DISCRETE-TIME LINEAR MODEL PREDICTIVE CONTROL

Addis Ababa University
College of Natural and Computational Sciences
Department of Mathematics

"In partial fulfillment of the requirements for the degree of Master of Science in Mathematics"

By: Dereje Tigabu

Stream: Optimization

Advisor: Dr. Semu Mitiku

Addis Ababa, Ethiopia

July 4, 2016

ADDIS ABABA UNIVERSITY
DEPARTMENT OF MATHEMATICS

The undersigned hereby certify that they have read and recommend to the department of mathematics for acceptance of this project entitled "**Discrete-Time Linear Model Predictive Control"** by Dereje Tigabu in partial fulfillment of the requirements for the degree of Master of Science in Mathematics.

Advisor: Dr. Semu Mitiku

Signature:_____

Date_____

Examiner 1: Dr. Berhanu G.

Signature:_____

Date_____

Examiner 2: Dr. Yirgalem T.

Signature:_____

Date_____

ADDIS ABABA UNIVERSITY

Author: Dereje Tigabu

Title: Discrete-Time Linear Model Predictive Control

Department: Mathematics

Degree: M.Sc.

Convocation: June

Year: 2016

Dereje Tigabu:

Signature:_____

Date_____

Acknowledgment

First of all, I am grateful to the almighty God for establishing me to complete this project report.

I would like to express my deepest gratitude and appreciation to my advisor Dr. Semu Mitiku for his unreserved help during my project report writing. I am also grateful to my family for all their encouragement and their successive help morally and financially in my study at Addis Ababa University.

Lastly, I would like to thank all my friends who comments and helping me in all activities.

Table of Contents

Abstract

Model predictive control is a form of control in which the current control action is obtained by solving on-line, at each sampling instant, a finite horizon optimal control problem, using the current state of the plant as the initial state; the optimization yields an optimal control sequence and the first control in this sequence is applied to the plant. An important advantage of this type of control is its ability to cope with hard constraints on controls and states. In this project, we discuss model predictive control(MPC) schemes for discrete-time linear time-invariant state space system with and without constraints on inputs, states and outputs. The constraints on inputs, states and outputs are in the form of bounds, that can be formulated as linear inequalities. In particular, the project focus on performance criteria based on quadratic form. And also, discuss some solution techniques of discrete-time model predictive control using quadratic programming solution techniques.

Chapter 1

Introduction

Model Predictive Control (MPC) is an advanced control tool that originates in the late seventies. Due to its simplicity, it quickly became the preferred control tool in many industrial applications[5]. Some of the fields where MPC is already considered to be a mature technique involve linear and rather slow systems like the ones usually encountered in the chemical process industry. However, the application of MPC to more complex systems, involving nonlinear, hybrid, or very fast processes is still in its infancy.

MPC does not designate a specific control strategy but rather an ample range of control methods which use a model of a process to obtain control actions by minimizing an objective function, possibly subject to given constraints. The various algorithms in the MPC family can be distinguished mainly by: (i) the model used to represent the process; (ii) the objective function to be minimized; (iii) the type of constraints. The most popular scheme applied in practice involves a linear time-invariant process model, linear constraints and quadratic objective function[12]. In general, it is referred to as linear MPC (LMPC). The computational burden associated with the application of LMPC is mainly due to forming and solving a quadratic program (QP) at each sampling interval.

Model-based predictive control is a relatively new method in control engineering. The basic idea of the method is to consider and optimize the relevant variables, not only at the current time point but also during their course in the future. This goal is achieved first by a heuristic choice of the manipulated variable sequence and simulation of the future course of the process variables. If the future course of the controlled and the constrained variables is not satisfactory, then new manipulated variable sequences are tried out until the control behavior becomes satisfactory. The expression predictive control arises from a forecast of the variables. A process model is necessary to simulate the process. In acquiring knowledge of the predicted process variables, constraints on the manipulated, controlled, and other variables can be simply taken into account. There is a fundamental difference between predictive control and conventional on- off or PID control[7]:

- A conventional controller observes only the current (and remembers the past) process variables.

- A predictive controller observes the current and also the future process variables (and remembers the past variables).

Predictive thinking is more natural in everyday life, for example, during car driving one observes the future shape of the road, brakes if one is approaching a curve, pushes the gas pedal if one is nearing a hill, and decreases the speed if another, slower car appears in the field of vision. Figure 1.1 compares the two driver philosophies.

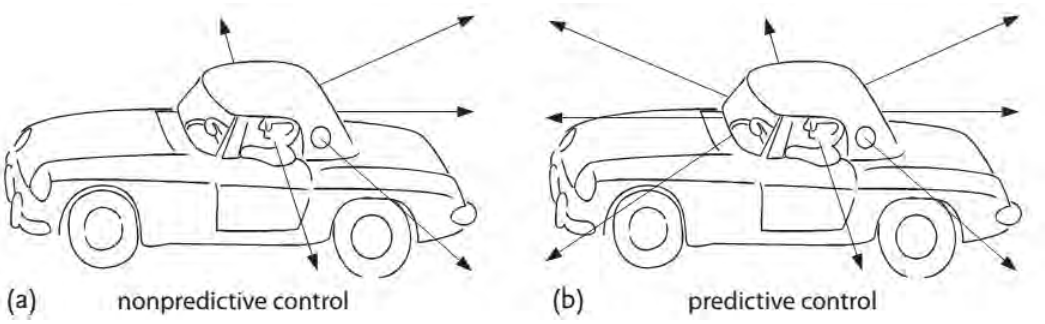


Figure 1.1: Car driving strategies.

- Conventional control in driving would mean a driving style where the car driver looks only through the side windows. In a curve the driver can correct the trace following the position only after having observed an error.
- Any real driver on the route is a predictive controller, because he/she drives depending on the curvature and what he/she sees in advance in front of the car.

The longer the preview distance, the better the position control, but the calculations are more time consuming. The horizon length has to be increased with the car speed. Beyond a certain preview distance the control would not be better. A minimum sampling time is necessary, otherwise the car cannot follow the drivers commands in due time.

In essence, the core technique in the design of discrete-time MPC is based on optimizing the future control trajectory subject to plant operational constraints. In the traditional predictive control, by assuming a finite control horizon N_c and prediction horizon N_p, the difference of the control signal $\Delta u(k)$ for $k = 0, 1, 2, ..., N_c - 1$ is captured by the control vector ΔU , while the rest of the $\Delta u(k)$ for $k = N_c, N_c + 1, \ldots, N_p$ is assumed to be zero.
It is fair to say that the majority of industrial control systems require integral action. In this project report, the design models are embedded with integrators to achieve this objective, which is similar to other classical predictive control systems. Because of the embedded integrators, the prediction horizon N_p as a design parameter which plays an important role in a predictive control system. Therefore, in this project work we consider discrete time linear

model predictive control with and without constraints on inputs, states and outputs.

This project report is organized as follows. Chapter 2 describes the main elements that appear in any MPC formulation and reviews the best known methods. In Chapter 3 we will use simple examples to show the principle of receding horizon control, which underpins predictive control. The solutions are limited to simple analytical forms and also, implementation strategies using observer and observability are discussed with examples.

In Chapter 4, the basic ideas about constrained control are introduced within the framework of receding horizon control. The key is to formulate the constrained control problem as a real-time optimization problem subject to inequality constraints. The solution to this problem relies on application of a quadratic programming procedure. we will also discuss optimization with equality and inequality constraints. The discussion is followed by the introduction of Hildreth's quadratic programming procedure, which offers simplicity and reliability in real-time implementation. Finally, conclusive remarks are presented in chapter 5.

Chapter 2

Preliminaries

2.1 Control

Controlling a system or an object means to influence the behaviour of the system or an object so as to achieve a desired goal[19].
Control problems are usually stated mathematically by the formula

$$g(x) = f(u),$$

where $x \in X$, is the state of the system, the unknown of the system that we want to control, and u is the control (or input) it take from some admissible set, called admissible controls set U, where X and U are vector spaces

$$g : X \to X$$
$$f : U \to X$$

where, g is the formula which must be satisfied by the state variable x and f is a formula which must be satisfied by the input variable u.
Controlled Variable: is the quantity or condition that is measured and controlled.
Control Signal or Manipulated Variable: is the quantity or condition that is varied by the controller so as to affect the value of the controlled variable. Normally, the controlled variable is the output of the system. Control means measuring the value of the controlled variable of the system and applying the control signal to the system to correct or limit deviation of the measured value from a desired value[17].
Control system: The system that includes at least the plant and its controller. It may include other subsystems and components (e.g., sensors, signal conditioning and modification)[4].
The essential components of a control system (figure 2.1) are:

1. The plant, the system which is to be controlled;
2. One or more sensors, which give information about the plant; and
3. The controller, the "heart" of the control system, which compares the measured values to their desired values and adjusts the input variables to the plant.

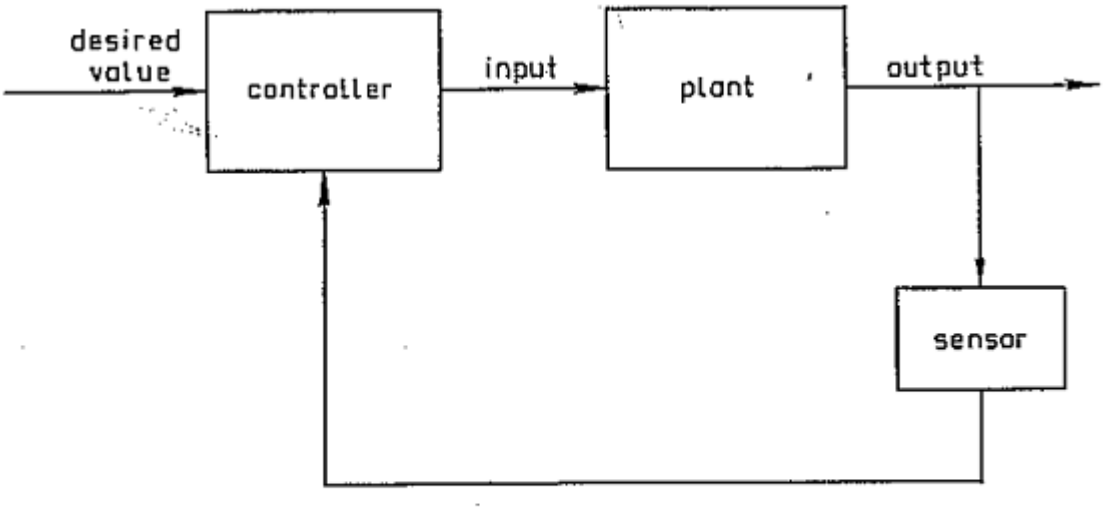


Figure 2.1: Schematic representation of a control system.

2.2 Open Loop and Closed Loop Control Systems

There are basically two types of control system: the open loop system and the closed loop system. They can both be represented by block diagrams. A block diagram uses blocks to represent processes, while arrows are used to connect different input, process and output parts.

2.2.1 Open Loop Control System

Fig.2.2 shows a simple open loop control system. Its operation is very simple, when an input signal directs the control element to respond, an output will be produced.

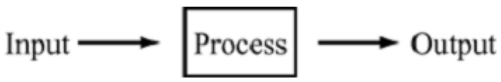


Figure 2.2: Block diagram of an open loop control system

The input and output of an open loop system are unrelated. An open-loop control system, does not measure (or feedback) the output to determine the control action; its accuracy depends on its calibration, and thus it is only used when there are no perturbations and the plant is perfectly known. A perturbation (or disturbance) is an unknown signal that tends

to adversely affect the output value of the plant. This undesired signal may be internal (endogenous) or external (exogenous) to the system. The drawback of an open loop control system is that it is incapable of making automatic adjustments. Even when the magnitude of the output is too big or too small, the system will not make the appropriate adjustments. For this reason, an open loop control system is not suitable for use as a complex control system. Sometimes it may even require monitoring and response from the user. For example, when a washing machine finishes cleaning the clothes, the user will need to check whether the clothes are clean or not; if they are not, they have to be put back into the machine and washed again.

2.2.2 Closed Loop Control System

Sometimes, we may use the output of the control system to adjust the input signal. This is called feedback. Feedback is a special feature of a closed loop control system. A closed loop control system compares the output with the expected result or command status, then it takes appropriate control actions to adjust the input signal. Therefore, a closed loop system is always equipped with a sensor, which is used to monitor the output and compare it with the expected result. Fig. 2.3 shows a simple closed loop system. The output signal is feedback to the input to produce a new output. A well-designed feedback system can often increase the accuracy of the output.
Feedback can be divided into positive feedback and negative feedback.

- **Positive feedback** causes the new output to deviate from the present command status. For example, an amplifier is put next to a microphone, so the input volume will keep increasing, resulting in a very high output volume.
- **Negative feedback** directs the new output towards the present command status, so as to allow more sophisticated control. For example, a driver has to steer continuously to keep his car on the right track.

One advantage of using the closed loop control system is that it is able to adjust its output automatically by feeding the output signal back to the input. When the load changes, the error signals generated by the system will adjust the output. However, closed loop control systems are generally more complicated and thus more expensive to make.

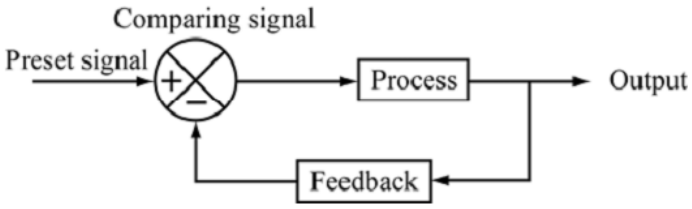


Figure 2.3: Block diagram of a closed loop control system

2.3 Receding Horizon Control(RHC)

Consider again the process of driving. It is usually to consider the road several hundred yards ahead and to anticipate any potential dangers. As we move along the road, the part of the road within our vision moves with us so that we can always see the next few hundred yards. This in essence is the receding horizon, that is the limit of our vision (say $200yds$ away) is always moving away at the same speed we are it is receding. This means that we are continually picking up new information from the far horizon and this information is used to update our control actions (decisions). Predictive control works just like this: it considers the predicted behaviour over some horizon into the future and therefore at each successive sampling instant, it predicts one further sample into the future. As new information comes available the input trajectory is automatically modified to take account of it. The resulting controller is referred to as **Receding Horizon Controller (RHC)**. The term receding horizon is introduced, since the horizon recedes as time proceeds[9].
The purpose of taking new measurement at each time step is to compensate for unmeasured disturbance and model in accuracy, both of which cause the system output to be different from the one predicted by the model.

The feedback control law is then obtained in a receding horizon manner by applying to the system only the first element of the computed sequence of optimal controls, and repeating the whole procedure at the next discrete-time step.

A receding horizon controller, where the finite time optimal control law is computed by solving an optimization problem on-line, is usually referred to as **Model Predictive Control (MPC).**

2.4 Model Predictive Control(MPC)

Model Predictive Control (MPC) is a control strategy which can be considered as a special case of the optimal control theory developed in the 1960 and later.

Suppose we are given a controlled process whose state $x(k)$ is measured at discrete time instants k, $k = 0, 1, 2, ...$ Controlled means that at each time instant we can select a control input $u(k)$ which influences the future behavior of the state of the system. In tracking control, the task is to determine the control inputs $u(k)$ such that $x(k)$ follows a given reference $r(k)$ as good as possible. This means that if the current state is far away from the reference then we want to control the system towards the reference and if the current state is already close to the reference then we want to keep it there[6].

The difficulty of the general optimal control problem arises because it is a function optimization problem. Instead of ordinary calculus, it needs the Calculus of Variations, to find the optimal input function among all possible ones[20].

The main idea of Predictive Control is to avoid this difficulty by restricting the set of possible inputs to such an extent that the optimization is performed only over a finite set of parameters or decision variables, rather than over a set of functions. Most commonly this is done by cutting up time into discrete intervals, optimising over a finite horizon, and hence over a finite number of intervals, and assuming that the input signal is held constant during each interval[11].

The key feature that distinguishes MPC from most other control strategies is the receding horizon principle. An MPC controller solves, at each sampling instant, a finite horizon optimal control problem. Only the first value of the resulting optimal control variable solution is then applied to the plant, and the rest of the solution is discarded. The same procedure is then repeated at each sampling instant, and the prediction horizon is shifted forward one step.

The plant models assumed in this work are linear or are linearized. If a linear model is used for prediction, the resulting MPC is called linear MPC. A version of MPC using nonlinear plant models is referred to as nonlinear MPC or NMPC. The idea of linear or nonlinear model predictive control is now to utilize a model of the process in order to predict and optimize the future system behavior.
The general design objective of model predictive control is to compute a trajectory of a future manipulated variable u to optimize the future behaviour of the plant output y. The optimization is performed within a limited time window by giving plant information at the start of the time window.
To understand the basic ideas that have been used in the design of predictive control, there are terms that we will use most frequently in the principle of model predictive control. They are introduced as below.

1. Moving horizon window: the time-dependent window from an arbitrary time k_i to k_i+N_p, where N_p is prediction horizon. The length of the window N_p remains constant. k_i defines the beginning of the optimization window.

2. Prediction horizon: dictates how far we wish the future to be predicted for. This parameter equals the length of the moving horizon window, N_p

3. Receding horizon control: although the optimal trajectory of future control signal is completely described within the moving horizon window, the actual control input to the plant only takes the first sample of the control signal, while neglecting the rest of the trajectory.

4. In the planning process, we need the information at time k_i in order to predict the future. This information is denoted as $x(k_i)$ which is a vector containing many relevant factors, and is either directly measured or estimated.

5. A given model that will describe the dynamics of the system is paramount in predictive control. A good dynamic model will give a consistent and accurate prediction of the future.

6. In order to make the best decision, a criterion is needed to reflect the objective. The objective is related to an error function based on the difference between the desired and the actual responses. This objective function is often called the cost function J, and the optimal control action is found by minimizing this cost function within the optimization window

MPC consists of an optimization problem at each time instant, k. The main point of this optimization problem is to compute a new control input vector, $\Delta u(k)$, to be feed to the system, and at the same time take process constraints into consideration (e.g. constraints on process variables). An MPC algorithm consists of

- **Cost function**
  A control objective, J, (or cost function) which is a scalar criterion measuring e.g., the difference between future outputs, and some specified (future) reference, and at the same time recognizing that the control, $\Delta u(k)$, is costly. The objective is a measure of how good is the process behaviour over the prediction horizon, N_p. This objective is minimized with respect to the future control vectors, $\Delta u(k+i|k)$, $i = 0, 1, ..., N_c$ and only the first control vector, $\Delta u(k)$, is actually used for control. This optimization process is solved again at the next time instant, i.e, at $k := k + 1$. This is sometimes called a receding horizon control problem.

- **Constraints**
  One of the main motivation behind MPC is that constraints on process variables can be treated simply. Common constraints as input amplitude constraints and input rate of change constraints can be treated far more efficient than in conventional control systems (PID-control). This usually leads to a simple inequality constraint, which is added to the optimization problem.

- **Prediction model(PM)**
  Any model that could be used to predict the future behaviour can be the system model in MPC, and it is usually called predictive model. MPC itself has no special request on the choice of model, the only need is that the model could predict the future behaviour of the system, no matter how we get the system model and how we obtain the future output by the model. But many researchers still classify MPC into different types by their models, since different model usually lead to quite different optimization method in solution of control law[23].
  The main drawback with MPC is that a model for the process, i.e., a model which describes the input to output behaviour of the process, is needed. The model is primarily used to predict the outputs and the states over the prediction horizon. The process model is usually used to construct a PM. The purpose of the PM is to describe the relationship between the future outputs and the future control inputs to be computed. The PM is a part of the optimization problem and is needed for this reason. In this study we will use state space model of the form

$$\begin{aligned} x_m(k+1) =& A_m x_m(k) + B_m u(k) \\ y(k) =& C_m x_m(k) \end{aligned}$$

where u is the manipulated variable or input variable; y is the process output; and x_m is the state variable vector with appropriate dimension. And A, B and C are matrices with appropriate dimensions. This system is a linear system that relate the controlled state x_m to the outputs signal y.

2.5 Implementation of MPC

Model predictive control (MPC) is a form of control (control strategy) in which the current control action is obtained by solving on-line, at each sampling instant, a finite horizon optimal control problem, using the current state of the plant as the initial state; the optimization yields an optimal control sequence and the first control in this sequence is applied to the plant[15]. This is its main difference from conventional control which uses a pre-computed control law. MPC is not a new method of control design. It essentially solves standard optimal control problems, in MPC, the optimal control problem is required to have a finite horizon. It differs from other controllers in that it solves the optimal control problem on-line for the current state of the plant, rather than determining off-line policy. From this point of view, MPC differs from other control methods merely in its implementation.
Predictive control law has the following components:

1. The control law depends on predicted behaviour.
2. The output predictions are computed using a process model.
3. The current input is determined by optimising some measure of predicted performance.
4. The receding horizon: the control input is updated at every sampling instant.

2.6 The Online Optimal Control Law

At time k, a sequence of future input will be solved $\Delta u(k), \Delta u(k+1), \Delta u(k+2), \Delta u(k+3), \ldots$ but only instant input $\Delta u(k)$ will carry out actually by the system. At the next sample time, time $k+1$, the whole process of prediction and optimization will be repeated and a new future input sequence is obtained. This is the essence of online optimization [23] .

This operation can introduce information in to the controller, such as the error between predictive output and real output, so model mismatch and other disturbances can be eliminated gradually. In some extent, the online optimization can be recognized as a kind of feedback control.

For linear systems, control law of MPC can often be obtained analytically, but for most nonlinear systems, we have to use numerical optimization algorithms to get the control solution. While, constraints (on input, on output or on both of them) may cause big trouble

in online optimization, for linear system, there is some method that can deal with simple constraints, but for complex constraints or for nonlinear systems, numerical methods are still the only usable means[23].

2.7 Model Linearization

Definition 1. *A state $x_e \in \mathbb{R}^n$ and an input $u_e \in \mathbb{R}^m$ are an equilibrium pair if for initial condition $x(0) = x_e$ and constant input $u(t) = u_e$ the state remains constant: $x(t) = x_e, \forall t \geq 0$.*

The above definition is equivalent to the statement: (x_e, u_e) is an equilibrium pair if $f(x_e, u_e) = 0$. In this case, x_e is called equilibrium state, u_e equilibrium input.
Real systems are usually nonlinear and they are represented by nonlinear analytical models consisting of nonlinear differential/difference equations. Linear systems (models) are in fact idealized representations, and are represented by linear differential/difference equations. Clearly, it is far more convenient to analyze, simulate, design, and control linear systems. For this reason, nonlinear systems are often approximated by linear models. A nonlinear analytical model may contain several nonlinear terms. The approach taken here is to linearize each nonlinear term by using the first order Taylor series approximation, which involves only the first derivative of the nonlinear terms (i.e., the tangent of its graphical representation). Note that a nonlinear term could be a function of more than one independent variable. In that case the first derivatives with respect to all its independent variables are needed in the linearization process (i.e., tangents along all orthogonal directions of the coordinate axes which represent the independent variables).

2.7.1 Linearization about an Operating Point (Equilibrium Point)

Linearization is carried out about some operating point. This is typically the normal operating condition of the system. By necessity, the normal operating condition (steady-state or the equilibrium state). In a steady-state, by definition, the rates of changes of the system variables are zero. Hence, the steady-state (equilibrium state) is determined by setting the time-derivative terms in the system equations to zero and then solving the resulting algebraic equations. This may lead to more than one solution, since the steady-state (algebraic) equations themselves are nonlinear. The steady-state (equilibrium) solutions can be:

1. Stable (given a slight shift, the system response will eventually return to the original steady-state).

2. Unstable (given a slight shift, the system response will continue to move away from the original steady-state), or

3. Neutral (given a slight shift, the system response will remain in the shifted condition).

2.7.2 State Space Model

The output of a system is affected by its state and, sometimes, directly by the input. The output can be the state or function of the state. The variation of the state is affected by the variation of the input. In order to emphasize the variation in the state, in general the mathematical model of a system can be represented as

$$\dot{x} = f(x, u, t), \quad y = g(x, u, t) \tag{2.1}$$

where x is the state, y the output, u the input, t the time, $\dot{x} = dx/dt$
Usually, x, u, y are vectors, $x = [x_1, x_2, \ldots, x_n]^T, y = [y_1, y_2, \ldots, y_r]^T, u = [u_1, u_2, \ldots, u_m]^T$. x_i is the i-th state. One can represent the state, output and input as

$$x \in \mathbb{R}^n, y \in \mathbb{R}^r, u \in \mathbb{R}^m$$

where $\mathbb{R}^n$ is the n-dimensional real-valued space.

If the solution of (2.1) exists, then this solution can be generally represented as

$$x(t) = \phi(t, t_0, x(t_0), u(t)), \quad y(t) = \varphi(t, t_0, x(t_0), u(t)), t \geq t_0 \tag{2.2}$$

If the system is relaxed at time t_0, i.e., $x(t_0) = 0$, then the solution can be generally represented as

$$x(t) = \phi_0(t, t_0, u(t)), \quad y(t) = \varphi_0(t, t_0, u(t)), t \geq t_0 \tag{2.3}$$

If the solution (2.3) satisfies the following superposition principle, then the system (2.1) is linear.

Superposition Principle Suppose $\phi_{0,a}(t, t_0, u_a(t))$ and $\phi_{0,b}(t, t_0, u_b(t))$ are the motions by applying the inputs $u_a(t)$ and $u_b(t)$, respectively. Then,the motion by applying $\alpha u_a(t)+\beta u_b(t)$ is

$$\phi_0(t, t_0, \alpha u_a(t) + \beta u_b(t)) = \alpha\phi_{0,a}(t, t_0, u_a(t)) + \beta\phi_{0,b}(t, t_0, u_b(t))$$

where α and β are arbitrary scalars.

For a linear system, if it is not relaxed, then the motion caused by the initial state $x(t_0)$ can be added to the motion with relaxed system, by applying the superposition principle. Satisfying the superposition principle is the fundamental approximation or assumption in some classical MPC algorithms.

If a system satisfies the superposition principle, then it can be further simplified as

$$\dot{x} = A(t)x + B(t)u, \quad y = C(t)x + D(t)u \tag{2.4}$$

where $A(t), B(t), C(t)$ and $D(t)$ are matrices with appropriate dimensions. If a linear system is time-invariant, then it can be represented by the following mathematical model:

$$\dot{x} = Ax + Bu, \quad y = Cx + Du. \tag{2.5}$$

A linear system satisfies the superposition principle, which makes the mathematical computations much more convenient and a general closed-form solution exists. In the system investigation and model application, complete results exist only for linear systems. However,

a real system is usually nonlinear. How to linearize a nonlinear system so as to analyze and design it according to the linearized model becomes a very important problem.

The basic idea for linearization is as follows. Suppose a real system moves around its equilibrium point (steady-state) according to various disturbances,and the magnitude of the motion is relatively small. In such a small range, the relationships between variables can be linearly approximated.

Mathematically, if a system is in a steady-state (x_e, y_e, u_e), then its state x is time-invariant, i.e., $\dot{x} = 0$. Hence, according to (2.1),

$$f(x_e, u_e, t) = 0, \quad y_e = g(x_e, u_e, t). \tag{2.6}$$

Let $x = x_e + \delta x, y = y_e + \delta y, u = u_e + \delta u$. Suppose the following matrices exist (called Jacobian matrices):

$$A(t) = \frac{\partial f}{\partial x}|_e = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

$$B(t) = \frac{\partial f}{\partial u}|_e = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdots & \frac{\partial f_1}{\partial u_m} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdots & \frac{\partial f_2}{\partial u_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \frac{\partial f_n}{\partial u_2} & \cdots & \frac{\partial f_n}{\partial u_m} \end{bmatrix}$$

$$C(t) = \frac{\partial g}{\partial x}|_e = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_r}{\partial x_1} & \frac{\partial g_r}{\partial x_2} & \cdots & \frac{\partial g_r}{\partial x_n} \end{bmatrix}$$

$$D(t) = \frac{\partial g}{\partial u}|_e = \begin{bmatrix} \frac{\partial g_1}{\partial u_1} & \frac{\partial g_1}{\partial u_2} & \cdots & \frac{\partial g_1}{\partial u_m} \\ \frac{\partial g_2}{\partial u_1} & \frac{\partial g_2}{\partial u_2} & \cdots & \frac{\partial g_2}{\partial u_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_r}{\partial u_1} & \frac{\partial g_r}{\partial u_2} & \cdots & \frac{\partial g_r}{\partial u_m} \end{bmatrix}$$

where e indicates "at the equilibrium point." Then in the neighborhood of (x_e, y_e, u_e) the system (2.1) can be approximated by

$$\delta\dot{x} = A(t)\delta x + B(t)\delta u, \quad \delta y = C(t)\delta x + D(t)\delta u. \tag{2.7}$$

If $(x_e, y_e, u_e) = (0, 0, 0)$, then equation (2.7) gives

$$\dot{x} = A(t)x + B(t)u, \quad y = C(t)x + D(t)u.$$

2.8 Discretization of Continuous-Time Systems

In the real applications, there is another kind of system whose every variable changes only after a certain time interval (e.g., bank interest counting system), rather than continuously with time, which is called discrete-time systems[2].
In digital control of continuous-time plants, we need to convert continuous-time state-space equation in to discrete-time state-space equation. Such conversion can be done by introducing fictitious samplers and fictitious holding devices in to continuous-time systems. The error introduced by discretization may be made negligible by using a sufficiently small sampling period compared with the significant time constant of the system[16]. The procedure that transforms an originally continuous-time system to a discrete-time system is called the discretization of this continuous-time system[2].

Suppose we are given the continuous time state space system and apply an input that changes only at discrete (equal) sampling intervals. We assume that the input vector $u(t)$ changes only at equally spaced sampling instants.
Note that the sampling operation here is fictitious. Simply denote kT as k, e.g., the state, output and input at time k are represented as $x(k), y(k)$ and $u(k)$, respectively. The following is utilized to approximate the derivative of x[2]:

$$\frac{dx}{dt} \approx \frac{x(k+1) - x(k)}{T}$$

Thus, corresponding to (2.1) we obtain

$$x(k+1) = x(k) + Tf(x(k), u(k), k), \quad y(k) = g(x(k), u(k), k) \tag{2.8}$$

and corresponding to (2.4),

$$x(k+1) = (I + A(k)T)x(k) + B(k)Tu(k), \quad y(k) = C(k)x(k) + D(k)u(k). \tag{2.9}$$

Many linear systems have their general solutions, by which the values at different sampling instants can be calculated and the exact result of the discretization is obtained. For the linear time-invariant system (2.5) the exact result for discretization is obtained as follow. We use the notation kT and $(k+1)T$ instead of k and $k+1$. It would be nice if we could find matrices G and H, independent of t or k so that we could obtain a discrete time model of the system,

$$x((k+1)T) = G(T)x(kT) + H(T)u(kT) \tag{2.10}$$

$$y(kT) = Cx(kT) + Du(kT) \tag{2.11}$$

Note that the matrices G and H depends on the sampling period T. Once a sampling period T is fixed, G and H are constant matrices. We will now determine the values of the matrices $G(T)$ and $H(T)$. We start by using the solution of (2.5) to calculate the values of the state x at times kT and $(k+1)T$. These are

$$x((k+1)T) = e^{A(k+1)T}x(0) + e^{A(k+1)T}\int_0^{(k+1)T} e^{-A\tau}Bu(\tau)d\tau \tag{2.12}$$

$$x(kT) = e^{AkT}x(0) + e^{AkT}\int_0^{kT} e^{-A\tau}Bu(\tau)d\tau \tag{2.13}$$

Multiplying equation (2.13) by e^{AT} and subtracting it from equation (2.12) gives us

$$x((k+1)T) = e^{AT}x(kT) + e^{A(k+1)T}\int_{kT}^{(k+1)T} e^{-A\tau}Bu(\tau)d\tau. \tag{2.14}$$

Next, we notice that within the interval from kT to $(k+1)T$, $\quad u(t) = u(kT)$ is constant, as is the matrix B, so we can take them out of the integral to obtain

$$x((k+1)T) = e^{AT}x(kT) + e^{A(k+1)T}\int_{kT}^{(k+1)T} e^{-A\tau}d\tau Bu(kT) \qquad \tau \in [kT, (k+1)T). \tag{2.15}$$

We can take the $e^{A(k+1)T}$ inside the integral to obtain

$$x((k+1)T) = e^{AT}x(kT) + \int_{kT}^{(k+1)T} e^{A[(k+1)T-\tau]}d\tau Bu(kT) \qquad \tau \in [kT, (k+1)T). \tag{2.16}$$

Now we see that as τ ranges from kT to $(k+1)T$ (the lower to the upper limit of integration) the exponent of e ranges from T to 0. Accordingly, lets define a new variable $\lambda = (k+1)T - \tau$. Then $d\lambda = -d\tau$ and λ ranges from T to 0 as τ ranges from kT to $(k+1)T$. Thus we have

$$x((k+1)T) = e^{AT}x(kT) - \int_{T}^{0} e^{A\lambda}d\lambda Bu(kT) \qquad \lambda \in [0, T). \tag{2.17}$$

or

$$x((k+1)T) = e^{AT}x(kT) + \int_{0}^{T} e^{A\lambda}d\lambda Bu(kT) \qquad \lambda \in [0, T). \tag{2.18}$$

We see that in (2.18) we have written the state update equation exactly in the form of (2.10) where

$$G(T) = e^{AT} \tag{2.19}$$

$$H(T) = \int_{0}^{T} e^{A\lambda}d\lambda B, \tag{2.20}$$

so we are done except that we would rather not leave the expression for $H(T)$ in the form of an integral. So long as A is invertible, we can easily integrate, using the fact that

$$\frac{de^{At}}{dt} = Ae^{At} = e^{At}A$$

to obtain

$$H(T) = A^{-1}\int_{0}^{T} Ae^{A\lambda}d\lambda B = A^{-1}e^{A\lambda}|_{\lambda=0}^{T}B \tag{2.21}$$

$$= A^{-1}(e^{AT} - I)B = (e^{AT} - I)A^{-1}B \tag{2.22}$$

2.9 Stability

The main goal of model predictive control, namely to control the state $x(k)$ of the system toward a reference trajectory $r(k)$ and then keep it close to this reference. In this section we formalize what we mean by toward and close to using concepts from stability theory of a systems. Consider the discrete-time nonlinear system

$$\begin{cases} x(k+1) &= f(x(k), u(k)) \\ y(k) &= g(x(k), u(k)) \end{cases}$$

Definition 2. *A state $x_e \in \mathbb{R}^n$ and an input $u_e \in \mathbb{R}^m$ are an **equilibrium pair** if for initial condition $x(0) = x_e$ and constant input $u(k) = u_e, \forall k \in \mathbb{N}$, the state remains constant: $x(k) = x_e, \forall k \in \mathbb{N}$*

The above definition is equivalent to the statement: (x_e, u_e) is an equilibrium pair if $f(x_e, u_e) = x_e$. In this case, x_e is called equilibrium state, u_e equilibrium input, and let x_e be an equilibrium state, so that $f(x_e, u_e) = x_e$.

Definition 3. *The equilibrium state x_e is said to be*

i. ***stable** if for each initial conditions $x(0)$ close enough to x_e , the corresponding trajectory $x(k)$ remains near x_e for all $k \in \mathbb{N}$ (i.e $\forall \epsilon > 0, \exists \delta > 0 : ||x(0) - x_e|| < \delta \Rightarrow ||x(k) - x_e|| < \epsilon, \forall k \in \mathbb{N}$).*

ii. ***asymptotically stable** if it is stable and $x(k) \to x_e$ for $k \to \infty$.*

*iii. otherwise,**unstable**.*

Time-invariant discrete-time linear systems

$$\begin{cases} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k) \end{cases}$$

can be tested for stability according to the following results.

Theorem 2.9.1. *Let $\lambda_1, \lambda_2, \ldots, \lambda_m, m \leq n$ be the eigenvalues of $A \in \mathbb{R}^{n \times n}$. The system $x(k+1) = Ax(k) + Bu(k)$ is*

i. ***asymptotically stable** iff $|\lambda_i| < 1, \quad \forall i = 1, \ldots, m$.*

ii. ***(marginally) stable** if $|\lambda_i| \leq 1, \quad \forall i = 1, \ldots, m$, and the eigenvalues with unit modulus have equal algebraic and geometric multiplicity.*

iii. ***unstable** if $\exists i$ such that $|\lambda_i| > 1$.*

The stability properties of a discrete-time linear system only depend on the modulus of the eigenvalues of matrix A.

2.10 Controllability and Observability

Definition 4. *(**Controllability**) The state x^* is said to be controllable if there is an input(a control) u that transforms the system from its initial state $x(0) = x_0$ into the state x^* within finite time t.*

*The system is said to be **controllable** if all the states of the system are controllable.*

Definition 5. *(**Observability**) A system with an initial state x_0 is said to be observable if the value of the initial state can be determined from the systems output that has been observed through the time interval $t_0 < t < t_f$.*

If the initial state cannot be, so determined, the system is unobservable.

For a linear time invariant discrete-time control system given by

$$\begin{aligned} x((k+1)T) &= Ax(kT) + Bu(kT) \\ y(kT) &= Cx(kT) \end{aligned} \tag{2.23}$$

where $x(kT)$ is the state vector at k^{th} sampling instant, $u(kT)$ control signal at k^{th} sampling instant, $y(kT)$ is the output vector, A, B and C are constant matrices of compatible dimensions.

Theorem 2.10.1. ***(Popov-Belevitch-Hautus(PBH) Test for Controllability and Observability)***
The system (2.23) *is said to be*

1. *Controllable iff the matrix $[\lambda I - A \quad B]$ has full rank for all $\lambda \in \mathbb{C}$.*
2. *Observable iff the matrix $\begin{bmatrix} \lambda I - A \\ C \end{bmatrix}$ has full rank for all $\lambda \in \mathbb{C}$.*

Proof. see [22] □

Remark 1. *If λ is not an eigenvalue of A, then $[\lambda I - A \quad B]$ and $\begin{bmatrix} \lambda I - A \\ C \end{bmatrix}$ are of a full rank matrix.*

So, to check non controllability and non observability take an eigenvalue λ, and check whether B and C fills out the null space of $\lambda I - A$.

Chapter 3

Discrete-Time MPC Without Constraint

3.1 Introduction

In this chapter, we will introduce the basic ideas about model predictive control. In Section 3.2, a single-input and single-output state-space model with an embedded integrator is introduced, which is used in the design of discrete-time predictive controllers with integral action in this project. In Section 3.3, we examine the design of predictive control within one optimization window. This is demonstrated by simple analytical examples. With the results obtained from the optimization, in Section 3.4, we discuss the ideas of receding horizon control, and state feedback gain matrices, and the closed-loop configuration of the predictive control system. The results are extended to multi-input and multi-output systems in Section 3.5. In a general framework of state-space design, an observer is needed in the implementation, and this is discussed in Section 3.6. With a combination of estimated state variables and the predictive controller, in Section 3.7 , we present state estimate predictive control.

3.2 State-space Models with Embedded Integrator

Model predictive control systems are designed based on a mathematical model of the plant. The model to be used in the control system design is taken to be a state-space model. By using a state-space model, the current information required for predicting ahead is represented by the state variable at the current time.

3.2.1 Single-input and Single-output System

For simplicity, we begin our study by assuming that the underlying plant is a single-input and single-output system, described by:

$$x_m(k+1) = A_m x_m(k) + B_m u(k) \tag{3.1}$$

$$y(k) = C_m x_m(k) \tag{3.2}$$

where u is the manipulated variable or input variable; y is the process output; and x_m is the state variable vector with assumed dimension n_1. Note that this plant model has $u(k)$ as its input. Thus, we need to change the model to suit our design purpose in which an integrator is embedded.

Note that a general formulation of a state-space model has a direct term from the input signal $u(k)$ to the output $y(k)$ as

$$y(k) = C_m x_m(k) + D_m u(k).$$

However, due to the principle of receding horizon control, where a current information of the plant is required for prediction and control, we have implicitly assumed that the input $u(k)$ cannot affect the output $y(k)$ at the same time. Thus, $D_m = 0$ in the plant model.

Taking a difference operation on both sides of (3.1), we obtain that

$$x_m(k+1) - x_m(k) = A_m(x_m(k) - x_m(k-1)) + B_m(u(k) - u(k-1)).$$

Let us denote the difference of the state variable by

$$\Delta x_m(k+1) = x_m(k+1) - x_m(k); \quad \Delta x_m(k) = x_m(k) - x_m(k-1)$$

and the difference of the control variable by

$$\Delta u(k) = u(k) - u(k-1)$$

These are the increments of the variables $x_m(k)$ and $u(k)$. With this transformation, the difference of the state-space equation is:

$$\Delta x_m(k+1) = A_m \Delta x_m(k) + B_m \Delta u(k) \tag{3.3}$$

Note that the input to the state-space model is $\Delta u(k)$. The next step is to connect $\Delta x_m(k)$ to the output $y(k)$. To do so, a new state variable vector is chosen to be

$$x(k) = [\Delta x_m(k)^T \quad y(k)]^T$$

where superscript ^{T} indicates matrix transpose. Note that

$$\begin{aligned} y(k+1) - y(k) &= C_m(x_m(k+1) - x_m(k)) = C_m \Delta x_m(k+1) \\ &= C_m A_m \Delta x_m(k) + C_m B_m \Delta u(k \end{aligned} \tag{3.4}$$

Putting together (3.3) with (3.4) leads to the following state-space model:

$$\overbrace{\begin{bmatrix}\Delta x_m(k+1)\\ y(k+1)\end{bmatrix}}^{x(k+1)} = \overbrace{\begin{bmatrix}A_m & 0_m^T\\ C_mA_m & 1\end{bmatrix}}^{A}\overbrace{\begin{bmatrix}\Delta x_m(k)\\ y(k)\end{bmatrix}}^{x(k)} + \overbrace{\begin{bmatrix}B_m\\ C_mB_m\end{bmatrix}}^{B}\Delta u(k)$$
$$y(k) = \overbrace{\begin{bmatrix}0_m & 1\end{bmatrix}}^{C}\begin{bmatrix}\Delta x_m(k)\\ y(k)\end{bmatrix} \tag{3.5}$$

where $0_m = \overbrace{\begin{bmatrix}0 & 0 & \dots & 0\end{bmatrix}}^{n_1}$. The triplet (A, B, C) is called the augmented model, which will be used in the design of predictive control. (A, B) is assumed to be a controllable pair and (A, C) is assumed to be an observable pair.

Example 3.1. *Consider a discrete-time model in the following form:*

$$\begin{aligned} x_m(k+1) =& A_m x_m(k) + B_m u(k)\\ y(k) =& C_m x_m(k)\end{aligned} \tag{3.6}$$

where the system matrices are $A_m = \begin{bmatrix}1 & 1\\ 0 & 1\end{bmatrix}; B_m = \begin{bmatrix}0.5\\ 1\end{bmatrix}; C_m = \begin{bmatrix}1 & 0\end{bmatrix}$.
Find the triplet matrices (A, B, C) in the augmented model (3.5).
Solution. *From (3.5),* $n_1 = 2$ *and* $0_m = \begin{bmatrix}0 & 0\end{bmatrix}$. *The augmented model for this plant is given by*

$$\begin{aligned} x(k+1) =& Ax(k) + B\Delta u(k)\\ y(k) =& Cx(k)\end{aligned} \tag{3.7}$$

where the augmented system matrices are

$$A = \begin{bmatrix}A_m & 0_m^T\\ C_mA_m & 1\end{bmatrix} = \begin{bmatrix}1 & 1 & 0\\ 0 & 1 & 0\\ 1 & 1 & 1\end{bmatrix}; \quad B = \begin{bmatrix}B_m\\ C_mB_m\end{bmatrix} = \begin{bmatrix}0.5\\ 1\\ 0.5\end{bmatrix};$$
$$C = \begin{bmatrix}0_m & 1\end{bmatrix} = \begin{bmatrix}0 & 0 & 1\end{bmatrix}.$$

3.3 Predictive Control within One Optimization Window

Upon formulation of the mathematical model, the next step in the design of a predictive control system is to calculate the predicted plant output with the future control signal as the adjustable variables. This prediction is described within an optimization window. This section will examine in detail the optimization carried out within this window. Here, we assume that the current time is k_i and the length of the optimization window is N_p as the number of samples. For simplicity, the case of single-input and single-output systems is considered first, then the results are extended to multi-input and multi-output systems.

3.3.1 Prediction of State and Output Variables

Assuming that at the sampling instant k_i, $\quad k_i > 0$, the state variable vector $x(k_i)$ is available through measurement, the state $x(k_i)$ provides the current plant information. The future control trajectory is denoted by

$$\Delta u(k_i), \Delta u(k_i+1), \Delta u(k_i+2), \ldots, \Delta u(k_i+N_c-1),$$

where N_c is called the control horizon dictating the number of parameters used to capture the future control trajectory. With given information $x(k_i)$, the future state variables are predicted forN_p number of samples, where N_p is called the prediction horizon. N_p is also the length of the optimization window. We denote the future state variables as

$$x(k_i+1|k_i), x(k_i+2|k_i), \ldots, x(k_i+m|k_i), \ldots, x(k_i+N_p|k_i),$$

where $x(k_i+m|k_i)$ is the predicted state variable at k_i+m with given current plant information $x(k_i)$. The control horizon N_c is chosen to be less than (or equal to) the prediction horizon N_p.

Based on the state-space model (A, B, C), the future state variables are calculated sequentially using the set of future control parameters:

$$\begin{aligned}
x(k_i+1|k_i) &= Ax(k_i) + B\Delta u(k_i) \\
x(k_i+2|k_i) &= Ax(k_i+1) + B\Delta u(k_i+1) \\
&= A^2x(k_i) + AB\Delta u(k_i) + B\Delta u(k_i+1) \\
&\vdots \\
x(k_i+N_p|k_i) &= A^{N_p}x(k_i) + A^{N_p-1}B\Delta u(k_i) + A^{N_p-2}B\Delta u(k_i+1) \\
&\quad + \ldots + A^{N_p-N_c}B\Delta u(k_i+N_c-1).
\end{aligned}$$

From the predicted state variables, the predicted output variables are, by substitution

$$\begin{aligned}
y(k_i+1|k_i) &= CAx(k_i) + CB\Delta u(k_i) \qquad (3.8) \\
y(k_i+2|k_i) &= CA^2x(k_i) + CAB\Delta u(k_i) + CB\Delta u(k_i+1) \\
y(k_i+3|k_i) &= CA^3x(k_i) + CA^2B\Delta u(k_i) + CAB\Delta u(k_i+1) \\
&\quad + CB\Delta u(k_i+2) \\
&\vdots \\
y(k_i+N_p|k_i) &= CA^{N_p}x(k_i) + CA^{N_p-1}B\Delta u(k_i) + CA^{N_p-2}B\Delta u(k_i+1) \\
&\quad + \ldots + CA^{N_p-N_c}B\Delta u(k_i+N_c-1). \qquad (3.9)
\end{aligned}$$

Note that all predicted variables are formulated in terms of current state variable information $x(k_i)$ and the future control movement $\Delta u(k_i+j)$, where $j = 0, 1, \ldots, N_c-1$.

Define vectors

$$Y = [y(k_i+1|k_i) \quad y(k_i+2|k_i) \quad y(k_i+3|k_i) \quad \ldots \quad y(k_i+N_p|k_i)]^T$$

$$\Delta U = [\Delta u(k_i) \quad \Delta u(k_i+1) \quad \Delta u(k_i+2) \quad \dots \quad \Delta u(k_i+N_c-1)]^T$$

where in the single-input and single-output case, the dimension of Y is N_p and the dimension of ΔU is N_c. We collect (3.8) and (3.9) together in a compact matrix form as

$$Y = Fx(k_i) + \Phi\Delta U, \tag{3.10}$$

where

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix}; \Phi = \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & & & & \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix}$$

3.3.2 Optimization

The control objectives can be of variable nature. For example, the main objective of the control of a passenger airplane is the flight to a defined airport along an assumed flight trajectory. Initially, basic objectives of the control of technological or economic processes in a market economy are of an economic nature gaining profit from a production or a commercial activity. Similarly, the initial objective of a telecommunications network control is economic in nature a long-term profit from the network operation. However, in order to achieve the basic economic objective effectively, it is essential to ensure the realization of a set of partial objectives, which condition the possibility of a safe realization of the basic objective and guarantee the required quality parameters of the offered products or services.
A cost function is an important part of a predictive controller because it is an indicator of the degree of optimality of a dynamic response, resulting a sequence of control inputs ΔU applied to the system. This degree of optimality may express how close we are to the desired output or state levels including how much effort is needed to get there, and in MPC this is referred to as the cost. In the controller itself the role of the cost function is reversed, and we are aiming to calculate the best series of control inputs ΔU which results in a minimal cost.

For a given set-point signal $r(k_i)$ at sample time k_i, within a prediction horizon the objective of the predictive control system is to bring the predicted output as close as possible to the set-point signal, where we assume that the set point signal remains constant in the optimization window. This objective is then translated into a design to find the best control parameter vector ΔU such that an error function between the set-point and the predicted output is minimized. The minimized objective function (the cost-function) usually consists of two parts. The first one takes into account the differences between the predicted trajectory of the output variable and the set-point trajectory (i.e. the predicted control errors) over the prediction horizon N_p. The role of the second part of the objective function (the penalty term) is to reduce excessive (and hence disadvantageous) changes of the manipulated variable.

Assuming that the data vector that contains the set-point information is

$$R_s^T = \overbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}}^{N_p} r(k_i)$$

The control law is determined from the minimisation of a 2-norm measure of predicted performance. A typical performance index (or cost function) is

$$\begin{aligned} J =& \sum_{j=1}^{N_p} ||r(k_i) - y(k_i + j)||_2^2 + r_w \sum_{j=0}^{N_c-1} ||\Delta u(k_i + j)||_2^2 \\ =& \sum_{j=1}^{N_p} ||e(k_i + j))||_2^2 + r_w \sum_{j=0}^{N_c-1} ||\Delta u(k_i + j)||_2^2 \\ =& (R_s - Y)^T (R_s - Y) + \Delta U^T \bar{R} \Delta U \end{aligned} \tag{3.11}$$

where $|| \cdot ||_2$ denotes the usual Euclidean norm and $r_w \geq 0$ is a weighting parameter for the control, which could also be chosen as 0 if no control penalization is desired.

The first term in J is linked to the objective of minimizing the errors between the predicted output and the set-point signal while the second term reflects the consideration given to the size of ΔU when the objective function J is made to be as small as possible. $\bar{R}$ is a diagonal matrix in the form that $\bar{R} = r_w I_{N_c \times N_c} (r_w \geq 0)$ where r_w is used as a tuning parameter for the desired closed-loop performance. For the case that $r_w = 0$, the cost function (3.11) is interpreted as the situation where we would not want to pay any attention to how large the ΔU might be and our goal would be solely to make the error $(R_s - Y)^T (R_s - Y)$ as small as possible. For the case of large r_w, the cost function (3.11) is interpreted as the situation where we would carefully consider how large the ΔU might be and cautiously reduce the error $(R_s - Y)^T (R_s - Y)$.
The terms in the performance index are easy to visualise in graphical form. Assume a sampling instant of 1 sec; then:

- Figure 3.1 illustrates the first four tracking errors e_i.
- Figure 3.2 illustrates the control moves for $N_c = 3$ (denoted by the double-sided arrows).

To find the optimal ΔU that will minimize J, by using (3.10), J is expressed as

$$\begin{aligned} J =& (R_s - Fx(k_i))^T (R_s - Fx(k_i)) - 2\Delta U^T \Phi^T (R_s - Fx(k_i)) \\ &+ \Delta U^T (\Phi^T \Phi + \bar{R}) \Delta U \end{aligned} \tag{3.12}$$

From the first derivative of the cost function J :

$$\frac{\partial J}{\partial \Delta U} = -2\Phi^T (R_s - Fx(k_i)) + 2(\Phi^T \Phi + \bar{R}) \Delta U \tag{3.13}$$

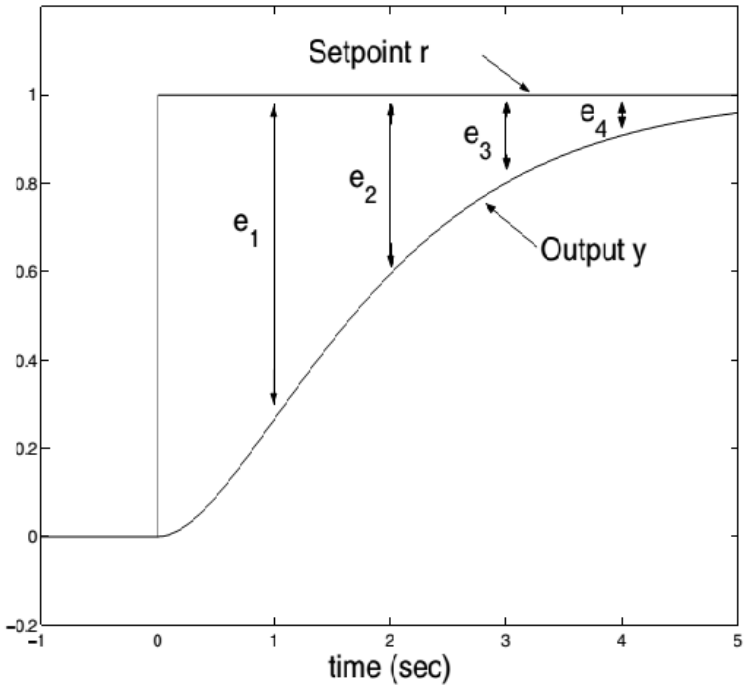


Figure 3.1: Tracking errors.

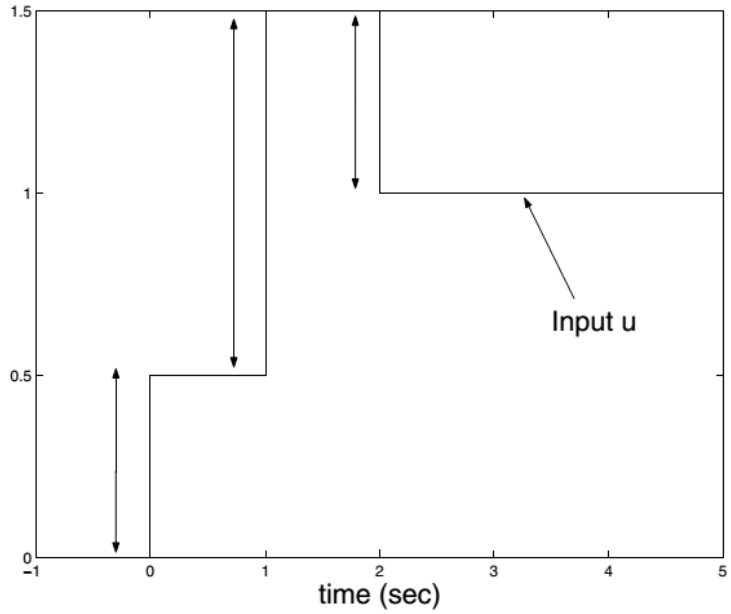


Figure 3.2: Input and input increments.

the necessary condition of the minimum J is obtained as

$$\frac{\partial J}{\partial \Delta U} = 0$$

from which we find the optimal solution for the control signal as

$$\Delta U = (\Phi^T \Phi + \bar{R})^{-1} \Phi^T (R_s - F x(k_i)) \tag{3.14}$$

with the assumption that $(\Phi^T \Phi + \bar{R})^{-1}$ exists. The matrix $(\Phi^T \Phi + \bar{R})^{-1}$ is called the Hessian matrix in the optimization literature. Note that R_s is a data vector that contains the set-point information expressed as

$$R_s = \overbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}^T}^{N_p} r(k_i) = \bar{R}_s r(k_i)$$

where

$$\bar{R}_s = \overbrace{\begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}^T}^{N_p}$$

The optimal solution of the control signal is linked to the set-point signal $r(k_i)$ and the state variable $x(k_i)$ via the following equation:

$$\Delta U = (\Phi^T \Phi + \bar{R})^{-1} \Phi^T (\bar{R}_s r(k_i) - F x(k_i)) \tag{3.15}$$

Example 3.2. *Suppose that a first-order system is described by the state equation:*

$$\begin{aligned} x_m(k+1) =& a x_m(k) + b u(k) \\ y(k) =& x_m(k) \end{aligned} \tag{3.16}$$

where $a = 0.8$ and $b = 0.1$ are scalars. Find the augmented state-space model. Assuming a prediction horizon $N_p = 10$ and control horizon $N_c = 4$, calculate the components that form the prediction of future output Y ,and the quantities $\Phi^T \Phi$, $\Phi^T F$ and $\Phi^T \bar{R}_s$. Assuming that at a time $k_i (k_i = 10$ for this example, but doesn't mean that $N_p = k_i$ in general case), $r(k_i) = 1$ and the state vector $x(k_i) = \begin{bmatrix} 0.1 & 0.2 \end{bmatrix}^T$, find the optimal solution ΔU with respect to the cases where $r_w = 0$ and $r_w = 10$, and compare the results.

Solution. *The augmented state-space equation is*

$$\begin{aligned} \begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} &= \begin{bmatrix} a & 0 \\ a & 1 \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} b \\ b \end{bmatrix} \Delta u(k) \\ y(k) &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} \end{aligned} \tag{3.17}$$

Based on (3.10), the F and Φ matrices take the following forms:

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ CA^4 \\ CA^5 \\ CA^6 \\ CA^7 \\ CA^8 \\ CA^9 \\ CA^{10} \end{bmatrix}; \Phi = \begin{bmatrix} CB & 0 & 0 & 0 \\ CAB & CB & 0 & 0 \\ CA^2B & CAB & CB & 0 \\ CA^3B & CA^2B & CAB & CB \\ CA^4B & CA^3B & CA^2B & CAB \\ CA^5B & CA^4B & CA^3B & CA^2B \\ CA^6B & CA^5B & CA^4B & CA^3B \\ CA^7B & CA^6B & CA^5B & CA^4B \\ CA^8B & CA^7B & CA^6B & CA^5B \\ CA^9B & CA^8B & CA^7B & CA^6B \end{bmatrix}$$

The coefficients in the F and Φ matrices are calculated as follows:

$$\begin{aligned} CA &= [s_1 \quad 1] \\ CA^2 &= [s_2 \quad 1] \\ CA^3 &= [s_3 \quad 1] \\ &\vdots \\ CA^k &= [s_k \quad 1], \end{aligned} \tag{3.18}$$

where $s_1 = a, \quad s_2 = a^2 + s_1, \quad \ldots, \quad s_k = a^k + s_{k1}$, and

$$\begin{aligned} CB &= g_0 = b \\ CAB &= g_1 = ab + g_0 \\ CA^2B &= g_2 = a^2b + g_1 \\ &\vdots \\ CA^kB &= g_k = a^kb + g_{k-1}, \end{aligned}$$

With the plant parameters $a = 0.8$ and $b = 0.1, \quad N_p = 10$ and $N_c = 4$, we calculate the quantities

$$\Phi^T\Phi = \begin{bmatrix} 1.1541 & 1.0407 & 0.9116 & 0.7726 \\ 1.0407 & 0.9549 & 0.8475 & 0.7259 \\ 0.9116 & 0.8475 & 0.7675 & 0.6674 \\ 0.7726 & 0.7259 & 0.6674 & 0.5943 \end{bmatrix}; \quad \Phi^TF = \begin{bmatrix} 9.2325 & 3.2147 \\ 8.3259 & 2.7684 \\ 7.2927 & 2.3355 \\ 6.1811 & 1.9194 \end{bmatrix};$$

$$\Phi^T\bar{R}_s = \begin{bmatrix} 3.2147 \\ 2.7684 \\ 2.3355 \\ 1.9194 \end{bmatrix}$$

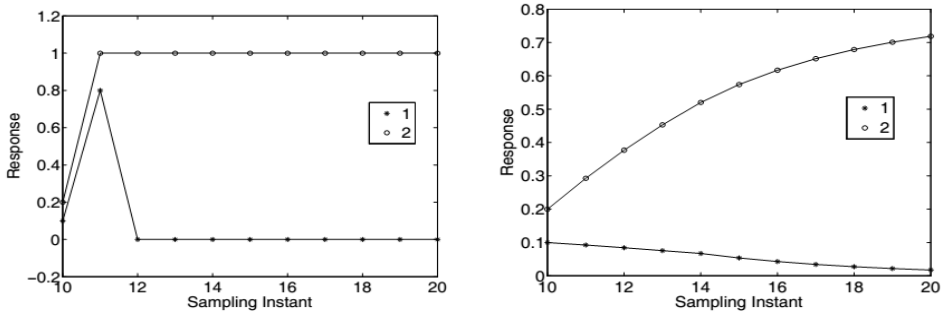
Note that the vector $\Phi^T\bar{R}_s$ is identical to the last column in the matrix Φ^TF. This is because the last column of F matrix is identical to $\bar{R}_s$.

At time $k_i = 10$, the state vector $x(k_i) = [0.1 \quad 0.2]^T$. In the first case, the error between

predicted Y and R_s is reduced without any consideration to the magnitude of control changes. Namely, $r_w = 0$. Then, the optimal ΔU is found through the calculation

$$\Delta U = (\Phi^T\Phi)^{-1}(\Phi^T R_s - \Phi^T F x(k_i)) = [7.2 \quad -6.4 \quad 0 \quad 0]^T$$

We note that without weighting on the incremental control, the last two elements $\Delta u(k_i+2) = 0$ and $u(k_i+3) = 0$, while the first two elements have a rather large magnitude. Figure 3.3a shows the changes of the state variables where we can see that the predicted output y has reached the desired set-point 1 while the Δx_m decays to zero. To examine the effect of the



(a) State variables with no weight on Δu (b) State variables with weight on Δu

Figure 3.3: Comparison of optimal solutions. Key: line (1) Δx_m; line (2) y

weight r_w on the optimal solution of the control, we let $r_w = 10$. The optimal solution of ΔU is given below, where I is a 4×4 identity matrix

$$\Delta U = (\Phi^T\Phi + 10 \times I)^{-1}(\Phi^T R_s - \Phi^T F x(k_i)) = [0.1269 \quad 0.1034 \quad 0.0829 \quad 0.065]^T \quad (3.19)$$

With this choice, the magnitude of the first two control increments is significantly reduced, also the last two components are no longer zero. Figure 3.3b shows the optimal state variables. It is seen that the output y did not reach the set-point value of 1, however, the Δx_m approaches zero.

An observation follows from the comparison study. It seems that if we want the control to move cautiously, then it takes longer for the control signal to reach its steady state (i.e., the values in ΔU decrease more slowly), because the optimal control energy is distributed over the longer period of future time. We can verify this by increasing N_c to 9, while maintaining $r_w = 10$. The result shows that the magnitude of the elements in ΔU is reducing, but they are significant for the first 8 elements:

$$\Delta U^T = [0.1227 \quad 0.0993 \quad 0.0790 \quad 0.0614 \quad 0.0463 \quad 0.0334 \quad 0.0227 \quad 0.0139 \quad 0.0072]$$

In comparison with the case where $N_c = 4$,we note that when $N_c = 9$, the first four parameters in ΔU are slightly different from the previous case.

3.4 Receding Horizon Control(RHC)

The terminology receding horizon is often applied to predictive control because one can imagine an MPC law as using a receding horizon, that is a horizon that is constantly moving

away, although at the same speed at which you are moving. Clearly the time span over which the optimisation takes place is always moving (receding). At the current sample one takes into account points that previously were beyond the time span. This is illustrated in table below where clearly the start point and end point of the costing horizon recede.

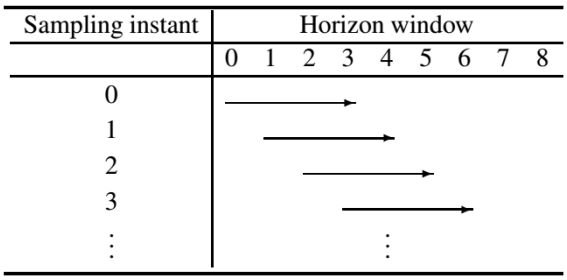


Illustration of receding horizon.

The Receding Horizon Control Principle

The idea of receding horizon principle is described as follows:

1. At time k_i and for the current state $x(k_i)$, solve an optimal control problem over a fixed future interval, say $[k_i, k_i + N_c - 1]$.
2. Apply only the first step in the resulting optimal control sequence.
3. Measure the state reached at time $k_i + 1$.
4. Repeat the fixed horizon optimization at time k_i+1 over the future interval $[k_i+1, k_i+N_c]$, starting from the current state $x(k_i + 1)$.

Although the optimal parameter vector ΔU contains the controls $\Delta u(k_i), \Delta u(k_i+1), \Delta u(k_i+2), \ldots, \Delta u(k_i + N_c - 1)$, with the receding horizon control principle, we only implement the first sample of this sequence, i.e.$\Delta u(k_i)$, while ignoring the rest of the sequence. When the next sample period arrives, the more recent measurement is taken to form the state vector $x(k_i + 1)$ for calculation of the new sequence of control signal. This procedure is repeated in real time to give the receding horizon control law.

Example 3.3. *We illustrate this procedure by continuing Example 3.2, where a first-order system with the state-space description*

$$x_m(k + 1) = 0.8x_m(k) + 0.1u(k)$$

is used in the computation. We will consider the case $r_w = 0$. *The initial conditions are* $x(10) = [0.1 \quad 0.2]^T$ *and* $u(9) = 0$.
Solution. *At sample time* $k_i = 10$, *the optimal control was previously computed as* $\Delta u(10) =$

7.2. *Assuming that* $u(9) = 0$*, then the control signal to the plant is* $u(10) = u(9) + \Delta u(10) = 7.2$ *and with* $x_m(10) = y(10) = 0.2$*, we calculate the next simulated plant state variable*

$$x_m(11) = 0.8x_m(10) + 0.1u(10) = 0.88 \tag{3.20}$$

At $k_i = 11$*, the new plant information is* $\Delta x_m(11) = 0.88 - 0.2 = 0.68$ *and* $y(11) = 0.88$*, which forms* $x(11) = [0.68 \quad 0.88]^T$*. Then we obtain*

$$\Delta U = (\Phi^T\Phi)^{-1}(\Phi^T R_s - \Phi^T F x(11)) = [-4.24 \quad -0.96 \quad 0.0000 \quad 0.0000]^T$$

This leads to the optimal control $u(11) = u(10) + \Delta u(11) = 2.96$*. This new control is implemented to obtain*

$$x_m(12) = 0.8x_m(11) + 0.1u(11) = 1 \tag{3.21}$$

At $k_i = 12$ *, the new plant information is* $\Delta x_m(12) = 1 - 0.88 = 0.12$ *and* $y(12) = 1$*, which forms* $x(12) = [0.12 \quad 1]$ *.We obtain*

$$\Delta U = (\Phi^T\Phi)^{-1}(\Phi^T R_s - \Phi^T F x(12)) = [-0.96 \quad 0.000 \quad 0.0000 \quad 0.0000]^T$$

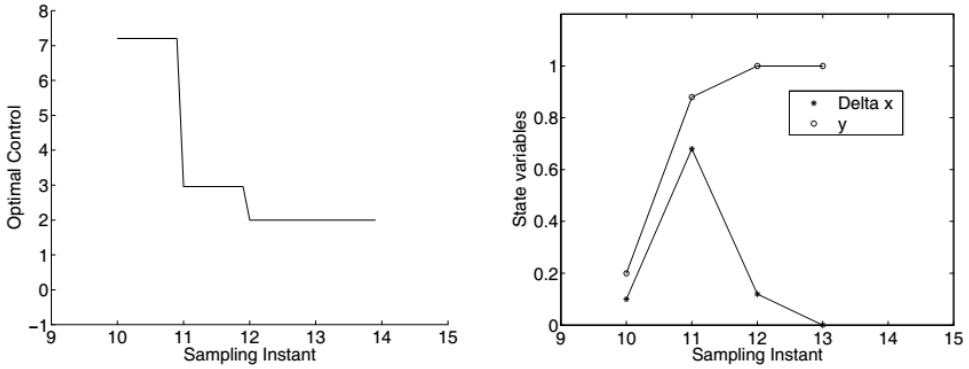
This leads to the control at $k_i = 12$ *as* $u(12) = u(11) - 0.96 = 2$ *. By implementing this control, we obtain the next plant output as*

$$x_m(13) = ax_m(12) + bu(12) = 1 \tag{3.22}$$

The new plant information is $\Delta x_m(13) = 1 - 1 = 0$ *and* $y(13) = 1$*. From this information, we obtain*

$$\Delta U = (\Phi^T\Phi)^{-1}(\Phi^T R_s - \Phi^T F x(13)) = [0.000 \quad 0.000 \quad 0.0000 \quad 0.0000]$$

Figure 3.4 shows the trajectories of the state variable Δx_m *and* y *,as well as the control*



(a) Optimal control (b) State variable

Figure 3.4: Receding horizon control

signal that was used to regulate the output. This example also illustrated the differences between the ΔU *parameter vectors at different time instances. We note that as the output response reaches the desired set-point signal, the parameters in the* ΔU *approach zero.*

3.4.1 Closed-loop Control System

There is another aspect that was illustrated by Example 3.3. If we examine this example carefully, then we find that at a given time k_i , the optimal parameter vector ΔU is solved using

$$\Delta U = (\Phi^T\Phi + \bar{R})^{-1}(\Phi^T R_s - \Phi^T F x(k_i)).$$

where $(\Phi^T\Phi+\bar{R})^{-1}\Phi^T R_s$ corresponds to the set-point change, while $-(\Phi^T\Phi+\bar{R})^{-1}\Phi^T F x(k_i)$ corresponds to the state feedback control within the framework of predictive control. Both depend on the system parameters, hence are constant matrices for a time-invariant system. Because of the receding horizon control principle, we only take the first element of ΔU at time k_i as the incremental control, thus

$$\begin{aligned}\Delta u(k_i) =& \overbrace{[1 \quad 0 \ldots 0]}^{N_c}(\Phi^T\Phi + \bar{R})^{-1}(\Phi^T\bar{R}_s r(k_i) - \Phi^T F x(k_i)) \\ =& K_y r(k_i) - K_{mpc} x(k_i), \end{aligned} \tag{3.23}$$

where K_y is the first element of

$$(\Phi^T\Phi + \bar{R})^{-1}\Phi^T\bar{R}_s$$

and K_{mpc} is the first row of

$$(\Phi^T\Phi + \bar{R})^{-1}\Phi^T F.$$

Equation (3.23) is in a standard form of linear time-invariant state feedback control. The state feedback control gain vector is K_{mpc}. Therefore, with the augmented design model

$$x(k+1) = Ax(k) + B\Delta u(k)$$

the closed-loop system is obtained by substituting (3.23) into the augmented system equation; changing index k_i to k, leading to the closed-loop equation

$$x(k+1) = Ax(k) - BK_{mpc}x(k) + BK_y r(k) \tag{3.24}$$

$$= (A - BK_{mpc})x(k) + BK_y r(k). \tag{3.25}$$

Thus, the closed-loop eigenvalues can be evaluated through the closed-loop characteristic equation:

$$det[\lambda I - (A - BK_{mpc})] = 0.$$

Because of the special structures of the matrices C and A, the last column of F is identical to $\bar{R}_s$, which is $[1 \quad 1 \quad \ldots 1]^T$, therefore K_y is identical to the last element of K_{mpc}. Noting that the state variable vector $x(k) = [\Delta x_m(k)^T \quad y(k)]^T$, and with the definition of K_y ,we can write $K_{mpc} = [K_x \quad K_y]$, where K_x corresponds to the feedback gain vector related to $\Delta x_m(k)$, and K_y corresponds to the feedback gain related to $y(k)$. Then, we obtain the closed-loop block diagram for the predictive control system as in Figure 3.5 where q^{-1} denotes the backward shift operator. The diagram shows the state feedback structure for the discrete model prediction control(DMPC) with integral action in which the module $\frac{1}{1-q^{-1}}$ denotes the discrete-time integrator.

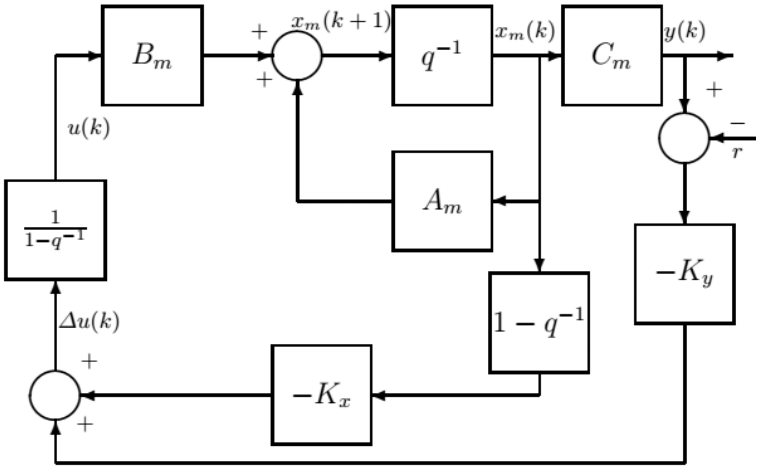


Figure 3.5: Block diagram of discrete-time predictive control system

Example 3.4. *This example will examine the closed-loop feedback gain matrices generated from Example 3.2 and the eigenvalues of the closed-loop system with weight* $r_w = 0$ *and* $r_w = 10$.

Solution. When the weight $r_w = 0$ *,we have*

$$K_y = [1 \quad 0 \quad 0 \quad 0](\Phi^T\Phi + \bar{R})^{-1}(\Phi^T \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}) = 10$$

$$K_{mpc} = [1 \quad 0 \quad 0 \quad 0](\Phi^T\Phi + \bar{R})^{-1}(\Phi^T F) = [8 \quad 10]$$

Hence, the eigenvalues of the closed-loop system are calculated by evaluating the eigenvalues of the closed-loop matrix $A - BK_{mpc}$*, where, from Example 3.2*

$$A = \begin{bmatrix} 0.8 & 0 \\ 0.8 & 1 \end{bmatrix}; B = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix}$$

They are $\lambda_1 = -6.409 \times 10^{-7}$ *and* $\lambda_2 = 6.409 \times 10^{-7}$ *, approximately on the origin of the complex plane. When the weight* $r_w = 10$*, we have*

$$K_y = [1 \quad 0 \quad 0 \quad 0](\Phi^T\Phi + \bar{R})^{-1}(\Phi^T \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}) = 0.2453$$

$$K_{mpc} = [1 \quad 0 \quad 0 \quad 0](\Phi^T\Phi + \bar{R})^{-1}(\Phi^T F) = [0.6939 \quad 0.2453]$$

With this gain vector, the eigenvalues of the closed-loop system are $\lambda_{1,2} = 0.8530 \pm 0.0542i$*, indicating that the dynamics of the closed-loop system have a much slower response than the one in the case when* $r_w = 0$.

3.5 Predictive Control of MIMO Systems

In the previous section, for simplicity of illustration the predictive control system was designed based on a single-input and single-output system. This design methodology can be readily extended to multi-input and multi-output systems without much additional effort, because of the state-space formulation.

3.5.1 General Formulation of the Model

Assume that the plant has m inputs, q outputs and n_1 states. We also assume that the number of outputs is less than or equal to the number of inputs (i.e.,$q \leq m$). If the number of outputs is greater than the number of inputs, we cannot hope to control each of the measured outputs independently with zero steady-state errors. In the general formulation of the predictive control problem, we also take the plant noise and disturbance into consideration.

$$x_m(k+1) = A_m x_m(k) + B_m u(k) + B_d \omega(k) \tag{3.26}$$

$$y(k) = C_m x_m(k) \tag{3.27}$$

where $\omega(k)$ is the input disturbance, assumed to be a sequence of integrated white noise. This means that the input disturbance $\omega(k)$ is related to a zero- mean, white noise sequence $\epsilon(k)$ by the difference equation

$$\omega(k) - \omega(k-1) = \epsilon(k) \tag{3.28}$$

Note that from (3.26), the following difference equation is also true:

$$x_m(k) = A_m x_m(k-1) + B_m u(k-1) + B_d \omega(k-1) \tag{3.29}$$

By defining $\Delta x_m(k) = x_m(k) - x_m(k-1)$ and $\Delta u(k) = u(k) - u(k-1)$, then subtracting (3.29) from (3.26) leads to

$$\Delta x_m(k+1) = A_m \Delta x_m(k) + B_m \Delta u(k) + B_d \epsilon(k) \tag{3.30}$$

In order to relate the output $y(k)$ to the state variable $\Delta x_m(k)$, we deduce that

$$\Delta y(k+1) = C_m \Delta x_m(k+1) = C_m A_m \Delta x_m(k) + C_m B_m \Delta u(k) + C_m B_d \epsilon(k)$$

where $\Delta y(k+1) = y(k+1) - y(k)$

Choosing a new state variable vector $x(k) = [\Delta x_m(k)^T \quad y(k)^T]^T$,we have

$$\begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} A_m & 0_m^T \\ C_m A_m & I_{q\times q} \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B_m \\ C_m B_m \end{bmatrix} \Delta u(k) + \begin{bmatrix} B_d \\ C_m B_d \end{bmatrix} \epsilon(k)$$

$$y(k) = \begin{bmatrix} 0_m & I_{q\times q} \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} \tag{3.31}$$

where $I_{q\times q}$ is the identity matrix with dimensions $q \times q$, which is the number of outputs; and 0_m is a $q \times n_1$ zero matrix. In (3.31),A_m, B_m and C_m have dimension $n_1 \times n_1, n_1 \times m$ and $q \times n_1$, respectively.

For notational simplicity, we denote (3.31) by

$$\begin{aligned} x(k+1) &= Ax(k) + B\Delta u(k) + B_\epsilon \epsilon(k) \\ y(k) &= Cx(k) \end{aligned} \tag{3.32}$$

where A, B and C are matrices corresponding to the forms given in (3.31). In the following, the dimensionality of the augmented state-space equation is taken to be $n(= n_1 + q)$.

3.5.2 Solution of Predictive Control for MIMO Systems

The extension of the predictive control solution is quite straightforward, and we need to pay attention to the dimensions of the state, control and output vectors in a multi-input, multi-output environment. Define the vectors Y and ΔU as

$$\Delta U = [\Delta u(k_i)^T \quad \Delta u(k_i+1)^T \quad \Delta u(k_i+2)^T \quad \dots \quad \Delta u(k_i+N_c-1)^T]^T$$

$$Y = [y(k_i+1|k_i)^T \quad y(k_i+2|k_i)^T \quad y(k_i+3|k_i)^T \quad \dots \quad y(k_i+N_p|k_i)^T]^T$$

Based on the state-space model (A, B, C), the future state variables are calculated sequentially using the set of future control parameters

$$\begin{aligned} x(k_i+1|k_i) &= Ax(k_i) + B\Delta u(k_i) + B_\epsilon \epsilon(k_i) \\ x(k_i+2|k_i) &= Ax(k_i+1|k_i) + B\Delta u(k_i+1) + B_\epsilon \epsilon(k_i+1|k_i) \\ &= A^2 x(k_i) + AB\Delta u(k_i) + B\Delta u(k_i+1) + AB_\epsilon \epsilon(k_i) \\ &\quad + B_\epsilon \epsilon(k_i+1|k_i) \\ &\vdots \\ x(k_i+N_p|k_i) &= A^{N_p} x(k_i) + A^{N_p-1} B\Delta u(k_i) + A^{N_p-2} B\Delta u(k_i+1) \\ &\quad + \cdots + A^{N_p-N_c} B\Delta u(k_i+N_c-1) + A^{N_p-1} B_\epsilon \epsilon(k_i) \\ &\quad + A^{N_p-2} B_\epsilon \epsilon(k_i+1|k_i) + \ldots + B_\epsilon \epsilon(k_i+N_p-1|k_i) \end{aligned}$$

With the assumption that $\epsilon(k)$ is a zero-mean white noise sequence, the predicted value of $\epsilon(k_i+i|k_i)$ at future sample i is assumed to be zero, hence, the noise effect to the predicted values being zero. For notational simplicity, the expectation operator is omitted without confusion.

Effectively, we have

$$Y = Fx(k_i) + \Phi\Delta U, \tag{3.33}$$

where

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix}; \Phi = \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & & & & \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix}$$

The incremental optimal control within one optimization window is given by

$$\Delta U = (\Phi^T\Phi + \bar{R})^{-1}(\Phi^T\bar{R}_s r(k_i) - \Phi^T F x(k_i)) \tag{3.34}$$

where matrix $\Phi^T\Phi$ has dimension $mN_c \times mN_c$ and $\Phi^T F$ has dimension $mN_c \times n$, and $\Phi^T\bar{R}_s$ equals the last q columns of $\Phi^T F$. The weight matrix $\bar{R}$ is a block matrix with m blocks and has its dimension equal to the dimension of $\Phi^T\Phi$.
The set-point signal is $r(k_i) = [r_1(k_i) \quad r_2(k_i) \dots r_q(k_i)]^T$ as the q set-point signals to the multi-output system.

Applying the receding horizon control principle, the first m elements in ΔU are taken to form the incremental optimal control:

$$\begin{aligned} \Delta u(k_i) &= \overbrace{[I_m \quad 0_m \cdots 0_m]}^{N_c}(\Phi^T\Phi + \bar{R})^{-1}(\Phi^T\bar{R}_s r(k_i) - \Phi^T F x(k_i)) \\ &= K_y r(k_i) - K_{mpc} x(k_i) \end{aligned} \tag{3.35}$$

where I_m and 0_m are, respectively, the identity and zero matrix with dimension $m \times m$.

3.6 State Estimation

In the design of model predictive controllers, we assumed that the information $x(k_i)$ is available at the time k_i . This assumes that all the state variables are measurable. In reality, with most applications, not all state variables are measured (or available). Some of them may be impossible to measure. One approach is to estimate the state variable $x(k)$ from the process measurement. The soft instrument used to estimate unknown state variables based on process measurement, in a control engineering context, is called an **observer**. The concept of an observer has been widely used in a noisy environment, a state observer can also act like a noise filter to reduce the effect of noise on the measurement. Our focus here is to use an observer in the design of predictive control.

3.6.1 Basic Ideas About an Observer

It would not be difficult to imagine that an observer is constructed based on a mathematical model of the plant. For instance, we assume the plant state-space model:

$$x_m(k+1) = A_m x_m(k) + B_m u(k). \tag{3.36}$$

Then we can use this model to calculate the state variable $\hat{x}_m(k), k = 1, 2, \ldots$ with an initial state condition $\hat{x}_m(0)$ and input signal $u(k)$ as

$$\hat{x}_m(k+1) = A_m\hat{x}_m(k) + B_m u(k) \tag{3.37}$$

This approach, in fact, would work after some transient time, if the plant model is stable and our guess of the initial condition is nearly correct. What could be the problems with this type of approach? Basically, it is an open-loop prediction. The error $\tilde{x}_m(k) = x_m(k) - \hat{x}_m(k)$ satisfies the difference equation:

$$\begin{aligned}\tilde{x}_m(k+1) =& A_m(x_m(k) - \hat{x}_m(k)) \\ =& A_m\tilde{x}_m(k).\end{aligned} \tag{3.38}$$

For a given initial error state $\tilde{x}_m(0) \neq 0$,we have

$$\tilde{x}_m(k) = A_m^k\tilde{x}_m(0). \tag{3.39}$$

If A_m has all eigenvalues inside the unit circle, then the error system (3.39) is stable and $||\tilde{x}_m(k)|| \to 0$ as $k \to \infty$, which means that the estimated state variable $\hat{x}_m(k)$ converges to $x_m(k)$. However, if A_m has one or more eigenvalues outside the unit circle, then the error system (3.39) is unstable and $||\tilde{x}_m(k)|| \to \infty$ as $k \to \infty$, which means that the prediction $\hat{x}_m(k)$ does not converge to $x_m(k)$. If A_m has one or more eigenvalues on the unit circle, the error states $||\tilde{x}_m(k)||$ will not converge to zero. The second point is that in the case of a stable plant model A_m, we have no control on the convergence rate of the error $||\tilde{x}_m(k)|| \to 0$, which is dependent on the location of the plant poles. Namely, if the plant poles are close to the origin of the complex plane, then the error converges at a fast rate to zero; otherwise, the convergence rate could be slow. The question is how to improve the estimate of $x_m(k)$. The solution is to use a feedback principle where an error signal is deployed to improve the estimation. The observer is constructed using the equation:

$$\hat{x}_m(k+1) = \overbrace{A_m\hat{x}_m(k) + B_m u(k)}^{model} + \overbrace{K_{ob}(y(k) - C_m\hat{x}_m(k))}^{correction \quad term} \tag{3.40}$$

where K_{ob} is the observer gain matrix. In the observer form, the state variable estimate $\hat{x}_m(k+1)$ consists of two terms. The first term is the original model, and the second term is the correction term based on the error between the measured output and the predicted output using the estimate $\hat{x}_m(k)$.

To choose the observer gain K_{ob}, we examine the closed-loop error equation. By substituting $y(k) = C_m x_m(k)$ into (3.40), with the definition of error state $\tilde{x}_m(k) = x_m(k) - \hat{x}_m(k)$, we obtain that

$$\begin{aligned}\tilde{x}_m(k+1) =& A_m\tilde{x}_m(k) - K_{ob}C_m\tilde{x}_m(k) \\ =& (A_m - K_{ob}C_m)\tilde{x}_m(k)\end{aligned} \tag{3.41}$$

Now, with given initial error $\tilde{x}_m(0)$, we have

$$\tilde{x}_m(k) = (A_m - K_{ob}C_m)^k\tilde{x}_m(0) \tag{3.42}$$

Comparing the observer error response given by (3.42) with the open-loop prediction (3.39), it is apparent that the observer gain K_{ob} can be used to manipulate the convergence rate of the error. If there is only a single output, a commonly used approach is to place the closed-loop eigenvalues of the error system matrix $A_m - K_{ob}C_m$ at a desired location of the complex plane. The following example shows how to select the observer gain K_{ob}.

Example 3.5. *The linearized equation of motion of a simple pendulum is*

$$\frac{d^2\theta}{dt^2} + \omega^2\theta = u \tag{3.43}$$

where θ is the angle of the pendulum. Design an observer that reconstructs the angle of the pendulum given measurements of $\frac{d\theta}{dt}$. Assume $\omega = 2rad/sec$ and sampling interval $\Delta t = 0.1(sec)$. The desired observer poles are chosen to be 0.1 and 0.2. Compare the open-loop estimation with the observer-based estimation.

Solution. *Let $x_1(t) = \theta$ and $x_2(t) = \dot{\theta}$, using the motion equation (3.43), the corresponding state-space model is*

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{bmatrix} = \begin{bmatrix}0 & 1\\ -\omega^2 & 0\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}u(t)$$
$$y(t) = \begin{bmatrix}0 & 1\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix} \tag{3.44}$$

With $\omega = 2rad/sec$ and sampling interval $\Delta t = 0.1(sec)$, the corresponding discrete-time state-space model is

$$\begin{bmatrix}x_1(k+1)\\ x_2(k+1)\end{bmatrix} = \begin{bmatrix}0.9801 & 0.0993\\ -0.3973 & 0.9801\end{bmatrix}\begin{bmatrix}x_1(k)\\ x_2(k)\end{bmatrix} + \begin{bmatrix}0.0050\\ 0.09930\end{bmatrix}u(k) \tag{3.45}$$
$$y(k) = \begin{bmatrix}0 & 1\end{bmatrix}\begin{bmatrix}x_1(k)\\ x_2(k)\end{bmatrix} \tag{3.46}$$

To begin this study, we investigate what happens if the pendulum model alone is used for the prediction of the angle $\theta(x_1)$. Assume that the input signal $u(k) = 0$ and the initial conditions of the state variables are $\theta(0) = x_1(0) = 1$ and $\dot{\theta}(0) = x_2(0) = 0$. The trajectories of movement for both θ and $\dot{\theta}$ are shown in Figure 3.6a. Both θ and $\dot{\theta}$ are sinusoidal signals. Now, suppose that we take a guess at the initial conditions of the state variables as $\hat{x}_1(0) = 0.3$ and $\hat{x}_2(0) = 0$. By using the state-space model (3.37), the estimates of θ and $\dot{\theta}$ are calculated and shown in comparison to the true trajectories (see Figure3.6a). It is seen that the estimate of θ, denoted $\hat{x}_1$, is not close to the true θ (see the top plot of Figure 3.6a). This study demonstrated that using the model alone is not sufficient to predict the angle of the pendulum. Let us design and implement an observer to predict the angle of the pendulum. Assume that the observer gain $K_{ob} = [j_1 \quad j_2]^T$. The closed-loop characteristic polynomial for the observer is

$$det(\lambda I - \begin{bmatrix}0.9801 & 0.0993 - j_1\\ -0.3973 & 0.9801 - j_2\end{bmatrix}) = (\lambda - 0.9801)(\lambda + j_2 - 0.9801)$$
$$- 0.3973 \times (j_1 - 0.0993),$$

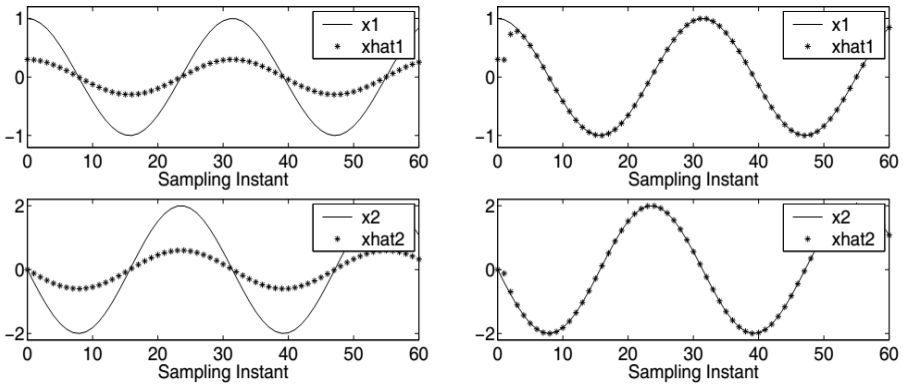
which is made to be equal to the desired closed-loop characteristic polynomial $(\lambda-0.1)(\lambda-0.2)$. *Namely,*

$$(\lambda - 0.9801)(\lambda + j_2 - 0.9801) - 0.3973 \times (j_1 - 0.0993) = (\lambda - 0.1)(\lambda - 0.2) \tag{3.47}$$

Solving equation (3.47) *gives us the observer gain as* $j_1 = -1.6284$ *and* $j_2 = 1.6601$. *The estimation of angle is carried out using the observer equation:*

$$\begin{bmatrix}\hat{x}_1(k+1)\\ \hat{x}_2(k+1)\end{bmatrix} = \begin{bmatrix}0.9801 & 0.0993\\ -0.3973 & 0.9801\end{bmatrix}\begin{bmatrix}\hat{x}_1(k)\\ \hat{x}_2(k)\end{bmatrix} + K_{ob}(x_2(k) - \hat{x}_2(k)) \tag{3.48}$$

with initial condition $\hat{x}_1(0) = 0.3$ *and* $\hat{x}_2(0) = 0$. *Figure 3.6b shows that the estimated angle converges to the true angle in about three steps.*



(a) Estimation without observer (b) Estimation with observer

Figure 3.6: Observer design and implementation

3.7 State Estimate Predictive Control

In the implementation of predictive control, an observer is used for the cases where the state variable $x(k_i)$ at time k_i is not measurable. Essentially, the state variable $x(k_i)$ is estimated via an observer of the form:

$$\hat{x}(k_i + 1) = A\hat{x}(k_i) + B\Delta u(k_i) + K_{ob}(y(k_i) - C\hat{x}(k_i)). \tag{3.49}$$

Note that in the implementation of predictive control using an observer, the control signal is $\Delta u(k_i)$ and the matrices (A, B, C) come from the augmented model used for the predictive control design. With the information of $\hat{x}(k_i)$ replacing $x(k_i)$, the predictive control law is then modified to find ΔU by minimizing

$$\begin{aligned} J =&(R_s - F\hat{x}(k_i))^T(\bar{R}_s r(k_i) - F\hat{x}(k_i)) - 2\Delta U^T\Phi^T(R_s - F\hat{x}(k_i))\\ &+ \Delta U^T(\Phi^T\Phi + \bar{R})\Delta U, \end{aligned} \tag{3.50}$$

where $\bar{R}_s, F, \Phi, \bar{R}$ and ΔU were defined in (3.33) and (3.34)

The optimal solution ΔU is obtained as,

$$\Delta U = (\Phi^T \Phi + \bar{R})^{-1} \Phi^T (R_s - F\hat{x}(k_i)) \tag{3.51}$$

Furthermore, application of the receding horizon control principle leads to the optimal solution of $\Delta u(k_i)$ at time k_i:

$$\Delta u(k_i) = K_y r(k_i) - K_{mpc}\hat{x}(k_i), \tag{3.52}$$

which is a standard state feedback control law with estimated $x(k_i)$. The closed-loop state feedback structure is illustrated in Figure 3.7. What about the closed-loop characteristic

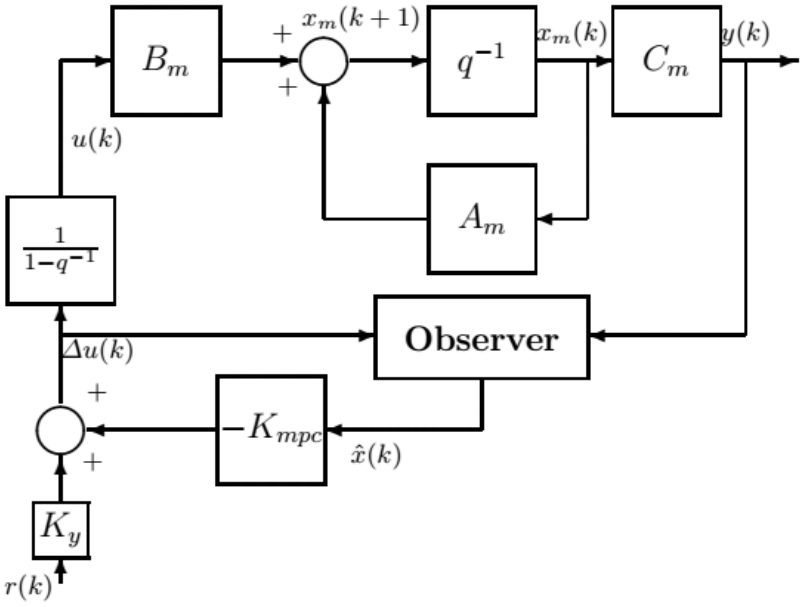


Figure 3.7: Block diagram of DMPC with observer

equation, and hence the eigenvalues of the closed-loop system? To investigate these issues, we form the closed-loop state-space equation as below:

$$\begin{aligned} x(k+1) =& Ax(k) + B\Delta u(k) \\ =& Ax(k) + BK_y r(k) - BK_{mpc}\hat{x}(k), \end{aligned} \tag{3.53}$$

where we substituted $\Delta u(k)$ with (3.52). Note that the closed-loop observer error equation is:

$$\tilde{x}(k+1) = (A - K_{ob}C)\tilde{x}(k), \tag{3.54}$$

where $\tilde{x}(k) = x(k) - \hat{x}(k)$. Replacing $\hat{x}(k)$ by $x(k) - \tilde{x}(k)$, (3.53) is rewritten as,

$$x(k+1) = (A - BK_{mpc})x(k) + BK_{mpc}\tilde{x}(k) + BK_y r(k). \tag{3.55}$$

Combination of (3.54) with (3.55) leads to:

$$\begin{bmatrix} \tilde{x}(k+1) \\ x(k+1) \end{bmatrix} = \begin{bmatrix} A - K_{ob}C & 0_{n\times n} \\ BK_{mpc} & A - BK_{mpc} \end{bmatrix} \begin{bmatrix} \tilde{x}(k) \\ x(k) \end{bmatrix} + \begin{bmatrix} 0_{n\times q} \\ BK_y \end{bmatrix} r(k) \tag{3.56}$$

where $0_{n\times n}$ is a $n \times n$ zero matrix and $0_{n\times q}$ is a $n \times q$ zero matrix. The characteristic equation of the closed-loop state-space system is determined by

$$det\left[\lambda I - \begin{bmatrix} A - K_{ob}C & 0_{n\times n} \\ BK_{mpc} & A - BK_{mpc} \end{bmatrix}\right] = 0$$

which is equivalent to

$$det(\lambda I - (A - K_{ob}C))det(\lambda I - (A - BK_{mpc})) = 0$$

because the system matrix in (3.56) has a lower block triangular structure. This effectively means that the closed-loop model predictive control system with state estimate has two independent characteristic equations

$$det(\lambda I - (A - K_{ob}C)) = 0 \tag{3.57}$$

$$det(\lambda I - (A - BK_{mpc})) = 0 \tag{3.58}$$

Since the closed-loop eigenvalues are the solutions of the characteristic equations, (3.57) and (3.58) indicate that the set of eigenvalues of the combined closed-loop system consists of predictive control-loop eigenvalues and observer-loop eigenvalues. This means that the design of the predictive control law and the observer can be carried out independently (or separately), yet when they are put together in this way, the eigenvalues remain unchanged.

Chapter 4

Discrete-Time MPC With Constraints

4.1 Introduction

In practice all processes are subject to restrictions. The actuators have a limited field of action as well as a determined Incremental Variation, as is the case of the valves, limited by the positions of totally open or closed and by the response rate. Constructive reasons, safety or environmental ones or even the sensor scopes themselves can cause limits in the process variables such as levels in tanks, flows in piping or maximum temperatures and pressures; moreover, the operational conditions are normally defined by the intersection of certain constraints for basically economic motives, so that the control system will operate close to the boundaries. All of these makes the introduction of constraints in the function to be minimized necessary. Many predictive algorithms intrinsically take in to account constraints and have therefore been very successful in industry, whilst others can incorporate them a posteriori. Normally bounds in the amplitude and in the Incremental Variation of the control signal and limits in the output/state will be considered:

$$\begin{aligned} u^{min} &\leq u(k) \leq u^{max} \quad \forall k. \\ \Delta u^{min} &\leq \Delta u(k) \leq \Delta u^{max} \quad \forall k. \\ y^{min} &\leq y(k) \leq y^{max} \quad \forall k. \\ x^{min} &\leq x(k) \leq x^{max} \quad \forall k. \end{aligned}$$

by adding these constraints to the objective function the minimization becomes more complex, so that the solution cannot be obtained explicitly as in the unrestricted case.

4.2 Hard and Soft Constraints

In practical problems it is common to find that the desirable constraints are inconsistent; that is, they cannot all be satisfied simultaneously. In the case that constraints do not admit a solution, then the MPC optimisation is ill posed and has no solution. Clearly one cannot

afford for this to happen on a real process as the resulting control would be arbitrary. Each process will build in its own fail safes for such an occurrence but a more generic procedure does exist.
Constraints should be placed into two categories:[18]

1. hard constraints, which cannot be violated, for example, the maximum flow in a pipe or valve,

2. soft constraints, these can be violated though at some penalty such as loss of product quality and safety values blowing. The soft limits should then be further prioritised into a ranking of importance. In the event that constraints cannot be satisfied, then one could gradually relax the soft constraints in the given priority order until a feasible solution exists.

4.3 Formulation of Constrained Control Problems

To this end, we need to formulate the predictive control problem as an optimization problem that takes into account the constraints present. This section discusses the operational constraints that are frequently encountered in the design of control systems. These operational constraints are presented as linear inequalities of the control and plant variables.

4.3.1 Frequently Used Operational Constraints

There are three major types of constraints frequently encountered in applications. The first two types deal with constraints imposed on the control variables $u(k)$, and the third type of constraint deals with output $y(k)$ or state variable $x(k)$ constraints. For clarity, we will discuss single-input, single output systems first and subsequently extend the cases to multi-input, multi-output systems.

Constraints on the Control Variable Incremental Variation

These are hard constraints on the size of the control signal movements, i.e.,on the rate of change of the control variables ($u(k)$). Suppose that for a single-input system, the upper limit is Δu^{max} and the lower limit is Δu^{min}.
The constraints are specified in the form

$$\Delta u^{min} \leq \Delta u(k) \leq \Delta u^{max} \tag{4.1}$$

Note that we use less than or equal to in (4.1), where the equality will play the critical role in the solution of the constrained control problem.
The rate of change constraints can be used to impose directional movement constraints on the control variables; for instance, if $u(k)$ can only increase, not decrease, then possibly,

we select $0 \leq \Delta u(k) \leq \Delta u^{max}$. The constraint on $\Delta u(k)$ can be used to cope with the cases where the rate of change of the control amplitude is restricted or limited in value. For example, in a control system implementation, assuming that the control variable $u(k)$ is only permitted to increase or decrease in a magnitude less 0.1 unit, then the operational constraint is

$$-0.1 \leq \Delta u(k) \leq 0.1$$

Constraints on the Amplitude of the Control Variable

These are the most commonly encountered constraints among all constraint types. For instance, we cannot expect a valve to open more than 100 percent nor a voltage to go beyond a given range. These are the physical hard constraints on the system. Simply, we demand that

$$u^{min} \leq u(k) \leq u^{max}$$

Here, we need to pay particular attention to the fact that $u(k)$ is an incremental variable, not the actual physical variable. The actual physical control variable equals the incremental variable u plus its steady-state value u_{ss}. A common mistake is to mix these two. For instance, if a valve is allowed to open in the range between 15% and 80% and the valves normal operating value is at 30%, then $u^{min} = 15\% - 30\% = -15\%$ and $u^{max} = 80\% - 30\% = 50\%$

Output Constraints

We can also specify the operating range for the plant output. For instance, supposing that the output $y(k)$ has an upper limit y^{max} and a lower limit y^{min}, then the output constraints are specified as

$$y^{min} \leq y(k) \leq y^{max} \tag{4.2}$$

Output constraints are often implemented as soft constraints in the way that a slack variable $s_v > 0$ is added to the constraints, forming

$$y^{min} - s_v \leq y(k) \leq y^{max} + s_v \tag{4.3}$$

There is a primary reason why we use a slack variable to form soft constraints for output. Output constraints often cause large changes in both the control and incremental control variables when they are enforced (we term them become active in the later sections). When that happens, the control or incremental control variables can violate their own constraints and the problem of constraint conflict occurs. In the situations where the constraints on the control variables are more essential to plant operation, the output constraints are often relaxed by selecting a larger slack variable s_v to resolve the conflict problem.

Similarly, we can impose constraints on the state variables if they are measurable or impose the constraints on observer state variables. They also need to be in the form of soft constraints for the same reasons as the output case above.

Constraints in a Multi-input and Multi-output Setting

If there is more than one input, then the constraints are specified for each input independently. In the multi-input case, suppose that the constraints are given for the upper limits as

$$[\Delta u_1^{max} \quad \Delta u_2^{max} \dots \Delta u_m^{max}]$$

and lower limits as

$$[\Delta u_1^{min} \quad \Delta u_2^{min} \dots \Delta u_m^{min}]$$

Each variable with rate of change is specified as

$$\begin{aligned} \Delta u_1^{min} \leq & \Delta u_1(k) \leq \Delta u_1^{max} \\ \Delta u_2^{min} \leq & \Delta u_2(k) \leq \Delta u_2^{max} \\ & \vdots \\ \Delta u_m^{min} \leq & \Delta u_m(k) \leq \Delta u_m^{max} \end{aligned} \tag{4.4}$$

Similarly, suppose that the constraints are given for the upper limit of the control signal as

$$[u_1^{max} \quad u_2^{max} \dots u_m^{max}]$$

and lower limit as

$$[u_1^{min} \quad u_2^{min} \dots u_m^{min}]$$

Then, the amplitude of each control signal is required to satisfy the constraints:

$$\begin{aligned} u_1^{min} \leq & u_1(k) \leq u_1^{max} \\ u_2^{min} \leq & u_2(k) \leq u_2^{max} \\ & \vdots \\ u_m^{min} \leq & u_m(k) \leq u_m^{max} \end{aligned} \tag{4.5}$$

Similarly, constraints are specified for each output and state variable if they are required. In short, the constraints for a multi-input and multi-output system are specified for each input and output independently .

4.3.2 Constraints as Part of the Optimal Solution

Having formulated the constraints as part of the design requirements, the next step is to translate them into linear inequalities, and relate them to the original model predictive control problem. The key here is to parameterize the constrained variables using the same parameter vector ΔU as the ones used in the design of predictive control. Therefore, the constraints are expressed in a set of linear equations based on the parameter vector ΔU . The vector ΔU is often called the decision variable in optimization literature. Since the predictive control problem is formulated and solved in the framework of receding horizon

control, the constraints are taken into consideration for each moving horizon window. This allows us to vary the constraints at the beginning of each optimization window, and also gives us the means to tackle the constrained control problem numerically. Based on this idea, if we want to impose the constraints on the rate of change of the control signal $\Delta u(k)$ at time k_i , the constraints at sample time k_i are expressed as

$$\Delta u^{min} \leq \Delta u(k_i) \leq \Delta u^{max}$$

From the time instance k_i , the predictive control scheme looks into the future. The constraints at future samples, for example on the first three samples, $\Delta u(k_i)$, $\Delta u(k_i + 1)$, $\Delta u(k_i + 2)$ are imposed as

$$\begin{aligned}\Delta u^{min} \leq &\Delta u(k_i) \leq \Delta u^{max}\\ \Delta u^{min} \leq &\Delta u(k_i + 1) \leq \Delta u^{max}\\ \Delta u^{min} \leq &\Delta u(k_i + 2) \leq \Delta u^{max}\end{aligned}$$

In principle, all the constraints are defined within the prediction horizon. However, in order to reduce the computational load, we sometimes choose a smaller set of sampling instants at which to impose the constraints, instead of all the future samples. The following example shows how to express the constraints from the design specification in terms of a function of ΔU.

Example 4.1. *In the motor control system, suppose that the input voltage variation is limited to $2V$ and $6V$. The steady state of the control signal is at $4V$. Assuming that the control horizon is selected to be $N_c = 4$, express the constraint on $\Delta u(k_i)$ and $\Delta u(k_i + 1)$ in terms of ΔU for the first two sample times.*

Solution. *The parameter vector to be optimized in the predictive control system at time k_i is $\Delta U = [\Delta u(k_i) \quad \Delta u(k_i + 1) \quad \Delta u(k_i + 2) \quad \Delta u(k_i + 3)]^T$*

Note that

$$\begin{aligned}u(k_i) =&u(k_i - 1) + \Delta u(k_i) = u(k_i - 1) + [1 \quad 0 \quad 0 \quad 0]\Delta U &(4.6)\\ u(k_i + 1) =&u(k_i) + \Delta u(k_i + 1) = u(k_i - 1) + \Delta u(k_i) + \Delta u(k_i + 1)\\ =&u(k_i - 1) + [1 \quad 1 \quad 0 \quad 0]\Delta U &(4.7)\end{aligned}$$

With the limits on the control variables, by subtracting the steady-state value of the control, as $u^{min} = 2 - 4 = -2$ and $u^{max} = 6 - 4 = 2$, the constraints are expressed as

$$\begin{bmatrix}-2\\-2\end{bmatrix} \leq \begin{bmatrix}1\\1\end{bmatrix} u(k_i - 1) + \begin{bmatrix}1 & 0 & 0 & 0\\ 1 & 1 & 0 & 0\end{bmatrix} \Delta U \leq \begin{bmatrix}2\\2\end{bmatrix} \tag{4.8}$$

Now, since we have expressed the constraints as the inequalities which is linear in-the-parameter ΔU, the next step is to combine the constraints with the original cost function J used in the design of predictive control. As the optimal solutions will be obtained using quadratic programming, the constraints need to be decomposed into two parts to reflect the lower limit, and the upper limit with opposite sign. Namely, for instance, the constraints

$$\Delta U^{min} \leq \Delta U \leq \Delta U^{max}$$

will be expressed by two inequalities:

$$-\Delta U \leq -\Delta U^{min} \tag{4.9}$$
$$\Delta U \leq \Delta U^{max} \tag{4.10}$$

In a matrix form, this becomes

$$\begin{bmatrix} -I \\ I \end{bmatrix} \Delta U \leq \begin{bmatrix} -\Delta U^{min} \\ \Delta U^{max} \end{bmatrix} \tag{4.11}$$

This procedure applies to all the constraints mentioned in this section, including control and output constraints.

Traditionally, the constraints are imposed for all future sampling instants, and all constraints are expressed in terms of the parameter vector ΔU. In the case of a manipulated variable constraint, we write:

$$\begin{bmatrix} u(k_i) \\ u(k_i+1) \\ u(k_i+2) \\ \vdots \\ u(k_i+N_c-1) \end{bmatrix} = \begin{bmatrix} I \\ I \\ I \\ \vdots \\ I \end{bmatrix} u(k_i-1) + \begin{bmatrix} I & 0 & 0 & \dots & 0 \\ I & I & 0 & \dots & 0 \\ I & I & I & \dots & 0 \\ \vdots & & & & \\ I & I & \dots & I & I \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \\ \vdots \\ \Delta u(k_i+N_c-1) \end{bmatrix} \tag{4.12}$$

Re-writing (4.12) in a compact matrix form, with

$$C_1 = \begin{bmatrix} I \\ I \\ I \\ \vdots \\ I \end{bmatrix}$$

and

$$C_2 = \begin{bmatrix} I & 0 & 0 & \dots & 0 \\ I & I & 0 & \dots & 0 \\ I & I & I & \dots & 0 \\ \vdots & & & & \\ I & I & \dots & I & I \end{bmatrix}$$

corresponding to the appropriate matrices, then the constraints for the control movement are imposed as,

$$-(C_1 u(k_i-1) + C_2 \Delta U) \leq -U^{min} \tag{4.13}$$
$$(C_1 u(k_i-1) + C_2 \Delta U) \leq U^{max} \tag{4.14}$$

where U^{min} and U^{max} are column vectors with N_c elements of u^{min} and u^{max}, respectively. Similarly, for the increment of the control signal, we have the constraints:

$$-\Delta U \leq -\Delta U^{min} \tag{4.15}$$
$$\Delta U \leq \Delta U^{max} \tag{4.16}$$

where ΔU^{min} and ΔU^{max} are column vectors with N_c elements of Δu^{min} and Δu^{max}, respectively. The output constraints are expressed in terms of ΔU:

$$Y^{min} \leq Fx(k_i) + \Phi \Delta U \leq Y^{max} \tag{4.17}$$

Finally, the model predictive control in the presence of hard constraints is proposed as finding the parameter vector ΔU that minimizes

$$\begin{aligned} J = &(R_s - Fx(k_i))^T (R_s - Fx(k_i)) - 2\Delta U^T \Phi^T (R_s - Fx(k_i)) \\ &+ \Delta U^T (\Phi^T \Phi + \bar{R}) \Delta U \end{aligned} \tag{4.18}$$

subject to the inequality constraints

$$\begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \Delta U \leq \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix} \tag{4.19}$$

where the data matrices are

$$M_1 = \begin{bmatrix} -C_2 \\ C_2 \end{bmatrix}; N_1 = \begin{bmatrix} -U^{min} + C_1 u(k_i - 1) \\ U^{max} - C_1 u(k_i - 1) \end{bmatrix}; M_2 = \begin{bmatrix} -I \\ I \end{bmatrix}; N_2 = \begin{bmatrix} -\Delta U^{min} \\ \Delta U^{max} \end{bmatrix};$$
$$M_3 = \begin{bmatrix} -\Phi \\ \Phi \end{bmatrix}; \quad N_3 = \begin{bmatrix} -Y^{min} + Fx(k_i) \\ Y^{max} - Fx(k_i) \end{bmatrix}$$

The matrix $\Phi^T \Phi + \bar{R}$ is the Hessian matrix and is assumed to be **positive definite.** Since the cost function J is a quadratic, and the constraints are linear inequalities, the problem of finding an optimal predictive control becomes one of finding an optimal solution to a standard quadratic programming problem. For compactness of expression, we denote (4.19) by

$$M \Delta U \leq \gamma \tag{4.20}$$

where M is a matrix reflecting the constraints, with its number of rows equal to the number of constraints and number of columns equal to the dimension of ΔU . When the constraints are fully imposed, M is of full rank matrix, the number of constraints is equal to $4 \times m \times N_c + 2 \times q \times N_p$, where m is the number of inputs and q is the number of outputs. The total number of constraints is, in general, greater than the dimension of the decision variable ΔU. Because the receding horizon control law implements the first control movement and ignores the rest of the calculated future control signals, a question naturally arises as to whether it is necessary to impose constraints on all future trajectories of both the control signals and system output.

4.4 Solution Methods Using Quadratic Programming

As was indicated in the previous section, the implementation of Model Predictive Controllers for processes with bounded signals requires the solution of a quadratic programming (QP)

problem; that is, an optimization problem with a quadratic objective function and linear constraints. This section is dedicated to revising some of the main QP techniques. It is not intended to be an exhaustive description of all QP methods. The required numerical solution for MPC is often viewed as an obstacle in the application of MPC. However, what we can do here is to understand the essence of quadratic programming so that we can produce the essential computational programs required. To be consistent with the literatures of quadratic programming, the decision variable is denoted by x.
In general, a quadratic programming optimization problem minimizes a multivariable quadratic function, which is subject to linear constraints on the variables. Let us assume x is in general a vector containing the optimization variables, while E is a symmetric and positive definite matrix and F is a vector. The objective function J and the constraints are expressed as [3]

$$J = \frac{1}{2}x^T Ex + x^T F \tag{4.21}$$

$$Mx \leq \gamma \tag{4.22}$$

where E, F, M and γ are compatible matrices and vectors in the quadratic programming problem. If E is a positive semidefinite matrix, then the function J is convex and it has a global minimizer, if there exists a feasible vector x. Feasibility means that the variable x satisfies all constraints. Given a positive definite E and a feasible x the global minimizer of the QP is unique.

4.4.1 Quadratic Programming for Equality Constraints

The simplest problem of quadratic programming is to find the constrained minimum of a positive definite quadratic function with linear equality constraints.
We consider a general approach to the constrained optimization with equality constraints.

Lagrange Multipliers

To minimize the objective function subject to equality constraints, let us consider the so-called Lagrange expression

$$L = \frac{1}{2}x^T Ex + x^T F + \lambda^T (Mx - \gamma) \tag{4.23}$$

$L(\cdot, \lambda)$ is convex function and the value of (4.23) subject to the equality constraints $Mx = \gamma$ being satisfied is the same as the original objective function. We now consider (4.23) as an objective function in $n + m$ variables x and λ, where n is the dimension of x and m is the dimension of λ. The procedure of minimization is to take the first partial derivatives with respect to the vectors x and λ, and then equate these derivatives to zero. This gives us the results

$$\frac{\partial L}{\partial x} = Ex + F + M^T \lambda = 0 \tag{4.24}$$

$$\frac{\partial L}{\partial \lambda} = Mx - \gamma = 0 \tag{4.25}$$

The linear equations (4.24) together with (4.25) contain $n + m$ variables x and λ, which are the necessary conditions for minimizing the objective function with equality constraints and also sufficient condition due to convexity of the objective function. The elements of the vector λ are called Lagrange multipliers.

The minimization of the Lagrange expression is straightforward. If M has full rank and E is positive definite, the optimal λ and x are found via the set of linear equations defined by (4.24) and (4.25) where

$$\lambda = -(ME^{-1}M^T)^{-1}(\gamma + ME^{-1}F) \tag{4.26}$$

$$x = -E^{-1}(M^T\lambda + F) \tag{4.27}$$

It is interesting to note that (4.27) can be written as two terms:

$$x = -E^{-1}F - E^{-1}M^T\lambda = x^0 - E^{-1}M^T\lambda$$

where the first term $x^0 = -E^{-1}F$ is the global optimal solution that will give a minimum of the original cost function J without constraints, and the second term is a correction term due to the equality constraints
The following example is used to illustrate the minimization with equality constraints.

Example 4.2. *Minimize*

$$J = \frac{1}{2}x^T Ex + x^T F$$

subject to

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ 3x_1 - 2x_2 - 3x_3 &= 1 \end{aligned} \tag{4.28}$$

where $E = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; F = \begin{bmatrix} -2 \\ -3 \\ -1 \end{bmatrix}$

Solution. *Without the equality constraints, the optimal solution is*

$$x^0 = -E^{-1}F = [2 \quad 3 \quad 1]^T$$

Writing the two equality constraints given by (4.28) in matrix form, we obtain the M *and* γ *matrices as*

$$M = \begin{bmatrix} 1 & 1 & 1 \\ 3 & -2 & -3 \end{bmatrix}; \gamma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

To use (4.26) the following quantities are required

$$ME^{-1}M^T = \begin{bmatrix} 3 & -2 \\ -2 & 22 \end{bmatrix}; ME^{-1}F = \begin{bmatrix} -6 \\ 3 \end{bmatrix}$$

Note that the determinant $det(ME^{-1}M^T) = 62$, *thus the matrix* $ME^{-1}M^T$ *is invertible. The* λ *vector is*

$$\lambda = -(ME^{-1}M^T)^{-1}(\gamma + ME^{-1}F) = \begin{bmatrix} 1.6452 \\ -0.0323 \end{bmatrix}$$

The x *vector that minimizes the objective function is*

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = x^0 - E^{-1}M^T\lambda = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 & 3 \\ 1 & -2 \\ 1 & -3 \end{bmatrix} \begin{bmatrix} 1.6452 \\ -0.0323 \end{bmatrix} = \begin{bmatrix} 0.4516 \\ 1.2903 \\ -0.7419 \end{bmatrix}$$

Example 4.3. *In this example, we examine what happens to the constrained optimal solution when the linear constraints are dependent. We assume that the objective function*

$$J = \frac{1}{2}x^T Ex + x^T F$$

where the matrices $E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; F = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$ *and the constraints are*

$$\begin{aligned} x_1 + x_2 &= 1 \\ 2x_1 + 2x_2 &= 6 \end{aligned} \tag{4.29}$$

Use graphs to demonstrate that there is no feasible solution of x_1 *and* x_2 *that will satisfy the equality constraints (4.29). In addition, demonstrate that the matrix* $M^T E^{-1} M$ *is not invertible.*

Solution. *The figure below shows that the two equality constraints are defined by two parallel lines on the* (x_1, x_2) *plane. Because the lines do not intersect, there does not exist a feasible set of parameters* x_1 *and* x_2 *that will simultaneously satisfy both linear equations. We can also examine what happens to the Lagrange multipliers when there is no feasible solution. The* M *and* γ *matrices are*

$$M = \begin{bmatrix} 1 & 1 \\ 2 & 2 \end{bmatrix}; \gamma = \begin{bmatrix} 1 \\ 6 \end{bmatrix}$$

Note that because of the linear dependency in the constraints, the matrix $ME^{-1}M^T = \begin{bmatrix} 2 & 4 \\ 4 & 8 \end{bmatrix}$ *has a zero determinant, and does not have an inverse.*

Fig 4.1.Illustration of no feasible solution of the constrained optimization problem. Solid-line $x_1 + x_2 = 1$*; darker-solid-line* $2x_1 + 2x_2 = 6$

In summary, in order to find the optimal constrained solution, the linear equality constraints are required to be linearly independent.

Example 4.4. *In this example, we will show how the number of equality constraints is also an issue in the constrained optimal solution. Once again, we use the same objective function as in Example 4.2 where* $E = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; F = \begin{bmatrix} -2 \\ -3 \\ -1 \end{bmatrix}$*, but add an extra constraint to the*

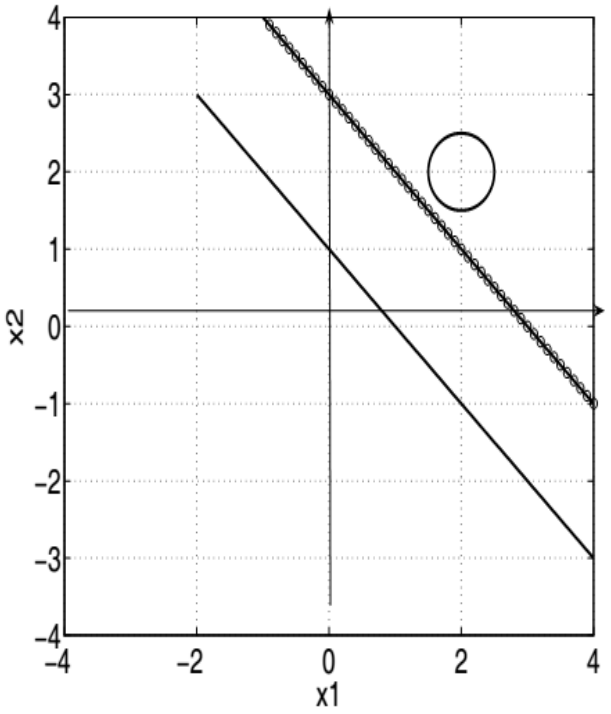


Figure 4.1:

original constraints so that

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ 3x_1 - 2x_2 - 3x_3 &= 1 \\ x_1 - 3x_2 + 2x_3 &= 1 \end{aligned} \tag{4.30}$$

Solution. *The additional constraint is independent of the first two original constraints. Now the only feasible solution that satisfies all the constraints is* $x = M^{-1}\gamma = [0.6552 \quad 0.069 \quad 0.2759]^T$, *which is the unique solution of the linear equations (4.30). There is no point in proceeding to the optimization of the objective function, because the only feasible solution is the constrained optimal solution.*

In summary, the number of equality constraints is required to be less than or equal to the number of decision variables (i.e., x). If the number of equality constraints equals the number of decision variables, the only feasible solution is the one that satisfies the constraints and there is no additional variable in x that can be used to optimize the original objective function. If the number of equality constraints is greater than the number of decision variables, then there is no feasible solution to satisfy the constraints. Alternatively, the situation is called infeasible.

4.4.2 Minimization with Inequality Constraints

In the minimization with inequality constraints, the number of constraints could be larger than the number of decision variables. The inequality constraints $Mx \leq \gamma$ as in (4.22) may comprise active constraints and inactive constraints. An inequality $M_i x \leq \gamma_i$ is said to be active if $M_i x = \gamma_i$ and inactive if $M_i x < \gamma_i$, where M_i together with γ_i form the ith inequality constraint and are the ith row of M matrix and the ith element of γ vector, respectively. We introduce the Kuhn-Tucker conditions, which define the active and inactive constraints in terms of the Lagrange multipliers.

Kuhn-Tucker Conditions

The necessary conditions for this optimization problem (Kuhn-Tucker conditions) are

$$\begin{aligned} Ex + F + M^T\lambda &= 0 \\ Mx - \gamma &\leq 0 \\ \lambda^T(Mx - \gamma) &= 0 \\ \lambda &\geq 0 \end{aligned} \tag{4.31}$$

where the vector λ contains the Lagrange multipliers. These conditions can be expressed in a simpler form in terms of the set of active constraints. Let S_{act} denote the index set of active constraints. Then the necessary conditions become

$$Ex + F + \sum_{i \in S_{act}} \lambda_i M_i^T = 0$$

$$M_i x - \gamma_i = 0 \quad i \in S_{act} \tag{4.32}$$

$$M_i x - \gamma_i < 0 \quad i \notin S_{act} \tag{4.33}$$

$$\lambda_i \geq 0 \quad i \in S_{act} \tag{4.34}$$

$$\lambda_i = 0 \quad i \notin S_{act} \tag{4.35}$$

where M_i is the ith row of the M matrix. In other words, (4.32) says that for the ith row, $M_i x - \gamma_i = 0$ means that this is an equality constraint, hence an active constraint. In contrast, $M_i x - \gamma_i < 0$ (see (4.33)) means that the constraint is satisfied, hence it is an inactive constraint. For an active constraint, the corresponding Lagrange multiplier is non-negative (see (4.34)), whilst the Lagrange multiplier is zero if the constraint is inactive (see (4.35)).

If the active set were known, the original problem could be replaced by the corresponding problem having equality constraints only.
Explicitly, supposing that M_{act} and λ_{act} are given, the optimal solution with inequality solution has the closed-form

$$\lambda_{act} = -(M_{act}E^{-1}M_{act}^T)^{-1}(\gamma_{act} + M_{act}E^{-1}F) \tag{4.36}$$

$$x = -E^{-1}(F + M_{act}^T\lambda_{act}) \tag{4.37}$$

Example 4.5. *In example(4.3) we showed that when optimizing with equality constraints, if the constraints are dependent, then there is no feasible solution. In this example, we examine what happens to the optimal solution if they are inequality constraints. We assume that the objective function*

$$J = \frac{1}{2}x^T Ex + x^T F \tag{4.38}$$

where the matrices $E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, F = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$ *and the constraints are*

$$x_1 + x_2 \leq 1 \tag{4.39}$$

$$2x_1 + 2x_2 \leq 6 \tag{4.40}$$

Solution. *Clearly the set of variables that satisfy inequality (4.39) will also satisfy the inequality (4.40). Thus, the constraint (4.39) is an active constraint, while the constraint (4.40) is an inactive constraint. We find the constrained optimum by minimizing J subject to equality constraint:* $x_1 + x_2 = 1$*,which is* $x_1 = 0.5$ *and* $x_2 = 0.5$*. We verify that indeed with this set of* x_1 *and* x_2 *values, inequality (4.40) is satisfied.*

Active Set Methods

The idea of active set methods is to define at each step of an algorithm a set of constraints, termed the working set, that is to be treated as the active set. The working set is chosen to be a subset of the constraints that are actually active at the current point, and hence the current point is feasible for the working set. The algorithm then proceeds to move on the surface defined by the working set of constraints to an improved point. At each step of the active set method, an equality constraint problem is solved. If all the Lagrange multipliers $\lambda_i \geq 0$, then the point is a local solution to the original problem. If, on the other hand, there exists a $\lambda_i < 0$, then the objective function value can be decreased by relaxing the i^{th} constraint (i.e., deleting it from the constraint equation). Thus, the i^{th} constraint is removed from the active set .(If more than one element of λ is negative, then the constraint corresponding to the most negative one is removed from the active set.) So now a new active set has been selected, and the whole process is repeated. During the course of minimization, it is necessary to monitor the values of the other constraints to be sure that they are not violated, since all points defined by the algorithm must be feasible. It often happens that while moving on the working surface, a new constraint boundary is encountered. It is necessary to add this constraint to the working set, then proceed to the re-defined working surface. If there are q constraints in the problem formulation (M has q rows) then there are 2^q possible sets of active constraints. Analysis of specific problems can often reduce the size of this set to some extent. For example , lower and upper limits on variable values cannot be active simultaneous.

Theorem 4.4.1. *Suppose that for every subset W of the constraint indices, the constrained problem*

$$\begin{aligned} &\textit{minimize } J(x) \\ &\textit{subject to } M_i x = \gamma_i \end{aligned}$$

is well-defined with a unique nondegenerate solution (that is, for all $i \in W, \quad \lambda_i \neq 0$*). Then the sequence of points generated by the basic active set strategy converges to the solution of the inequality constrained problem* (4.21)-(4.22)*.*

Proof. After the solution corresponding to one working set is found, a decrease in the objective is made, and hence it is not possible to return to that working set. Since there are only a finite number of working sets, the process must terminate[10]. □

To illustrate the basic idea of active set methods, let us look at the example below.

Example 4.6. *Optimize the objective function where*

$$E = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; F = \begin{bmatrix} -2 \\ -3 \\ -1 \end{bmatrix}$$

subject to the inequality constraints:

$$\begin{aligned} x_1 + x_2 + x_3 &\leq 1 \\ 3x_1 - 2x_2 - 3x_3 &\leq 1 \\ x_1 - 3x_2 + 2x_3 &\leq 1 \end{aligned} \tag{4.41}$$

Solution. *The feasible solution of equality constraints (4.41) exists, which is the solution of the linear equations*

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ 3x_1 - 2x_2 - 3x_3 &= 1 \\ x_1 - 3x_2 + 2x_3 &= 1 \end{aligned} \tag{4.42}$$

Thus, the three equality constraints are taken as the first working set. We calculate the Lagrange multiplier for the three constraints leading to

$$\lambda = -(ME^{-1}M^T)^{-1}(\gamma + ME^{-1}F) = \begin{bmatrix} 1.6873 \\ 0.0309 \\ -0.4352 \end{bmatrix}$$

the third element in λ *is negative, therefore, the third constraint is an inactive constraint and will be dropped from the constrained equation set. We take the first two constraints as the active constraints, and solve the optimization problem as minimizing*

$$J = \frac{1}{2}x^T Ex + x^T F,$$

subject to

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ 3x_1 - 2x_2 - 3x_3 &= 1 \end{aligned} \tag{4.43}$$

which is the identical problem to the one given in Example 4.2. We solve this equality constraint problem, obtaining as before

$$\lambda = \begin{bmatrix} 1.6452 \\ -0.0323 \end{bmatrix}$$

Clearly the second element in λ is negative. We drop the second constraint and solve the optimization problem as

$$J = \frac{1}{2} x^T E x + x^T F,$$

subject to

$$x_1 + x_2 + x_3 = 1$$

Once more, we solve this equality constrained optimization problem, and obtain $\lambda = \frac{5}{3}$, leading to $x = [0.3333 \quad 1.3333 \quad -0.6667]^T$. Clearly, the optimal solution x satisfies the equality constraint. We also check whether the rest of the inequality constraints (4.41) are satisfied. They are all indeed satisfied.
There are several comments as follows.

1. In the case of equality constraints, the maximum number of equality constraints equals the number of decision variables. In this example, it is 3, and the only feasible solution x is to satisfy the equality constraints (see (4.42)). In contrast, in the case of inequality constraints, the number of inequality constraints is permitted to be larger than the number of decision variables, as long as they are not all active. In this example, only one constraint becomes active so it becomes an equality constraint. Once the optimal solution is found against this active constraint, the rest of the inequalities are automatically satisfied.

2. It is clear that an iterative procedure is required to solve the optimization problem with inequality constraints, because we did not know which constraints would become active constraints. If the active set could be identified in advance, then the iterative procedure would be shortened.

3. Note that the conditions for the inequality constraints are more relaxed than the case of imposing equality constraints. For instance, the number of constraints is permitted to be greater than the number of decision variables, and the set of inequality constraints is permitted to be linearly dependent. However, these relaxations are only permitted to the point that the active constraints need to be linearly independent and the number of active constraints needs to be less than or equal to the number of decision variables.

4.4.3 Primal-Dual Method

The family of active methods belongs to the group of primal methods, where the solutions are based on the decision variables (also called primal variables in the literature). In the active set methods, the active constraints need to be identified along with the optimal decision variables. If there are many constraints, the computational load is quite large. Also, the programming of an active method is not a straightforward task, as we illustrated through Example 4.6. A dual method can be used systematically to identify the constraints that are not active. They can then be eliminated in the solution. The Lagrange multipliers are called dual variables in the optimization literature. This method will lead to very simple programming procedures for finding optimal solutions of constrained minimization problems.

The dual problem to the original primal problem is derived as follows. Assuming feasibility (i.e.,there is an x such that $Mx < \gamma$), the primal problem is equivalent to

$$\max_{\lambda \geq 0} \min_{x} [\frac{1}{2} x^T E x + x^T F + \lambda^T (Mx - \gamma)] \tag{4.44}$$

The minimization over x is unconstrained and is attained by

$$x = -E^{-1}(F + M^T \lambda) \tag{4.45}$$

Substituting this in (4.44), the dual problem is written as

$$\max_{\lambda \geq 0} (-\frac{1}{2} \lambda^T H \lambda - \lambda^T K - \frac{1}{2} F^T E^{-1} F) \tag{4.46}$$

where the matrices H and K are given by

$$H = M E^{-1} M^T \tag{4.47}$$

$$K = \gamma + M E^{-1} F \tag{4.48}$$

Thus, the dual is also a quadratic programming problem with λ as the decision variable. Equation (4.46) is equivalent to

$$\min_{\lambda \geq 0} (\frac{1}{2} \lambda^T H \lambda + \lambda^T K + \frac{1}{2} F^T E^{-1} F) \tag{4.49}$$

Note that the dual problem may be much easier to solve than the primal problem because the constraints are simpler (see Hildreths quadratic programming procedure in the next subsection).

The set of optimal Lagrange multipliers that minimize the dual objective function

$$J = (\frac{1}{2} \lambda^T H \lambda + \lambda^T K + \frac{1}{2} F^T E^{-1} F) \tag{4.50}$$

subject to $\lambda \geq 0$, are denoted as λ_{act}, and the corresponding constraints are described by M_{act} and γ_{act}. Once the dual problem is solved yielding an optimal λ_{act}, the primal problem can be solved by minimizing the corresponding Lagrangian. With the values of λ_{act} and M_{act} , the primal variable vector x is obtained using

$$x = -E^{-1} F - E^{-1} M_{act}^T \lambda_{act} \tag{4.51}$$

where the constraints are treated as equality constraints in the computation.

4.4.4 Hildreth's Quadratic Programming Procedure

Hildreth's quadratic programming procedure was proposed for solving this dual problem. In this algorithm, the direction vectors were selected to be equal to the basis vectors $e_i = [0 \quad 0 \quad \dots \quad 1 \quad \dots \quad 0 \quad 0]^T$. Then, the λ vector can be varied one component at a time. At a given step in the process, having obtained a vector $\lambda \geq 0$, we fix our attention on a single component λ_i. The objective function may be regarded as a quadratic function in this single component. We adjust λ_i to minimize the objective function. If that requires $\lambda_i < 0$, we set $\lambda_i = 0$. In either case, the objective function is decreased. Then, we consider the next component λ_{i+1}. If we consider one complete cycle through the components to be one iteration taking the vector λ^k to λ^{k+1}, the method can be expressed explicitly as

$$\lambda_i^{k+1} = max(0, w_i^{k+1}), \tag{4.52}$$

with

$$w_i^{k+1} = -\frac{1}{h_{ii}}[d_i + \sum_{j=1}^{i-1} h_{ij}\lambda_j^{k+1} + \sum_{j=i+1}^{n} h_{ij}\lambda_j^k], \tag{4.53}$$

where the scalar h_{ij} is the $ijth$ element in the matrix $H = ME^{-1}M^T$, and d_i is the ith element in the vector $K = \gamma + ME^{-1}F$. Also note that in (4.53) there are two sets of λ values in the computation: one involves λ^k and one involves the updated λ^{k+1} .

Because the converged λ^* vector contains either zero or positive values of the Lagrange multipliers, by (4.45), we have

$$x = -E^{-1}(F + M^T\lambda^*). \tag{4.54}$$

Hildreth's quadratic programming algorithm is based on an element-by-element search, therefore, it does not require any matrix inversion. As a result, if the active constraints are linearly independent and their number is less than or equal to the number of decision variables, then the dual variables will converge. However, if one or both of these requirements are violated, then the dual variables will not converge to a set of fixed values. The iteration will terminate when the iterative counter reaches its maximum value. Because there is no matrix inversion, the computation will continue without interruption. As we will observe in the coming examples, the algorithm will give a compromised, near-optimal solution with constraints if the situation of conflict constraints arises. This is one of the key strengths of using this approach in real-time applications, because the algorithms ability to automatically recover from an ill-conditioned constrained problem is paramount for the safety of plant operation.

When the conditions are satisfied, the one-dimensional search technique in Hildreth's quadratic programming procedure has been shown to converge to the set of λ^*, where λ^* contains zeros for inactive constraints and the positive components corresponding to the active constraints. The positive component collected as a vector is called λ_{act}^* with its value defined by

$$\lambda_{act}^* = -(M_{act}E^{-1}M_{act}^T)^{-1}(\gamma_{act} + M_{act}E^{-1}F), \tag{4.55}$$

where M_{act} and γ_{act} are the constraint data matrix and vector with the deletion of the row elements that corresponding to the zero elements in λ^*. The proof of the convergence relies on the existence of a set of bounded λ_{act}^* . This is virtually determined by the existence of the $(M_{act}E^{-1}M_{act}^T)^{-1}$.

Example 4.7. *Minimize the cost function:*

$$J = \frac{1}{2}x^T Ex + F^T x$$

where $E = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}; F = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$ The constraints are $0 \leq x_1, 0 \leq x_2$ and

$$3x_1 + 2x_2 \leq 4$$

Solution. We form the linear inequality constraints

$$\begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ 4 \end{bmatrix} \tag{4.56}$$

The global optimal solution without constraints is

$$\begin{bmatrix} x_1^0 \\ x_2^0 \end{bmatrix} = -E^{-1}F = -\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

By substituting the global optimal solution into (4.56), we note that the inequality constraints are violated, with respect to the third constraint (i.e.,$3 + 2 > 4$).
To find the optimal λ^* using the algorithm stated, we form

$$H = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} -1 & 0 & 3 \\ 0 & -1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -5 \\ 1 & 2 & -7 \\ -5 & -7 & 29 \end{bmatrix} \tag{4.57}$$

$$K = \gamma + ME^{-1}F = \begin{bmatrix} 0 \\ 0 \\ 4 \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} \tag{4.58}$$

The iteration starts at $m = 0$, with the initial conditions of $\lambda_1^0 = \lambda_2^0 = \lambda_3^0 = 0$. At $m = 1$,

$$w_1^1 + 1 = 0 \tag{4.59}$$
$$\lambda_1^1 + 2w_2^1 + 1 = 0 \tag{4.60}$$
$$-5\lambda_1^1 - 7\lambda_2^1 + 29w_3^1 - 1 = 0 \tag{4.61}$$

Solving (4.59) gives $\lambda_1^1 = \max(0, w_1^1) = 0$, solving (4.60) gives $\lambda_2^1 = \max(0, w_2^1) = 0$, and solving (4.61) gives $\lambda_3^1 = \max(0, w_3^1) = 0.0345$.
At $m = 2$,

$$w_1^2 + \lambda_2^1 - 5\lambda_3^1 + 1 = 0 \tag{4.62}$$
$$\lambda_1^2 + 2w_2^2 - 7\lambda_3^1 + 1 = 0 \tag{4.63}$$
$$-5\lambda_1^2 - 7\lambda_2^2 + 29w_3^2 - 1 = 0 \tag{4.64}$$

This gives $\lambda_1^2 = \max(0, w_1^2) = 0, \lambda_2^2 = \max(0, w_2^2) = 0$ and $\lambda_3^2 = \max(0, w_3^2) = 0.0345$. Since $\lambda^2 = \lambda^1$, the iterative procedure has converged. The optimal solution of λ is $\lambda_1^* = 0, \lambda_2^* = 0$, and $\lambda_3^* = 0.0345$. The optimal solution of x is given by

$$x^* = \begin{bmatrix} x_1^0 \\ x_2^0 \end{bmatrix} - E^{-1}M^T\lambda^* = \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.1724 \\ 0.2414 \end{bmatrix} = \begin{bmatrix} 0.8276 \\ 0.7586 \end{bmatrix} \tag{4.65}$$

From (4.65), it is seen that the constrained optimal solution consists of two parts. One is identical to the global optimal solution, and the second part is a correction term due to the active constraint.

4.4.5 Closed-form Solution of λ^*

We noticed that the Hildreth's quadratic programming procedure produces the optimal λ^* that has zeros and the components corresponding to the active constraints. The converged vector is called λ_{act}^* with its value defined by

$$\lambda_{act}^* = -(M_{act}E^{-1}M_{act}^T)^{-1}(\gamma_{act} + M_{act}E^{-1}F) \tag{4.66}$$

where M_{act} and γ_{act} are the constraint data matrix and vector with the deletion of the row elements that correspond to the zero elements in λ^*. Thus, if we could correctly identify apriori the active constraints, then we can compute the closed-form solution of the constrained optimization problem.

Example 4.8. *Suppose that with apriori knowledge, the third constraint in Example 4.7 is known to be active. Find the optimal solution using the closed-form solution.*
Solution. *With the given information, we let $\lambda_1^* = \lambda_2^* = 0$. We form*

$$M_{act} = [3 \quad 2]; \quad \gamma_{act} = 4$$

$$\lambda_{act}^* = -(M_{act}E^{-1}M_{act}^T)^{-1}(\gamma_{act} + M_{act}E^{-1}F) = \frac{1}{29} = 0.0345.$$

The results are identical to those obtained in Example 4.7.

This technique of guessing active constraints is useful in the situation when the computational speed is critical for an application, with which we can resort to the closed-form solution of a constrained control problem. Also, the computational speed is increased if we only need to search for part of the active constraints set, while the other part comes from guesswork.

4.5 Predictive Control with Constraints on Input Variable Examples

This section will present worked examples of predictive control showing how constraints on the input variable $u(k)$ are imposed. The constraints include those on rate of change and amplitude constraints. These constraints are commonly encountered in industrial applications.

Constraints on Rate of Change

We present two worked examples of predictive control here. The first example is to show how to incorporate constraints for the first element in ΔU, and the second example is to show how to incorporate constraints for all the elements in ΔU.

Example 4.9. *A discrete-time plant is described by a state model*

$$\begin{aligned} x_m(k+1) &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_m(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u(k) \\ y(k) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x_m(k) \end{aligned}$$

Assuming a control horizon $Nc = 3$, a prediction horizon $Np = 5$, at time $k_i = 1, \quad r(k_i) = 1$, the state vector $x(k_i) = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, and the weight on the control signal is $r_w = 1$. The limit on the rate of change on the control signal is specified as

$$-1 \leq \Delta u(k) \leq 0.1$$

For this example, we only consider the case of imposing the constraints on the first element of ΔU. Find the optimal solution ΔU.

Solution. From (3.5), $n_1 = 2$ and $0_m = \begin{bmatrix} 0 & 0 \end{bmatrix}$. The augmented model for this plant is given by

$$\begin{aligned} x(k+1) &= Ax(k) + B\Delta u(k) \\ y(k) &= Cx(k) \end{aligned}$$

where the augmented system matrices are

$$A = \begin{bmatrix} A_m & 0_m^T \\ C_m A_m & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}; \quad B = \begin{bmatrix} B_m \\ C_m B_m \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}; \quad C = \begin{bmatrix} 0_m & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}.$$

We obtain the objective function:

$$J = \Delta U^T(\Phi^T\Phi + \bar{R})\Delta U - 2\Delta U^T\Phi^T(R_s - Fx(k_i)) + (R_s - Fx(k_i))^T(R_s - Fx(k_i))$$

where

$$F = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 1 \\ 3 & 6 & 1 \\ 4 & 10 & 1 \\ 5 & 15 & 1 \end{bmatrix}, \quad \Phi = \begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 6 & 3 & 1 \\ 10 & 6 & 3 \\ 15 & 10 & 6 \end{bmatrix}$$

$$\Phi^T\Phi = \begin{bmatrix} 371 & 231 & 126 \\ 231 & 146 & 81 \\ 126 & 81 & 46 \end{bmatrix}, \Phi^T F = \begin{bmatrix} 140 & 371 & 35 \\ 85 & 231 & 20 \\ 45 & 126 & 10 \end{bmatrix}, \Phi^T R_s = \begin{bmatrix} 35 \\ 20 \\ 10 \end{bmatrix}$$

Based on the procedures presented in Section 4.3, the constraints are translated to the two linear inequalities as

$$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} \leq \begin{bmatrix} 0.1 \\ 1 \end{bmatrix} \tag{4.67}$$

To demonstrate how the solution evolves, we illustrate the first two steps in the computational procedure.

1. At $k_i = 1$, without constraints, the global optimal solution of ΔU is $[0.3634 \quad -0.3063 \quad -0.2336]^T$. Thus, the constraint (4.67) is violated. The problem is solved using Hildreth's quadratic programming to obtain
$$\Delta U = [0.1000 \quad 0.1865 \quad -0.3767]^T.$$
2. With the first component $\Delta u(1) = 0.1$, and $y(1) = 0$ the state variable is updated to yield $x(2) = [0.1 \quad 0.1 \quad 0.1]^T$.
3. Again, the global optimal solution is $\Delta U = [0.1737 \quad -0.2764 \quad -0.1616]^T$ which violates the constraint. The optimization problem is again solved using the Hildreth's quadratic programming to obtain
$$\Delta U = \begin{bmatrix} 0.1000 & -0.1385 & -0.2017 \end{bmatrix}^T$$
4. With updated information on $y(2) = 0.1$ and $\Delta u(2) = 0.1$, the state variable is updated with value $x(3) = [0.3 \quad 0.2 \quad 0.4]^T$.
5. The global optimal solution is $\Delta U = [-0.1541 \quad -0.1634 \quad -0.0009]^T$, which satisfies the constraint.
6. The computation continues in the same way.

Example 4.10. *This example will investigate the scenario where the constraints are imposed for all elements in ΔU, which is the case often referred to in the predictive control literature. The nominal design of predictive control remains the same as in Example 4.9.*

Solution. From Section 4.3, when the constraints are fully imposed on all the components in ΔU, they are translated to the six linear inequalities as

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} \leq \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{4.68}$$

Since there are three decision variables ($Nc = 3$), the number of active constraints should not be greater than 2 at any given time in order to have an optimal solution. We illustrate the solutions for the first two cycles of the computation.

1. At $k_i = 1$, without constraints, the global optimal solution of ΔU is $[0.3634 \quad -0.3063 \quad -0.2336]^T$. Thus, the constraint (4.68) is violated. The problem is solved using Hildreth's quadratic programming to obtain $\Delta U = [0.1000 \quad 0.1000 \quad -0.2277]^T$. In comparison with the results in Example 4.9, the first solution has two active constraints. Namely, the first two elements in ΔU are the results from active constraints.

2. At $k_i = 2$, with the first component $\Delta u(1) = 0.1$, and $y(1) = 0$ the state variable is updated to yield $x(2) = [0.1 \quad 0.1 \quad 0.1]^T$.

3. Again, the global optimal solution is $\Delta U = [0.1737 \quad -0.2764 \quad -0.1616]^T$ which violates the constraint. The optimization problem is again solved using the Hildreth's quadratic programming to obtain $\Delta U = \begin{bmatrix} 0.1000 & -0.1385 & -0.2017 \end{bmatrix}^T$, where only the first constraint becomes activated, therefore the solution is identical to the solution in the second step of Example 4.9.

4. With updated information on $y(2) = 0.1$ and $\Delta u(2) = 0.1$, the state variable is updated with value $x(3) = [0.3 \quad 0.2 \quad 0.4]^T$.

5. The global optimal solution is $\Delta U = [-0.1541 \quad -0.1634 \quad -0.0009]^T$, which satisfies the constraint.

Because the constraints are imposed on all the element of ΔU, there are possibilities that non-essential constraints become activated.

Constraints on Amplitude of the Control

The amplitude of the control signal $u(k)$ is another important object for imposing constraint. The examples below illustrate how to impose constraints on the amplitude of the control signal. We will consider two examples. The first example is to impose the constraints on the first sample of the control signal and the second example is to impose constraints on all elements of the control signal.

Example 4.11. *We will consider the same system given in Example 4.9 with identical design specification, except that the constraints are changed to*

$$-2 \leq u(k) \leq 0.2$$

Again, in this example, we will consider the case of imposing the constraints on the first sample of control.

Solution. Note that at sample time k_i

$$u(k_i) = \Delta u(k_i) + u(k_i - 1); \quad \Delta u(k_i) = [1 \quad 0 \quad 0]\Delta U.$$

Here, ΔU is the parameter vector to be optimized. Therefore, from Section 4.3, the inequality constraints are translated into

$$-2 \leq [1 \quad 0 \quad 0]\Delta U + u(k_i - 1) \leq 0.2$$

That is

$$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} \leq \begin{bmatrix} 0.2 - u(k_i-1) \\ 2 + u(k_i-1) \end{bmatrix} \tag{4.69}$$

Note that the value of past control signal $u(k_i-1)$ is embedded into the constraint equations, thus the constraint equations need to be updated as the control system is implemented in real time.
Let us examine the first one cycles of the implementation.

1. When $k_i = 1$, without constraints, the global optimal solution of ΔU is $[0.3634 \quad -0.3063 \quad -0.2336]^T$. which gives $u(k_i) = 0.3634$, where $u(k_i - 1) = 0$ is assumed as the initial condition. Therefore, the constraint is violated. The quadratic program procedure finds the optimal solution as $\Delta U = [0.2000 \quad -0.0006 \quad -0.3224]^T$, which leads to $u(1) = 0.2$ satisfying the constraint.

2. With the updated information on the state variable and the information $u(1) = 0.2$, with constraint on the control amplitude, the optimal solution for $k_i = 2$ is $\Delta U = [-0.0160 \quad -0.2465 \quad -0.0897]^T$, which gives the optimal control as $u(2) = 0.1840$ satisfying the constraint.

3. As time progresses, the predictive control system finds the optimal control at each sampling period with respect to the specified constraints on the control signal.

Example 4.12. *In this example, we will continue the Example 4.11 by considering the case* $-2 \leq u(k) \leq 0.2$*, however, where the constraints are imposed on all elements of the control signal.*

Solution. Note that

$$\begin{aligned} u(k_i+1) =& \Delta u(k_i+1) + u(k_i) = \Delta u(k_i+1) + \Delta u(k_i) + u(k_i-1) \\ u(k_i+2) =& \Delta u(k_i+2) + \Delta u(k_i+1) + \Delta u(k_i) + u(k_i-1). \end{aligned}$$

In matrix vector form, the expression for the three elements of control in terms of ΔU is

$$\begin{bmatrix} u(k_i) \\ u(k_i+1) \\ u(k_i+2) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} + \begin{bmatrix} u(k_i-1) \\ u(k_i-1) \\ u(k_i-1) \end{bmatrix} \tag{4.70}$$

Now, with constraints to be imposed on both upper and lower limits of the control signals, the linear inequalities are formulated into the matrix vector form

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ -1 & 0 & 0 \\ -1 & -1 & 0 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} \leq \begin{bmatrix} 0.2 - u(k_i-1) \\ 0.2 - u(k_i-1) \\ 0.2 - u(k_i-1) \\ 2 + u(k_i-1) \\ 2 + u(k_i-1) \\ 2 + u(k_i-1) \end{bmatrix} \tag{4.71}$$

Since there are three decision variables ($N_c = 3$), the number of active constraints should not be greater than 2 at any given time in order to have an optimal solution. We illustrate the solutions for the first two cycles of the computation.

1. At $k_i = 1$, without constraints, the global optimal solution of ΔU is $[0.3634 \quad -0.3063 \quad -0.2336]^T$. which gives $u(k_i) = 0.3634, u(k_i + 1) = 0.0571, u(k_i + 2) = -0.1775$, where $u(k_i - 1) = 0$ is assumed as the initial condition. Therefore, the constraint (4.71) is violated. The quadratic program procedure finds the optimal solution as $\Delta U = [0.2000 \quad -0.0006 \quad -0.3224]^T$, which leads to $u(1) = 0.2, u(2) = 0.1994, u(3) = -0.1230$ satisfying the constraint.

2. At $k_i = 2$, with the first component $\Delta u(1) = 0.2$, and $y(1) = 0$ the state variable is updated to yield $x(2) = [0.2 \quad 0.2 \quad 0.2]^T$.

3. Again, the global optimal solution is $\Delta U = \begin{bmatrix} 0.1634 & -0.5063 & -0.4336 \end{bmatrix}^T$, which gives $u(k_i) = 0.3634$. Therefore, the constraint (4.71) is violated. The quadratic program procedure finds the optimal solution as $\Delta U = \begin{bmatrix} -0.0160 & -0.2465 & -0.0897 \end{bmatrix}^T$, which leads to $u(2) = 0.1840, u(3) = -0.0625, u(4) = -0.1522$ satisfying the constraint.

4. With updated information on $y(2) = 0.2$ and $\Delta u(2) = -0.0160$, the state variable is updated with value $x(3) = [0.384 \quad 0.184 \quad 0.584]^T$.

5. With the updated information on the state variable and the information u(2) = 0.1840, with constraint on the control amplitude, the optimal solution for $k_i = 3$ is $\Delta U = \begin{bmatrix} -0.2619 & -0.0851 & 0.0764 \end{bmatrix}^T$, which gives the optimal control as $u(3) = -0.0779, u(4) = -0.1630, u(5) = -0.0866$ satisfying the constraint.

6. As time progresses, the predictive control system finds the optimal control at each sampling period with respect to the specified constraints on the control signal.

Constraints on Amplitude and Rate of Change

It is also a common practice that constraints are imposed on both the amplitude and the rate of change of control signal. If this is required in the design specification, both constraints will be combined together to form a larger set of linear inequalities. Two examples are given below to illustrate the design and implementation procedure. Again, we will consider first the case where constraints are imposed at the sampling instant k_i, and secondly extend the constraints to all future control movements.

Example 4.13. *We consider the same system as in Example 4.9 with constraints on*

$$-1 \leq \Delta u(k) \leq 0.1, \quad -2 \leq u(k) \leq 0.2$$

Solution. Assuming that at the sampling instant k_i the constraints are only imposed on $\Delta u(k_i)$ and $u(k_i)$, the set of linear inequality constraints is formulated into the matrix vector

form

$$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} \leq \begin{bmatrix} 0.1 \\ 1 \\ 0.2 - \Delta u(k_i-1) \\ 2 + \Delta u(k_i-1) \end{bmatrix} \tag{4.72}$$

The first two rows are used for the rate constraints and the last two rows are for the amplitude constraints. We solve the constrained optimization problem by minimizing the objective function J subject to the constraints given by (4.72). Note that the value of past control signal $u(k_i - 1)$ is embedded into the constraint equations, thus the constraint equations need to be updated as the control system is implemented in real time.
Let us examine the first two cycles of the implementation.

1. When $k_i = 1$, without constraints, the global optimal solution is:

   $$\Delta U = \begin{bmatrix} 0.3634 & -0.3063 & -0.2336 \end{bmatrix}^T$$

   , which gives $u(k_i) = 0.3634$, and $\Delta u(k_i) = 0.3634$, where $u(k_i - 1) = 0$ is assumed as the initial condition. Therefore, the constraint is violated. The quadratic program procedure finds the optimal solution as $\Delta U = \begin{bmatrix} 0.1000 & 0.1865 & -0.3767 \end{bmatrix}^T$, which leads to $\Delta u(1) = 0.1, u(1) = 0.1$ satisfying the constraint.

2. At $k_i = 2$, with the first component $\Delta u(1) = 0.1$, and $y(1) = 0$ the state variable is updated to yield $x(2) = [0.1 \quad 0.1 \quad 0.1]^T$.

3. Again, the global optimal solution is $\Delta U = [0.1737 \quad -0.2764 \quad -0.1616]^T$ which violates the constraint. The optimization problem is again solved using the Hildreth's quadratic programming to obtain $\Delta U = \begin{bmatrix} 0.1000 & 0.5115 & -0.5517 \end{bmatrix}^T$, which leads to $\Delta u(2) = 0.1, u(2) = \Delta u(2) + u(1) = 0.2$ satisfying the constraint.

4. With updated information on $y(2) = 0.1$ and $\Delta u(2) = 0.1$, the state variable is updated with value $x(3) = [0.3 \quad 0.2 \quad 0.4]^T$.

5. The global optimal solution is $\Delta U = \begin{bmatrix} -0.1541 & -0.1634 & -0.0009 \end{bmatrix}^T$, which satisfies the constraint.

6. As time progresses, the predictive control system finds the optimal control at each sampling period with respect to the specified constraints on amplitude and rate of change.

Example 4.14. *In this example, we consider the case*

$$-1 \leq \Delta u(k) \leq 0.1, \quad -2 \leq u(k) \leq 0.2$$

However, the constraints will be imposed on all future components of ΔU *and* $u(k_i), u(k_i + 1), u(k_i + 2)$ *and the results will be compared with those presented in Example 4.13.*

Solution. With all the constraints imposed on both the rate of change and the amplitude of the control signal, the inequalities are translated into the matrix vector form

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ -1 & 0 & 0 \\ -1 & -1 & 0 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} \leq \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 1 \\ 1 \\ 1 \\ 0.2 - u(k_i-1) \\ 0.2 - u(k_i-1) \\ 0.2 - u(k_i-1) \\ 2 + u(k_i-1) \\ 2 + u(k_i-1) \\ 2 + u(k_i-1) \end{bmatrix} \tag{4.73}$$

Since there are three decision variables (Nc = 3), the number of active constraints should not be greater than 2 at any given time in order to have an optimal solution. We illustrate the solutions for the first two cycles of the computation.

1. At $k_i = 1$, without constraints, the global optimal solution of ΔU is $[0.3634 \quad -0.3063 \quad -0.2336]^T$. Thus, the constraint (4.73) is violated, because $\Delta u(1) = 0.3634 > 0.1, u(1) = 0.3634 > 0.2$, where $u(k_i - 1) = 0$ is assumed as the initial condition. The problem is solved using Hildreth's quadratic programming to obtain $\Delta U = [0.1000 \quad 0.1000 \quad -0.2277]^T$, which leads to $\Delta u(1) = 0.1, \Delta u(2) = 0.1, \Delta u(3) = -0.2277, u(1) = 0.1, u(2) = 0.2$, and $u(3) = -0.0277$ satisfying the constraint.

2. At $k_i = 2$, with the first component $\Delta u(1) = 0.1$, and $y(1) = 0$ the state variable is updated to yield $x(2) = [0.1 \quad 0.1 \quad 0.1]^T$.

3. Again, the global optimal solution is $\Delta U = [0.1737 \quad -0.2764 \quad -0.1616]^T$ which violates the constraint. The optimization problem is again solved using the Hildreth's quadratic programming to obtain $\Delta U = \begin{bmatrix} 0.1000 & -0.1385 & -0.2017 \end{bmatrix}^T$, which leads to $\Delta u(2) = 0.1, \Delta u(3) = -0.1385, \Delta u(3) = -0.2017, u(2) = 0.2, u(3) = 0.0615$, and $u(4) = -0.1402$ satisfying the constraint.

4. With updated information on $y(2) = 0.1$ and $\Delta u(2) = 0.1$, the state variable is updated with value $x(3) = [0.3 \quad 0.2 \quad 0.4]^T$.

5. The global optimal solution is $\Delta U = \begin{bmatrix} -0.1541 & -0.1634 & -0.0009 \end{bmatrix}^T$, which satisfies the constraint.

6. As time progresses, the predictive control system finds the optimal control at each sampling period with respect to the constraints on amplitude and rate of change.

Constraints on the Output Variable

In this section, we investigate the case where the constraints are imposed on the output variable.

Example 4.15. *We consider the same system as in Example 4.9 with constraints on*

$$0 \leq y(k) \leq 1$$

Solution. Based on the procedures presented in Section 4.3, the constraints are translated to the linear inequalities as

$$\begin{bmatrix} -1 & 0 & 0 \\ -3 & -1 & 0 \\ -6 & -3 & -1 \\ -10 & -6 & -3 \\ -15 & -10 & -6 \\ 1 & 0 & 0 \\ 3 & 1 & 0 \\ 6 & 3 & 1 \\ 10 & 6 & 3 \\ 15 & 10 & 6 \end{bmatrix} \begin{bmatrix} \Delta u(k_i) \\ \Delta u(k_i+1) \\ \Delta u(k_i+2) \end{bmatrix} \leq \begin{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 1 \\ 3 & 6 & 1 \\ 4 & 10 & 1 \\ 5 & 15 & 1 \end{bmatrix} x(k_i) \\ \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 1 \\ 3 & 6 & 1 \\ 4 & 10 & 1 \\ 5 & 15 & 1 \end{bmatrix} x(k_i) \end{bmatrix} \tag{4.74}$$

We illustrate the solutions for first two cycles of the computation.

1. At $k_i = 1$, without constraints, the global optimal solution of ΔU is $[0.3634 \quad -0.3063 \quad -0.2336]^T$. Thus, the constraint (4.74) is violated. The problem is solved using Hildreth's quadratic programming to obtain $\begin{bmatrix} 0.0981 & -0.1994 & 0.0870 \end{bmatrix}^T$. Thus, the constraint (4.74) is satisfied.

2. With the first component $\Delta u(1) = 0.0981$, and $y(1) = 0$ the state variable is updated to yield $x(2) = [0.0981 \quad 0.0981 \quad 0.0981]^T$. Thus, the output variable is updated to yield $y(2) = 0.0981$.

3. At $k_i = 2$, the global optimal solution is

   $$\Delta U = [0.1773 \quad -0.2770 \quad -0.1630]^T.$$

   Thus, the constraint (4.74) is violated. The problem is solved using Hildreth's quadratic programming to obtain

   $$\Delta U = \begin{bmatrix} 0.1992 & -0.3413 & -0.2693 \end{bmatrix}^T.$$

   Thus, the constraint (4.74) is satisfied.

4. With the first component $\Delta u(2) = 0.1992$, and the state variable is updated to yield $x(3) = [0.2973 \quad 0.2973 \quad 0.4935]^T$. Thus, the output variable is updated to yield $y(3) = 0.4935$.

5. The global optimal solution is $\Delta U = [-0.2719 \quad -0.1573 \quad 0.0262]^T$, which satisfies the constraint. Thus, $\Delta u(3) = -0.2719$,$x(4) = \begin{bmatrix} 0.3227 & 0.0254 & 0.8162 \end{bmatrix}^T$, and $y(4) = 0.8162$.

6. As time progresses, the predictive control system finds the optimal control at each sampling period with respect to the constraints on the output variable.

   When the constrains are fully imposed, the same procedure is implemented to find the optimal solution.

Chapter 5

Conclusion

This project report has discussed the basic ideas about discrete-time model predictive control. With the current system information represented by the state variable vector $x(k_i)$, the prediction of the future behaviour of the system output relies on the state-space model where the optimal control trajectory is captured by the set of parameters that define the incremental control movement as: $\Delta u(k_i), \Delta u(k_i+1), \ldots, \Delta u(k_i+N_c-1)$. Within the optimization window, the objective of the control system is expressed in terms of the error function between the desired set-point signal and the predicted output signal. With a specific choice of an error function as the measurement of the objective, an optimal solution is obtained for the set of incremental control movement: $\Delta u(k_i), \Delta u(k_i+1), \ldots, \Delta u(k_i+N_c-1)$. Although the optimal control trajectory is calculated for N_c future samples, the implementation of the predictive control uses only the first sample, $\Delta u(k_i)$ while ignoring the rest of the trajectory. The optimization procedure repeats itself when the next sample period arrives. This is based on the receding horizon control principle, where feedback is naturally incorporated in the control system design.

The design model used here is an augmented system model with embedded integrator(s). By doing so, the control signal to be optimized is the sequence of $\Delta u(k_i+m), m = 0, 1, 2, \ldots,$ instead of the sequence of the control signal $u(k_i+m)$. An integrator is naturally embedded into the design, leading to the predictive control system tracking constant references and rejecting constant disturbances without steady-state errors. Another significant advantage of this approach is that in implementation, it neither requires the steady-state information about the control ($u(k) = u(k-1) + \Delta u(k)$) nor the information about the steady state of the state variable x_m(Δx_m at steady state is zero). This simplified information requirement becomes more important for a system having many inputs and many outputs.

Discrete-time model predictive control with constraints, imposing constraints in the design and implementation of predictive control system involves the following steps:

1. Defining system operational limits, including limits on the input variables, the incremental change of the input variables, state variables and system output variables.

2. Expressing these limits as parameters for the minimum and maximum of $u, \Delta u, x_m$ and y, with consideration of steady-state information.

3. With parameterization of the future control trajectory, these minimum and maximum values are expressed in the form of inequalities with variables $\Delta u(k_i), \Delta u(k_i + 1), \ldots, \Delta u(k_i + N_c - 1)$.

4. The design objective of model predictive control becomes the minimization of the original error function subject to the inequality constraints, where the set of parameters $\Delta u(k_i), \Delta u(k_i + 1), \ldots, \Delta u(k_i + N_c - 1)$ become decision variables.

5. Solving the constrained optimization problem using a quadratic programming procedure at every sampling instance to obtain the optimal solution of the decision variables.

Because the constraints are expressed in terms of inequalities (constraints may or may not be violated at a particular time), in general there is no closed-form solution of the constrained control problem, unless the set of active constraints are known. If the active constraints are known, the optimal solution of the decision variables is expressed in a closed-form. In chapter 4, instead of solving the decision variables iteratively, Hildreth's programming procedure was used to identify the active constraints via Lagrange multipliers (or the dual variables). Although it is still a quadratic programming problem on the dual variables, the constraints are much simplified ($\lambda \geq 0$) so a simple iterative procedure was used to obtain the optimal solution of the multipliers. Perhaps even more importantly, the iterative solution does not involve matrix inversion, so in the situation of conflicting constraints, the algorithm still delivers a compromised, sub-optimal solution without being numerically unstable. This is particularly important in the real-time implementation of the predictive control system.

References

[1] D. Atherton, Control Enigeeniring: An Introduction with the Use of Matlab, Bookboon.com, Brighton, Second Edition, 2013.

[2] D. Bao-Cang, Modern Predictive Control, CRC Press, Boca Raton, 2010.

[3] E. F. Camacho and C. Bordons, Modern Predictive Control(Advanced Textbooks in Control and Signal Processing), Springer-Verlag, London, Second Edition, 2007.

[4] C. W. de Silva, Modeling and Control of Engineering Systems, CRC Press, Boca Raton, 2009.

[5] M. Diehl, F. Glineur, E. Jarlebring and W. Michiels, Recent Advances in Optimization and its Applications in Engineering, Springer-Verlag, Berlin Heidelberg, 2010.

[6] L. Grüne and J. Pannek. Nonlinear Model Predictive Control Theory and Algorithms(Communications and Control Engineering), Springer-Verlag, London Limited, 2011.

[7] R. Haber, R. Bars,and U. Schmitz, Predictive Control in Process Engineering: From the Basics to the Applications, WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim, Germany, 2011.

[8] H. Kwakernaak and R. Sivan, Linear Optimal Control Systems, John Wiley and Sons, New York, 1972.

[9] W. H. Kwon and S. Han, Receding Horizon Control: Model Predictive Control for State Models(Advanced Textbooks in Control and Signal Processing), Springer-Verlag, London Limited , 2005.

[10] D. G. Luenberger and Yinyu Ye, Linear and Nonlinear Programming, Springer, New York, Third Edition, 2008.

[11] J. M. Maciejowski, Predictive Control(A Lecture Course Given in the Aerospace Engineering Faculty TU Delft), Delft University Press, Netherlands, 1998

[12] J. M. Maciejowski, Predictive Control with Constraints, Prentice Hall, New York, 2001.

[13] D. Q. Mayne, and H. Michalska, Receding Horizon Control of Nonlinear System, IEEE Trans. On Autom. Control, Vol. 35, PP 814 - 824, 1990.

[14] D. Q. Mayne, Nonlinear Model Predictive Control: An Assessment, Chemical Process Control, Vol 5, PP 217-231, 1997.

[15] D. Q. Mayne, J. B. Rawlings, C.V. Rao and P. O. M. Scokaert, Constrained Model Predictive Control: Stability and Optimality, Auto., Vol. 36, PP 789-814, 2000.

[16] K. Ogata, Discrete-Time Control Systems, Prentice-Hall, Englewood Cliffs, New Jersey, Second Edition, 1995.

[17] K. Ogata, Modern Control Engineering, Prentice-Hall, New York, Fifth Edition, 2010.

[18] J. A. Rossiter, Model Based Predictive Control: A Practical Approach(Control Series), CRC Press, Boca Raton, London, New York, Washington D.C., 2004.

[19] E. D. Sontag, Mathematical Control Theory: Deterministic Finite Dimensional Systems, Springer, New York, Second Edition, 1998.

[20] H. J. Sussmann and J. C. Willems, 300 Years of Optimal Contral: From the Brachystochrone to the Maximum Principle. IEEE Control Systems Magazine, Vol 17, PP 32-44, 1997.

[21] G. Takcs and B. Rohal-Ilkiv, Model Predictive Vibration Control: Efficient Constrained MPC Vibration Control for Lightly Damped Mechanical Structures, Springer, London Limited 2012.

[22] R. L. Williams II and D. A. Lawrence, Linear State-Space Control Systems, John Wiley & Sons, Hoboken, New Jersey, 2007.

[23] T. Zheng, Model Predictive Control: Basic Characters, In Tech, Rijeka, Croatia, 2012.