Therefore the variance should be a compromise between the two.

7.2.3 Sensitivity Measures

The term robustness in general refers to a controllers ability to uphold an acceptable performance under uncertainty. A controller is said to be robust if the control performance does not change much if the controller is applied to a system which is different from the one used in the synthesis of the controller. For most industrial application the model of the process used to design the controller is different from the actual process. This is especially critical to MPC, since the controller design relies heavily on the observer model.

In classical control theory, robustness of closed-loop control systems is ensured by requiring sufficiently large gain and phase margins. This is in practice done by visual inspection of open-loop Bode plots.

According to [SP01] the maximum sensitivity M_S, which corresponds to the maximum magnitude of the sensitivity function $S(z)$ in the frequency domain

$$M_S = \max_{\omega} \left| S(e^{j\omega T_s}) \right| \tag{7.6}$$

can be used to quantify robustness and performance for closed-loop SISO control systems. Furthermore, the gain and phase margins are related to the maximum peak of the sensitivity function $S(z)$ by

$$GM \geq \frac{M_S}{M_S - 1}, \quad PM \geq 2arcsin\left(\frac{1}{2M_S}\right) \geq \frac{1}{M_S} \tag{7.7}$$

This implies that the maximum sensitivity M_S can be used to ensure acceptable gain and phase margins simultaneously.

For MIMO systems the sensitivity function $\boldsymbol{S}(z)$ is a matrix, describing the sensitivity for each controlled (or measured) output due to disturbances and/or measurement noise. For these systems the notion of system direction is important. That the system has a direction mainly describes the system's gain for different input directions. The system might have directions where it is very sensitive to noise and others where it is insensitive. This complicates the description of maximum sensitivity of MIMO systems.

The singular values of the frequency response matrix $\boldsymbol{S}(e^{j\omega T_s})$ for the sensitivity function can be interpreted as gains of the system for different frequencies ω [SP01]. Furthermore, the maximum gain, for any input direction, can be expressed as the maximum singular value for a given frequency ω.

According to [DS81], the maximum of the maximal singular values of the frequency response for the sensitivity function is a good measure of the worst case sensitivity of a MIMO system. This is commonly denoted as the H_∞ norm of the sensitivity function $S(z)$ and is defined as

$$M_S = \|\boldsymbol{S}(z)\|_\infty = \max_\omega \bar{\sigma}(\boldsymbol{S}(e^{j\omega T_s})) \tag{7.8}$$

where $\bar{\sigma}(\cdot)$ denotes the maximum singular value and T_s is the sample time. The singular values of $\boldsymbol{S}(e^{j\omega T_s})$ for each frequency ω can be obtained by a singular value decomposition (SVD).

7.3 The Tuning Problem

The tuning of a model predictive controller for a $(n_y \times n_u)$ MIMO system consists of determining the n_y diagonal elements of the reference weight matrix $\boldsymbol{Q}_y$ and the n_u diagonal elements of the control input weight matrix $\boldsymbol{S}_u$ in the objective function:

$$\min_{\{\boldsymbol{u}_{k+j|k}\}_{j=0}^{N-1}} \quad \phi = \frac{1}{2}\sum_{j=0}^{N-1} \left\|\hat{\boldsymbol{y}}_{k+1+j|k} - \boldsymbol{r}_{k+1+j|k}\right\|_{\boldsymbol{Q}_y}^2 + \left\|\Delta\boldsymbol{u}_{k+j|k}\right\|_{\boldsymbol{S}_u}^2$$

Furthermore, the n_d parameters of the disturbance model, should also be determined.

In general the prediction horizon N is also regarded as a tuning parameter. However, it is assumed that the prediction horizon is chosen sufficiently long, such that an infinite horizon controller is approximated and hereby eliminating this parameter from the tuning problem.

7.3.1 Optimization based Tuning

In order to determine the tuning parameters a constrained optimization problem is formulated. The objective function of the optimization problem is related to the performance measures for the closed-loop control system presented in Section 7.2.

To ensure robustness of the resulting control system, an upper bound $M_{S,max}$ is defined for the maximum peak of the sensitivity function (7.8). Furthermore, bounds are defined on the tuning parameters.

The constrained optimization problem can be stated as

$$\min_{\boldsymbol{x}} \quad J(\boldsymbol{x}) \tag{7.9}$$

$$s.t. \quad M_S(\boldsymbol{x}) \leq M_{S,max} \tag{7.10}$$

$$\boldsymbol{x}_{min} \leq \boldsymbol{x} \leq \boldsymbol{x}_{max} \tag{7.11}$$

where $J(\boldsymbol{x})$ represents a specific performance measure and $\boldsymbol{x}$ is a vector containing the tuning parameters. The tuning parameters are organized in the vector as $\boldsymbol{x} = [\alpha_1, \ldots, \alpha_{n_d}, s_1, \ldots, s_{n_u}, q_1, \ldots, q_{n_y}]^T$.

It should be noted that this optimization problem is non-linear, due to the constraint (7.10). This type of constrained optimization problem is also referred to as a Non-Linear Program (NLP).

Stating the tuning of the MPC as a constrained optimization problem has both advantages and disadvantages. The main advantage is that all the tuning parameters can be determined simultaneously and that only solutions that result in sufficiently robust controllers are obtained. The primary disadvantage is that the optimization problem cannot in general be expected to be convex. This implies that a global minimum of the optimization problem can't be guaranteed.

7.3.2 Tuning Algorithm

The optimization problem is solved using the built-in MATLAB solver `fmincon`. To use this solver two functions have to be provided: a function for the evaluation of the objective function $J(\boldsymbol{x})$ and a function which evaluates the constraint $M_S(\boldsymbol{x})$. Furthermore, the solver also requires a starting point $\boldsymbol{x}_0$.

The evaluation function of the objective function and the function computing the maximum sensitivity are both computed based on simulation results of the closed-loop state-space model.

The algorithm for the computation of the closed-loop state-space model is based on the deterministic-stochastic model described in Section 3.2. The deterministic part of the model is obtained from the system identification step and subsequently combined with the stochastic part.

Based on the deterministic-stochastic model an unconstrained MPC is designed and the explicit control law is derived. The explicit control law is then combined with the Kalman filter recursions for the controller model, into a controller state-space. Finally, the controller state-space is combined with the process model to form the closed-loop state-space model.

Using the closed-loop state-space model, scenario simulation can be conducted and the sensitivity function can be set up and evaluated. The evaluation functions are implemented in the two function objfun and confun as seen in Algorithm 1.

`fmincon` allows the user to specify the internal solver algorithm. For the Matlab version (R2009b Mac) used for this thesis the available algorithms are an active set or an interior point algorithm. These are specified in the field `options`. In `options`, different solver termination conditions can also be specified. These includes the maximum number of iterations and function evaluations allowed, minimum step size for the variable $\boldsymbol{x}$ and tolerance for the minimum change in the objective function. All options have default values and can be changed if necessary.

Algorithm 1 illustrates in pseudo code how `fmincon` is used to solve the optimization problem .

Algorithm 1 Tuning Algorithm

```
Defining plant and model:
plant = $(A_p, B_p, C_p)$
model = $(A, B, C, T_s, N)$
sys = [plant; model]
Setting bounds on tuning parameters:
lb = $x_{min}$
ub = $x_{max}$
Computing solution:
x = fmincon(@(x)objfun(sys, x), x0, lb, ub, @(x)confun(sys, x), options);
```

All Matlab functions used in the tuning approach and in the thesis in general are included in Appendix A.

7.4 Tuning based on Identified Models

In this thesis it is assumed that the transfer function from input to output $\hat{\boldsymbol{G}}(s)$ is available from system identification, while the transfer functions from disturbance to output and from process noise to output are not available. This assumption in principle implies that scenario simulation can be conducted for reference scenarios only.

It is furthermore assumed that the distributions of the process and measurement noise are unknown. This implies that the closed-loop covariance matrices cannot be computed and subsequently that the optimization problem can't be based on objective functions related to the stochastic properties of the closed-loop system.
The objectives considered for the tuning procedure are the deterministic measures.

In the following two alternative approaches for the formulation of the closed-loop system are considered. The approaches incorporate the assumptions stated above.

7.4.1 Approach I

In this approach the identified model is used for the controller design and a perturbation of this model is used for the plant and furthermore to formulate the deterministic part of the closed-loop system.

To have offset free control for the resulting tuning, unknown disturbances are emulated by plant-model mismatch. The plant-model mismatch is obtained using the identified model and the perturbation of this model for the plant.

The perturbation is conducted by modifying the gains, time constants or delays of the identified model. It is crucial to use a modified plant model in order to ensure offset free controllers are generated from the optimization problem. Using the correct model would result in a situation where the output predictions match the plant model and hereby eliminate the need for integration [OJ13].

The perturbed model representing the plant is realized as a discrete-time state-space model

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_p \boldsymbol{x}_k + \boldsymbol{B}_p \boldsymbol{u}_k \tag{7.12}$$

$$\boldsymbol{y}_k = \boldsymbol{C}_p \boldsymbol{x}_k \tag{7.13}$$

and subsequently used to formulate the closed-loop system.

Using this approach to conduct scenario simulation, it is necessary that the controller is equipped with integrators to provide offset free control, since the plant-model mismatch has been introduced.

Using this approach the objective function of the optimization problem is based on the reference scenario simulation only. However, due to the plant-model mismatch offsets are now present and integrators will have to compensate for these offsets, hereby incorporating disturbances in the simulation.

7.4.2 Approach II

In some situations the identified model might be very accurate despite the fact that it has been identified under noisy conditions. In such a case it is of interest to use the identified model both in the synthesis of the controller and as the process model used in the closed-loop formulation.

To ensure offset free control for the resulting tuning using this approach, input and output disturbances are added to the process model. This is done, as in approach I, to create a plant-model mismatch. Due to the fact that the true disturbance(s) are unknown, pure input and output disturbances are used. The reason being that these only require knowledge of the identified part of the process model, to be simulated.

These disturbances might not represent the true disturbances well, but they will ensure that the output predictions do not match the process model and that the optimization problem results in parameters which ensure offset free control.

The linear state-space model used to represent the plant is in this case given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}(\boldsymbol{u}_k + \bar{\boldsymbol{u}}_k) \tag{7.14}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \bar{\boldsymbol{y}}_k \tag{7.15}$$

where $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ are the matrices of the identified model and $\bar{\boldsymbol{u}}_k$, $\bar{\boldsymbol{y}}_k$ are step input and output disturbances, respectively.

In this approach, it is also necessary that the controller is equipped with integrators to provide offset free control.

Using this approach the optimization can be based on both reference scenario and disturbance scenario simulation.

7.5 Summary

In this chapter the importance of tuning was introduced. To quantify the performance of a control system different performance measures have been presented and discussed. These have furthermore been used to formulate a tuning approach which is based on constrained optimization. In the optimization problem a bound on the maximum sensitivity was formulated to ensure robustness of the resulting control system. Finally, two approaches to conduct the tuning based on deterministic identified models were presented.

Part II

Application

Chapter 8

Case Study - Introduction

The second part of the thesis presents a case study. In this case study a tank process is considered and will be used to illustrate the theory studied in the first part of the thesis.

The first chapter introduces the tank process and presents a first principle non-linear model governing the process. Furthermore, the objective of regulating the process is presented and the different variables of the process are classified.

In the second chapter a linear model of the tank process is obtained by system identification using input-output data from the process. The identification is conducted for two cases: a case without process- and measurement noise and a case where noise is present.

The last chapter combines the identified linear model with a disturbance model in order to obtain offset free control. Furthermore, this chapter designs the MPC based on the combined model. The chapter also considers the tuning of the designed MPC. Finally, the designed model predictive controller with specified tuning parameters is tested in closed-loop simulations with the non-linear tank process.

Chapter 9

Case Study - The Modified 4-tank System

In this chapter, the modified 4-tank system is presented. The modified 4-tank system is a modification of the 4-tank system considered in [Joh97], [JN98], [JHWH99] and [Joh00]. The main modification in comparison to the original 4-tank system is the addition of two unknown disturbances entering the top tanks (tank 3 and tank 4 in Figure 9.1).

9.1 Process Description

The modified 4-tank system is an interconnected tank system consisting of 4 water tanks, 2 pumps and a reservoir. A process diagram of the system is shown in Figure 9.1.

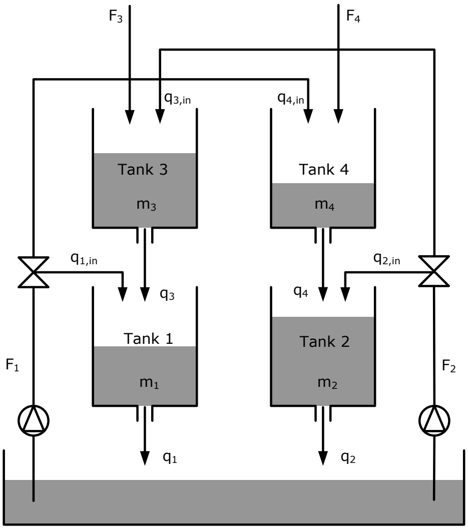

Figure 9.1: Process diagram of the modified 4-tank process.

The flow rates F_1 and F_2 are associated to water stream 1 and 2. For stream 1 the fraction γ_1 of the flow goes into tank 1 while the remaining fraction $1 - \gamma_1$ flows into tank 3. Similarly, the fraction γ_2 of stream 2 flows into tank 2 while the remaining fraction $1 - \gamma_2$ flows into tank 3. The outflow from tank 3 goes flows into tank 1, and the outflow from tank 4 flows into tank 2.

The modification is the addition of stream 3 and stream 4 with the respective flow rates, F_3 and F_4. In this thesis, these flow rates are disturbances that are unknown to the controller.

9.2 Process Model

The equations governing the modified 4-tank system are given by a set of ordinary differential equations (ODEs). The equations are obtained by performing a mass balance on each tank and are given by [Jør12]

$$\frac{dm_1(t)}{dt} = \rho q_{1,in}(t) + \rho q_3(t) - \rho q_1(t) \tag{9.1}$$

$$\frac{dm_2(t)}{dt} = \rho q_{2,in}(t) + \rho q_4(t) - \rho q_2(t) \tag{9.2}$$

$$\frac{dm_3(t)}{dt} = \rho F_3(t) + \rho q_{3,in}(t) - \rho q_3(t) \tag{9.3}$$

$$\frac{dm_4(t)}{dt} = \rho F_4(t) + \rho q_{4,in}(t) - \rho q_4(t) \tag{9.4}$$

with initial conditions

$$m_i(t_0) = m_{i,0}, \quad i = 1,2,3,4 \tag{9.5}$$

and where

$$q_{1,in}(t) = \gamma_1 F_1(t) \tag{9.6}$$

$$q_{2,in}(t) = \gamma_2 F_2(t) \tag{9.7}$$

$$q_{3,in}(t) = (1-\gamma_2) F_2(t) \tag{9.8}$$

$$q_{4,in}(t) = (1-\gamma_1) F_1(t) \tag{9.9}$$

are the inflows into the individual tanks. Here γ_1 and γ_2 are flow distribution constants for valves 1 and 2. Furthermore, $F_1(t)$ and $F_2(t)$ are the input flow rates (volume per time) coming from the two pumps.

The outflow from each tank is given by

$$q_i(t) = a_i\sqrt{2gh_i(t)}, \quad i = 1,2,3,4 \tag{9.10}$$

where a_i is the area of outlet pipe i, g is the gravitational force and $h_i(t)$ is the water height (level) in tank i. The water height $h_i(t)$ in each tank is described by

$$h_i(t) = \frac{1}{\rho A_i} m_i(t), \quad i = 1,2,3,4 \tag{9.11}$$

where A_i is the cross sectional area of tank i and ρ is the density of water. The mass of the water in each tank (as a function of time) is denoted by $m_i(t)$.

The terms $F_3(t)$ and $F_4(t)$ are flow rates describing the flows entering tanks 3 and tank 4, these flows enter the system as disturbances. This means that these are not controlled by the operator of the system. The disturbances can be either deterministic or stochastic (or a combination).

It is noted that the modified 4-tank system is a non-linear process, due to the non-linear relationship in (9.10).

Equations (9.1)-(9.11) constitute the process model of the modified 4-tank system. For this process the manipulated variables (MVs) are the two pump flow rates $F_1(t)$ and $F_2(t)$ and the controlled variables (CVs) are for this case study the levels in tank 1 and tank 2, that is $h_1(t)$ and $h_2(t)$. Furthermore, the disturbance variables (DVs) are the two additional flow rates $F_3(t)$ and $F_4(t)$.

The model of the process can be expressed as a system of first-order ordinary differential equations

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)) \tag{9.12}$$

$$\boldsymbol{y}(t) = g(\boldsymbol{x}(t)) \tag{9.13}$$

$$\boldsymbol{z}(t) = h(\boldsymbol{x}(t)) \tag{9.14}$$

with initial condition

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{9.15}$$

and vectors defined by

$$\boldsymbol{x}(t) = \begin{bmatrix} m_1(t) \\ m_2(t) \\ m_3(t) \\ m_4(t) \end{bmatrix}, \quad \boldsymbol{u}(t) = \begin{bmatrix} F_1(t) \\ F_2(t) \end{bmatrix}, \quad \boldsymbol{d}(t) = \begin{bmatrix} F_3(t) \\ F_4(t) \end{bmatrix} \tag{9.16}$$

$$\boldsymbol{y}(t) = \begin{bmatrix} h_1(t) \\ h_2(t) \end{bmatrix}, \quad \boldsymbol{z}(t) = \begin{bmatrix} h_1(t) \\ h_2(t) \end{bmatrix} \tag{9.17}$$

Equation (9.12) is referred to as the process equation, while equations (9.13)-(9.14) are referred to as the measurement equation and the output equation, respectively.

The parameters of the modified 4-tank system are listed in Table 9.1. These parameters are the same as used in [Jør12].

Table 9.1: Parameters for the modified 4-tank system.

Description	Symbol	Value	Unit
Pipe cross sectional area	a_i	1.2272	cm^2
Tank cross sectional area	A_i	380.1327	cm^2
Acceleration of gravity	g	981	$\mathrm{cm/s}^2$
Density of water	ρ	1.00	$\mathrm{g/cm}^3$
Valve distribution constant	γ_1	0.45	
Valve distribution constant	γ_2	0.40	

The valve distribution constants could be varied, however, in this case study these are assumed to be fixed.

9.3 Simulation of the Modified 4-Tank Process

For the purpose of simulating the modified 4-tank process the continuous-time process model is transformed into a discrete-time process model.

The state can at discrete times t_k be expressed as

$$\boldsymbol{x}(t_k) = \boldsymbol{x}_k$$

Assuming that the manipulated variables and the disturbance variables are constant in each sample interval

$$\begin{aligned} \boldsymbol{u}(t) &= \boldsymbol{u}_k, \quad t_k \leq t < t_{k+1} \\ \boldsymbol{d}(t) &= \boldsymbol{d}_k, \quad t_k \leq t < t_{k+1} \end{aligned}$$

where the sample times are given by $t_k = t_0 + kT_s$ and T_s is the sample time. The state at the next discrete time $\boldsymbol{x}(t_{k+1}) = \boldsymbol{x}_{k+1}$ can be expressed as

$$\boldsymbol{x}_{k+1} = F(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{d}_k) \tag{9.18}$$

where

$$F(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{d}_k) = \boldsymbol{x}_k + \int_{t_k}^{t_{k+1}} f(\boldsymbol{x}(t), \boldsymbol{u}_k, \boldsymbol{d}_k) dt \tag{9.19}$$

and measurement and output can be expressed as

$$\boldsymbol{y}_k = g(\boldsymbol{x}_k) \tag{9.20}$$

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k) \tag{9.21}$$

9.3.1 Stochastic Simulation Model

The model (9.18)-(9.21) is a deterministic system. For the purpose of representing the process more realistically, the process model is extended to include process- and measurement noise.

In this project the disturbance is modeled as having a deterministic and a stochastic component. The deterministic component, denoted by $\boldsymbol{d}_k^{det}$, represents the mean value of the flow rates $F_3(t)$ and $F_4(t)$ for the flows entering tanks 3 and tank 4 and may change stepwise. The stochastic component represents a random variation of the mean and is assumed to be Gaussian white-noise. The stochastic component is considered the process-noise in this case study and is denoted by $\boldsymbol{w}_k$. Furthermore, it is assumed to be constant in each sample interval.

The discrete-time disturbance used for simulation of the process is given by

$$\boldsymbol{d}_k = \boldsymbol{d}_k^{det} + \boldsymbol{w}_k \tag{9.22}$$

where

$$\boldsymbol{w}_k \sim \mathcal{N}_{iid}(\mathbf{0}, \boldsymbol{Q}) \tag{9.23}$$

Furthermore, the disturbance is modeled as only being able to enter the tanks, that is the disturbance should not be able to remove water from the tanks. This requirement implies that the process-noise $\boldsymbol{w}_k$ should not make $\boldsymbol{d}_k^{det}$ negative and can be ensured by choosing the covariance matrix accordingly.

The measurements of the water levels (heights) in tanks 1 and 2 are assumed to be influenced by measurement noise. The measurement noise is assumed to be Gaussian white-noise. The measurements are given by

$$\boldsymbol{y}_k = g(\boldsymbol{x}_k) + \boldsymbol{v}_k \tag{9.24}$$

where

$$\boldsymbol{v}_k \sim \mathcal{N}_{iid}(\mathbf{0}, \boldsymbol{R}) \tag{9.25}$$

The discrete-time simulation model for the modified 4-tank process may be expressed as

$$\boldsymbol{x}_{k+1} = F(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{d}_k) \tag{9.26}$$

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k) \tag{9.27}$$

$$\boldsymbol{y}_k = \boldsymbol{z}_k + \boldsymbol{v}_k \tag{9.28}$$

where

$$F(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{d}_k) = \boldsymbol{x}_k + \int_{t_k}^{t_{k+1}} f(\boldsymbol{x}(t), \boldsymbol{u}_k, \boldsymbol{d}_k) dt \tag{9.29}$$

$$\boldsymbol{w}_k \sim \mathcal{N}_{iid}(\mathbf{0}, \boldsymbol{Q}) \tag{9.30}$$

$$\boldsymbol{v}_k \sim \mathcal{N}_{iid}(\mathbf{0}, \boldsymbol{R}) \tag{9.31}$$

9.4 Control Objective

The objective of controlling the modified 4-tank system is to regulate the water heights $h_i(t)$ in tank 1 and tank 2, to water height references $r_i(t)$. For the situation where $F_3(t)$ and $F_4(t)$ are unmeasured disturbances entering tanks 3 and 4.

In the discrete-time setting, this corresponds to regulating the measured output $\boldsymbol{y}_k$ to the reference vector $\boldsymbol{r}_k$.

Chapter 10

Case Study - System Identification

In this chapter a linear model for the modified 4-tank process is identified from input-output data. In order to obtain the input-output data, step tests are conducted on the modified 4-tank process.

10.1 Step Tests

For the purpose of generating the process data the non-linear modified 4-tank process model (9.26)-(9.29) is used to represent the true process. It is assumed that the process model is completely unknown (a black box model) for the purpose of identification. The reason for this assumption is to make the identification procedure as real as possible to achieve what would be encountered in real industrial applications.

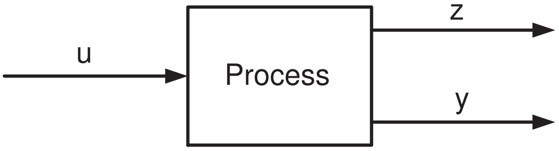

Figure 10.1: Black box model representation of the process.

Figure 10.1 illustrates the black box assumption. z represents the true output of the process. However, this is not what is actually measured due to measurement noise. y is the measured output of the process.

The input-output data are generated by conducting individual steps in the manipulated variables and subsequently recording the measured output $\boldsymbol{y}(t_k) = \boldsymbol{y}_k$ of the process.

The manipulated variables are piecewise constant in each sample interval

$$\boldsymbol{u}(t) = \boldsymbol{u}_k, \quad t_k \leq t < t_{k+1}$$

The sample instances are defined by $t_k = t_0 + kT_s$ where t_0 is the start time and T_s is the sample time.

10.1.1 Test 1 - No Process and Measurement Noise

The first set of step tests are conducted without the presences of process- and measurement noise. This is mainly done to clearly illustrate the identification procedure and to have an ideally identified model to compare to models obtained under the presence of process- and measurement noise.

The process is simulated for 3000 seconds starting at $t_0 = 0$ and the output of the process is measured with the sample-time $T_s = 4$ $[s]$. The resulting number of sampled measurements is $N_{sim} = 750$. For the first 1000 seconds both manipulated variables are kept constant at

$$u_1 = 300 \ [cm^3/s]$$
$$u_2 = 300 \ [cm^3/s]$$

At time $t = 1000$ seconds a step change of 45 $[cm^3/s]$ is introduced in u_1, while u_2 remains constant. The response of the two measured variables y_1 and y_2 is recorded and presented with the manipulated variables in Figure 10.2.

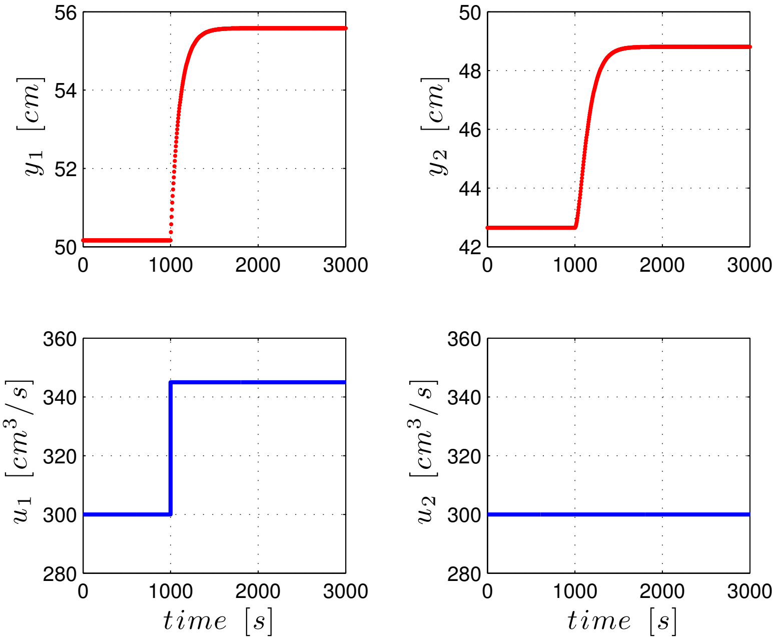

Figure 10.2: Step test data for a step change in u_1.

Figure 10.2 illustrates the response of the process. It is observed that when the step change in u_1 occurs at $t = 1000$, the measured outputs respond instantaneously. This illustrates that there is no time delay present between the input u_1 and the outputs y_1 and y_2. The figure furthermore shows that both outputs clearly respond to the step change in u_1.

The procedure is then repeated for a step change in u_2, while u_1 is kept constant. At time $t = 1000$ seconds, a step change of 45 $[cm^3/s]$ is introduced in u_2. The response of the two measured variables is recorded and presented with the manipulated variables in Figure 10.3.

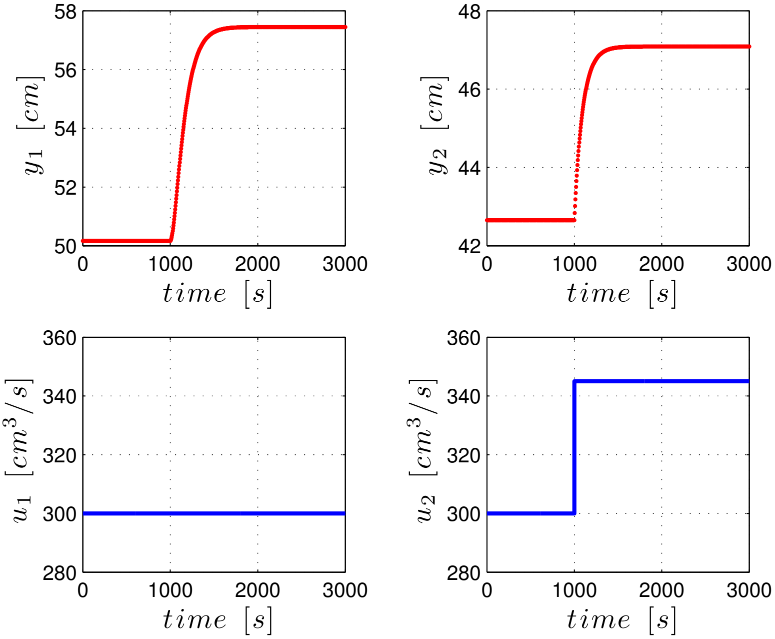

Figure 10.3: Step test data for a step change in u_2.

Figure 10.3 illustrates the response of the process. It is observed that when the step change in u_2 occurs at $t = 1000$, the measured outputs respond instantaneously. This illustrates that there is no time delay present between the input u_2 and the outputs y_1 and y_2. The figure furthermore shows that both outputs also clearly respond to the step change in u_2.

Prior to the step change the outputs are steady at the level $y_{i,s}$, $i = 1, 2$, which is the steady state output corresponding to the constant inputs. When the step change is introduced at time $t = 1000$ the output responds, and after a transient period reaches a new steady state output level. The output data recorded from $t = 1000$ constitute the step response data.

The input-output data obtained from the two step tests are next transformed to unit step response data. This corresponds to the response of the process to a unit step when the process is at steady state. This is done by subtracting the initial steady-state output data from the output response data and dividing by the size of the step in the input

$$\frac{y_i - y_{is}}{\Delta u_j} \tag{10.1}$$

where $i = 1, 2$ is the index of the output and $j = 1, 2$ is the index of the stepped input. The unit step response data corresponding to the step tests conducted above are reported in Figure 10.4.

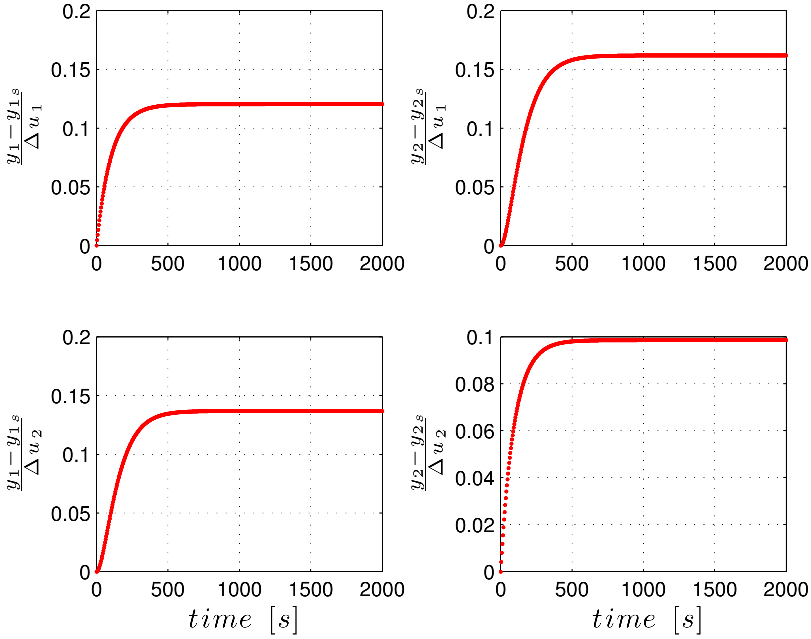

Figure 10.4: Unit step response data.

The unit step response data are then fitted to the analytical expression for the unit step response of a first-order plus dead time system and for a second-order plus dead time system. This is done using non-linear least-squares estimation.

The parameter estimates obtained from the least-squares estimation are presented in Table 10.1 and Table 10.2. Figure 10.5 illustrates the unit step response data along with the estimated unit step response based on the FOPDT and SOPDT models.

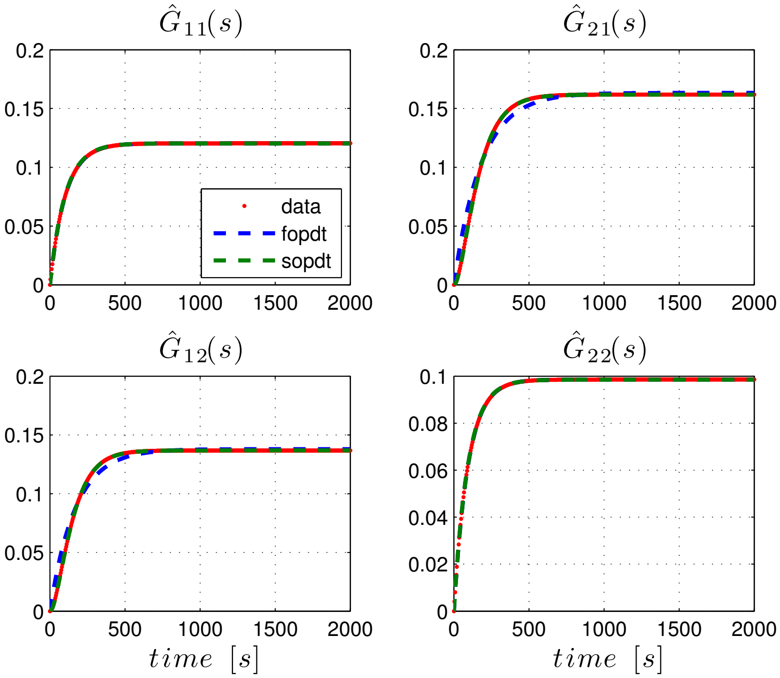

Figure 10.5: Estimated unit step response based on the FOPDT and SOPDT models.

Figure 10.5 shows as expected that the estimated unit step responses are very similar to the data, since no process- and measurement noise is present. It is seen that there is a perfect fit between the data and the estimates for $\hat{G}_{11}(s)$ and $\hat{G}_{22}(s)$ and that this is the case for both the FOPDT and SOPDT models. This illustrates that these are in fact first-order models. For $\hat{G}_{21}(s)$ and $\hat{G}_{12}(s)$ estimated based on the FOPDT structure it is seen there is a small difference from the data and furthermore that this is not the case for the SOPDT. This illustrates that these are in fact second-order models.

Table 10.1: Estimated parameters for the first-order plus dead time models based on the unit step response data.

	$\hat{K}_{ij}$	$\hat{\tau}_{ij}$	$\hat{\theta}_{ij}$
$\hat{G}_{11}(s)$	0.1203	102.9214	0
$\hat{G}_{12}(s)$	0.1633	182.0347	0
$\hat{G}_{21}(s)$	0.1379	168.6652	0
$\hat{G}_{22}(s)$	0.0985	94.7909	0

Table 10.2: Estimated parameters for the second-order plus dead time models based on the unit step response data.

	$\hat{K}_{ij}$	$\hat{\tau}_{ij}^{(1)}$	$\hat{\tau}_{ij}^{(2)}$	$\hat{\theta}_{ij}$
$\hat{G}_{11}(s)$	0.1203	4.4934e-05	101.7178	0
$\hat{G}_{12}(s)$	0.1619	60.9551	112.0959	0
$\hat{G}_{21}(s)$	0.1368	57.1322	103.5558	0
$\hat{G}_{22}(s)$	0.0985	1.4511e-05	94.0700	0

10.1.2 Test 2 - Process and Measurement Noise

The second set of step tests are conducted in the presence of process- and measurement noise. In the tests the covariance matrices for the process- and measurement noise are given by

$$\boldsymbol{Q} = \begin{bmatrix} 20^2 & 0 \\ 0 & 20^2 \end{bmatrix}, \quad \boldsymbol{R} = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$$

There is not cross correlation between the process noise and the measurement noise. Both the two process noise components and the two measurement noise components are independent.

The information of the noise characteristics and the presence of the disturbance is not used in the actual system identification.

The process is again simulated for 3000 seconds starting at $t_0 = 0$ and the output of the process is measured with the sample-time $T_s = 4$ $[s]$. Similarly, the resulting number of sampled measurements is $N_{sim} = 750$. For the first 1000 seconds both manipulated variables are kept constant at

$$u_1 = 300 \ \ [cm^3/s]$$
$$u_2 = 300 \ \ [cm^3/s]$$

At time $t = 1000$ seconds a step change of 45 $[cm^3/s]$ is introduced in u_1, while u_2 remains constant. The response of the two measured variables y_1 and y_2 is recorded and presented with the manipulated variables in Figure 10.6.

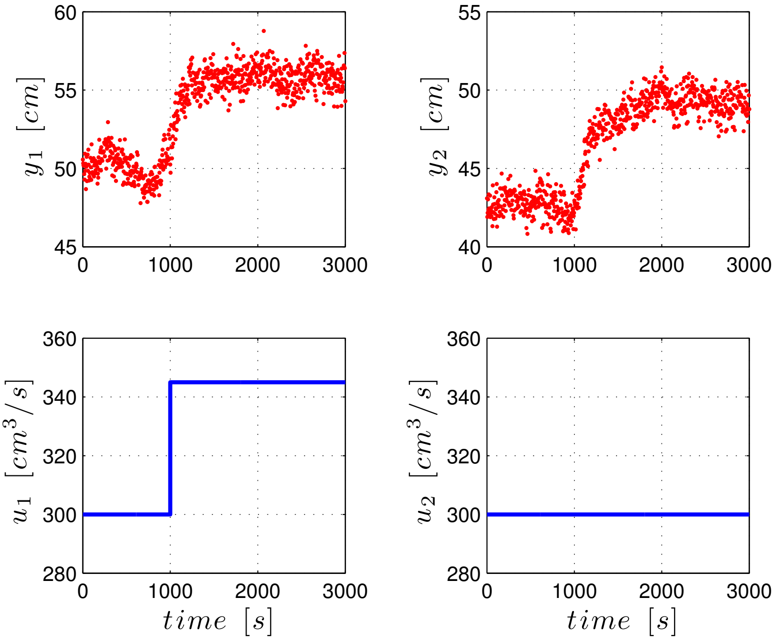

Figure 10.6: Step test data for a step change in u_1, in the presence of process- and measurement noise.

Figure 10.6 illustrates the response of the process to a step change in u_1, in the presence of process- and measurement noise. It is observed that when the step change in u_1 occurs at $t = 1000$, both the measured outputs still respond in a relatively clear way. However, it is seen that before the step change occurs the process is varying significantly and the presence of noise makes it much harder to identify the steady state output levels.

Figure 10.7 illustrates the response of the process to a step change in u_2, in the presence of process- and measurement noise.

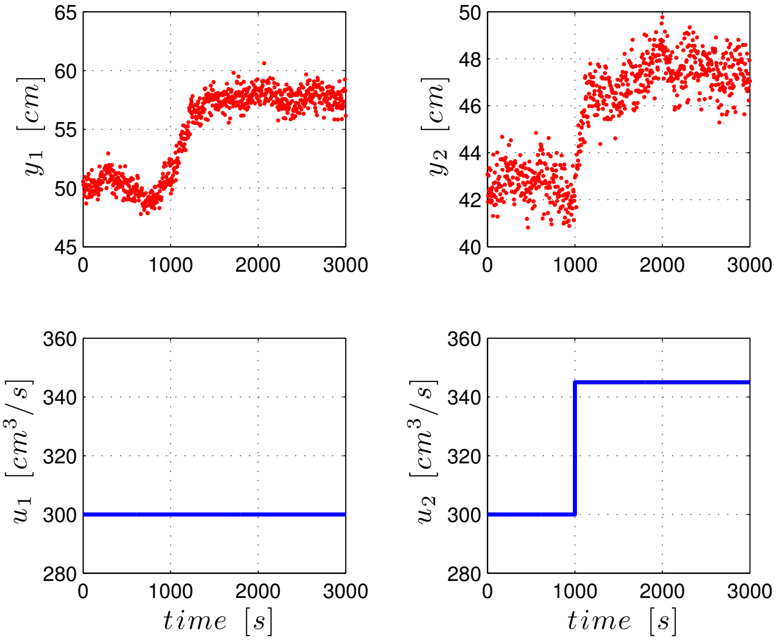

Figure 10.7: Step test data for a step change in u_2, in the presence of process- and measurement noise.

Figure 10.7 illustrates similar behavior to a step in u_1. Since the two step tests are conducted separately the noise contributions are different for the two cases. In this step test y_1 behaves very nicely and a clear trend of the response is seen. On the other hand, the response of y_2 is much affected by the presence of the noise.

In both Figure 10.6 and Figure 10.7 there are no indications of delays being present from the inputs to the outputs.

In order to get an estimate of the steady state output levels, the sample mean of output data recorded from $t_0 = 0$ to $t = 1000$ is computed. This is then used to transform the input-output data obtained from the two step tests to unit step response data, as done in the case without noise. The unit step response data are reported in Figure 10.8.

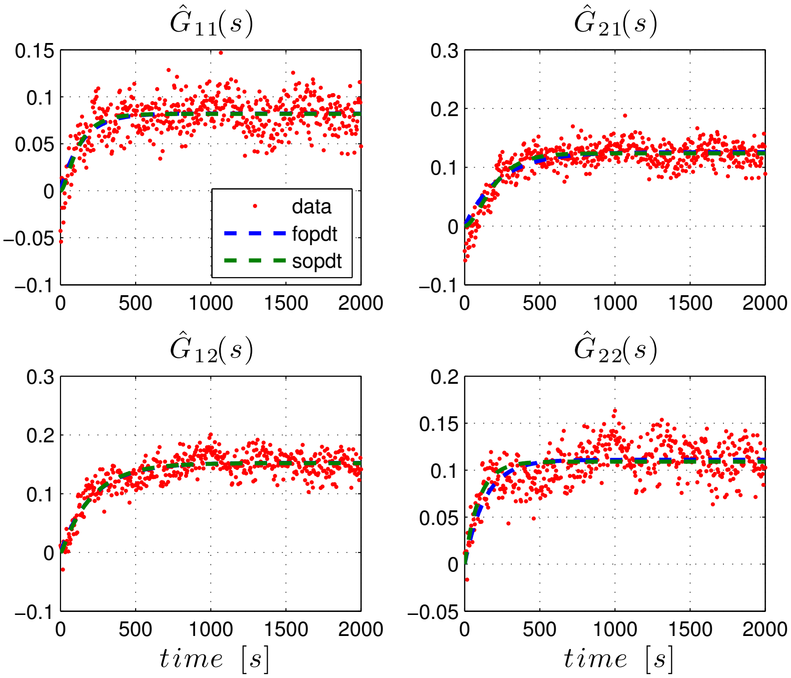

Figure 10.8: Unit step response data obtained in the presence of process- and measurement noise.

The unit step response data are then fitted to the analytical expression for the unit step response of a first-order plus dead time system and for a second-order plus dead time system. This is again done using non-linear least-squares estimation. The parameter estimates obtained from the least-squares estimation are presented in Table 10.3 and Table 10.4.

Table 10.3: Estimated parameters for the first-order plus dead time models based on the unit step response data.

	$\hat{K}_{ij}$	$\hat{\tau}_{ij}$	$\hat{\theta}_{ij}$
$\hat{G}_{11}(s)$	0.0822	140.2627	0
$\hat{G}_{12}(s)$	0.1258	231.0847	0
$\hat{G}_{21}(s)$	0.1523	216.4180	0
$\hat{G}_{22}(s)$	0.1110	137.5618	0

Table 10.4: Estimated parameters for the second-order plus dead time models based on the unit step response data.

	$\hat{K}_{ij}$	$\hat{\tau}_{ij}^{(1)}$	$\hat{\tau}_{ij}^{(2)}$	$\hat{\theta}_{ij}$
$\hat{G}_{11}(s)$	0.0818	25.3224	108.7843	0
$\hat{G}_{12}(s)$	0.1238	104.5181	104.4942	0
$\hat{G}_{21}(s)$	0.1520	7.6230	206.9884	0
$\hat{G}_{22}(s)$	0.1091	9.3216e-06	96.1746	0

Figure 10.9 illustrates the unit step response data along with the estimated unit step response based on the FOPDT and SOPDT models.

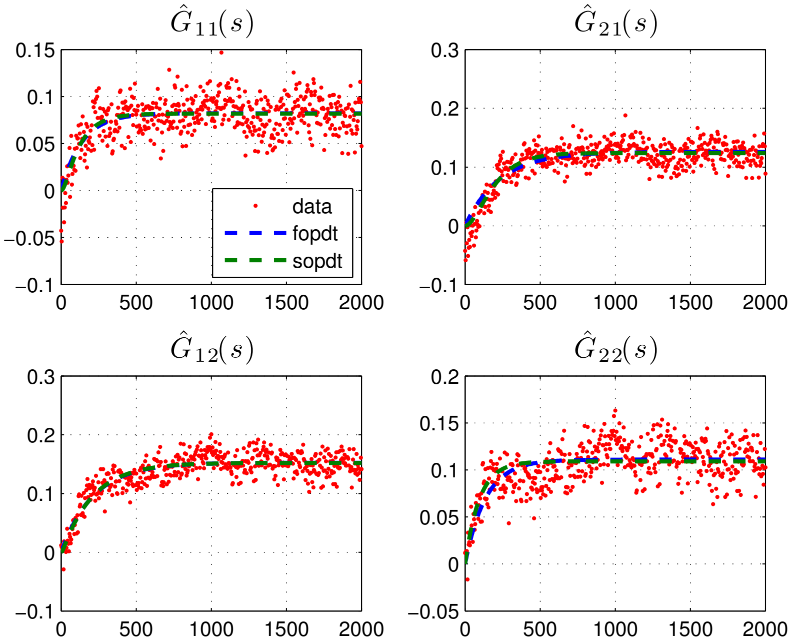

Figure 10.9: Estimated unit step response based on the FOPDT and SOPDT models.

Figure 10.9 illustrates the unit step response data and the estimated unit step responses of the FOPDT and SOPDT models. It is seen that the estimates of the FOPDT and SOPDT models are almost indistinguishable from each other. This observation indicates that it is not possible, based on the estimated responses, to conclude whether first or second order models are most suitable.

In this case study the *correct* model structures are known, however, this would in general not be the case for industrial applications and this knowledge will therefore not be utilized.

The identified linear input-output model for the modified 4-tank process is given by

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \end{bmatrix} = \begin{bmatrix} \hat{G}_{11}(s) & \hat{G}_{12}(s) \\ \hat{G}_{21}(s) & \hat{G}_{22}(s) \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} \tag{10.2}$$

where the transfer functions $\hat{G}_{11}(s)$, $\hat{G}_{12}(s)$, $\hat{G}_{21}(s)$ and $\hat{G}_{22}(s)$ are given in Table 10.3 for the case where all transfer functions are modeled as first order systems and Table 10.4 for the case where all transfer functions are modeled as second order systems.

It should be noted that this identified model is a purely deterministic model and that no models of the noise and/or the disturbances have been identified.

10.2 Summary

In this chapter a linear model for the modified 4-tank system has been obtained by system identification. The identification was based on step tests conducted using the non-linear simulation model for the process. Both a first-order and second-order input-output model were obtained.

Chapter 11

Case Study - Numerical Results

The main purpose of this chapter is to illustrate the optimization based tuning approach on the identified models for the modified 4-tank system. Furthermore, based on the obtained sets of tuning parameters to conduct closed-loop simulation using the non-linear process model for the modified 4-tank system as the plant model.

11.1 Tuning based on Optimization

In this section, the tuning of the MPC based on the identified models is considered. The tuning will be conducted mainly based on the first-order identified model. However, for the purpose of comparison also some tuning tests will be conducted for the second-order identified model.

For all tuning tests the prediction horizon for the MPC is selected to $N = 500$. The choice provides a good approximation to an infinite horizon controller for the entire range of tuning parameters. The designed controllers are all based on the deterministic-stochastic model.

For the modified 4-tank system the tuning parameters to be determined are the diagonal elements q_1, q_2, s_1 and s_2 of tuning weights $\boldsymbol{Q}_y$ and $\boldsymbol{S}_u$ along with the disturbance model parameters of the deterministic-stochastic model α_1 and α_2. This means that the optimization problem contains 6 tuning variables to be determined.

For the evaluation of M_S a vector of frequencies in the range $10^{-4} \leq \omega \leq \pi/T_s$ with a 1000 linearly spaced frequencies is used for all the test. This provides a consistent estimate for the maximum sensitivity peak for all the cases considered.

The results are organized as follows. In the first part of the section tuning is conducted based on the first-order identified model. The second part of the section conducts the tuning based on the second-order identified model.

11.1.1 Tuning based on first-order identified model

This section considers the tuning based on the first-order identified model. The tuning will be conducted for both the approaches described in Section 7.4.
The first part of this section conducts the tuning based on approach I.

Approach I In approach I the plant-model mismatch is conducted by modifying the identified model. This modification is done for the model gains $\hat{K}_{ij}$ and time constants $\hat{\tau}_{ij}$ separately and is obtained by multiplying all the gains or time constants by the same deviation factor δ. The tuning can subsequently be conducted for different levels of deviation.
In this approach only reference scenario simulations are conducted. The reference scenarios used are generated by introducing a step in the individual references for the two outputs.

In the first tuning test the optimization problem is solved for different choices of the maximum sensitivity bound $M_{S,max}$. In this way it is investigated what sensitivity level gives reasonable tuning parameters for the controllers based on the first-order identified model.

The test is conducted based on a gain deviation scenario, where all gains of the plant model are 90 % of the nominal values of the identified first-order model. That is $\delta = 0.9$.
The reference step used to conduct the reference scenario is $\boldsymbol{r}_k = [1 \;\; 1]^T$ and the objective of the optimization J is the sum of the IAE associated with the individual reference scenarios.

The result of the optimizations has been produced using one fixed starting point for all values of the maximum sensitivity bound $M_{S,max}$. The starting point used is given by $\boldsymbol{x}_0 = [0.7 \ 0.7 \ 100 \ 100 \ 1 \ 1]$. The algorithm utilized by `fmincon` is the interior-point algorithm. In this test the default options for the solver is used.

The tuning parameters generated by solution of the optimization problem based on the first-order identified model are listed in Table 11.1.

$M_{S,max}$	α_1, α_2	s_1, s_2	q_1, q_2	δ	J
1.10	$6.1 \cdot 10^{-8}$ $3.4 \cdot 10^{-8}$	109.1 116.8	0.0616 0.0656	0.9	1129.9
1.15	$3.4 \cdot 10^{-9}$ $2.0 \cdot 10^{-9}$	110.8 102.8	0.3495 0.3629	0.9	664.9
1.20	$5.0 \cdot 10^{-5}$ $3.9 \cdot 10^{-5}$	37.9 166.8	1.3395 1.3876	0.9	377.4
1.25	$6.9 \cdot 10^{-5}$ $1.2 \cdot 10^{-4}$	116.8 $2.4 \cdot 10^{4}$	361.2 382.6	0.9	285.1
1.30	$6.5 \cdot 10^{-7}$ 0.9980	128.7 0.0011	36.6 90.5	0.9	130.6
1.50	$5.5 \cdot 10^{-6}$ $1.2 \cdot 10^{-5}$	44.0 $3.1 \cdot 10^{5}$	$3.0 \cdot 10^{5}$ $3.1 \cdot 10^{5}$	0.9	79.1

Table 11.1: Tuning parameters and associated objective function values obtained by optimization based on gain deviation scenario with $\delta = 0.9$ for different values of the maximum sensitivity bound $M_{S,max}$.

From Table 11.1 it is seen that the tuning parameters obtained for all cases expect for one case have $\alpha_1, \alpha_2 \approx 0$, corresponding to full integration. In the exception case, $M_{S,max} = 1.30$, the first output y_1 has full integration $\alpha_1 \approx 0$ while the second output y_2 has almost no integration $\alpha_2 \approx 1$. However, the integrator is not completely disabled so offset free control can be achieved for this case.

It is observed that for $M_{S,max} = 1.25$, $M_{S,max} = 1.30$ and $M_{S,max} = 1.50$ the ratio between the tuning parameters s_1 and s_2 is very large. This will in practice imply that one actuator is changing the control signal very aggressively while the other is only changed very slowly. Combined with the fact that q_1 and q_2 are relatively large, these parameter sets will most likely result in controllers that are useless in practice. The behavior will be investigated later in this chapter, when closed-loop simulations are conducted.

Tests have also been conducted for $M_{S,max} > 1.50$. In these tests similar behavior is observed. In some cases $s_1 < s_2$ and others $s_2 < s_1$ while q_1 and q_1 are always the same order of magnitude and relatively large.

As expected the objective function J increases when the sensitivity bound $M_{S,max}$ is decreased. In particular it is seen that J increases very fast when the value of the bound gets close to 1.

The solutions obtained for $M_{S,max} \leq 1.25$ seem to provide reasonable and intuitive tuning parameters. That is relatively high values for the input rate movement weights s_1 and s_2 and lower values for the reference tracking weights q_1 and q_2.

The results in Table 11.1 are obtained based on the IAE of the individual reference scenarios. Similar tuning parameters are obtained when using the ISE to asses the scenarios.

In order to investigate the effect of the size of the deviation, next a tuning test based on a gain deviation scenario where $\delta = 0.5$ is conducted. This test is done only for three levels of the sensitivity bound $M_{S,max}$. The tuning parameters generated by solution of the optimization problem for this case are listed in Table 11.2.

$M_{S,max}$	α_1, α_2	s_1, s_2	q_1, q_2	δ	J
1.10	$5.1 \cdot 10^{-7}$ $3.0 \cdot 10^{-7}$	129.1 121.3	0.6587 0.6888	0.5	760.0
1.15	0.0040 0.0031	4.2491 204.0406	4.5824 4.6849	0.5	443.0
1.20	$4.5 \cdot 10^{-7}$ $3.8 \cdot 10^{-7}$	$3.8 \cdot 10^{-4}$ 177.9	21.6650 22.4084	0.5	295.7

Table 11.2: Tuning parameters and associated objective function values obtained by optimization based on gain deviation scenario with $\delta = 0.5$ for different values of the maximum sensitivity bound $M_{S,max}$.

From Table 11.2 it is seen that the solution obtained for $M_{S,max} = 1.20$ is very different compared to the solution found in Table 11.1. This solution shares the same characteristics as the solutions obtained for higher values of $M_{S,max}$. When $M_{S,max}$ is lowered it is again seen that much more reasonable tuning parameters are obtained. This behavior suggests that the higher the deviation is chosen the more sensitive the closed-loop becomes. As a consequence a lower bound $M_{S,max}$ is needed to obtain acceptable tuning parameters.

It is also seen that the objective function J for all cases is smaller when compared to the respective cases in Table 11.1. This can be contributed to the deviation level.

Last, a test is conducted based on a time constant deviation scenario, where all the time constants of the plant model are 90 % of the nominal values of the identified first-order model. The tuning parameters generated by solution of the optimization problem for this case are listed in Table 11.3.

$M_{S,max}$	α_1, α_2	s_1, s_2	q_1, q_2	δ	J
1.10	$9.6 \cdot 10^{-9}$ $5.5 \cdot 10^{-9}$	102.1 107.4	0.0573 0.0573	0.9	1120.5
1.20	$4.9 \cdot 10^{-5}$ $3.7 \cdot 10^{-5}$	54.8 149.5	1.3047 1.3523	0.9	357.4
1.30	$2.2 \cdot 10^{-5}$ 1.0000	83.0 0.002	18.7335 48.4741	0.9	96.0

Table 11.3: Tuning parameters and associated objective function values obtained by optimization based on time constant deviation scenario with $\delta = 0.9$ for different values of the maximum sensitivity bound $M_{S,max}$.

From Table 11.3 it is seen that the solutions obtained for $M_{S,max} = 1.10$ and $M_{S,max} = 1.20$ are very similar compared to the solution found in Table 11.1. This indicates that the time constant deviation and the gain deviation have similar effect on the optimization problem. It is furthermore seen that $M_{S,max} = 1.30$ results in a situation where one of the integrators is turned off. The inactive integrator makes this tuning unable to suppress disturbances and ensure offset free control. This case also suffers from the behavior seen previously related to the input weights.

The tests conducted based on Approach I illustrate that the obtained tuning parameters to a large extend depend on the level of deviation and that it is not obvious exactly how this level should be chosen. However, it is seen for all the cases studied that it is possible to obtain reasonable tuning parameters. The tests suggest that the system based on the first-order identified model is very sensitive and consequently $M_{S,max}$ should be chosen very low.

The next part of this section tests the tuning procedure based on approach II. Mainly the same tuning tests as conducted above will be carried out. This is done such that the two approaches can be compared.

Approach II In approach II the optimization is based on a combination of an input disturbance scenario, an output disturbance scenario and a reference scenario. Each of these scenarios is conducted by introducing a step with a specified magnitude in the associated input to the closed-loop system.

The objective J of the optimization is the sum of the IAE associated with the individual scenarios. These are denoted by J_u, J_y and J_r.

In the first tuning test the optimization problem is solved for different choices of the maximum sensitivity bound $M_{S,max}$. In this test the step for the input disturbance is $\bar{\boldsymbol{u}}_k = [1 \;\; 1]^T$, the step in the output disturbance is $\bar{\boldsymbol{y}}_k = [0.1 \;\; 0.1]^T$ and the reference step is $\boldsymbol{r}_k = [0.1 \;\; 0.1]^T$. In terms of scaling this choice seems to provide a good balance between the three components of the objective function. That is each component has the same order of magnitude.

The result of the optimizations has also been produced using one fixed starting point for all values of the maximum sensitivity bound $M_{S,max}$. The starting point used is also in this test given by $\boldsymbol{x}_0 = [0.7 \;\; 0.7 \;\; 100 \;\; 100 \;\; 1 \;\; 1]$. Also for this test the default option values are used.

The tuning parameters generated by solution of the optimization problem based on the first-order identified model are listed in Table 11.4.

$M_{S,max}$	α_1, α_2	s_1, s_2	q_1, q_2	J_u	J_y	J_r	J
1.10	$3.7 \cdot 10^{-8}$ $2.3 \cdot 10^{-8}$	102.1 99.7	0.0428 0.0432	217.2	351.2	336.3	904.7
1.15	$2.0 \cdot 10^{-8}$ $1.3 \cdot 10^{-8}$	107.3 99.0	0.2181 0.2221	111.2	166.5	127.5	405.1
1.20	$6.9 \cdot 10^{-5}$ $4.7 \cdot 10^{-5}$	62.1 143.7	0.9891 1.0073	53.7	69.2	41.7	164.6
1.25	$5.9 \cdot 10^{-5}$ $3.4 \cdot 10^{-5}$	6.9811 198.7575	2.0463 1.9726	33.5	50.9	30.4	114.8
1.50	$4.0 \cdot 10^{-4}$ $4.0 \cdot 10^{-4}$	0.0001 170.333	47.145 52.223	9.0	21.2	9.9	40.1
1.70	0.9013 $2.1 \cdot 10^{-5}$	$1.0 \cdot 10^{-6}$ $1.0 \cdot 10^{-6}$	381.7822 341.9258	4.0	6.0	0.2	10.2

Table 11.4: Tuning parameters and associated objective function values obtained by optimization for different values of the maximum sensitivity bound $M_{S,max}$.

Table 11.4 presents the tuning parameters obtained by optimization for $M_{S,max} = \{1.10, 1.15, 1.20, 1.25, 1.50, 1.70\}$. The table shows that for this approach the tunings for all cases, except $M_{S,max} = 1.7$, have $\alpha_1, \alpha_2 \approx 0$ corresponding to full integration. This behavior is what was also observed for Approach I.

It is observed that when the maximum sensitivity $M_{S,max}$ is chosen larger than $M_{S,max} \geq 1.25$, the tuning parameter s_1 is determined to be very small in comparison to s_2 or both parameters are very small. This again corresponds to a situation where the first control input can vary rapidly while the second control input can only change very slowly as previously observed. This behavior indicates that the closed-loop system based on the first-order identified model is very sensitive and a small bound is necessary to obtain useful tuning parameters. A possible explanation for this behavior could be that the first-order model is too simple and cannot in an appropriate fashion minimize the objective when the sensitivity bound is allowed to be relatively high. The tuning parameters obtained for $M_{S,max} = \{1.10, 1.15, 1.20, (1.25)\}$ all appear to be reasonable choices. The objective function J clearly illustrates the tradeoff between these sets of tuning parameter with both respect to reference tracking and disturbance rejection.

Overall it is seen that the two considered approaches behave in a similar fashion when applied for the first-order identified model.
The results in Table 11.4 are obtained based on the IAE of the individual scenarios. Similar tuning parameters are obtained when using the ISE.

To investigate the effect of the scaling of the terms in the objective function additional tests have been conducted. In these tests the ratios between the size of input and output disturbance steps and the reference step have been varied. The main conclusion of these tests is that the scaling only has little effect on the obtained tuning parameters and that similar behavior to the previous results is observed.

Based on the fact that almost all the sets of tuning parameters obtained by the optimization result in full integration, it was tested which tuning parameters are obtained if the disturbance model parameters α_1 and α_2 are fixed in the optimization problem. This appeared to cause problems for the solver and often resulted in no feasible solution when α_1 and α_2 was fixed between $0.6 - 0.999$. For lower values the resulting tunings were similar to the ones seen so far.

11.1.1.1 Summary

This section have illustrated the optimization based tuning approach on the first-order identified model. The tuning parameters obtained for $M_{S,max} \geq 1.3$ show large differences for the two input rate weights s_1 and s_2 allowing one actuator to use excessive control action. For these cases q_1 and q_2 attain relative high values and consequently result in controllers that are too aggressive and most likely are useless in practice. The tuning parameters obtained for $M_{S,max} < 1.3$ appear to result in reasonable controllers.

11.1.2 Tuning based on second-order identified model

This section considers the tuning based on the second-order identified model. The tuning will be conducted based on both Approach I and Approach II.

In the tests the optimization problem is solved for different choices of the maximum sensitivity bound $M_{S,max}$. In this way it is investigated what sensitivity level gives reasonable tuning parameters for the controllers based on the second-order identified model. These tests will be the basis for the comparison to the results obtained based on the first-order identified model.

Approach I The tuning parameters generated by solution of the optimization problem based on tuning Approach I are listed in Table 11.5.

$M_{S,max}$	α_1, α_2	s_1, s_2	q_1, q_2	δ	J
1.10	$6.1 \cdot 10^{-8}$ $3.4 \cdot 10^{-8}$	109.1 116.8	0.0616 0.0656	0.9	1129.9
1.15	$3.4 \cdot 10^{-9}$ $2.0 \cdot 10^{-9}$	110.8 102.8	0.3495 0.3629	0.9	664.9
1.20	$5.0 \cdot 10^{-5}$ $3.9 \cdot 10^{-5}$	37.9 166.8	1.3395 1.3876	0.9	377.4
1.25	$6.9 \cdot 10^{-5}$ $1.2 \cdot 10^{-4}$	116.8 $2.4 \cdot 10^{4}$	361.2 382.6	0.9	285.1
1.30	$6.5 \cdot 10^{-7}$ 0.9980	128.7 0.0011	36.6 90.5	0.9	130.6
1.50	$5.5 \cdot 10^{-6}$ $1.2 \cdot 10^{-5}$	44.0 $3.1 \cdot 10^{5}$	$3.0 \cdot 10^{5}$ $3.1 \cdot 10^{5}$	0.9	79.1

Table 11.5: Tuning parameters and associated objective function values obtained by optimization based on gain deviation scenario with $\delta = 0.9$ for different values of the maximum sensitivity bound $M_{S,max}$.

Approach II The tuning parameters generated by solution of the optimization problem based on the second-order identified model are listed in Table 11.6.

$M_{S,max}$	α_1, α_2	s_1, s_2	q_1, q_2	J_u	J_y	J_r	J
1.10	$4.0 \cdot 10^{-8}$ $2.8 \cdot 10^{-8}$	105.2840 108.2985	0.0431 0.0449	218.4	356.0	341.3	915.7
1.15	$2.2 \cdot 10^{-8}$ $1.6 \cdot 10^{-8}$	95.0781 104.5402	0.1553 0.1656	128.7	201.4	163.1	493.3
1.20	$3.8 \cdot 10^{-6}$ $2.5 \cdot 10^{-6}$	90.9645 109.8823	0.5512 0.6247	69.2	93.0	57.2	219.3
1.25	0.0001 0.0001	26.4586 169.8823	1.0393 1.0852	49.7	67.5	38.6	155.8
1.30	0.0001 0.0001	8.1340 191.5077	1.7094 1.7438	38.0	55.5	30.8	124.3
1.40	0.0002 0.0001	0.2895 197.1328	3.8313 4.4009	23.4	41.5	21.7	86.5
1.50	0.0003 0.0002	0.0025 184.1745	8.1583 9.3094	17.1	33.7	15.3	66.1
1.70	0.0074 0.0041	248.5293 $9.8 \cdot 10^{-5}$	81.6342 101.7485	10.3	19.5	8.4	38.2
2.00	0.0638 0.0201	$1.0 \cdot 10^{-6}$ $1.0 \cdot 10^{-6}$	251.7017 309.3501	8.6	15.4	0.2	24.2

Table 11.6: Tuning parameters and associated objective function values obtained by optimization for different values of the maximum sensitivity bound $M_{S,max}$.

From Table 11.5 and Table 11.6 it is seen that the solutions obtained based on the second-order model for both tuning approaches have much the same characteristics as the solutions obtained based on the first-order model. This indicates that the behavior experienced in the previous test does not stem from the fact that the optimization is based on the first-order model but rather is a general feature of the process.

11.1.2.1 Summary

This section has illustrated the optimization based tuning approach on the second-order identified model. The tuning parameters obtained for $M_{S,max} < 1.4$ appear to result in reasonable controllers.

11.2 Closed-Loop Simulations

In this section closed-loop simulations for the modified 4-tank process are conducted.

The closed-loop simulations are conducted using the tuning parameters obtained by the optimization procedure in the previous section. The simulations are done in order to illustrate the importance of proper tuning of industrial control systems.

In the simulations the non-linear modified 4-tank process model (9.26)-(9.29) is used to represent the plant. The covariance matrices for the process- and measurement noise used in the simulation are given by

$$\boldsymbol{Q} = \begin{bmatrix} 20^2 & 0 \\ 0 & 20^2 \end{bmatrix}, \quad \boldsymbol{R} = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$$

The process is sampled using a sample time $T_s = 4$ $[s]$. The prediction horizon used in the simulations is $N = 500$, corresponding to a prediction of the future dynamic behavior of the process for 2000 $[s]$. It has been chosen to be long to approximate an infinite horizon controller, hereby reducing the effect of differences between open-loop and closed-loop solutions.

In the simulations the nominal value for the deterministic part of the disturbance is $\boldsymbol{d}_k^{det} = [70 \;\; 70]^T$. It should be noted that these are the same conditions which the identified models were obtained under. This corresponds to what would be considered normal operation conditions in the case study.

In the simulations a simulation profile where the process is in a steady state is used. The initial water level (height) in Tank 1 is 50.2 cm and the water level in Tank 2 is 42.7 cm.

11.2.1 Simulations based on first-order identified model

As already mentioned, it is not evident that all the tuning parameters obtained in the previous section result in satisfactory behavior of the control system. To investigate what level of sensitivity results in acceptable behavior, simulations are conducted for different values of $M_{S,max}$.

The first simulation is conducted based on the results in Table 11.1. A simulation based on the tuning parameters obtained for $M_{S,max} = 1.25$ and $M_{S,max} = 1.30$ is presented in Figure 11.1.

In the simulation the reference for the water level in Tank 1 is at $t = 170$ [min] increased to 58 cm, while the reference for Tank 2 is unchanged. Furthermore, at $t = 35$ [min] a step change in the deterministic part of the disturbance into Tank 2 is introduced with a relative change of $+40$ $[cm^3/s]$.

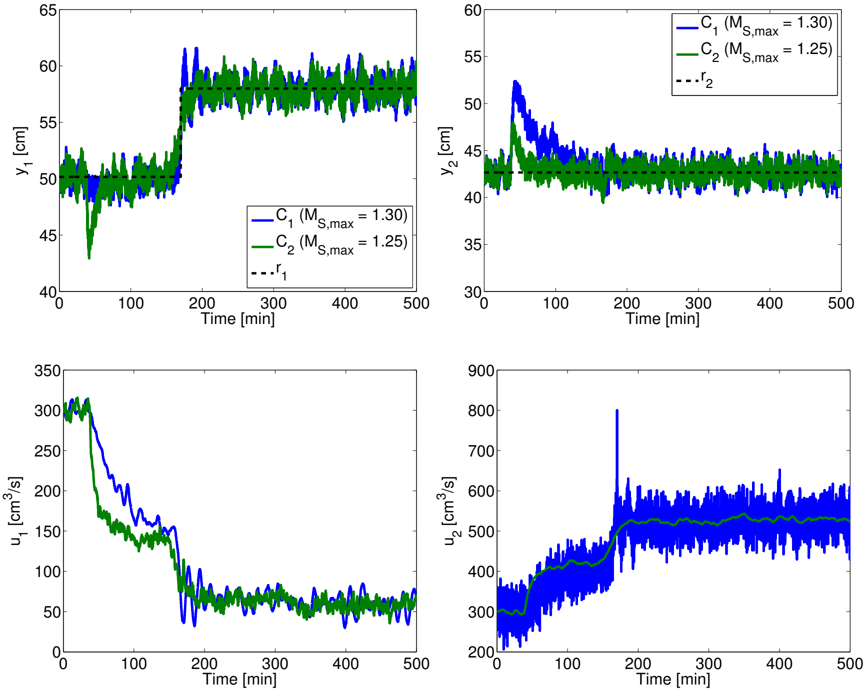

Figure 11.1: Closed-loop simulation based on first-order identified model with tuning parameters obtained using $M_{S,max} = 1.25$ and $M_{S,max} = 1.30$.

The simulation in Figure 11.1 shows that these tunings provide good reference tracking for both outputs, despite the disturbance and the high level of process and measurement noise. It is seen that the disturbance rejection for y_2 is most efficient for $M_{S,max} = 1.25$, but at the same time that y_1 reacts quite strongly as well.

Furthermore, it is clearly seen that the variance of the control signal u_2 for $M_{S,max} = 1.30$ is very high and that this set of tuning parameters is not acceptable and useless in practice. The parameters obtained for $M_{S,max} = 1.25$ seem to provide an acceptable performance.

Next it is investigated what performance is obtained when the sensitivity bound is reduced. To illustrate this, a test is conducted for $M_{S,max} = 1.20$ and $M_{S,max} = 1.15$ using the same simulation scenario as used above. The simulation is presented in Figure 11.2.

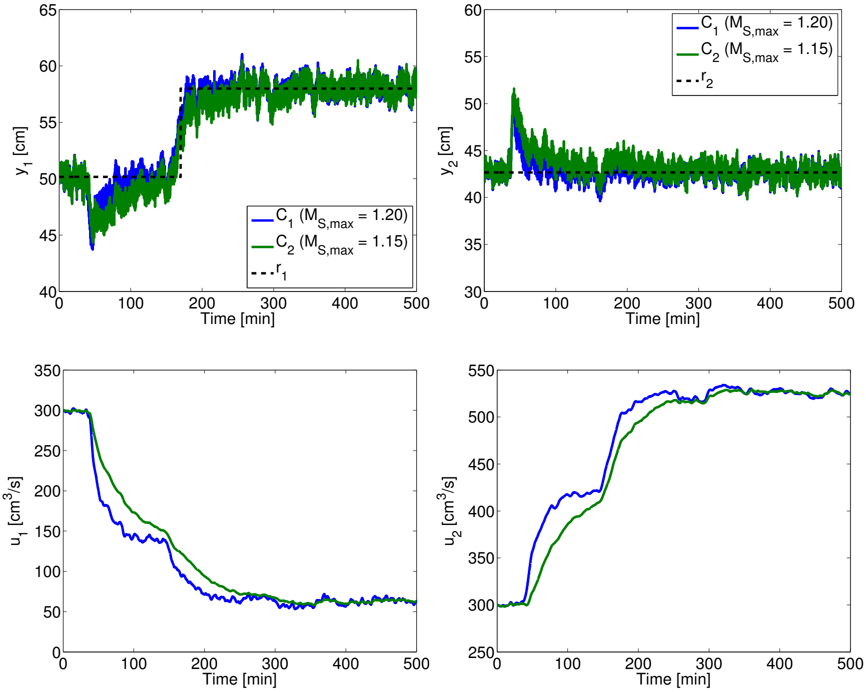

Figure 11.2: Closed-loop simulation based on first-order identified model with tuning parameters obtained using $M_{S,max} = 1.15$ and $M_{S,max} = 1.20$.

From the simulation in Figure 11.2 it is seen that these sets of tuning parameters provide a marginally better and smooth overall reference tracking for both outputs compared to the previous simulation. However, it is seen that these tunings both react more slowly to the reference step. A nice feature related to these tunings is that both outputs are relatively insensitive to the high level of process noise.

The disturbance rejection is similar for both sets of tuning parameters, but $M_{S,max} = 1.20$ provides a slightly faster rejection without causing any side effects.

A very important fact about these tunings is that the variances on the control inputs are very small. In particular it is seen that the variance of the control signal u_2 is reduced substantially for both tunings in comparison to the case where $M_{S,max} = 1.30$. This nicely illustrates the benefits of lowering the sensitivity bound.

It will now be investigated if the performance can be increased even more by further reducing the sensitivity bound. This simulation is conducted for $M_{S,max} = 1.20$ and the lowest sensitivity bound $M_{S,max} = 1.15$. The simulation for these sets of tuning parameters is presented in Figure 11.3.

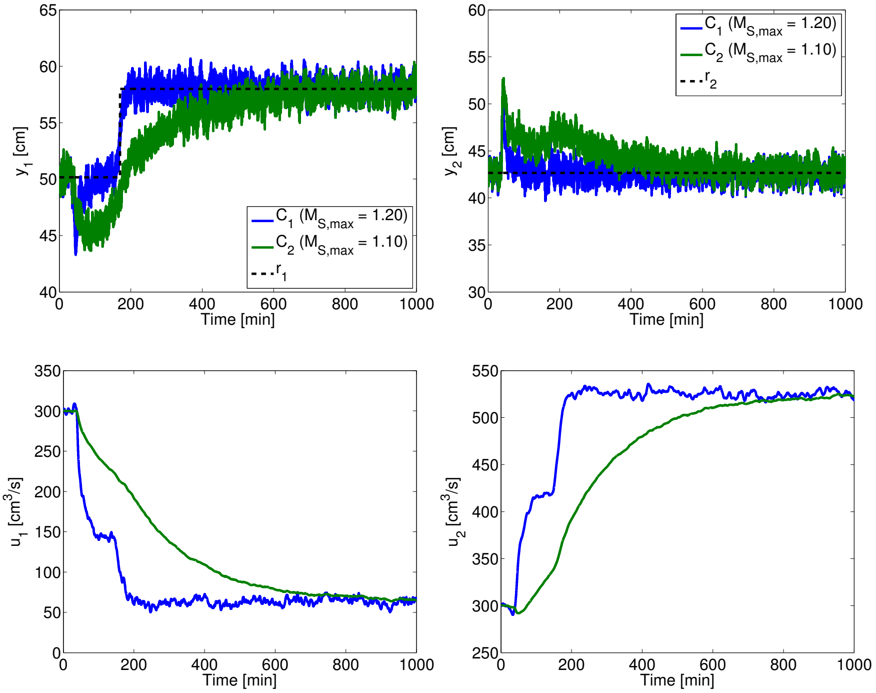

Figure 11.3: Closed-loop simulation based on first-order identified model with tuning parameters obtained using $M_{S,max} = 1.10$ and $M_{S,max} = 1.20$.

The simulation in Figure 11.3 very clearly illustrates the trade-off between disturbance rejection and sensitivity of the system. It is seen that the disturbance introduced in y_2 at $t = 35$ [min] is rejected very slowly for $M_{S,max} = 1.10$ and that this has a substantial impact in the simulation.

When compared to the tuning obtained for $M_{S,max} = 1.20$ it is seen that this tuning is in a sense *too* robust and probably not that useful in practice due to its overall performance.

From this simulation it can be concluded that there is no practical benefit in selecting $M_{S,max} = 1.10$ as this heavily increases the disturbance rejection time and does not offer a better tracking for the system. Based on the simulations conducted so far the best trade-off between low input variance and disturbance rejections seems to be obtained for a tuning using $M_{S,max} = 1.20$ or $M_{S,max} = 1.15$.

The simulations conducted so far are all based on the tuning parameters in Table 11.1 obtained using tuning Approach I. Since the tuning parameters obtained for other deviation scenarios are very similar, simulations will not be illustrated for the tuning parameters in Table 11.2 and Table 11.3. It should however be noted that similar performance and behavior are observed for such simulations.

Next simulation based on the results in Table 11.4 is considered. That is the tuning parameters obtained using tuning Approach II.

As already stated, the tuning parameters obtained based on Approach II are very similar to that of Approach I. Based on simulation studies it was seen that the tuning parameters resulted in similar closed-loop performance. Therefore only one simulation is presented for this approach. The simulation is conducted for $M_{S,max} = 1.25$ and $M_{S,max} = 1.50$ and is reported in Figure 11.4.

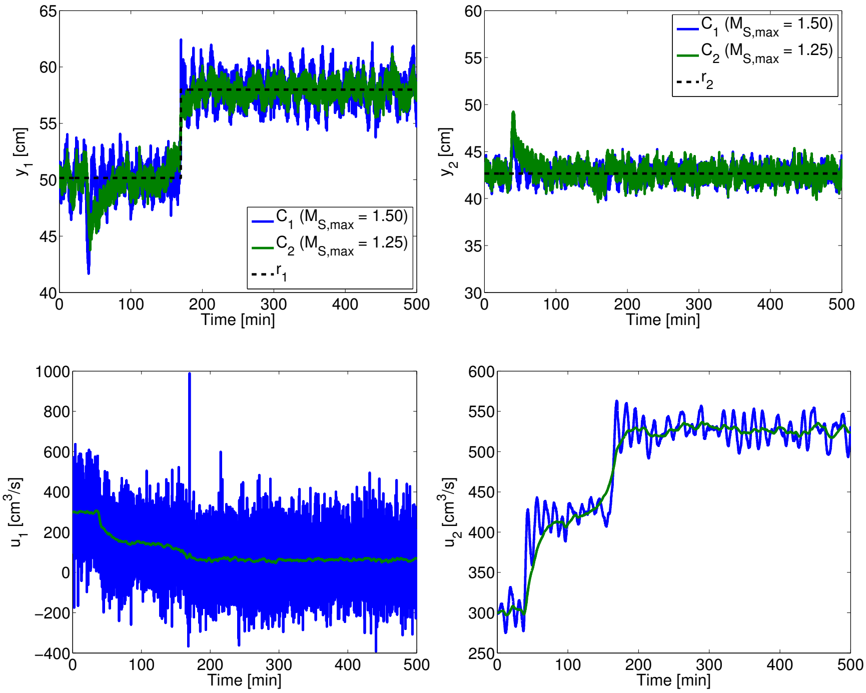

Figure 11.4: Closed-loop simulation based on first-order identified model with tuning parameters obtained using $M_{S,max} = 1.25$ and $M_{S,max} = 1.50$.

The simulation in Figure 11.4 shows that the tuning for $M_{S,max} = 1.25$ provides good reference tracking for both outputs, despite the disturbance and the high level of process and measurement noise. For $M_{S,max} = 1.50$ this is not the case. It is seen that especially y_1 is heavily affected by the high level of process and measurement noise and as a result oscillates around the reference. The disturbance rejection for y_2 is surprisingly very efficiently rejected for both tunings.

Furthermore, it is clearly seen that the variance of the control signal u_1 for $M_{S,max} = 1.50$ is very high and that this set of tuning parameters is not acceptable and useless in practice. It is noted that this is the opposite situation of what was seen in Figure 11.1. The parameters obtained for $M_{S,max} = 1.25$ seem to provide an overall acceptable performance.

11.2.2 Simulations based on second-order identified model

In this section simulations based on the second-order identified model will be conducted. Based on the tuning parameters obtained in Section 11.1.2 it is expected that the closed-loop performance based on the second-order identified model to a large extend will be similar to what was observed for the first-order model. The simulation will be conducted based on the results in Table 11.6 obtained using tuning Approach II.

The first simulation is conducted based on the tuning parameters obtained for $M_{S,max} = 1.30$ and $M_{S,max} = 1.40$ and is presented in Figure 11.5.

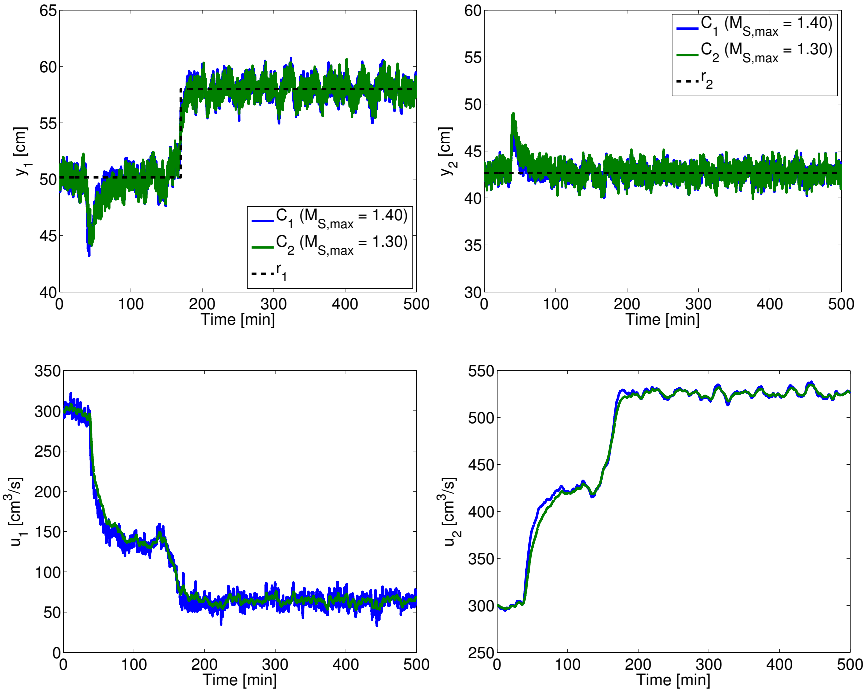

Figure 11.5: Closed-loop simulation based on second-order identified model with tuning parameters obtained using $M_{S,max} = 1.30$ and $M_{S,max} = 1.40$.

The simulation in Figure 11.5 shows that for the second-order model reasonable tuning parameters are obtained for both $M_{S,max} = 1.30$ and $M_{S,max} = 1.40$, as opposed to the first-order model where this level of sensitivity resulted in useless controllers. This indicates that the closed-loop system based on the second-order identified model is less sensitive. It is seen that both the reference tracking and the disturbance rejection for these tunings are slightly better than what was seen for the first-order model for even lower levels of $M_{S,max}$.

In Figure 11.6 a simulation using the tuning parameters obtained for $M_{S,max} = 1.10$ and $M_{S,max} = 1.25$ is shown.

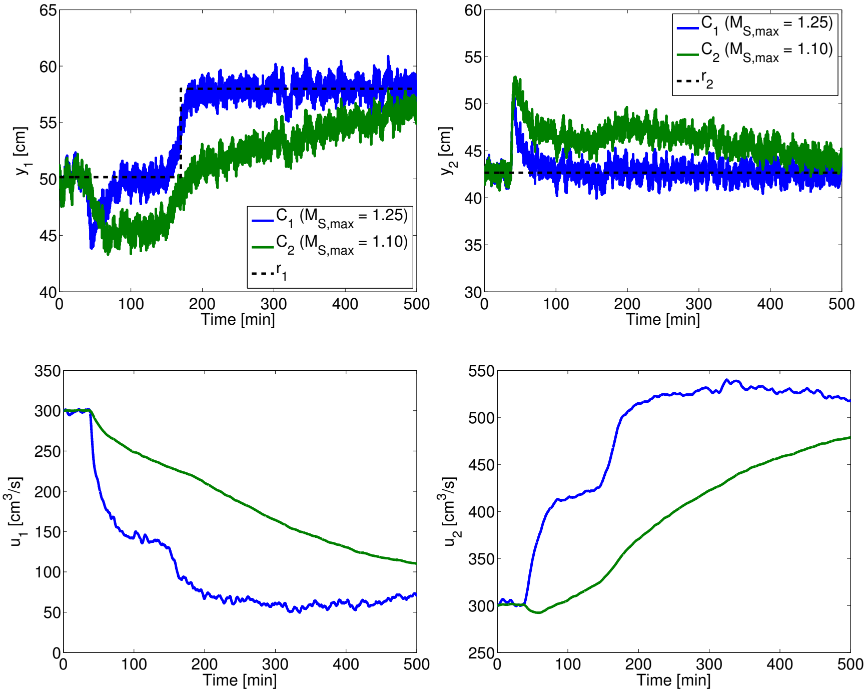

Figure 11.6: Closed-loop simulation based on second-order identified model with tuning parameters obtained using $M_{S,max} = 1.10$ and $M_{S,max} = 1.25$.

The simulation in Figure 11.6 clearly illustrates the trade-off between disturbance rejection and sensitivity of the system for the second-order model. It is seen that the disturbance introduced in y_2 at $t = 35$ [min] is rejected even more slowly than was seen in the case for the first-order model and that this has a very strong influence in the simulation. The level of robustness is simply too low and the performance obtained is not acceptable and completely useless in practice.

Based on the simulations conducted for the second-order model, the best trade-off between low input variance and disturbance rejections seems to be obtained for a tuning using $M_{S,max} = 1.25$ or $M_{S,max} = 1.30$.

11.2.3 Comparison

In this section the closed-loop performance obtained using the first-order and the second-order identified models are compared. The comparison is based on the tuning parameters obtained using Approach II for both models. In this way the performance can be compared strictly based on $M_{S,max}$.

The comparison is based on the first-order model controller with $M_{S,max} = 1.25$ and the second-order model controller with $M_{S,max} = 1.40$. The simulation is shown in Figure 11.7.

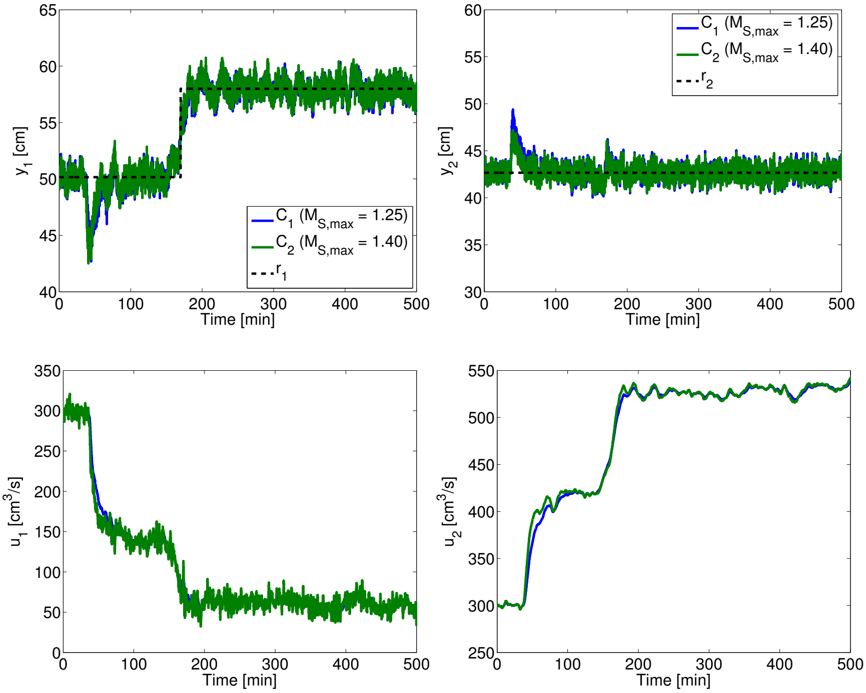

Figure 11.7: Closed-loop simulation based on first-order model controller (C_1) with $M_{S,max} = 1.25$ and second-order model controller (C_2) with $M_{S,max} = 1.40$.

The simulation in Figure 11.7 shows that despite the fact that the controller based on the second-order model is less robust, almost the same closed-loop performance is obtained. This is interesting and indicates that the controller based on the second-order model is more flexible and less sensitive.

A direct comparison for the two model with the same $M_{S,max}$ value is probably a more suitable test to determine if there is any real advantage in using a second-order over a first-order model in the controller design.

This comparison is based on the first-order model and the second-order model controller both with $M_{S,max} = 1.25$. The simulation is shown in Figure 11.8.

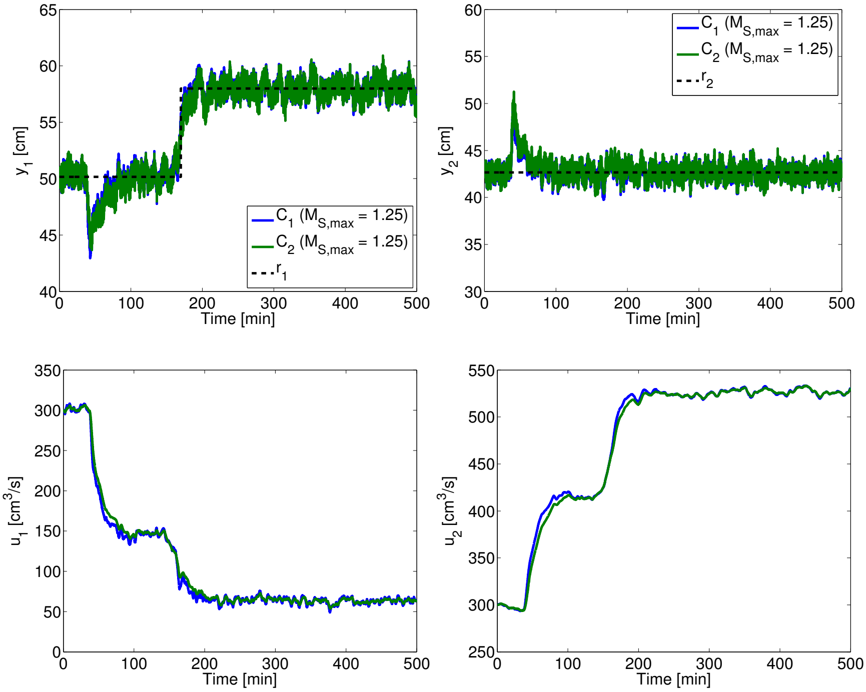

Figure 11.8: Closed-loop simulation based on first-order model controller (C_1) with $M_{S,max} = 1.25$ and second-order model controller (C_2) with $M_{S,max} = 1.25$.

The simulation in Figure 11.8 shows the closed-loop performance for the two controllers are almost the same for $M_{S,max} = 1.25$. This illustrates there is no real advantage in using a second-order over a first-order model if the obtained performances is considered acceptable. In this case the performance is acceptable especially from the point of view of the high level of noise in the system.

Part III

Conclusion

Chapter 12

Conclusion

An industrial MPC approach has been considered in this thesis. The approach included the identification of a linear model for the process to be used in the MPC and the setup of the MPC based on the identified model extended with a disturbance model, such that offset free control is ensured. In addition, an optimization based tuning procedure for the MPC has been developed.

In the first part of the thesis the theoretical background was developed and analyzed. This part constitutes a large part of the thesis and many different topics have been studied.

A central part has been to set up the MPC. It has been shown here that the unconstrained MPC regulation problem can be transformed into a convex quadratic program (QP). Based on the QP an explicit expression for the optimal control has been derived. Since the expressions for the optimal control and the state estimator for the controller model are linear it was possible to express the controller as a linear state-space model.

The key concept for the tuning of the MPC has been to formulate a closed-loop state-space description for the process and the controller state-space model. In the formulation of the closed-loop system two different plant models were used in order to base the tuning problem only on the deterministic linear model obtained by system identification.

Furthermore, for the closed-loop state-space system sensitivity functions have been derived. The sensitivity function was used as a measure of robustness and in the formulation of the optimization problem to obtain the tuning parameters of the MPC.

The optimization problem is based on an objective function related to scenario simulation results obtained using the closed-loop system. The scenarios used are deterministic reference and disturbance scenarios. The tuning parameters are obtained by minimization of the absolute integrated error (IAE) or integrated squared error (ISE) for the individual scenarios.

The designed controllers used in the tuning approach were all based on the deterministic-stochastic model.

To illustrate the developed MPC approach, a case study for the modified 4-tank process was conducted. The process has been used as a representative of real life industrial processes, since it shares some of the general features encountered in industry. In the case study system identification based on step tests was carried out and a linear model for the process obtained. From the identification, it can be concluded that models capturing the general behavior of the process can be determined, even in cases where a high level of noise is present. In particular a first-order and second-order continuous-time transfer model was identified.

The optimization problem has been used to tune the MPC based on both the first-order and the second-order identified models. From the numerical tests it can overall be concluded that the optimization based tuning approach has and can produce some reasonably good tuning parameters for the modified 4-tank process, even under a high level of noise in the system.

The optimization algorithm has mainly been tested using the interior point algorithm with default options for one fixed starting point in all conducted tests. All tests have resulted in feasible solutions. A key element to this behavior was the use of a fixed number of frequencies in the evaluation of $M_S(\boldsymbol{x})$.

For high values of the sensitivity bound $M_{S,max}$ it can be concluded that useless tuning parameters were in general found. The tunings resulted in too much stress on the actuators and only modest reference tracking. For $M_{S,max} \leq 1.25$ acceptable tuning parameters were determined for the first-order models, while $M_{S,max} \leq 1.40$ in general provided acceptable tunings for the second-order model.

It can be concluded that there is no real advantage in using a second-order over a first-order model in the controller design for this process and that it mainly comes down to the tuning parameters. This conclusion is based on the fact that the second-order model has more parameters to be identified but does not offer significantly better performance.

In general the two tuning approaches have produced similar results and it is hard to conclude if one is better. The tests conducted based on Approach I have illustrated that the obtained tuning parameters to a large extend depend on the level of deviation and that it is not obvious exactly how this level should be chosen. Approach II has shown that the obtained tuning parameters depend little on the size of the step disturbances used in the simulation scenarios. It is suggested by the obtained results that the size of the step disturbances in general can be chosen as unit steps. However, whether both input and output disturbance scenarios should be included in the objective is hard to conclude.

From the tests it can also be concluded that it does not have any significant influence on the obtained tuning parameters whether the IAE or the ISE is used to asses the simulation scenarios in the objective function.

Almost all the tuning tests resulted in full integration ($\alpha_1, \alpha_2 \approx 0$). An explanation for this behavior, has not be found despite the fact that a number of tests have been conducted to investigate this.

Chapter 13

Discussion and Future Work

The optimization based tuning approach has shown potential, however, there are a number of aspects which should be studied in further detail. It should be investigated further why the tuning procedure for most cases result in full integration. This could be due to the deterministic-stochastic model used in the controller design or related to the structure of the process.

The optimization problem based on Approach II is very appealing since it uses only the identified model and user defined disturbances for the optimization. This makes it very useful in industrial application where very little is known about the true disturbances. This approach should be the main focus of future research.

To investigate the optimality of the optimization problem it would be interesting to develop an efficient (if possible) parameter sweep approach for MIMO systems for comparison. This would allow us to determine, which of the studied optimization approaches have the best overall performance.

Another interesting future study would be to conduct a comparison study between a tuning approach based on deterministic-stochastic model and the MISO ARX model. This study would help to illustrate to pros and cons of the two approaches to offset free control.

A natural extension of the work conducted in the thesis, would be to investigate how the tuning approach will perform on systems with more inputs and outputs. Furthermore, so would it be to test the entire MPC procedure on a real small scale industrial process.

Part IV

Appendix

Appendix A

Matlab code

In this appendix all the Matlab code implemented in thesis is included. The code is organized as follows.

- Setup and simulation files for the non-linear modified 4-tank system
- MPC setup files
  - Unconstrained
  - Constrained
- Tuning files
  - Approach I
  - Approach II
- System identification files

A.1 Setup and Simulation Files for the Modified 4-Tank System

```
function xdot = ModifiedFourTankSystem(t,x,u,d,p)
% MODIFIEDFOURTANKSYSTEM    Model dx/dt = f(t,x,u,d,p) for modified
% 4-tank System
%
% This function implements a differential equation model for the
% modified 4-tank system.


m     = x;          % Mass of liquid in each tank [g]
F     = u;          % Flow rates in pumps [cm3/s]
D     = d;          % Disturbance flow rates [cm3/s]
a     = p(1:4,1);   % Pipe cross sectional areas [cm2]
A     = p(5:8,1);   % Tank cross sectional areas [cm2]
gamma = p(9:10,1);  % Valve positions [-]
g     = p(11,1);    % Acceleration of gravity [cm/s2]
rho   = p(12,1);    % Density of water [g/cm3]

% Inflows
qin = zeros(4,1);
qin(1,1)  = gamma(1)*F(1);     % Inflow from valve 1 to tank 1 [cm3/s]
qin(2,1)  = gamma(2)*F(2);     % Inflow from valve 2 to tank 2 [cm3/s]
qin(3,1)  = (1-gamma(2))*F(2);% Inflow from valve 2 to tank 3 [cm3/s]
qin(4,1)  = (1-gamma(1))*F(1);% Inflow from valve 1 to tank 4 [cm3/s]

% Outflows
h = m./(rho*A);                % Liquid level in each tank [cm]
qout = a.*sqrt(2*g*h);         % Outflow from each tank [cm3/s]

% Differential equations
xdot = zeros(4,1);
xdot(1,1)  = rho*(qin(1,1)+qout(3,1)-qout(1,1));% Mass balance Tank 1
xdot(2,1)  = rho*(qin(2,1)+qout(4,1)-qout(2,1));% Mass balance Tank 2
xdot(3,1)  = rho*(D(1)+qin(3,1)-qout(3,1));     % Mass balance Tank 3
xdot(4,1)  = rho*(D(2)+qin(4,1)-qout(4,1));     % Mass balance Tank 4
```

```
function y = ModifiedFourTankSystemSensor(x,p)

% measurement of tank 1 and tank 2 only !
rho = p(12,1);
A = p(5:6,1);

y = x(1:2)./(rho*A);
```

```
function xdot = ModifiedFourTankSystemWrap(x,u,d,p)
         xdot = ModifiedFourTankSystem(0,x,u,d,p);
```

```
function xknew = nonlinearstate(xk,uk,dk,wk,Ts,p)
% Computes new state of the non—linear modified 4—tank system
%
% Inputs:
% xk      — Current non—linear state
% uk      — Current control input
% dk      — Disturbance
% wk      — Process noise
% Ts      — Sample—time
% p       — Parameter vector for the modified 4—tank system
%
% Output:
% xknew   — Next non—linear state

[Tk,Xk] = ode15s(@ModifiedFourTankSystem,[0 Ts],xk,[],uk,(dk+wk),p);
xknew = Xk(end,:)';
```

```
function yk = nonlinearmeasurement(xk,vk,p)
% Computes measurement of the non—linear modified
% 4—tank system with measurement noise
%
% Inputs:
% xk      — Current non—linear state
% vk      — Measurement noise
% p       — Parameter vector for the modified 4—tank system
%
% Output:
% yk      — Measurement of the modified 4—tank system


yk = ModifiedFourTankSystemSensor(xk,p) + vk;
```

```
function [A B E C xs ys p] = getlinearizedModified4TankSystem...
                                (us,ds,gamma1,gamma2,xs0)
% linearizes the non—linear model of modified 4—tank system
% around a steady state
%
% Inputs:
% us       — Steady control input vector
% ds       — Steady disturbance input vector
% gamma1   — Flow distribution constant valve 1
% gamma2   — Flow distribution constant valve 2
% xs0      — initial guess for steady state
%
% Outputs:
% A        — Continuous—time state matrix
% B        — Continuous—time input matrix
% E        — Continuous—time disturbance matrix
% C        — Continuous—time measurement matrix
% xs       — steady state vector
% ys       — steady measurement vector
% p        − parameter vector


% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
% Parameters for the modified 4−tank system
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

a1 = 1.2272; %[cm2] Area of outlet pipe 1
a2 = 1.2272; %[cm2] Area of outlet pipe 2
a3 = 1.2272; %[cm2] Area of outlet pipe 3
a4 = 1.2272; %[cm2] Area of outlet pipe 4
ap = [a1; a2; a3; a4]; %[cm2] Pipe cross sectional areas

A1 = 380.1327; %[cm2] Cross sectional area of tank 1
A2 = 380.1327; %[cm2] Cross sectional area of tank 2
A3 = 380.1327; %[cm2] Cross sectional area of tank 3
A4 = 380.1327; %[cm2] Cross sectional area of tank 4
At = [A1; A2; A3; A4]; %[cm2] Tank cross sectional areas

g = 981;    %[cm/s2] The acceleration of gravity
rho = 1.00; %[g/cm3] Density of water

if nargout > 5
p = [ap; At; gamma1; gamma2; g; rho];
end

% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
% Steady State and measurement
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

if nargin ≤ 4
xs0 = [5000; 5000; 5000; 5000]; % [g] Initial guess on xs
end

xs = fsolve(@ModifiedFourTankSystemWrap,xs0,[],us,ds,p);
ys = ModifiedFourTankSystemSensor(xs,p);

% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
% Continuous−time linearized model
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

hs = xs./(rho∗At);
T = (At./ap).∗sqrt(2∗hs/g);

A = [−1/T(1) 0 1/T(3) 0;
    0 −1/T(2) 0  1/T(4);
    0 0 −1/T(3) 0;
    0 0 0 −1/T(4)];

B = [rho∗gamma1 0;0 rho∗gamma2; 0 rho∗(1−gamma2); rho∗(1−gamma1) 0];
E = [0 0; 0 0; rho 0; 0 rho];

Ctemp = diag(1./(rho∗At));
C = Ctemp(1:2,:);
```

```
function p = getnonlinearsystemparameters(gamma1,gamma2)

a1 = 1.2272; %[cm2] Area of outlet pipe 1
a2 = 1.2272; %[cm2] Area of outlet pipe 2
a3 = 1.2272; %[cm2] Area of outlet pipe 3
a4 = 1.2272; %[cm2] Area of outlet pipe 4
ap = [a1; a2; a3; a4]; %[cm2] Pipe cross sectional areas

A1 = 380.1327; %[cm2] Cross sectional area of tank 1
A2 = 380.1327; %[cm2] Cross sectional area of tank 2
A3 = 380.1327; %[cm2] Cross sectional area of tank 3
A4 = 380.1327; %[cm2] Cross sectional area of tank 4
At = [A1; A2; A3; A4]; %[cm2] Tank cross sectional areas

g = 981;    %[cm/s2] The acceleration of gravity
rho = 1.00; %[g/cm3] Density of water

p = [ap; At; gamma1; gamma2; g; rho];
```

```
function [u y z] = ClosedloopSimulationUnconstrainedNonlinear3 ...
                   (Nsim,Systemnonlinear,Controller,x0,d,v,w,r)
% Computes the closed—loop control inputs, measurements and
% controlled outputs for the unconstrained MPC problem with
% non—linear plant, given an initial plant state x0 and
% simulation scenario (d,v,w,r)
%
% Inputs:
% Nsim          — Number of simulation steps
% System        — Struct holding plant/system
% Controller    — Struct holding controller model
% x0            — Initial state for plant
% d             — Disturbance scenario
% v             — Measurement noise scenario
% w             — Process noise scenario
% r             — Reference scenario
%
% Outputs:
% u             — closed—loop control inputs
% y             — closed—loop measurements
% z             — closed—loop controlled outputs

% ————————————————————————————————————————————————————————
% Non—linear plant parameters
% ————————————————————————————————————————————————————————

p  = Systemnonlinear.p;
Ts = Systemnonlinear.Ts;

% ————————————————————————————————————————————————————————
% Controller model
% ————————————————————————————————————————————————————————

% control model
Ad = Controller.Abar;
Bd = Controller.Bbar;
Cd = Controller.Cbar;
Kd = Controller.Kbar;

% controller gains
Lx = Controller.Lx;
Le = Controller.Le;
LR = Controller.LR;
Lu = Controller.Lu;

%
N = Controller.N;

% size of controller model
ny = size(Cd,1);
nu = size(Bd,2);


% ----------------------------------------------------------
% Initialization
% ----------------------------------------------------------

% initial state for the plant
xk = x0;

% initial control input u_{0|-1}
%ukm1 = zeros(nu,1);

% steady initial control input
ukm1 = 300*ones(nu,1);

% initial state estimate xhat_{0|-1}
%xhatkkm1 = zeros(nx,1);

% steady initial state
xhatkkm1 = 1.0e+03*[-1.4085 -0.0691 -0.0770 0.1981 ...
                                  -0.0122 -0.0363]';

% preallocation
y = zeros(ny,Nsim);
z = zeros(ny,Nsim);
u = zeros(nu,Nsim);

% -----------------------------------------------------------
% Closed-loop simulation
% -----------------------------------------------------------

for k = 1:Nsim

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%             PLANT             %%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % NON-LINEAR PLANT

    % plant measurement in physical variables
    yk = nonlinearmeasurement(xk,v(:,k),p);

    % plant output in physical variables
    zk = nonlinearmeasurement(xk,0*v(:,k),p);


    y(:,k) = yk;
    z(:,k) = zk;

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%          CONTROLLER              %%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % select the set-points for the current MPC horizon
    Rtemp = r(:,k:(k+N)-1);
    Rk = Rtemp(:);

    % compute optimal control input in physical variables
    [uhatkk xhatkp1k] = MPCcomputeUnconstrainedInnovation ...
               (xhatkkm1,ukm1,yk,Rk,Lx,Le,LR,Lu,Ad,Bd,Cd,Kd);

    u(:,k) = uhatkk;


    % Updating

    % update state estimate
    xhatkkm1 = xhatkp1k;

    % update control input
    ukm1 = uhatkk;


    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%             PLANT              %%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % NON-LINEAR PLANT

    % state update in physical variables
    xk = nonlinearstate(xk,u(:,k),d(:,k),w(:,k),Ts,p);

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end
```

```
function [u y z] = ClosedloopSimulationUnconstrainedNonlinear4 ...
                  (Nsim,Systemnonlinear,Controller,x0,d,v,w,r)
% Computes the closed-loop control inputs, measurements and
% controlled outputs for the unconstrained MPC problem with
% non—linear plant, given an initial plant state x0 and
% simulation scenario (d,v,w,r)
%
% Inputs:
% Nsim         — Number of simulation steps
% System       — Struct holding plant/system
% Controller   — Struct holding controller model
% x0           — Initial state for plant/system
% d            — Disturbance scenario
% v            — Measurement noise scenario
% w            — Process noise scenario
% r            — Reference scenario
%
% Outputs:
% u            — closed—loop control inputs
% y            — closed—loop measurements
% z            — closed—loop controlled outputs

% ——————————————————————————————————————————————————————
% Plant non—linear
% ——————————————————————————————————————————————————————

p  = Systemnonlinear.p;
Ts = Systemnonlinear.Ts;

% ——————————————————————————————————————————————————————
% Controller model
% ——————————————————————————————————————————————————————

% control model
Ad = Controller.Abar;
Bd = Controller.Bbar;
Cd = Controller.Cbar;
Kd = Controller.Kbar;

% controller gains
Lx = Controller.Lx;
Le = Controller.Le;
LR = Controller.LR;
Lu = Controller.Lu;

%
N = Controller.N;

% size of controller model
ny = size(Cd,1);
nu = size(Bd,2);
nx = size(Ad,1);



% ——————————————————————————————————————————————————————
% Initialization
% ——————————————————————————————————————————————————————

% initial state for the plant
xk = x0;

% initial control input u_{0|-1}
%ukm1 = zeros(nu,1);

% steady initial control input
ukm1 = 300*ones(nu,1);

% initial state estimate xhat_{0|-1}
%xhatkkm1 = zeros(nx,1);

% steady initial state
xhatkkm1 = 1.0e+03 * [ -1.3488 0.0228 -0.1901 -0.0879 ...
    -0.0458 0.0284 0.0027 -0.0115 -0.0357]';


% preallocation
y = zeros(ny,Nsim);
z = zeros(ny,Nsim);
u = zeros(nu,Nsim);

% -------------------------------------------------------
% Closed-loop simulation
% -------------------------------------------------------

for k = 1:Nsim

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%             PLANT              %%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % NON-LINEAR PLANT

    % plant measurement in physical variables
    yk = nonlinearmeasurement(xk,v(:,k),p);

    % plant output in physical variables
    zk = nonlinearmeasurement(xk,0*v(:,k),p);


    y(:,k) = yk;
    z(:,k) = zk;

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%          CONTROLLER              %%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % select the set-points for the current MPC horizon
    Rtemp = r(:,k:(k+N)-1);
    Rk = Rtemp(:);

    % compute optimal control input in physical variables
    [uhatkk xhatkp1k] = MPCcomputeUnconstrainedInnovation ...
            (xhatkkm1,ukm1,yk,Rk,Lx,Le,LR,Lu,Ad,Bd,Cd,Kd);

    u(:,k) = uhatkk;


    % Updating

    % update state estimate
    xhatkkm1 = xhatkp1k;

    % update control input
    ukm1 = uhatkk;


    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%             PLANT               %%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % NON-LINEAR PLANT

    % state update in physical variables
    xk = nonlinearstate(xk,u(:,k),d(:,k),w(:,k),Ts,p);

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end
```

```
clear all
close all
clc

addpath('Realization')

% Closed-loop simulation of modified 4-tank system
% based on first order identified model - Unconstrained case
%
% Plant model: Non-linear modified 4-tank system
% Controller model: Deterministic-Stochastic model


% ---------------------------------------------------------------
%  parameters
% ---------------------------------------------------------------

% sample-time [s]
Ts = 4;

% time of simulation in minutes
Tf = 500;

% Prediction horizon
N = 500;


```

```

% -------------------------------------------------------------------
% steady state for simulation

gamma1 = 0.45;    % Flow distribution constant. Valve 1
gamma2 = 0.40;    % Flow distribution constant. Valve 2

us = [300;300];
ds = [70;70];

[Axx Bxx Exx Cxx xs ys] =  ...
getlinearizedModified4TankSystem(us,ds,gamma1,gamma2);

% -------------------------------------------------------------------


% set tuning parameters

% MSmax = 1.20;
x1 = [6.9023e-05 4.7833e-05 62.1234 143.7291 0.9891 1.0073];

% MSmax = 1.15;
x2 = [2.0499e-08 1.2682e-08 107.3115 98.9778 0.2181 0.2221];

% -------------------------------------------------------------------
% Simulation details
% -------------------------------------------------------------------

% disturbance
d1 = 0;
d2 = 40 ;


% sample of disturbance start
d1time = 76;
d2time = 551;


% sample-time [min]
Tstep = Ts/60;

% time interval
T = 0:Tstep:Tf;

% number of simulation steps
Nsim = length(T);


% -------------------------------------------------------------------
%  Noise
% -------------------------------------------------------------------

% noise variance
sigma2w = (20)^2;
sigma2v = (0.5)^2;
```

```

% covariance matrices
Qw = sigma2w*eye(2,2);
Rv = sigma2v*eye(2,2);

Swv = zeros(2,2);

% combined covariance matrix
Z = [Qw Swv; Swv Rv];

LZ = chol(Z,'lower');
wv = LZ*randn(4,Nsim+N);

w = wv(1:2,:);    % process noise
v = wv(3:4,:);    % measurement noise



% tuning 1
alpha1 = x1(1);
alpha2 = x1(2);

s1 = x1(3);
s2 = x1(4);

q1 = x1(5);
q2 = x1(6);

% tuning2
alpha11 = x2(1);
alpha21 = x2(2);

s11 = x2(3);
s21 = x2(4);

q11 = x2(5);
q21 = x2(6);

% ---------------------------------------------------------------
% Non-linear modified 4-tank plant
% ---------------------------------------------------------------

p = getnonlinearsystemparameters(gamma1,gamma2);

SystemNonlinear.p  = p;
SystemNonlinear.Ts = Ts;

% ---------------------------------------------------------------
% Controller
% ---------------------------------------------------------------

% first-order identified model
% (identified under process and measurement noise)
% Continuous-time transfer functions

```

```matlab

% G11
num11 = 0.0822;
den11 = [140.2627 1.0000];

% G12
num12 = 0.1258;
den12 = [231.0847 1.0000];

% G21
num21 = 0.1523;
den21 = [216.4180 1.0000];

% G22
num22 = 0.1110;
den22 = [137.5618 1.0000];


% set up state-space realization of identified model

num = cell(2,2); den = cell(2,2); tau = zeros(2,2);

num{1,1} = num11; num{1,2} = num12;
num{2,1} = num21; num{2,2} = num22;

den{1,1} = den11; den{1,2} = den12;
den{2,1} = den21; den{2,2} = den22;


Nmax = 100;
tol = 1e-8;

[A,B,C,D,sH] = mimoctf2dss(num,den,tau,Ts,Nmax,tol);



% -----------------------------------------------------------------
% Controller model - deterministic-stochastic model
% -----------------------------------------------------------------

% size of system state
nxmodel = size(A,1);


% disturbance model
As = eye(2,2);
Ks = [(1-alpha1) 0; 0 (1-alpha2)];
Cs = eye(2,2);

% combined model
Abar = [A zeros(nxmodel,2); zeros(2,nxmodel) As];
Bbar = [B; zeros(2,2)];
Kbar = [zeros(nxmodel,2); Ks];
Cbar = [C Cs];

```

```
% controller struct
Controller.Abar = Abar;
Controller.Bbar = Bbar;
Controller.Cbar = Cbar;
Controller.Kbar = Kbar;


% size of controller model
ny = size(Cbar,1);
nu = size(Bbar,2);


% ————————————————————————————————————————————————————————————
% Controller model 1 — deterministic—stochastic model
% ————————————————————————————————————————————————————————————

% disturbance model
As1 = eye(2,2);
Ks1 = [(1—alpha11) 0; 0 (1—alpha21)];
Cs1 = eye(2,2);

% combined model
Abar1 = [A zeros(nxmodel,2); zeros(2,nxmodel) As1];
Bbar1 = [B; zeros(2,2)];
Kbar1 = [zeros(nxmodel,2); Ks1];
Cbar1 = [C Cs1];

% controller struct
Controller1.Abar = Abar1;
Controller1.Bbar = Bbar1;
Controller1.Cbar = Cbar1;
Controller1.Kbar = Kbar1;


% ————————————————————————————————————————————————————————————
% Reference
% ————————————————————————————————————————————————————————————

% reference height in tank 1
refh1 =  ys(1);

% reference height in tank 2
refh2 =  ys(2);

% reference vector
ref = [refh1;refh2];

r = repmat(ref,1,(Nsim+N));

% step in reference
r(1,2551:end) = ys(1) + (58 — ys(1));


% ————————————————————————————————————————————————————————————
% Disturbance
% ---------------------------------------------------------------

d3 = ds(1)*ones(1,Nsim+N);
d4 = ds(2)*ones(1,Nsim+N);

% step in tank 3
d3(d1time:end) = ds(1) + d1;

% step in tank 4
d4(d2time:end) = ds(2) + d2;

d = [d3;d4];

% ---------------------------------------------------------------
%  Weights for controller
% ---------------------------------------------------------------

% weight matrix Q (reference)
Q = eye(ny,ny);
Q(1,1) = q1;
Q(2,2) = q2;

% weight matrix S (control input)
S = eye(nu,nu);
S(1,1) = s1;
S(2,2) = s2;

% ---------------------------------------------------------------
% Controller gains - deterministic-stochastic model
% ---------------------------------------------------------------

% design matrices
[phix phie gammau H varphi I0 Qcal Scal] = ...
          MPCDesign(N,Abar,Bbar,Cbar,Kbar,Q,S);

% gain matrices for explicit MPC solution
[Lx Le LR Lu] = MPCDesignUnconstrained ...
    (H,phix,phie,gammau,varphi,I0,Qcal,Scal);


% controller gains
Controller.Lx = Lx;
Controller.Le = Le;
Controller.LR = LR;
Controller.Lu = Lu;

Controller.N = N;


% ---------------------------------------------------------------
%  Weights of controller 1
% ---------------------------------------------------------------

% weight matrix Q (reference)
Q1 = eye(ny,ny);
```

```
Q1(1,1) = q11;
Q1(2,2) = q21;

% weight matrix S (control input)
S1 = eye(nu,nu);
S1(1,1) = s11;
S1(2,2) = s21;

% ---------------------------------------------------------------
% Controller gains 1 - deterministic-stochastic model
% ---------------------------------------------------------------

% design matrices
[phix1 phie1 gammau1 H1 varphi1 I01 Qcal1 Scal1] =  ...
             MPCDesign(N,Abar1,Bbar1,Cbar1,Kbar1,Q1,S1);

% gain matrices for explicit MPC solution
[Lx1 Le1 LR1 Lu1] = MPCDesignUnconstrained ...
    (H1,phix1,phie1,gammau1,varphi1,I01,Qcal1,Scal1);

% controller gains
Controller1.Lx = Lx1;
Controller1.Le = Le1;
Controller1.LR = LR1;
Controller1.Lu = Lu1;

Controller1.N = N;

% ---------------------------------------------------------------
% Closed-loop simulations
% ---------------------------------------------------------------

% initial state
x0 = xs;


% closed-loop simulation with nonlinear plant

% for tuning 1
[uk yk zk]    = ClosedloopSimulationUnconstrainedNonlinear3 ...
                  (Nsim,SystemNonlinear,Controller,x0,d,v,w,r);

% for tuning 2
[uk1 yk1 zk1] = ClosedloopSimulationUnconstrainedNonlinear3 ...
                 (Nsim,SystemNonlinear,Controller1,x0,d,v,w,r);

% ---------------------------------------------------------------
% Closed-loop plots
% ---------------------------------------------------------------

figure
plot(T,yk(1,:),T,yk1(1,:),T,r(1,1:Nsim),'--k','LineWidth',3)
ylim([40 65])
xlabel('Time [min]','FontSize',20);
ylabel('y_1 [cm]','FontSize',20);
```

```

set(gca,'FontSize',20);
legend('C_1 (M_{S,max} = 1.50)','C_2 (M_{S,max} = 1.25)','r_1',4);

figure
stairs(T,[uk(1,:)' uk1(1,:)'],'LineWidth',3)
xlabel('Time [min]','FontSize',20);
ylabel('u_1 [cm^3/s]','FontSize',20);
set(gca,'FontSize',20);
ylim([0 350])

figure
plot(T,yk(2,:),T,yk1(2,:),T,r(2,1:Nsim),'--k','LineWidth',3)
ylim([30 60])
xlabel('Time [min]','FontSize',20);
ylabel('y_2 [cm]','FontSize',20);
set(gca,'FontSize',20);
legend('C_1 (M_{S,max} = 1.50)','C_2 (M_{S,max} = 1.25)','r_2',1);

figure
stairs(T,[uk(2,:)' uk1(2,:)'],'LineWidth',3)
xlabel('Time [min]','FontSize',20);
ylabel('u_2 [cm^3/s]','FontSize',20);
set(gca,'FontSize',20);
```

```
clear all
close all
clc

addpath('Realization')

% Closed-loop simulation of modified 4-tank system
% based on second order identified model - Unconstrained case

% Plant model: Non-linear modified 4-tank system
% Controller model: Deterministic-Stochastic model


% ----------------------------------------------------------------
%  parameters
% ----------------------------------------------------------------

% sample-time [s]
Ts = 4;

% time of simulation in minutes
Tf = 500;

% Prediction horizon
N = 500;

% ----------------------------------------------------------------
% steady state for simulation
```

```
us = [300;300];
ds = [70;70];

gamma1 = 0.45;    % Flow distribution constant. Valve 1
gamma2 = 0.40;    % Flow distribution constant. Valve 2

[Axx Bxx Exx Cxx xs ys] = getlinearizedModified4TankSystem ...
                                    (us,ds,gamma1,gamma2);

% ---------------------------------------------------------------


%MSmax = 1.25;
x1 = [0.0001 0.0001 26.4586 169.6454 1.0393 1.0852];

%MSmax = 1.10;
x2 = [4.0062e-08 2.7689e-08 105.2840 108.2985 0.0431 0.0449];


% ---------------------------------------------------------------
% Simulation details
% ---------------------------------------------------------------

% disturbance
d1 = 0;
d2 = 40;

% sample of disturbance start
d1time = 551;
d2time = 551;

% sample-time [min]
Tstep = Ts/60;

% time interval
T = 0:Tstep:Tf;

Nsim = length(T);


% ---------------------------------------------------------------
%  Noise
% ---------------------------------------------------------------

% noise variance
sigma2w = (20)^2;
sigma2v = (0.5)^2;

% covariance matrices
Qw = sigma2w*eye(2,2);
Rv = sigma2v*eye(2,2);

Swv = zeros(2,2);

% combined covariance matrix
```

```
Z = [Qw Swv; Swv Rv];

LZ = chol(Z,'lower');
wv = LZ*randn(4,Nsim+N);

w = wv(1:2,:);    % process noise
v = wv(3:4,:);    % measurement noise



% tuning 1
alpha1 = x1(1);
alpha2 = x1(2);

s1 = x1(3);
s2 = x1(4);

q1 = x1(5);
q2 = x1(6);


% tuning2
alpha11 = x2(1);
alpha21 = x2(2);

s11 = x2(3);
s21 = x2(4);

q11 = x2(5);
q21 = x2(6);



% -----------------------------------------------------------------
% Non-linear modified 4-tank plant
% -----------------------------------------------------------------

p = getnonlinearsystemparameters(gamma1,gamma2);

SystemNonlinear.p  = p;
SystemNonlinear.Ts = Ts;


% -----------------------------------------------------------------
% Controller
% -----------------------------------------------------------------

% second-order identified model
% (under process and measurement noise)
% Continuous-time transfer functions

% G11
num11 = 0.0818;
den11 = conv([25.3224 1],[108.7843 1]);

```

```
% G12
num12 = 0.1238;
den12 = conv([104.5181 1],[104.4942 1]);

% G21
num21 = 0.1520;
den21 = conv([7.6230 1],[206.9884 1]);

% G22
num22 = 0.1091;
den22 = conv([9.3216e-6 1],[96.1746 1]);


% set up state-space realization of identified model

num=cell(2,2); den=cell(2,2); tau = zeros(2,2);

num{1,1}=num11; num{1,2}=num12;
num{2,1}=num21; num{2,2}=num22;

den{1,1}=den11; den{1,2}=den12;
den{2,1}=den21; den{2,2}=den22;


Nmax = 100;
tol = 1e-8;

[A,B,C,D,sH] = mimoctf2dss(num,den,tau,Ts,Nmax,tol);


% ---------------------------------------------------------------
% Controller model - deterministic-stochastic model
% ---------------------------------------------------------------

nxmodel = size(A,1);

% disturbance model
As = eye(2,2);
Ks = [(1-alpha1) 0; 0 (1-alpha2)];
Cs = eye(2,2);

% combined model
Abar = [A zeros(nxmodel,2); zeros(2,nxmodel) As];
Bbar = [B; zeros(2,2)];
Kbar = [zeros(nxmodel,2); Ks];
Cbar = [C Cs];

% controller struct
Controller.Abar = Abar;
Controller.Bbar = Bbar;
Controller.Cbar = Cbar;
Controller.Kbar = Kbar;

% size of controller model
ny = size(Cbar,1);
```

```
nu = size(Bbar,2);

% ————————————————————————————————————————————————————————————
% Controller model 1 — deterministic—stochastic model
% ————————————————————————————————————————————————————————————

% disturbance model
As1 = eye(2,2);
Ks1 = [(1—alpha11) 0; 0 (1—alpha21)];
Cs1 = eye(2,2);

% combined model
Abar1 = [A zeros(nxmodel,2); zeros(2,nxmodel) As1];
Bbar1 = [B; zeros(2,2)];
Kbar1 = [zeros(nxmodel,2); Ks1];
Cbar1 = [C Cs1];

% controller struct
Controller1.Abar = Abar1;
Controller1.Bbar = Bbar1;
Controller1.Cbar = Cbar1;
Controller1.Kbar = Kbar1;


% ————————————————————————————————————————————————————————————
% Reference
% ————————————————————————————————————————————————————————————

% reference height in tank 1
refh1 =  ys(1);

% reference height in tank 2
refh2 =  ys(2);

% reference vector
ref = [refh1;refh2];

r = repmat(ref,1,(Nsim+N));

% reference step
r(1,2551:end) = ys(1) + (58 — ys(1));

% ————————————————————————————————————————————————————————————
% Disturbance
% ————————————————————————————————————————————————————————————

d3 = ds(1)*ones(1,Nsim+N);
d4 = ds(2)*ones(1,Nsim+N);

% step in tank 3
d3(d1time:end) = ds(1) + d1;

% step in tank 4
d4(d2time:end) = ds(2) + d2;

```

```
d = [d3;d4];

% --------------------------------------------------------------
%  Weights
% --------------------------------------------------------------

% weight matrix Q (reference)
Q = eye(ny,ny);
Q(1,1) = q1;
Q(2,2) = q2;

% weight matrix S (control input)
S = eye(nu,nu);
S(1,1) = s1;
S(2,2) = s2;


% --------------------------------------------------------------
% Controller gains — deterministic—stochastic model
% --------------------------------------------------------------

% design matrices
[phix phie gammau H varphi I0 Qcal Scal] = ...
                           MPCDesign(N,Abar,Bbar,Cbar,Kbar,Q,S);

% gain matrices for explicit MPC solution
[Lx Le LR Lu] = MPCDesignUnconstrained ...
                       (H,phix,phie,gammau,varphi,I0,Qcal,Scal);

% controller gains
Controller.Lx = Lx;
Controller.Le = Le;
Controller.LR = LR;
Controller.Lu = Lu;

Controller.N = N;


% --------------------------------------------------------------
%  Weights of controller 1
% --------------------------------------------------------------

% weight matrix Q (reference)
Q1 = eye(ny,ny);
Q1(1,1) = q11;
Q1(2,2) = q21;

% weight matrix S (control input)
S1 = eye(nu,nu);
S1(1,1) = s11;
S1(2,2) = s21;

% --------------------------------------------------------------
% Controller gains — deterministic—stochastic model
% --------------------------------------------------------------

% design matrices
[phix1 phie1 gammau1 H1 varphi1 I01 Qcal1 Scal1] = ...
                    MPCDesign(N,Abar1,Bbar1,Cbar1,Kbar1,Q1,S1);

% gain matrices for explicit MPC solution
[Lx1 Le1 LR1 Lu1] = MPCDesignUnconstrained ...
              (H1,phix1,phie1,gammau1,varphi1,I01,Qcal1,Scal1);

% controller gains
Controller1.Lx = Lx1;
Controller1.Le = Le1;
Controller1.LR = LR1;
Controller1.Lu = Lu1;

Controller1.N = N;


% --------------------------------------------------------------
% Closed-loop simulations
% --------------------------------------------------------------

% initial state
x0 = xs;

% closed-loop simulation with nonlinear plant

% for tuning 1
[uk yk zk] = ClosedloopSimulationUnconstrainedNonlinear4 ...
               (Nsim,SystemNonlinear,Controller,x0,d,v,w,r);

% for tuning 2
[uk1 yk1 zk1] = ClosedloopSimulationUnconstrainedNonlinear4 ...
                 (Nsim,SystemNonlinear,Controller1,x0,d,v,w,r);

% --------------------------------------------------------------
% Closed-loop plots
% --------------------------------------------------------------

figure
plot(T,yk(1,:),T,yk1(1,:),T,r(1,1:Nsim),'--k','LineWidth',3)
ylim([40 65])
xlabel('Time [min]','FontSize',20);
ylabel('y_1 [cm]','FontSize',20);
set(gca,'FontSize',20);
legend('C_1 (M_{S,max} = 1.25)','C_2 (M_{S,max} = 1.10)','r_1',4);

figure
stairs(T,[uk(1,:)' uk1(1,:)'],'LineWidth',3)
xlabel('Time [min]','FontSize',20);
ylabel('u_1 [cm^3/s]','FontSize',20);
set(gca,'FontSize',20);
ylim([0 350])

figure
```

```
plot(T,yk(2,:),T,yk1(2,:),T,r(2,1:Nsim),'--k','LineWidth',3)
ylim([30 60])
xlabel('Time [min]','FontSize',20);
ylabel('y_2 [cm]','FontSize',20);
set(gca,'FontSize',20);
legend('C_1 (M_{S,max} = 1.25)','C_2 (M_{S,max} = 1.10)','r_2',1);

figure
stairs(T,[uk(2,:)' uk1(2,:)'],'LineWidth',3)
xlabel('Time [min]','FontSize',20);
ylabel('u_2 [cm^3/s]','FontSize',20);
set(gca,'FontSize',20);
```

A.2 MPC Setup Files

```
function [phix gammau] = phigammampc(A,B,C,N)
% setup matrices phix and gammau for the function
% MPCDesign

% INPUTS:
% A           - discrete-time system matrix
% B           - discrete-time control input matrix
% C           - discrete-time measurement matrix
% N           - prediction horizon

% Outputs:
% phix        - MPC design matrix - dimension (N*ny) x (nx)
% gammau      - MPC design matrix - dimension (N*ny) x (N*nu)


% dimensions
nx = size(A,1);
nu = size(B,2);
ny = size(C,1);

% preallocation
phix = zeros(N*ny,nx);
gammau = zeros(N*ny,N*nu);

% counters
k1 = 1;
k2 = ny;

% temporary
T = C;

% assemble phix and first block column of gammau
for i = 1:N
    gammau(k1:k2,1:nu) = T*B;

    T = T*A;

    phix(k1:k2,1:nx) = T;

    k1 = k1 + ny;
    k2 = k2 + ny;
end

% counters
c1 = nu+1;
c2 = 2*nu;

r1 = ny+1;
r2 = ny;

% assemble remaining block columns of gammau
for j = 1:N-1
    gammau(r1:end,c1:c2) = gammau(1:end-r2,1:nu);

    c1 = c1 + nu;
    c2 = c2 + nu;

    r1 = r1 + ny;
    r2 = r2 + ny;
end
```

```
function [phix phie gammau H varphi I0 Qcal Scal] = ...
                         MPCDesign(N,A,B,C,K,Q,S)
% Design MPC  Design the matrices for the MPC setup

% INPUTS:
% A    - discrete-time system matrix
% B    - discrete-time control input matrix
% C    - discrete-time measurement matrix
% K    - discrete-time innovation matrix

% N    - prediction horizon
% Q    - weight matrix on reference tracking
% S    - weight matrix on control input rate movement


% dimension of control input u_k and controlled output y_k
nu = size(B,2);
ny = size(C,1);

% weight matrix on reference tracking
Qcal = kron(eye(N,N),Q);

% weight matrix on input rate movement
Scal = kron(eye(N,N),S);

% setup phix and gammau
```

```
[phix gammau] = phigammampc(A,B,C,N);

% setup phie
temp = [C;phix];
temp = temp(1:N*ny,1:end);
phie = temp*K;

% setup varphi
varphistencil = diag(-1*ones(N-1,1),-1) + eye(N);
varphi = kron(varphistencil,eye(nu));

% setup I0
I0 = eye(N*nu,nu);

% setup H
H = (gammau'*Qcal*gammau + varphi'*Scal*varphi);
```

```
function [Lx Le LR Lu] = MPCDesignUnconstrained ...
        (H,phix,phie,gammau,varphi,I0,Qcal,Scal)
% MPCDesignUnconstrained
%   Computes the gain matrices for the unconstrained MPC
%   with controller model in innovation form

% Inputs:
% H       - Hessian matrix for unconstrained MPC QP
% phix    - MPC design matrix
% phie    - MPC design matrix
% gammau  - MPC design matrix
% varphi  - MPC design matrix
% I0      - MPC design matrix
% Qcal    - Weight matrix on reference
% Scal    - Weight matrix on control input

% Outputs:
% Lx      - Gain matrix for state estimate
% Le      - Gain matrix for innovation
% LR      - Gain matrix for reference
% Lu      - Gain matrix for on previous control input


% Cholesky factorization of H matrix
[R,p] = chol(H);

if p > 0
error('H is not positive definite')
end

I0t = I0';

% temporary variables
temp = gammau'*Qcal;

Lx0 = temp*phix;
```

```
Le0 = temp*phie;
LR0 = -temp;
Lu0 = -varphi'*Scal*I0;

% solve linear systems to obtain gains for the
%optimal control input Uk for the whole MPC horizon
Lx1 = -(R\(R'\Lx0));
Le1 = -(R\(R'\Le0));
LR1 = -(R\(R'\LR0));
Lu1 = -(R\(R'\Lu0));

% gains for the optimal control input uk
Lx = I0t*Lx1;
Le = I0t*Le1;
LR = I0t*LR1;
Lu = I0t*Lu1;
```

```
function [uhatkk xhatkp1k] = MPCcomputeUnconstrainedInnovation ...
                            (xhatkkm1,ukm1,yk,Rk,Lx,Le,LR,Lu,A,B,C,K)
% MPCcomputeunconstrained
% Compute optimal control input uk for the unconstrained MPC problem
% with controller model in innovation form
%
% Inputs:
% xhatkkm1         - One-step state prediction
% ukm1             - previous control input
% yk               - Plant measurement
% Rk               - Reference of the current MPC horizon
% Lx, Le, LR, Lu   - Gain matrices
% A, B, C, K       - Controller model matrices
%
% Outputs:
% uhatkk           - Optimal control input
% xhatkp1k         - Updated one-step state prediction


% innovation
ek = yk - C*xhatkkm1;

% optimal control input
uhatkk = Lx*xhatkkm1 + Le*ek + LR*Rk + Lu*ukm1;

% one-step state prediction
xhatkp1k = A*xhatkkm1 + B*uhatkk + K*ek;
```

```
function [Umin Umax dUmin dUmax] = MPCDesignConstraints ...
                               (N,umin,umax,dumin,dumax)
% Setup hard constraints on control inputs and control
% input rate movement for the constrained MPC problem

% Inputs:
```

```
% umin      — lower bound on control input
% umax      — upper bound on control input
% dumin     — lower bound on control input rate movement
% dumax     — upper bound on control input rate movement
% N         — prediction horizon

% Outputs:
% Umin      — lower bound on control input
%             for MPC horizon — N
%
% Umax      — upper bound on control input
%             for MPC horizon — N
%
% dUmin     — lower bound on control input rate movement
%             for MPC horizon — N
%
% dUmax     — upper bound on control input rate movement
%             for MPC horizon — N

% dimensions of outputs (N*nu) x 1


% hard constraints on control inputs
Umin = repmat(umin,N,1);
Umax = repmat(umax,N,1);

% hard constraints on control input rate movement
dUmin = repmat(dumin,N,1);
dUmax = repmat(dumax,N,1);
```

```
function [uhatkk xhatkp1k info U] = MPCcomputeConstrainedInnovation...
(xhatkkm1,ukm1,yk,Rk,A,B,C,K,phix,phie,varphi,...
Umin,Umax,dUmin,dUmax,I0,H,Uinit,Z1,Z2)
% MPCcomputeConstrainedInnovation
% Computes optimal control input uk for the constrained
% MPC problem with controller model in innovation form
%
% Inputs:
% yk          — Plant measurement
% xhatkkm1    — One—step state prediction
% A           — Measurement matrix
% B           — Measurement matrix
% C           — Measurement matrix
% K           — Measurement matrix
% Rk          — Reference vector for current MPC horizon
% phix        — MPC design matrix
% phie        — MPC design matrix
% varphi      — MPC design matrix
% Umin        — lower bound on control inputs
% Umax        — upper bound on control inputs
% dUmin       — lower bound on control input rate movement
% dUmax       — upper bound on control input rate movement
% I0          — MPC design matrix
% H          - MPC design matrix
% Z1         - MPC design matrix
% Z2         - MPC design matrix
% Uinit      - initial guess on U for QP solver

% Outputs:
% uhatkk      - Optimal control input
% xhatkp1k    - Updated one-step state prediction
% info        - QP convergence information
% U           - Optimal control inputs over the
%               MPC prediction horizon

% innovation
ek = yk - C*xhatkkm1;

% g in the objective function is updated
bk = phix*xhatkkm1 + phie*ek;

g = Z1*(bk - Rk) - Z2*ukm1;

% constraints for the control input rate movement are updated
bl = dUmin + I0*ukm1;
bu = dUmax + I0*ukm1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%       QPsolver         %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[U,info] = qpsolver(H,g,Umin,Umax,varphi,bl,bu,Uinit);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% size of control input
sizeu = size(ukm1);

% optimal control input
uhatkk = U(1:sizeu(1),1);

% one-step state prediction
xhatkp1k = A*xhatkkm1 + B*uhatkk + K*ek;
```

```
function [x,fval,exitflag,output,lambda] = ...
               qpsolver(H,g,l,u,A,bl,bu,xinit)

Abar = [A;-A];
bbar = [bu;-bl];

options = optimset('Algorithm','interior-point','MaxIter',500);

[x,fval,exitflag,output,lambda] = ...
               quadprog(H,g,Abar,bbar,[],[],l,u,xinit,options);
```

A.3 Tuning Files

```
clear all
close all
clc

addpath('Realization')

% TUNING APPROACH 1

% Test file for tuning procedure of modified 4—tank system
% based on tuning approach 1 for first—order identified
% model


% sample—time [s]
Ts = 4;

% Prediction horizon
N = 500;


% ——————————————————————————————————————————————————————————————
% Linear plant model: identified model + deviation
% ——————————————————————————————————————————————————————————————

% deviation
dev = 0.90;


% G11
num11p = 0.0822*dev;
den11p = [140.2627 1.0000];

% G12
num12p = 0.1258*dev;
den12p = [231.0847 1.0000];

% G21
num21p = 0.1523*dev;
den21p = [216.4180 1.0000];

% G22
num22p = 0.1110*dev;
den22p = [137.5618 1.0000];


% set up state—space realization of plant model

nump = cell(2,2); denp = cell(2,2); taup = zeros(2,2);

nump{1,1}=num11p; nump{1,2}=num12p;
nump{2,1}=num21p; nump{2,2}=num22p;
```

```

denp{1,1}=den11p; denp{1,2}=den12p;
denp{2,1}=den21p; denp{2,2}=den22p;


Nmax = 100; tol = 1e-8;

[Ap,Bp,Cp,Dp,sHp] = mimoctf2dss(nump,denp,taup,Ts,Nmax,tol);



% ---------------------------------------------------------------
% Controller model: identified first-order model
% ---------------------------------------------------------------

% first-order identified model
% (identified under process and measurement noise)


% G11
num11 = 0.0822;
den11 = [140.2627 1.0000];

% G12
num12 = 0.1258;
den12 = [231.0847 1.0000];

% G21
num21 = 0.1523;
den21 = [216.4180 1.0000];

% G22
num22 = 0.1110;
den22 = [137.5618 1.0000];


% set up state-space realization of controller model

num=cell(2,2); den=cell(2,2); tau = zeros(2,2);

num{1,1}=num11; num{1,2}=num12;
num{2,1}=num21; num{2,2}=num22;

den{1,1}=den11; den{1,2}=den12;
den{2,1}=den21; den{2,2}=den22;


[A,B,C,D,sH] = mimoctf2dss(num,den,tau,Ts,Nmax,tol);



% ---------------------------------------------------------------
% Solver and solver parameters
% ---------------------------------------------------------------


% sensitivity bound
MSmax = 1.10;

% solver settings
options = optimset('Algorithm','interior-point','display' ...
                                  ,'iter','MaxFunEvals',2200);

% bounds on tuning parameters
lb = [0*ones(2,1); 0*ones(2,1); 0*ones(2,1)];
ub = [1*ones(2,1); 1e6*ones(2,1); 1e6*ones(2,1)];

% start point
x0 = [0.7*ones(2,1); 100*ones(2,1); 1*ones(2,1)];


% compute solution
x = fmincon(@(x)ObjectiveFun1(x,Ap,Bp,Cp,A,B,C,N,Ts),x0,[],[],[],[],...
    lb,ub,@(x)ConFun1(x,Ap,Bp,Cp,A,B,C,N,MSmax,Ts),options);
```

```
function J = ObjectiveFun1(x,Ap,Bp,Cp,A,B,C,N,Ts)
% ObjectiveFun1
% Evaluates objective function for tuning optimization problem
% for tuning approach 1


% number of measurements
ny =size(C,1);

% number of control inputs
nu =size(B,2);

% disturbance model parameters
alpha = x(1:ny);

% weight matrix Su
S = diag(x(ny+1:ny+nu));

% weight matrix Qy
Q = diag(x(ny+nu+1:end));


% -----------------------------------------------------------
% Parameters
% -----------------------------------------------------------

% time of simulation in minutes
Tf = 80;

% sample-time [min]
Tstep = Ts/60;
T = 0:Tstep:Tf;
Nsim = length(T);
```

```

rstep = 1;
rtime = 601;


% -----------------------------------------------------------
% Controller model - deterministic-stochastic model
% -----------------------------------------------------------

% size of identified model state
nx = size(A,1);

% disturbance model
As = eye(ny,ny);
Ks = diag(ones(ny,1)-alpha);
Cs = eye(ny,ny);

% combined model
Abar = [A zeros(nx,ny); zeros(ny,nx) As];
Bbar = [B; zeros(ny,nu)];
Kbar = [zeros(nx,ny); Ks];
Cbar = [C Cs];


% --------------------------------------------------------------
% Controller gains - deterministic-stochastic model
% --------------------------------------------------------------

% design matrices for controller
[phix phie gammau H varphi I0 Qcal Scal] = ...
                           MPCDesign(N,Abar,Bbar,Cbar,Kbar,Q,S);

% gain matrices for explicit MPC solution
[Lx Le LR Lu] = MPCDesignUnconstrained ...
                       (H,phix,phie,gammau,varphi,I0,Qcal,Scal);

% --------------------------------------------------------------
% Controller state-space
% --------------------------------------------------------------

Ac = [((Abar - Kbar*Cbar) + Bbar*(Lx - Le*Cbar)) Bbar*Lu;
                                         Lx - Le*Cbar Lu];

Bcy = [Kbar + Bbar*Le; Le];
Bcr = [Bbar*LR; LR];

Cc = [Lx-Le*Cbar Lu];

Dcy = Le;
Dcr = LR;


% state size of controller state-space
nxc = size(Ac,1);

```

```
% ---------------------------------------------------------------
% Closed-loop systen
% ---------------------------------------------------------------

Acl = [Ap + Bp*Dcy*Cp Bp*Cc; Bcy*Cp Ac];

%Bdcl = [Ep; zeros(8,2)];
%Bwcl = [Gp; zeros(8,2)];

%Bvcl = [Bp*Dcy;Bcy];
Brcl = [Bp*Dcr;Bcr];

Ccl = [Cp zeros(2,nxc)];
%Cucl = [Dcy*Cp Cc];

%Dvcl = Dcy;
%Drcl = Dcr;




% ---------------------------------------------------------------
% Integrated Absolute Error
% ---------------------------------------------------------------


% reference
r = zeros(2,Nsim+N);


% IAE for reference scenario 1
% ---------------------------------------------------------------


r1 = r;
r1(1,rtime:end) = rstep;


% initial state
xkclr1 = zeros(size(Acl,1),1);

for j = 1:Nsim
    Rtempr1 = r1(:,j:(j+N)-1);
    Rkr1 = Rtempr1(:);

    xkclr1next = Acl*xkclr1 + Brcl*Rkr1;
    yr1(:,j) = Ccl*xkclr1;

    xkclr1 = xkclr1next;
end


% tank 1, step in reference 1
e11r = yr1(1,:) - r1(1,1:Nsim);

```

```
% tank 2, step in reference 1
e21r = yr1(2,:) - r1(2,1:Nsim);


% IAE
Jr11 = sum(abs(e11r));
Jr21 = sum(abs(e21r));


% IAE for reference scenario 2
% ---------------------------------------------------------------

r2 = r;
r2(2,rtime:end) = rstep;


% initial state
xkclr2 = zeros(size(Acl,1),1);

for j = 1:Nsim
    Rtempr2 = r2(:,j:(j+N)-1);
    Rkr2 = Rtempr2(:);


    xkclr2next = Acl*xkclr2 + Brcl*Rkr2;
    yr2(:,j) = Ccl*xkclr2;

    xkclr2 = xkclr2next;
end


% tank 1, step in reference 2
e12r = yr2(1,:) - r2(1,1:Nsim);

% tank 2, step in reference 2
e22r = yr2(2,:) - r2(2,1:Nsim);


% IAE
Jr12 = sum(abs(e12r));
Jr22 = sum(abs(e22r));


% Objective function
J = Jr11 + Jr21 + Jr12 + Jr22;
```

```
function [c,ceq] = ConFun1(x,Ap,Bp,Cp,A,B,C,N,MSmax,Ts)
% ConFun1
% Evaluates constraint for tuning optimization problem
% for tuning approach 1


% number of measurements
```

```
ny =size(C,1);

% number of control inputs
nu =size(B,2);

% disturbance model parameters
alpha = x(1:ny);

% weight matrix Su
S = diag(x(ny+1:ny+nu));

% weight matrix Qy
Q = diag(x(ny+nu+1:end));


% --------------------------------------------------------------
% Controller model - deterministic-stochastic model
% --------------------------------------------------------------

% size of identified model state
nx = size(A,1);

% disturbance model
As = eye(ny,ny);
Ks = diag(ones(ny,1)-alpha);
Cs = eye(ny,ny);

% combined model
Abar = [A zeros(nx,ny); zeros(ny,nx) As];
Bbar = [B; zeros(ny,nu)];
Kbar = [zeros(nx,ny); Ks];
Cbar = [C Cs];

% --------------------------------------------------------------
% Controller gains - deterministic-stochastic model
% --------------------------------------------------------------

% design matrices
[phix phie gammau H varphi I0 Qcal Scal] =  ...
                           MPCDesign(N,Abar,Bbar,Cbar,Kbar,Q,S);

% gain matrices for explicit MPC solution
[Lx Le LR Lu] = MPCDesignUnconstrained ...
                       (H,phix,phie,gammau,varphi,I0,Qcal,Scal);


% --------------------------------------------------------------
% Controller in state-space form
% --------------------------------------------------------------


Ac = [((Abar - Kbar*Cbar) + Bbar*(Lx - Le*Cbar)) Bbar*Lu;
                                        Lx - Le*Cbar Lu];

Bcy = [Kbar + Bbar*Le; Le];
```

```
Bcr = [Bbar*LR; LR];

Cc = [Lx—Le*Cbar Lu];

Dcy = Le;
Dcr = LR;


nxc = size(Ac,1);

% ————————————————————————————————————————————————————————————————
% Closed—loop state—space: Controller + Linear plant
% ————————————————————————————————————————————————————————————————



Acl = [Ap + Bp*Dcy*Cp Bp*Cc; Bcy*Cp Ac];

%Bwcl = [Gp; zeros(8,2)]; % unknown
%Bdcl = [Ep; zeros(8,2)]; % unknown

Bvcl = [Bp*Dcy;Bcy];
%Brcl = [Bp*Dcr;Bcr];

Ccl = [Cp zeros(2,nxc)];
%Cucl = [Dcy*Cp Cc];

%Dvcl = Dcy;
%Drcl = Dcr;


% ————————————————————————————————————————————————————————————————
% Sensitivity
% ————————————————————————————————————————————————————————————————

w = 0:0.0001:(pi/Ts);
sv = sigma(ss(Acl,Bvcl,Ccl,eye(ny),Ts),w);

MS = max(max(sv))

% constraints
c   = MS — MSmax;
ceq = [];
```

```
clear all
close all
clc

addpath('Realization')

% TUNING APPROACH 2

% Test file for tuning procedure of modified 4—tank system
```

```matlab
% based on tuning approach 2 for first-order or second-order
% identified model


% sample-time [s]
Ts = 4;

% Prediction horizon
N = 500;


% ------------------------------------------------------------
% Identified model
% ------------------------------------------------------------

% 1)
% first-order identified model
% (identified under process and measurement noise)

% G11
num11 = 0.0822;
den11 = [140.2627 1.0000];

% G12
num12 = 0.1258;
den12 = [231.0847 1.0000];

% G21
num21 = 0.1523;
den21 = [216.4180 1.0000];

% G22
num22 = 0.1110;
den22 = [137.5618 1.0000];

% 2)
% second-order identified model
% (identified under process and measurement noise)


% % G11
% num11 = 0.0818;
% den11 = conv([25.3224 1],[108.7843 1]);
%
% % G12
% num12 = 0.1238;
% den12 = conv([104.5181 1],[104.4942 1]);
%
% % G21
% num21 = 0.1520;
% den21 = conv([7.6230 1],[206.9884 1]);
%
% % G22
% num22 = 0.1091;
% den22 = conv([9.3216e-6 1],[96.1746 1]);


% set up state-space realization of identified model

num = cell(2,2); den = cell(2,2); tau = zeros(2,2);

num{1,1}=num11; num{1,2}=num12;
num{2,1}=num21; num{2,2}=num22;

den{1,1}=den11; den{1,2}=den12;
den{2,1}=den21; den{2,2}=den22;


Nmax=100; tol=1e-8;

[A,B,C,D,sH] = mimoctf2dss(num,den,tau,Ts,Nmax,tol);




% -----------------------------------------------------------
% Solver and solver parameters
% -----------------------------------------------------------


% sensitivity bound
MSmax = 1.20;


% solver settings
options = optimset('Algorithm','interior-point','display' ...
                                 ,'iter','MaxFunEvals',2200);


% bounds on tuning parameters
lb = [zeros(2,1); 0*ones(2,1); 0*ones(2,1)];
ub = [ ones(2,1); 1e6*ones(2,1); 1e6*ones(2,1)];

% start point
x0 = [0.7*ones(2,1); 100*ones(2,1); 1*ones(2,1)];


% compute solution
x = fmincon(@(x)ObjectiveFun2(x,A,B,C,N),x0,[],[],[],[] ...
           ,lb,ub,@(x)ConFun2(x,A,B,C,N,MSmax,Ts),options);
```

```
function J = ObjectiveFun2(x,A,B,C,N)
% ObjectiveFun2
% Evaluates objective function for tuning optimization problem
% for tuning approach 2


% number of measurements
```

```matlab
ny =size(C,1);

% number of control inputs
nu =size(B,2);

% disturbance model parameters
alpha = x(1:ny);

% weight matrix Su
S = diag(x(ny+1:ny+nu));

% weight matrix Qy
Q = diag(x(ny+nu+1:end));


% -----------------------------------------------------------------
% Parameters
% -----------------------------------------------------------------

% time of simulation in minutes
Tf = 200;

% sample-time [s]
Ts = 4;

% sample-time [min]
Tstep = Ts/60;
T = 0:Tstep:Tf;
Nsim = length(T);




rstep = 0.1;
rtime = 601;

ubartime = 601;
ubarstep = 1;

ybartime = 601;
ybarstep = 0.1;




% -----------------------------------------------------------------
% Controller model - deterministic-stochastic model
% -----------------------------------------------------------------

% size of identified model state
nx = size(A,1);

% disturbance model
As = eye(ny,ny);
Ks = diag(ones(ny,1)-alpha);
```

```
Cs = eye(ny,ny);

% combined model
Abar = [A zeros(nx,ny); zeros(ny,nx) As];
Bbar = [B; zeros(ny,nu)];
Kbar = [zeros(nx,ny); Ks];
Cbar = [C Cs];



% --------------------------------------------------------------
% Controller gains - deterministic-stochastic model
% --------------------------------------------------------------

% design matrices for controller
[phix phie gammau H varphi I0 Qcal Scal] = ...
                           MPCDesign(N,Abar,Bbar,Cbar,Kbar,Q,S);

% gain matrices for explicit MPC solution
[Lx Le LR Lu] = MPCDesignUnconstrained ...
                       (H,phix,phie,gammau,varphi,I0,Qcal,Scal);


% --------------------------------------------------------------
% Controller state-space
% --------------------------------------------------------------

Ac = [((Abar - Kbar*Cbar) + Bbar*(Lx - Le*Cbar)) Bbar*Lu;
                                        Lx - Le*Cbar Lu];

Bcy = [Kbar + Bbar*Le; Le];
Bcr = [Bbar*LR; LR];

Cc = [Lx-Le*Cbar Lu];

Dcy = Le;
Dcr = LR;


% --------------------------------------------------------------
% Closed-loop systen
% --------------------------------------------------------------

Acl = [A + B*Dcy*C B*Cc; Bcy*C Ac];


sizeAcl = size(Acl,2);
sizeC = size(C,2);

sizediff = sizeAcl - sizeC;


%Bdcl = [Ep; zeros(8,2)];
%Bwcl = [Gp; zeros(8,2)];

```

```
%Bvcl = [B*Dcy;Bcy];
Brcl = [B*Dcr;Bcr];

Ccl = [C zeros(2,sizediff)];
%Cucl = [Dcy*C Cc];

%Dvcl = Dcy;
%Drcl = Dcr;

%%%%%%%%%% TEST %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% closed-loop input disturbance matrix
Bubarcl = [B; zeros(sizediff,2)];


% closed-loop output disturbance matrix
Bybarcl = [B*Dcy;Bcy];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% --------------------------------------------------------------
% Integrated Absolute Error
% --------------------------------------------------------------

% reference
r = zeros(2,Nsim+N);


% IAE for input disturbance scenario 1
% --------------------------------------------------------------

% ubar
ubar = zeros(2,Nsim+N);

ubar1 = ubar;
ubar1(1,ubartime:end) = ubarstep;


% initial state
xkclubar1 = zeros(size(Acl,1),1);

for j = 1:Nsim
    Rtemp = r(:,j:(j+N)-1);
    Rk = Rtemp(:);

    xkclubar1next = Acl*xkclubar1 + Brcl*Rk + Bubarcl*ubar1(:,j);
    yubar1(:,j)   = Ccl*xkclubar1;


    xkclubar1 = xkclubar1next;
end

```

```

% tank 1, step in reference 1
e11u = yubar1(1,:) - r(1,1:Nsim);

% tank 2, step in reference 1
e21u = yubar1(2,:) - r(2,1:Nsim);


% IAE
Ju11 = sum(abs(e11u));
Ju21 = sum(abs(e21u));


% IAE for input disturbance scenario 2
% ---------------------------------------------------------------

ubar2 = ubar;
ubar2(2,ubartime:end) = ubarstep;


% initial state
xkclubar2 = zeros(size(Acl,1),1);

for j = 1:Nsim
    Rtemp = r(:,j:(j+N)-1);
    Rk = Rtemp(:);

    xkclubar2next = Acl*xkclubar2 + Brcl*Rk + Bubarcl*ubar2(:,j);
    yubar2(:,j) =   Ccl*xkclubar2;

    xkclubar2 = xkclubar2next;
end


% tank 1, step in reference 1
e12u = yubar2(1,:) - r(1,1:Nsim);

% tank 2, step in reference 1
e22u = yubar2(2,:) - r(2,1:Nsim);

% IAE
Ju12 = sum(abs(e12u));
Ju22 = sum(abs(e22u));


Ju = Ju11 + Ju21 + Ju12 + Ju22;


% IAE for output disturbance scenario 1
% ---------------------------------------------------------------

% ybar
ybar = zeros(2,Nsim+N);


```

```

ybar1 = ybar;
ybar1(1,ybartime:end) = ybarstep;


% initial state
xkclybar1 = zeros(size(Acl,1),1);

for j = 1:Nsim
    Rtemp = r(:,j:(j+N)-1);
    Rk = Rtemp(:);

    xkclybar1next = Acl*xkclybar1 + Brcl*Rk + Bybarcl*ybar1(:,j);
    yybar1(:,j) =   Ccl*xkclybar1 +                   ybar1(:,j);

    xkclybar1 = xkclybar1next;
end


% tank 1, step in reference 1
e11y = yybar1(1,:) - r(1,1:Nsim);

% tank 2, step in reference 1
e21y = yybar1(2,:) - r(2,1:Nsim);

% IAE
Jy11 = sum(abs(e11y));
Jy21 = sum(abs(e21y));


% IAE for output disturbance scenario 2
% ---------------------------------------------------------------


ybar2 = ybar;
ybar2(2,ybartime:end) = ybarstep;


% initial state
xkclybar2 = zeros(size(Acl,1),1);

for j = 1:Nsim
    Rtemp = r(:,j:(j+N)-1);
    Rk = Rtemp(:);

    xkclybar2next = Acl*xkclybar2 + Brcl*Rk + Bybarcl*ybar2(:,j);
    yybar2(:,j) =   Ccl*xkclybar2 +                   ybar2(:,j);

    xkclybar2 = xkclybar2next;
end


% tank 1, step in reference 1
e12y = yybar2(1,:) - r(1,1:Nsim);

```

```
% tank 2, step in reference 1
e22y = yybar2(2,:) - r(2,1:Nsim);

% IAE
Jy12 = sum(abs(e12y));
Jy22 = sum(abs(e22y));


Jy = Jy11 + Jy21 + Jy12 + Jy22;


% IAE for reference scenario 1
% ---------------------------------------------------------------


r1 = r;
r1(1,rtime:end) = rstep;


% initial state
xkclr1 = zeros(size(Acl,1),1);

for j = 1:Nsim
    Rtempr1 = r1(:,j:(j+N)-1);
    Rkr1 = Rtempr1(:);

    xkclr1next = Acl*xkclr1 + Brcl*Rkr1;
    yr1(:,j) = Ccl*xkclr1;

    xkclr1 = xkclr1next;
end


% tank 1, step in reference 1
e11r = yr1(1,:) - r1(1,1:Nsim);

% tank 2, step in reference 1
e21r = yr1(2,:) - r1(2,1:Nsim);

% IAE
Jr11 = sum(abs(e11r));
Jr21 = sum(abs(e21r));


% IAE for reference scenario 2
% ---------------------------------------------------------------

r2 = r;
r2(2,rtime:end) = rstep;


% initial state
xkclr2 = zeros(size(Acl,1),1);

for j = 1:Nsim
```

```
    Rtempr2 = r2(:,j:(j+N)-1);
    Rkr2 = Rtempr2(:);

    xkclr2next = Acl*xkclr2 + Brcl*Rkr2;
    yr2(:,j) = Ccl*xkclr2;

    xkclr2 = xkclr2next;
end


% tank 1, step in reference 2
e12r = yr2(1,:) - r2(1,1:Nsim);

% tank 2, step in reference 2
e22r = yr2(2,:) - r2(2,1:Nsim);

% IAE
Jr12 = sum(abs(e12r));
Jr22 = sum(abs(e22r));

Jr = Jr11 + Jr21 + Jr12 + Jr22;


% Objective function
J = Ju + Jy + Jr;
```

```
function [c,ceq] = ConFun2(x,A,B,C,N,MSmax,Ts)
% ConFun2
% Evaluates constraint for tuning optimization problem
% for tuning approach 2


% number of measurements
ny =size(C,1);

% number of control inputs
nu =size(B,2);

% disturbance model parameters
alpha = x(1:ny);

% weight matrix Su
S = diag(x(ny+1:ny+nu));

% weight matrix Qy
Q = diag(x(ny+nu+1:end));


% ---------------------------------------------------------------
% Controller model - deterministic-stochastic model
% ---------------------------------------------------------------

% size of identified model state
```

```
nx = size(A,1);

% disturbance model
As = eye(ny,ny);
Ks = diag(ones(ny,1)-alpha);
Cs = eye(ny,ny);

% combined model
Abar = [A zeros(nx,ny); zeros(ny,nx) As];
Bbar = [B; zeros(ny,nu)];
Kbar = [zeros(nx,ny); Ks];
Cbar = [C Cs];

% ---------------------------------------------------------------
% Controller gains - deterministic-stochastic model
% ---------------------------------------------------------------

% design matrices
[phix phie gammau H varphi I0 Qcal Scal] = ...
                           MPCDesign(N,Abar,Bbar,Cbar,Kbar,Q,S);

% gain matrices for explicit MPC solution
[Lx Le LR Lu] = MPCDesignUnconstrained ...
                       (H,phix,phie,gammau,varphi,I0,Qcal,Scal);

% ---------------------------------------------------------------
% Controller in state-space form
% ---------------------------------------------------------------


Ac = [((Abar - Kbar*Cbar) + Bbar*(Lx - Le*Cbar)) Bbar*Lu;
                                        Lx - Le*Cbar Lu];

Bcy = [Kbar + Bbar*Le; Le];
Bcr = [Bbar*LR; LR];

Cc = [Lx-Le*Cbar Lu];

Dcy = Le;
Dcr = LR;




% ---------------------------------------------------------------
% Closed-loop state-space: Controller + Linear plant
% ---------------------------------------------------------------

% ---------------------------------------------------------------
% Closed-loop systen
% ---------------------------------------------------------------

Acl = [A + B*Dcy*C B*Cc; Bcy*C Ac];


```

```
sizeAcl = size(Acl,2);
sizeC = size(C,2);

sizediff = sizeAcl - sizeC;


%Bdcl = [Ep; zeros(8,2)];
%Bwcl = [Gp; zeros(8,2)];

Bvcl = [B*Dcy;Bcy];
%Brcl = [B*Dcr;Bcr];

Ccl = [C zeros(2,sizediff)];
%Cucl = [Dcy*C Cc];

%Dvcl = Dcy;
%Drcl = Dcr;

%%%%%%%%%% TEST %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% closed-loop input disturbance matrix
%Bubarcl = [B; zeros(sizediff,2)];


% closed-loop output disturbance matrix
%Bybarcl = [B*Dcy;Bcy];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% ---------------------------------------------------------------
% Sensitivity
% ---------------------------------------------------------------

w = 0:0.0001:(pi/Ts);
sv = sigma(ss(Acl,Bvcl,Ccl,eye(ny),Ts),w);

MS = max(max(sv))

% constraints
c   = MS - MSmax;
ceq = [];
```

A.4 System Identification

```
close all
clear all
```

```
clc

addpath('Realization')

% System identification Script for modified 4-tank system

% Setup non-linear model

% --------------------------------------------------------------
% Parameters
% --------------------------------------------------------------

a1 = 1.2272;     %[cm2] Area of outlet pipe 1
a2 = 1.2272;     %[cm2] Area of outlet pipe 2
a3 = 1.2272;     %[cm2] Area of outlet pipe 3
a4 = 1.2272;     %[cm2] Area of outlet pipe 4

A1 = 380.1327;   %[cm2] Cross sectional area of tank 1
A2 = 380.1327;   %[cm2] Cross sectional area of tank 2
A3 = 380.1327;   %[cm2] Cross sectional area of tank 3
A4 = 380.1327;   %[cm2] Cross sectional area of tank 4

gamma1 = 0.45;   % Flow distribution constant. Valve 1
gamma2 = 0.40;   % Flow distribution constant. Valve 2

g = 981;         %[cm/s2] The acceleration of gravity
rho = 1.00;      %[g/cm3] Density of water

p = [a1; a2; a3; a4; A1; A2; A3; A4; gamma1; gamma2; g; rho];


% --------------------------------------------------------------
% Simulation Setup
% --------------------------------------------------------------

t0 = 0.0;            % [s] Initial time
tf = 50*60;          % [s] Final time

Ts = 4;              % [s] Sample Time

t = (t0:Ts:tf)';     % [s] Sample instants
N = length(t);

F1 = 300;            % [cm3/s] Flow rate from pump 1
F2 = 300;            % [cm3/s] Flow rate from pump 2


% steady state
us = [300;300];   % [cm3/s] Flow rates
ds = [70;70];     % [cm3/s] Flow rates

[Axx Bxx Exx Cxx xs ys] = ...
       getlinearizedModified4TankSystem(us,ds,gamma1,gamma2);

% initial state
```

```
x0 = xs;

stepindexu1 = 251;

u1 = F1*ones(1,N);
u1(stepindexu1:end) = 1.15*F1;
%u1(stepindexu1:end) = 0.85*F1;

u2 = F2*ones(1,N);

u = [u1;u2];

% Disturbance
d = 70*ones(2,N);

% Process Noise
Q = (20^2)*eye(2);
Lq = chol(Q,'lower');
w = Lq*randn(2,N);

% Measurement Noise
R = 0.5*eye(2);
Lr = chol(R,'lower');
v = Lr*randn(2,N);

% simulate data
y = nonlinearmodified4tanksystemSIM(x0,u,d,v,w,t,N,p);

% Plot simulated process data (non-linear process)
figure
subplot(221)
plot(t,y(1,:),'.r')
ylabel('$y_1 \enskip [cm]$','interpreter','latex','fonts',14)
grid on

subplot(223)
stairs(t,u1,'-b','LineWidth',2)
ylim([280 360])
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$u_1 \enskip [cm^3/s]$','interpreter','latex','fonts',14)
grid on

subplot(222)
plot(t,y(2,:),'.r')
ylabel('$y_2 \enskip [cm]$','interpreter','latex','fonts',14)
grid on

subplot(224)
stairs(t,u2,'-b','LineWidth',2)
ylim([280 360])
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$u_2 \enskip [cm^3/s]$','interpreter','latex','fonts',14)
grid on


```

```
%%%%% step data for step in u1

tdata =  t(stepindexu1:end);
u1data = u1(stepindexu1:end)';
y1data = y(1,stepindexu1:end)';
y2data = y(2,stepindexu1:end)';


%%%%%%% treated data


tsamples = length(tdata)*Ts - Ts;
tnew = 0:Ts:tsamples;

Δu1 = u1data(1) - u1(250);

% data y1
y1ss = y(1,stepindexu1-5);
y1datanew = (y1data - y1ss*ones(size(y1data)))./Δu1;

% data y2
y2ss = y(2,stepindexu1-5);
y2datanew = (y2data - y2ss*ones(size(y2data)))./Δu1;


figure
plot(tnew,y1datanew,'.r','LineWidth',2)
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$\frac{y_1 - y_{s1}}{\Delta u_1}$',...
    'interpreter','latex','fonts',14)
grid on

figure
plot(tnew,y2datanew,'.r','LineWidth',2)
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$\frac{y_2 - y_{s2}}{\Delta u_1}$',...
    'interpreter','latex','fonts',14)
grid on


%%%%%% PARAMATER ESTIMATION


options = optimset('TolFun',1e-10,'MaxFunEvals',1000);

% first-order estimate
[y1u1,resnorm1] = lsqcurvefit(@myfun,[1;1],tnew ...
    ,y1datanew',[],[],options);

[x2,resnorm2] = lsqcurvefit(@myfun,[1;1],tnew ...
    ,y2datanew',[],[],options);

% second-order estimate
[model_y1u1,resnormy1u1] = lsqcurvefit ...
    (@secondordermodel,[1;30;40],tnew,y1datanew',[],[],options);
```

```

[model_y2u1,resnormy2u1] = lsqcurvefit ...
    (@secondordermodel,[1;30;40],tnew,y2datanew',[],[],options);

y1fit = myfun(y1u1,tnew);
y2fit = myfun(x2,tnew);

y1u1fitSO = secondordermodel(model_y1u1,tnew);
y2u1fitSO = secondordermodel(model_y2u1,tnew);



% plot estimates

figure
plot(tnew,y1datanew,'.r',tnew,y1fit,'—',tnew ...
    ,y1u1fitSO,'—','LineWidth',2)

xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
title('$G_{11}(s)$','interpreter','latex','fonts',14)
grid on
h1 = legend('data','fopdt','sopdt');
set(h1,'interpreter','latex','fontsize',12);


figure
plot(tnew,y2datanew,'.r',tnew,y2fit,'—',tnew ...
    ,y2u1fitSO,'—','LineWidth',2)

xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
title('$\hat G_{21}(s)$','interpreter','latex','fonts',14)
grid on
h2 = legend('data','fopdt','sopdt');
set(h2,'interpreter','latex','fontsize',12);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       STEP IN U2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

stepindexu2 = 251;


u12 = F1*ones(1,N);

u22 = F2*ones(1,N);
u22(stepindexu2:end) = 1.15*F2;


u2 = [u12;u22];
y2 = nonlinearmodified4tanksystemSIM(x0,u2,d,v,w,t,N,p);


% Plot simulated process data (non—linear process)
figure
```

```

subplot(221)
plot(t,y2(1,:),'.r','LineWidth',2)
ylabel('$y_1 \enskip [cm]$','interpreter','latex','fonts',14)
grid on

subplot(223)
stairs(t,u12,'-b','LineWidth',2)
ylim([280 360])
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$u_1 \enskip [cm^3/s]$','interpreter','latex','fonts',14)
grid on

subplot(222)
plot(t,y2(2,:),'.r','LineWidth',2)
ylabel('$y_2 \enskip [cm]$','interpreter','latex','fonts',14)
grid on

subplot(224)
stairs(t,u22,'-b','LineWidth',2)
ylim([280 360])
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$u_2 \enskip [cm^3/s]$','interpreter','latex','fonts',14)
grid on

%%%%% step data

tdata =  t(stepindexu2:end);
u2data = u22(stepindexu2:end)';
y12data = y2(1,stepindexu2:end)';
y22data = y2(2,stepindexu2:end)';

%%%%%%% treated data
tsamples2 = length(tdata)*Ts - Ts;
tnew2 = 0:Ts:tsamples2;

∆u2 = u2data(1) - u2(stepindexu2-1);

y1ss2 = y2(1,stepindexu2-5);
y1datanew2 = (y12data - y1ss2*ones(size(y12data)))./∆u2;

y2ss2 = y2(2,stepindexu2-5);
y2datanew2 = (y22data - y2ss2*ones(size(y22data)))./∆u2;


% PARAMATER ESTIMATION


options = optimset('TolFun',1e-10,'MaxFunEvals',1000);

% first-order estimates
[x12,resnorm12] = lsqcurvefit(@myfun,[1;1] ...
    ,tnew2,y1datanew2',[],[],options);

[x22,resnorm22] = lsqcurvefit(@myfun,[1;1]...
    ,tnew2,y2datanew2',[],[],options);

% second-order estimates
[model_y1u2,resnormy1u2] = lsqcurvefit(@secondordermodel ...
                    ,[1;30;40],tnew2,y1datanew2',[],[],options);
[model_y2u2,resnormy2u2] = lsqcurvefit(@secondordermodel ...
                    ,[1;30;40],tnew2,y2datanew2',[],[],options);

y1fit2 = myfun(x12,tnew2);
y2fit2 = myfun(x22,tnew2);

y1u2fitSO2 = secondordermodel(model_y1u2,tnew2);
y2u2fitSO2 = secondordermodel(model_y2u2,tnew2);


figure
plot(tnew2,y1datanew2,'.r',tnew2,y1fit2,'--',tnew2 ...
    ,y1u2fitSO2,'--','LineWidth',2)

xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
title('$G_{12}(s)$','interpreter','latex','fonts',14)
grid on
h1 = legend('data','fopdt','sopdt');
set(h1,'interpreter','latex','fontsize',12);


figure
plot(tnew2,y2datanew2,'.r',tnew2,y2fit2,'--',tnew2 ...
    ,y2u2fitSO2,'--','LineWidth',2)

xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
title('$G_{22}(s)$','interpreter','latex','fonts',14)
grid on
h2 = legend('data','fopdt','sopdt');
set(h2,'interpreter','latex','fontsize',12);

%--------------- Continuos Time transfer functions

num11 = model_y1u1(1);                                  % B11
den11 = conv([model_y1u1(2) 1],[model_y1u1(3) 1]);      % A11
tau11  = 0;

num12 = model_y1u2(1);                                  % B12
den12 = conv([model_y1u2(2) 1],[model_y1u2(3) 1]);      % A12
tau12 = 0;

num21 = model_y2u1(1);                                  % B21
den21 = conv([model_y2u1(2) 1],[model_y2u1(3) 1]);      % A21
tau21  = 0;

num22 = model_y2u2(1);                                  % B22
den22 = conv([model_y2u2(2) 1],[model_y2u2(3) 1]);      % A22
tau22 = 0;

num=cell(2,2); den=cell(2,2); tau=zeros(2,2);
```

```

num{1,1}=num11; num{1,2}=num12;
num{2,1}=num21; num{2,2}=num22;

den{1,1}=den11; den{1,2}=den12;
den{2,1}=den21; den{2,2}=den22;


Nmax=100; tol=1e-8;

[Ad1,Bd1,Cd1,Dd1,sH]=mimoctf2dss(num,den,tau,Ts,Nmax,tol);

figure

subplot(221)
hold on
plot(tnew,y1datanew,'.r') % ,'MarkerSize',10)
plot(tnew,y1fit,'--',tnew,y1u1fitSO,'--','LineWidth',2)
hold off
grid on
title('$\hat G_{11}(s)$','interpreter','latex','fonts',14)
h1 = legend('data','fopdt','sopdt','fontsize',10);
%set(h1,'interpreter','latex','fontsize',10);
box on

subplot(223)
plot(tnew,y2datanew,'.r',tnew,y2fit,'--',tnew ...
    ,y2u1fitSO,'--','LineWidth',2)

grid on
title('$\hat G_{12}(s)$','interpreter','latex','fonts',14)
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)


subplot(222)
plot(tnew2,y1datanew2,'.r',tnew2,y1fit2,'--',tnew2 ...
    ,y1u2fitSO2,'--','LineWidth',2)

grid on
title('$\hat G_{21}(s)$','interpreter','latex','fonts',14)


subplot(224)
plot(tnew2,y2datanew2,'.r',tnew2,y2fit2,'--',tnew2 ...
    ,y2u2fitSO2,'--','LineWidth',2)

grid on
title('$\hat G_{22}(s)$','interpreter','latex','fonts',14)
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)



figure

subplot(221)
```

```
plot(tnew,y1datanew,'.r')
ylabel('$\frac{y_1 - y_{1s}}{\Delta u_1}$',...
    'interpreter','latex','fonts',14)

grid on

subplot(223)
plot(tnew,y2datanew,'.r')
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$\frac{y_1 - y_{1s}}{\Delta u_2}$' ...
    ,'interpreter','latex','fonts',14)

grid on

subplot(222)
plot(tnew2,y1datanew2,'.r')
ylabel('$\frac{y_2 - y_{2s}}{\Delta u_1}$', ...
    'interpreter','latex','fonts',14)

grid on

subplot(224)
plot(tnew2,y2datanew2,'.r')
xlabel('$time \enskip [s]$','interpreter','latex','fonts',14)
ylabel('$\frac{y_2 - y_{2s}}{\Delta u_2}$', ...
    'interpreter','latex','fonts',14)

grid on
```

```
function y = nonlinearmodified4tanksystemSIM(x0,u,d,v,w,t,N,p)
% nonlinearmodified4tanksystemSIM
% Simulates the non-linear 4-tank system for generation of
% input-output for use in system identification

nx = 4; nu = 2; ny = 2; nz = 2;

x = zeros(nx,N);
y = zeros(ny,N);
z = zeros(nz,N);

X = zeros(0,nx);
T = zeros(0,1);

x(:,1) = x0;

for k = 1:N-1
    y(:,k) = ModifiedFourTankSystemSensor(x(:,k),p)+v(:,k);
    if any(d(:,k)+w(:,k)) < 0
        'negative'
    end
    [Tk,Xk] = ode15s(@ModifiedFourTankSystem ...
        ,[t(k) t(k+1)],x(:,k),[],u(:,k),(d(:,k)+w(:,k)),p);

    x(:,k+1) = Xk(end,:)';
    T = [T; Tk];
    X = [X; Xk];
end

k = N;
    y(:,k) = ModifiedFourTankSystemSensor(x(:,k),p)+v(:,k);
```

Bibliography

[DS81] J. C. Doyle and G. Stein. Multivariable feedback design: Concepts for a classical/modern synthesis. *IEEE Transactions on Automatic Control*, AC-26(1):4–16, 1981.

[HPJJ12] Jakob Kjøbsted Huusom, Niels Kjølstad Poulsen, Sten Bay Jørgensen, and John Bagterp Jørgensen. Tuning siso offset-free model predictive control based on arx models. *Journal of Process Control*, 22(10):1997 – 2007, 2012.

[JHR11] John Bagterp Jørgensen, Jakob Kjøbsted Huusom, and James B Rawlings. Finite horizon mpc for systems in innovation form. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC),*, pages 1896–1903. IEEE, 2011.

[JHWH99] Karl Henrik Johansson, Alexander Horch, Olle Wijk, and Anders Hansson. Teaching multivariable control using the quadruple-tank process. In *Proceedings of the 38th Conference on Decision and Control*, pages 807–812, Phoenix, Arizona, December 1999.

[JN98] Karl Henrik Johansson and José Luís Rocha Nunes. A multivariable laboratory process with an adjustable zero. In *Proceedings of the American Control Conference*, pages 2045–2049, Philadelphia, Pennsylvania, USA, June 1998.

[Joh97] K. H. Johansson. *Relay feedback and multivariable control.* PhD thesis, Department of Automatic Control, Lund University, 1997.

[Joh00] K. H. Johansson. The quadruple-tank process: a multivariable laboratory process with an adjustable zero. *IEEE Transactions on Control Systems Technology*, 8(3):456–465, 2000.

[Jør04] John Bagterp Jørgensen. *Linear Model Predictive Control Toolbox - User's Guide.* 2-control ApS (www.2-control.com), 2004.

[Jør12] John Bagterp Jørgensen. *Modified Four Tank System: Benchmark Problem for Control Design. Lecture notes for 02619 Model Predictive Control.* DTU, 2012.

[Kal60] R. Kalman. A new approach to linear filtering and prediction problems. In *Transactions of the ASME - Journal of basic Engineering*, number 82, pages 35–45, 1960.

[Lju99] Lennart Ljung. *System Identification. Theory for the User.* Prentice-Hall, second edition edition, 1999.

[Mac02] J. Maciejowski. *Predictive Control with Constraints.* Prentice-Hall, 2002.

[MH97] Henrik Madsen and Jan Holst. *Lecture notes for Advanced Time Series Analysis.* DTU, 1997.

[OJ13] Jakob Kjøbsted Olesen, Daniel Haugaard Huusom and John Bagterp Jørgensen. A tuning procedure for arx-based mpc. In *Proceedings of the European Control Conference (ECC)*, 2013.

[Øks00] Bernt Øksendal. *Stochastic Differential Equations: An Introduction with Applications.* Springer-Verlag Heidelberg New York, 5 edition, 2000.

[Pou07] N. K. Poulsen. *Stokastisk Adaptiv Regulering: stokastiske systemer, identifikation og regulering.* IMM - DTU, 2007.

[PR03] Gabriele Pannocchia and James B Rawlings. Disturbance models for offset-free model-predictive control. *AIChE Journal*, 49(2):426–437, 2003.

[QB03] S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11:733–764, 2003.

[RRQ09] Murali R Rajamani, James B Rawlings, and S Joe Qin. Achieving state estimation equivalence for misassigned disturbances in offset-free model predictive control. *AIChE Journal*, 55(2):396–407, 2009.

[SP01] Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control - Analysis and design.* John Wiley and Sons, 2 edition, 2001.

[Was10] Harald Siegfried Waschl. Self tuning strategies for model predictive control of integral gas engines. Master's thesis, JOHANNES KEPLER UNIVERSITAT, Austria, 2010.

[Zhu01] Yucai Zhu. *Multivariable System Identification for Process Control.* Pergamon, 2001.

[Zhu06] Yucai Zhu. System identification for process control: Recent experience and outlook. *IFAC Symposium on System Identification*, 14, 2006.