공학박사 학위논문

A Novel Iterative Learning Control Method Combined with Model Predictive Control for Tracking Specific Points

특정 점의 추적을 위한 모델예측제어가 결합된 새로운 반복학습제어 기법

2017년 2월

서울대학교 대학원
화학생물공학부
오 세 규

A Novel Iterative Learning Control Method Combined with Model Predictive Control for Tracking Specific Points

지도교수 이 종 민

이 논문을 공학박사 학위논문으로 제출함

2016년 12월

서울대학교 대학원

화학생물공학부

오 세 규

오 세 규의 공학박사 학위논문을 인준함

2016년 12월

위 원 장 _____ (인)

부위원장 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

위 원 _____ (인)

Abstract

A Novel Iterative Learning Control Method Combined with Model Predictive Control for Tracking Specific Points

Se-Kyu Oh
School of Chemical and Biological Engineering
The Graduate School
Seoul National University

In this thesis, we study an iterative learning control (ILC) technique combined with model predictive control (MPC), called the iterative learning model predictive control (ILMPC), for constrained multivariable control of batch processes. Although the general ILC makes the outputs converge to reference trajectories under model uncertainty, it uses open-loop control within a batch; thus, it cannot reject real-time disturbances. The MPC algorithm shows identical performance for all batches, and it highly depends on model quality because it does not use previous batch information. We integrate the advantages of the two algorithms. In many batch or repetitive processes, the output does not need to track all points of a reference trajectory. We propose a novel ILMPC method which can only consider the desired reference points, not an entire reference trajectory. It does not require to generate a reference trajectory which passes through the specific

desired points. Numerical examples are provided to demonstrate the performances of the suggested approach on point-to-point tracking, iterative learning, constraints handling, and real-time disturbance rejection.

Keywords: Iterative Learning Control, Model Predictive Control, Point-to-Point Tracking
Student Number: 2013-30282

Contents

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

A controller in a continuous process generally aims to converge an output to a constant set-point. In many cases, the regulation problem can be solved with the proportional-integral-derivative (PID) controller or the linear model-based controller. The linear model-based controller uses a linearized process model at an operating point. A controller in a batch, cyclic, repetitive or iteration process aims to make the output track a time-varying reference trajectory defined over a finite time interval. The PID controller only uses the error of the previous time; thus, it cannot prepare for the future time-varying reference trajectory. Model predictive control (MPC) has become the accepted standard for complex constrained multivariable control problems in the process industry. MPC uses the predictive model to calculate the input trajectory that minimize future output errors. The batch process dynamics, however, is highly nonlinear because of its wide operation range; thus, it is difficult to obtain an appropriate linear model. Although nonlinear MPC (NMPC) is used, perfect tracking is impossible because of the model-plant mismatch. Despite the above difficulties, the batch process has an important characteristic of re-

peating the same task, that is, the information from the previous batch steps and the previous time steps are available. The conventional control techniques use the model and the information of the previous time steps. Thus, a new control technique that can learn from the information of the previous batch is needed.

Iterative learning control (ILC) is a control technique that learns from previous experience such as previous batch, cycle, repetition or iteration. The basic idea of ILC can be found in [1] and the first research paper on ILC was written by Uchiyama [2]. It was written in Japanese; therefore, it has not received much attention from control community. In 1984, Arimoto *et al.* first introduced ILC in English [3]. ILC was originally introduced for robot manipulators, which repeat the same task from trial and trial. The basic algorithm of ILC is to use the information from the previous trial to control the current trial. However, it does not use the information of previous time steps at the current trial, that is, it is not a real-time feedback controller. Thus, ILC should be combined with a real-time control technique to reject real-time disturbances.

Among the advanced process control techniques, MPC is the most standard advanced real-time control technique. Many ILC techniques combined with MPC, often referred to as iterative learning model predictive control (ILMPC), have been proposed to handle real-time disturbances. Most studies of ILMPC use an augmented state-space model where a state vector consists of the entire error sequences of a batch, and a prediction horizon is fixed as the entire batch horizon [4, 5, 6, 7, 8]. In addition, the resulting formulation of such an algorithm employs linear time-varying (LTV) models in the state augmentation step, even if a system is linear time-invariant

(LTI). Thus, additional calculation burdens for LTI systems are added in the algorithm. Also, it has a different formulation from general ILC or MPC; thus, additional modification may be needed to incorporate other techniques applicable to general ILC or MPC such as advanced state estimation theory or point-to-point tracking technique into the controller. Two-stage approaches have also been proposed for combining ILC with real-time feedback controller [6, 7, 9]. The two-stage approaches have difficulties in system analysis and parameter tuning and require two optimization steps. In addition, the time-wise feedback controllers of the approaches are not offset-free control; thus, offset occurs in the early batches until the batch-wise controller shows convergence. Above all, the two-stage approaches do not consider constraints. ILC combined with dynamic matrix control (DMC) for LTI system [10] has been proposed, but the DMC algorithm without an observer cannot handle unknown disturbance and measurement noise effectively [11].

ILMPC should contain the following all advantages of MPC. (1) ILMPC should guarantee offset-free control. (2) It should have a single optimization step, not two optimization steps for both ILC part and MPC part separately. (3) It should consider constraints and ensure that a feasible solution will always be found. (4) Prediction horizon should be able to be adjusted to reduce the computational load. (5) If the model is LTI, it should use LTI model directly. (6) The form of prediction model and algorithm procedure should be similar to those of MPC.

General control techniques including PID, ILC and MPC should have an entire reference trajectory or set-points for all control time steps. If it is important for the output to converge to specific points,

an arbitrary reference trajectory passing through those points should be prepared first. This process adds additional burden. In addition, if parameters or constraints are modified, it is necessary to find a new reference trajectory. Tracking an entire reference trajectory is not always necessary in many applications such as a robotic "pick and place" task, crane control, rapid thermal process, and chemical batch reactor [12, 13, 14]. An ILC technique that considers only the desired reference points is called point-to-point ILC (PTP ILC) and has been studied recently [15, 16, 17]. Terminal ILC (TILC) has been also studied for tracking terminal point only [12, 18, 19]. It is a special case of the PTP ILC problem. These types of PTP ILC algorithms are open-loop control within a batch; thus, they cannot reject real-time disturbances. If real-time disturbances should be rejected, the PTP ILC algorithm needs integrating with a real-time feedback controller.

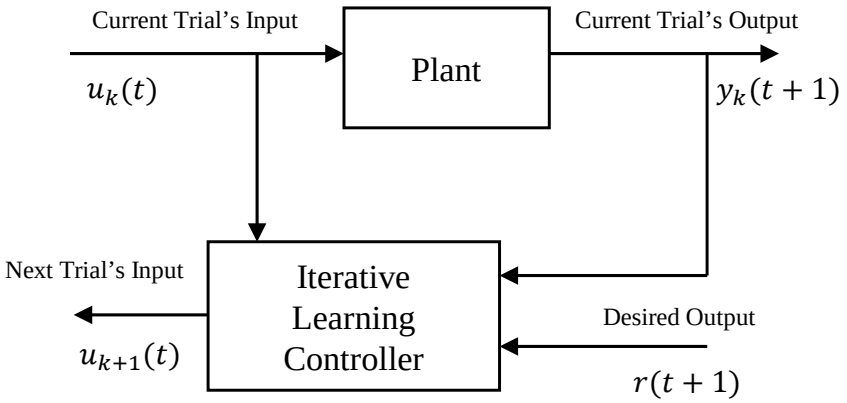
The main objective of the study is to propose a standard form of ILMPC that includes all the advantages mentioned above. Then, we propose a novel ILMPC technique that can track specific points without generating an arbitrary reference trajectory passing through the specific points.

1.2 Literature Review

1.2.1 Iterative Learning Control

The basic idea of the ILC is illustrated in Figure 1.1. For simplicity, we consider the LTI system in this section. In classical ILC, the following postulates are required.

- Every trial (pass, cycle, batch, iteration, repetition) ends in a



$$u_{k+1}(t) = f\big(u_k(t), y_k(t+1), r(t+1)\big)$$

Figure 1.1: The basic scheme of iterative learning control

fixed time of duration.

- Repetition of the initial setting is satisfied. That is, the initial state $x_k(0)$ of the objective system can be set to the same point at the beginning of each iteration.
- Invariance of the system dynamics is ensured throughout the repetition.
- The output $y_k(t)$ is measured in a deterministic way.

In recent ILC studies, the above postulates can be relaxed. Let us consider the following continuous LTI system:

$$\begin{aligned} \dot{x}_k(t) &= Ax_k(t) + Bu_k(t) \\ y_k(t) &= Cx_k(t) \end{aligned} \tag{1.1}$$

where $x_k(t)$ is the state, $u_k(t)$ is the input, $y_k(t)$ is the output, t is the time index, k is the batch, cycle, repetition or iteration index. That is, $u_k(t)$ is the system input at time t of the k-th batch.

The first learning control scheme, called "Arimoto-type" ILC, was proposed in 1984 [3, 20].

$$u_{k+1}(t) = u_k(t) + \Gamma \dot{e}_k(t) \tag{1.2}$$

where $e_k(t) = r(t) - y_k(t)$ and $r(t)$ is the reference trajectory. Consider the plant (1.1) and the input update law (1.2), the output $y_k(t) \to r(t)$ for all t as $k \to \infty$ if the learning gain matrix Γ satisfies the following condition.

$$\|I - CB\Gamma\|_i < 1 \tag{1.3}$$

where i is an operator norm and $i \in \{1,\ 2,\ \cdots,\ \infty\}$. Arimoto also proposed more general PID-type input update law in 1986 [21]. In this paper, Arimoto referred to this technique as "Iterative Learning Control". He used the term "Bettering Operation" in his previous papers.

$$u_{k+1}(t) = u_k(t) + \Phi e_k(t) + \Gamma \dot{e}_k(t) + \Psi \int_0^t e_k(\tau) d\tau \tag{1.4}$$

In industrial application, digital controller are used to control systems and to store the information obtained in the course of learning process. Thus, many ILC studies are based on discrete-time system.

$$\begin{aligned} x_k(t+1) &= Ax_k(t) + Bu_k(t) \\ y_k(t) &= Cx_k(t) \end{aligned} \tag{1.5}$$

In 1985, Togai and Yamano [22] proposed discrete optimal learning control algorithm based on state variable technique and they used the following objective function to obtain optimal learning gain matrix.

$$\min_{u_k(t)} \frac{1}{2} e_k^T(t+1) Q e_k(t+1) \tag{1.6}$$

where Q is an appropriate weighting matrix. The input update law is

$$u_{k+1}(t) = u_k(t) + G e_k(t+1) \tag{1.7}$$

and they proposed three types of learning gain matrix. Note that this paper considers $C = I$.

1. Steepest Descent

$$G = -KB^T \ \ (K : \text{constant}) \tag{1.8}$$

2. Newton-Raphson

$$G = -\frac{\|e_k(t+1)\|^2}{\|B^T B e_k(t+1)\|^2} B^T \tag{1.9}$$

3. Gauss-Newton

$$G = -(B^T B)^{-1} B^T \tag{1.10}$$

Most model-based ILC algorithms were based on the notion of direct model inversion [22, 23, 24, 25, 26, 27]. The learning gain matrix of the algorithms based on direct model inversion is very sensitive to high-frequency components in $e_k(t)$. Tao *et al.* [28] proposed a discrete-time ILC algorithm based on the following objective func-

tion with an input penalty term to reduce the noise sensitivity [5].

$$\min_{u_k(t)} \frac{1}{2} \left\{ e_k(t+1)^T Q e_k(t+1) + u_k(t)^T R u_k(t) \right\} \tag{1.11}$$

Sogo and Adachi [29] also proposed a continuous-time ILC algorithm based on the similar objective function.

ILC is basically an open-loop control. It is not necessary to obtain the input trajectory of the current batch in real time. Many ILC studies uses the lifted vector form. Each lifted vector consists of values of input and output for all time steps. Thus, an input trajectory for all time steps at the current batch is calculated with a single calculation if the lifted vector is used. Eq. (1.5) can be expressed the following form.

$$\begin{bmatrix} y_k(1) \\ y_k(2) \\ \vdots \\ y_k(N) \end{bmatrix} = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2}B & \cdots & CB \end{bmatrix} \begin{bmatrix} u_k(0) \\ u_k(1) \\ \vdots \\ u_k(N-1) \end{bmatrix} + \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} x_k(0) \tag{1.12}$$

Let us define Eq. (1.12) as follows.

$$\mathbf{y}_k = \mathbf{G}\mathbf{u}_k + \mathbf{F}x_k(0) \tag{1.13}$$

Amann *et al.* [30, 31, 32] and Lee *et al.* [33] independently proposed discrete-time ILC algorithms based on the following objective function with a penalty term on input change.

$$\min_{\mathbf{u}_k} \frac{1}{2} \left\{ \mathbf{e}_{k+1}^T \mathbf{Q} \mathbf{e}_{k+1} + \Delta \mathbf{u}_k^T \mathbf{R} \Delta \mathbf{u}_k \right\} \tag{1.14}$$

where $\Delta \mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_{k-1}$. Amann *et al.* and Lee *et al.* suggested different solutions.

$$\begin{aligned} \mathbf{u}_{k+1} =& \mathbf{u}_k + \mathbf{R}^{-1} \mathbf{G}^T \mathbf{Q} \mathbf{e}_k \quad (\text{Amann } \textit{et al.} \text{ [30]}) \\ \mathbf{u}_{k+1} =& \mathbf{u}_k + \left(\mathbf{G}^T \mathbf{Q} \mathbf{G} + \mathbf{R} \right)^{-1} \mathbf{G}^T \mathbf{Q} \mathbf{e}_k \quad (\text{Lee } \textit{et al.} \text{ [33]}) \end{aligned} \tag{1.15}$$

In all of the above control techniques, the error converges to zero. The following is more general form of the learning control algorithm [25].

$$\mathbf{u}_{k+1} = \mathbf{T}_u \mathbf{u}_k + \mathbf{T}_e \mathbf{e}_k \tag{1.16}$$

If the plant is $\mathbf{y}_k = \mathbf{T}_s \mathbf{u}_k$ with zero initial condition, the condition for convergence is $\| \mathbf{T}_u - \mathbf{T}_e \mathbf{T}_s \|_i < 1$. This is much less restrictive than $\| \mathbf{I} - \mathbf{T}_e \mathbf{T}_s \|_i < 1$ in Eq. (1.3). In this case, the error does not converge to zero. The final error is as follows.

$$\mathbf{e}^* = \lim_{k \to \infty} \mathbf{e}_k = \left(\mathbf{I} - \mathbf{T}_s \left(\mathbf{I} - \mathbf{T}_u + \mathbf{T}_e \mathbf{T}_s \right)^{-1} \mathbf{T}_e \right) \mathbf{r}. \tag{1.17}$$

If $\mathbf{T}_u = \mathbf{I}$ and the plant matrix is invertible, the error goes to zero as $k \to \infty$.

Several researchers have considered higher-order ILC (HOILC) [34, 35, 36, 37, 38]. HOILC uses up to the n-th previous batch, not just the previous batch. The following is the input update law of

HOILC.

$$\begin{aligned}\mathbf{u}_{k+1} =& \Lambda_k \mathbf{u}_k + \Lambda_{k-1}\mathbf{u}_{k-1} + \cdots + \Lambda_{k-n}\mathbf{u}_{k-n} \\ &+ \Gamma_k \mathbf{e}_k + \Gamma_{k-1}\mathbf{e}_{k-1} + \cdots + \Gamma_{k-n}\mathbf{e}_{k-n}\end{aligned} \tag{1.18}$$

Studies on ILC have also been conducted for applications in various systems. In the early days of ILC research, it was mainly applied to robot and mechatronic systems [20, 23, 39, 40, 41, 42, 43]. Then, ILC began to be applied to other systems such as chemical batch processes [44, 45, 46, 13, 47, 48, 49, 50, 51, 52, 53], injection molding processes [54, 55, 56] and semiconductor industry [57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69]. In many processes, especially in chemical batch processes, ILC combined with real-time feedback controller is used because disturbance rejection is an important issue. MPC is the most accepted standard real-time feedback control technique for complex constrained multivariable control problem in the process industry. Thus, Many ILC techniques combined with MPC have been studied.

Example 1.1

Consider the system and the reference trajectory as shown in Fig. 1.2.

$$\begin{aligned} x(t+1) &= \begin{bmatrix} -0.8 & -0.2 \\ 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 1 & 0.5 \end{bmatrix} \end{aligned} \tag{1.19}$$

Terminal time is 100 and the system has zero initial condition. The following three models are available.

1st model

$$\begin{aligned} x_k^1(t+1) &= \begin{bmatrix} -0.7 & -0.1 \\ 0.9 & 0 \end{bmatrix} x_k^1(t) + \begin{bmatrix} 0.4 \\ 0.9 \end{bmatrix} u_k^1(t) \\ y_k^1(t) &= \begin{bmatrix} 0.9 & 0.4 \end{bmatrix} x_k^1(t) \end{aligned} \tag{1.20}$$

2nd model

$$\begin{aligned} x_k^2(t+1) &= \begin{bmatrix} -0.8 & -0.5 \\ 1.3 & 0 \end{bmatrix} x_k^2(t) + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u_k^2(t) \\ y_k^2(t) &= \begin{bmatrix} 10 & 0.5 \end{bmatrix} x_k^2(t) \end{aligned} \tag{1.21}$$

3rd model

$$\begin{aligned} x_k^3(t+1) &= \begin{bmatrix} -0.7 & -0.1 \\ 0.9 & 0 \end{bmatrix} x_k^3(t) + \begin{bmatrix} 0.4 \\ 0.9 \end{bmatrix} u_k^3(t) \\ y_k^3(t) &= \begin{bmatrix} -1 & 0.4 \end{bmatrix} x_k^3(t) \end{aligned} \tag{1.22}$$

We use the following control law ($\mathbf{T}_u = \mathbf{I}$ in Eq. (1.16)).

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{T}_e^i \mathbf{e}_k \tag{1.23}$$

Convergence condition is $\|\mathbf{I} - \mathbf{T}_e^i \mathbf{T}_s\|_2 < 1$ where $\mathbf{T}_s$ is the step response matrix of the plant (1.19) and $\mathbf{T}_e^i$ is the step response matrix of the i-th model. The values of the convergence conditions of

the three models are as follows.

$$\begin{aligned} \|\mathbf{I} - \mathbf{T}_e^1 \mathbf{T}_s\|_2 &= 0.5745 \\ \|\mathbf{I} - \mathbf{T}_e^2 \mathbf{T}_s\|_2 &= 0.9977 \\ \|\mathbf{I} - \mathbf{T}_e^3 \mathbf{T}_s\|_2 &= 2.7664 \end{aligned} \tag{1.24}$$

If the 1st or 2nd model is used, the output trajectory is convergent to the reference trajectory as show in Figs. (1.3) and (1.4). If the 3rd model is used, the output trajectory diverges as shown in Fig. (1.5).

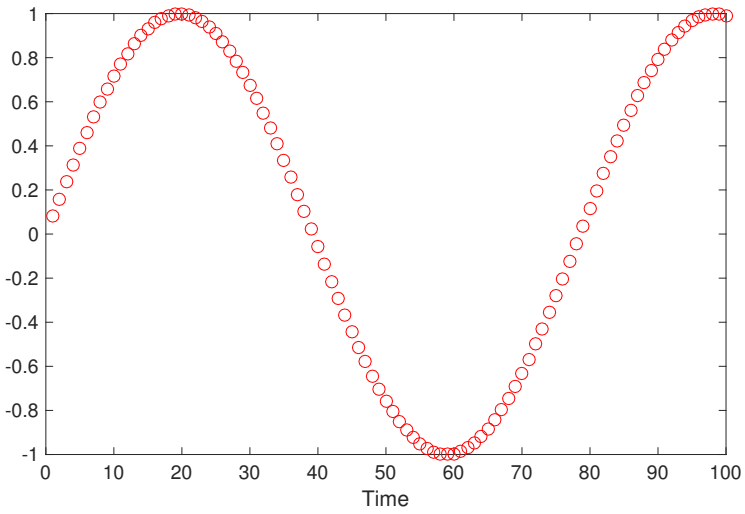


Figure 1.2: Reference trajectory

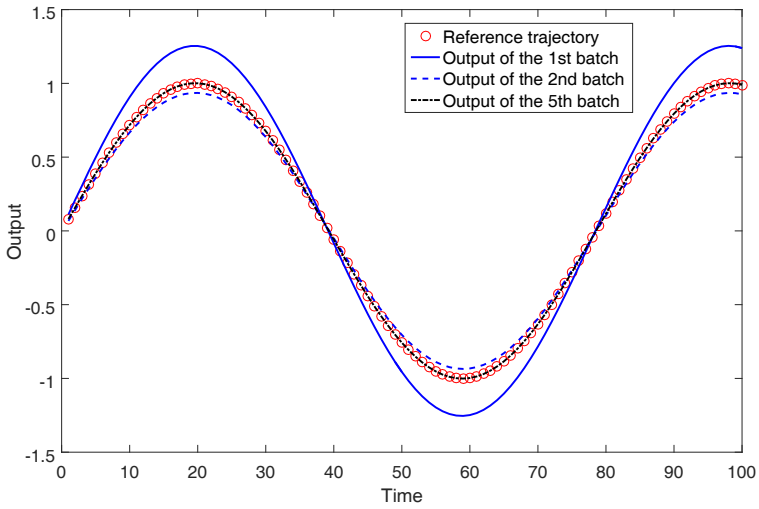


Figure 1.3: Result with the 1st model

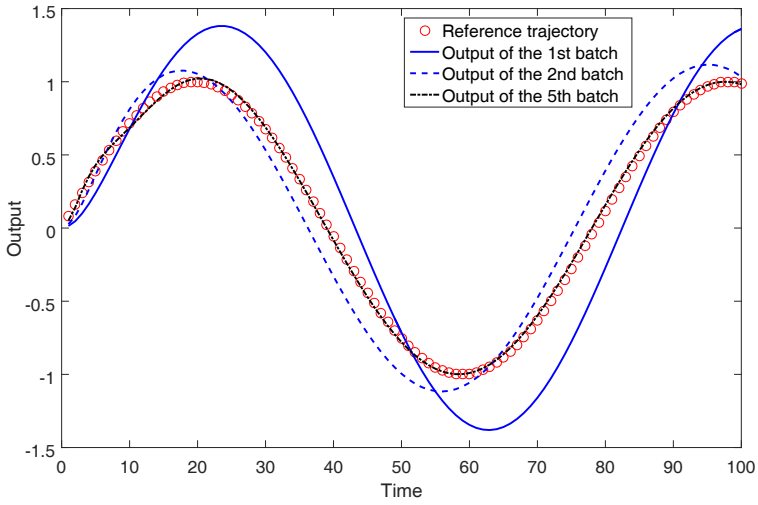


Figure 1.4: Result with the 2nd model

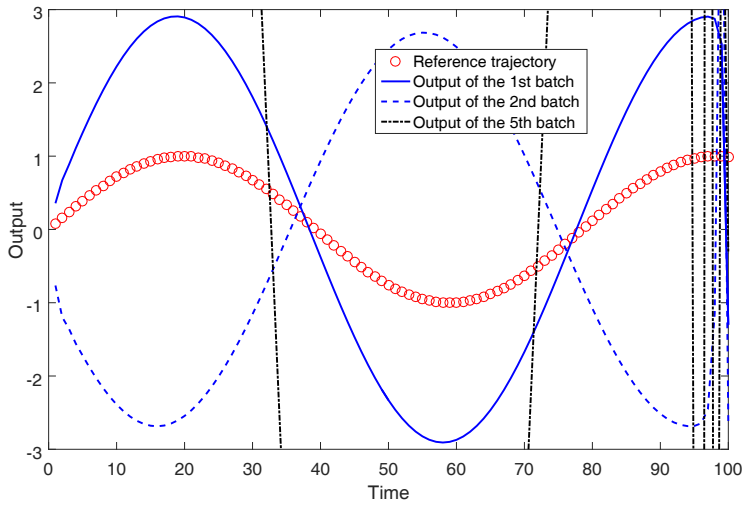


Figure 1.5: Result with the 3rd model

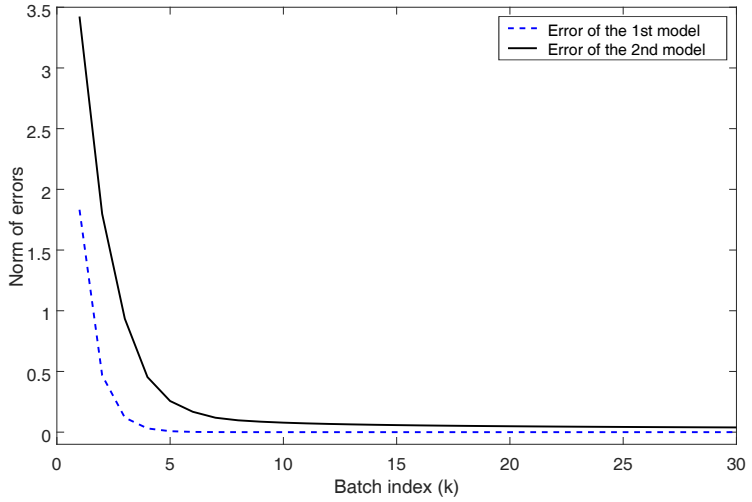


Figure 1.6: Convergence performance with the 1st and 2nd models

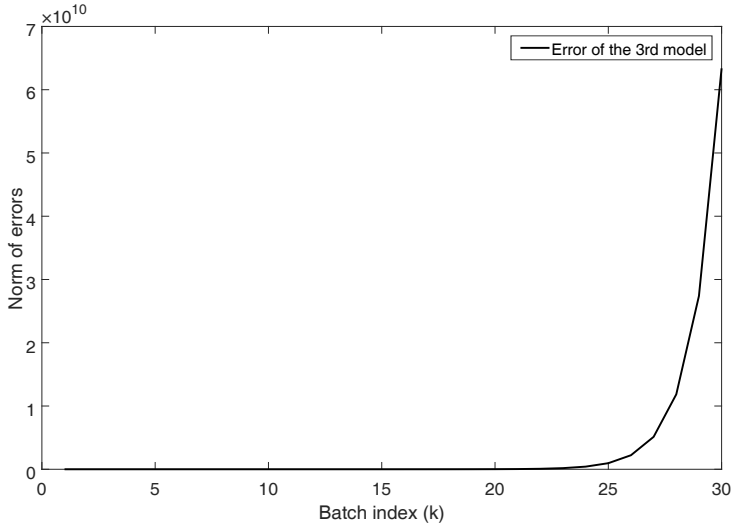


Figure 1.7: Convergence performance with the 3rd model

1.2.2 Iterative Learning Control Combined with Model Predictive Control

ILC was originally developed for robot manipulator control. Unlike the robot system, chemical process has overdamped nonlinear dynamics, significant interactions, large model errors, large disturbance and active constraints [4]. For these reasons, ILC has not been widely applied in the field of chemical process. To overcome these issues, many papers that propose an ILC combined with MPC have been published [70, 4, 71, 6, 72, 55, 73, 74, 10, 9, 75]. The first rigorous paper for ILC combined with MPC was proposed by Lee *et al.* [4, 71] and the technique is called batch MPC (BMPC). BMPC uses

the following periodically time-varying state-space model.

$$\begin{aligned}\begin{bmatrix}\bar{\mathbf{e}}_k(t+1)\\ \mathbf{e}_k(t+1)\end{bmatrix} &= \begin{bmatrix}\mathbf{I} & 0\\ 0 & \mathbf{I}\end{bmatrix}\begin{bmatrix}\bar{\mathbf{e}}_k(t)\\ \mathbf{e}_k(t)\end{bmatrix} - \begin{bmatrix}G(t)\\ G(t)\end{bmatrix}\Delta u_k(t)\\ e_k(t) &= \begin{bmatrix}0 & H(t)\end{bmatrix}\begin{bmatrix}\bar{\mathbf{e}}_k(t)\\ \mathbf{e}_k(t)\end{bmatrix}\end{aligned} \tag{1.25}$$

where $G(t)$ and $H(t)$ are defined as ($\mathbf{G}$ is the same as $\mathbf{G}$ in Eq. (1.13))

$$\mathbf{G} = \begin{bmatrix}G(0) & G(1) & \cdots & G(N-1)\end{bmatrix} \tag{1.26}$$

$$H(t) = \begin{bmatrix}0 & I & 0\end{bmatrix} \tag{1.27}$$

where I of $H(t)$ is located at t-th block column.

Example 1.2

If $N = 4$, $t = 2$ and the system has single-input single-output, $G(t)$ and $H(t)$ become

$$\begin{aligned}G(2) &= \begin{bmatrix}0 & 0 & CB & CAB\end{bmatrix}^T\\ H(2) &= \begin{bmatrix}0 & 1 & 0 & 0\end{bmatrix}\end{aligned} \tag{1.28}$$

BMPC uses the above state-space model to create a prediction model for the objective function. BMPC has been successfully applied to many systems [6, 7, 8]. However, even if the original system is time-invariant, BMPC should use time-varying parameters and the formulation of BMPC is somewhat complicate to combine with other techniques.

Two-stage approach has also been studied to combine ILC with real-time feedback controller [6, 7, 9]. This technique calculates the input of the ILC part and the input of the MPC part separately, then the sum of the two inputs is used as the actual input. It is effective for non-repetitive disturbances but requires two optimization steps. Also, constraints were not considered in this approach.

1.2.3 Iterative Learning Control for Point-to-Point Tracking

In many repetitive processes, the output does not need to track all points of a reference trajectory. PTP ILC was proposed by Lucibello [76] in 1992. TILC, a special case of PTP ILC, aims to track terminal point and has mainly been applied for quality control of systems which cannot measure the output in real time [77, 12, 13, 78, 19]. PTP ILC for tracking multiple points has been studied relatively recently compared to other ILC techniques [15, 79, 80, 81, 14, 16, 17]. For PTP tracking, there are two main approaches. The first is reference trajectory update-based approach [15]. In this approach, the reference trajectory that passes through desired points is updated at each batch, then the output tracks the updated reference trajectory. The second is direct tracking approach [16]. In this approach, the output converges to the desired points without the reference trajectory. All of the above PTP ILC algorithms are open-loop control within a batch. In order to apply PTP ILC to various applications, a real-time feedback controller should be integrated.

1.3 Major Contributions of This Thesis

The major contributions of this thesis are listed in the following:

- The standard form of ILC technique combined with MPC is proposed. The formulation and algorithm procedure of the proposed ILMPC is similar to conventional MPC; thus, various techniques for MPC can be applied to the proposed ILMPC without particular modification. Additional advantages include the simplicity of the formulation and low entry barriers.

- The case where the error converges to non-zero is studied. Most existing ILMPC techniques are designed to have an zero error as $k \rightarrow \infty$. However, zero error is not always preferable to non-zero error. An input trajectory for perfect tracking including vertices of a reference trajectory has a non-smooth trajectory. A penalty term for a smooth input trajectory is added to the objective function of the propose ILMPC and convergence analysis is performed.

- A novel ILMPC technique for tracking specific points is proposed. The all existing PTP ILC is open-loop control and the controller cannot reject disturbances. The PTP tracking problem in the ILMPC form is addressed by introducing an extraction matrix that extracts only the components related to specific points. If all the points of the whole operation time are regarded as specific points, PTP ILMPC becomes the same as ILMPC. Thus, the proposed PTP ILMPC includes all advantages of the proposed ILMPC and ILMPC can be seen as a special case of PTP ILMPC.

- Adaptive ILC schemes for discrete LTI stochastic system with batch-varying reference trajectory (BVRT) is proposed. If reference trajectories change every batch, ILC shows a different convergence property from that of the identical reference trajectory. This technique is not directly related to ILMPC, but can be integrated with ILMPC with minor modification. The limitation of this technique is that only linear system is considered.

1.4 Outline of This Thesis

The remainder of this thesis is organized as follows.

In Chapter 2, we propose a MPC technique combined with ILC for constrained multivariable control of batch processes. Although the general ILC makes the outputs converge to reference trajectories under model uncertainty, it uses open-loop control within a batch; thus, it cannot reject real-time disturbances. The MPC algorithm shows identical performance for all batches, and it highly depends on model quality because it does not use previous batch information. We integrate the advantages of the two algorithms. The proposed ILMPC formulation is based on general MPC and incorporates an iterative learning function into MPC. Thus, it is easy to handle various issues for which the general MPC is suitable, such as constraints, time-varying systems, disturbances, and stochastic characteristics. Simulation examples are provided to show the effectiveness of the proposed ILMPC.

In Chapter 3, the case in which the output error converges to non-zero value is studied. The existing ILMPC techniques make the error converge to zero. However, if the error converges to zero, an

impractical input trajectory may be calculated. We use a generalized objective function to independently tune weighting factors of manipulated variable change with respect to both the time index and batch horizons. If the generalized objective function is used, output error converges to non-zero values. We provide convergence analysis for both cases of zero convergence and non-zero convergence.

In Chapter 4, we propose a point-to-point ILMPC technique which can only consider the desired reference points, not an entire reference trajectory. It does not require to generate a reference trajectory which passes through the desired reference values. The existing ILMPC techniques aim to track a reference trajectory of repetitive process on a finite time interval while rejecting real-time disturbances. In many repetitive processes, however, the output does not need to track all points of a reference trajectory. In order to guarantee the convergence of tracking error, the suggested approach requires the error between measured and estimated outputs go to zero for all time as the number of iterations goes to infinity. However, neither classical observer nor Kalman filter guarantees the estimation error converge to zero for all time points. To overcome this issue, iterative learning observer (ILO) is applied to the algorithm and it can ensure that the estimation error go to zero for all time as the number of iterations goes to infinity. Numerical examples are provided to demonstrate the performances of the suggested approach on point-to-point tracking, iterative learning, constraints handling, and real-time disturbance rejection.

In Chapter 5, we present adaptive ILC schemes for discrete LTI stochastic system with BVRT. In this case, if the state noise and measurement noise exist,convergence rate and tracking performance are degraded because the controller considers the difference arising from

the noise as tracking error. To deal with such a problem, we propose two approaches. The first is based on a batch-domain Kalman filter, which uses the difference between the current output trajectory and the next reference trajectory as a state vector, while the second is based on a time-domain Kalman filter. In the second approach, the system is identified at the end of each batch in an iterative fashion using the observer/Kalman filter identification (OKID). Then, the stochastic problem is handled using Kalman filter with a steady-state Kalman gain obtained from the identification. Therefore, the second approach can track the reference trajectories of discrete LTI stochastic system using only the input–output information. Simulation examples are provided to show the effectiveness of the proposed schemes.

Finally, Conclusions and possible directions for further work are given in Section 6.

Chapter 2

Iterative Learning Control Combined with Model Predictive Control

In this chapter, we propose a MPC technique combined with ILC for constrained multivariable control of batch processes. Although the general ILC makes the outputs converge to reference trajectories under model uncertainty, it uses open-loop control within a batch; thus, it cannot reject real-time disturbances. The MPC algorithm shows identical performance for all batches, and it highly depends on model quality because it does not use previous batch information. We integrate the advantages of the two algorithms. The proposed ILMPC formulation is based on general MPC and incorporates an iterative learning function into MPC. Thus, it is easy to handle various issues for which the general MPC is suitable, such as constraints, time-varying systems, disturbances, and stochastic characteristics. Simulation examples are provided to show the effectiveness of the proposed ILMPC.

2.1 Introduction

Iterative learning control (ILC) is an effective control technique for improving the tracking performance of a batch process under

model uncertainty. ILC was originally introduced for robot manipulators [3] and has been implemented in many industrial processes, such as semiconductor manufacturing and chemical batch processes [12, 52]. In many ILC algorithms, the input sequences for the current batch are calculated using the tracking error sequences of the previous batch. This type of ILC algorithm uses open-loop control within a batch and cannot handle real-time disturbances. ILC should be integrated with real-time feedback control to reject real-time disturbances.

Model predictive control (MPC) has become the accepted standard for complex constrained multivariable control problems in the process industry. Some studies about ILC formulations combined with MPC, called iterative learning model predictive control (ILMPC), have been studied for handling real-time disturbances in batch processes. In case of combining ILC with MPC, it should include fundamental advantages of MPC as well as real-time feedback function. The following characteristics of MPC should be included in ILMPC. 1) ILMPC should guarantee offset-free control. 2) It should have a single optimization step, not two optimization steps for both ILC part and MPC part separately. 3) It should consider constraints and ensure that a feasible solution will always be found. 4) Prediction horizon should be able to be adjusted to reduce the computational load. 5) If the model is linear time-invariant (LTI), it should use LTI model directly. However, there are no studies about ILMPC algorithms which contain above all advantages. Most studies of ILMPC use a state-space model where a state vector consists of the entire error sequences of a batch, and a prediction horizon is fixed as the entire batch horizon [5, 6, 7, 8]. Thus, the control calculations may

not be performed within a sampling interval if a process has output constraints, a small sample time, long operation time, and many outputs. The prediction horizon should be adjusted to reduce the excessive computational load. In addition, the resulting formulation of such an algorithm employs linear time-varying (LTV) models in the state augmentation step, even if a system is LTI. Thus, additional calculation burdens for LTI systems are added in the algorithm. Two-stage approaches have also been proposed for combining ILC with real-time feedback [6, 7, 9]. The two-stage approaches have difficulties in system analysis and parameter tuning and requires two optimization steps. In addition, the time-wise feedback controllers of the approaches are not offset-free control; thus, offset occurs in the early batches until the batch-wise controller shows convergence. Above all, the two-stage approaches do not consider constraints. ILC combined with dynamic matrix control (DMC) for LTI system [10] has been proposed, but the DMC algorithm without an observer cannot handle unknown disturbance and measurement noise effectively [11].

In this chapter, we proposed ILMPC which contains fundamental advantages of MPC. 1) The proposed ILMPC guarantees offset-free control by introducing incremental state-space model. Therefore, outputs can track reference trajectories while rejecting disturbances at the first batch even if this is not perfect tracking. 2) This is one-stage approach and has single optimization step. 3) It considers constraints and includes slack variable; thus this algorithm ensures that a feasible solution will always be found. 4) Prediction and control horizon can be adjusted to reduce the computational load. 5) If the system is LTI, this algorithm uses LTI parameters directly. Finally, the formulation and algorithm procedure of the proposed ILMPC is similar to conven-

tional MPC; thus various techniques for MPC can be applied for the proposed ILMPC without particular modification. The reason which these advantages can be included in the proposed algorithm is that a prediction model formulated by an input-output model between two adjacent batches is directly applied to the algorithm in an identical way as a conventional MPC.

2.2 Prediction Model for Iterative Learning Model Predictive Control

2.2.1 Incremental State-Space Model

2.2.1.1 Delta input formulation

Fisrt, we consider the following linear discrete time-invariant system which operates on an interval $t \in [0, N]$:

$$\begin{aligned} \bar{x}_k(t+1) &= \bar{A}\bar{x}_k(t) + \bar{B}u_k(t) \\ y_k(t) &= \bar{C}\bar{x}_k(t) \end{aligned} \tag{2.1}$$

where $\bar{x}_k(t) \in \mathbb{R}^{n_x}$ is the state vector; $u_k(t) \in \mathbb{R}^{n_u}$ is the input vector; $y_k(t) \in \mathbb{R}^{n_y}$ is the output vector; t is the time index; k is the batch index; and the matrices $\bar{A},\ \bar{B}$, and $\bar{C}$ are real matrices of appropriate dimensions. An incremental state-space model uses the control increment instead of the control signal. This model can be written in the general state-space form with $\delta u_k(t) = u_k(t) - u_k(t-1)$. The follow-

ing representation is the augmented incremental state-space model:

$$\begin{aligned}\begin{bmatrix}\bar{x}_k(t+1)\\ u_k(t)\end{bmatrix} &= \overbrace{\begin{bmatrix}\bar{A} & \bar{B}\\ 0 & I\end{bmatrix}}^{A}\overbrace{\begin{bmatrix}\bar{x}_k(t)\\ u_k(t-1)\end{bmatrix}}^{x_k(t)} + \overbrace{\begin{bmatrix}\bar{B}\\ I\end{bmatrix}}^{B}\delta u_k(t)\\ y_k(t) &= \underbrace{\begin{bmatrix}\bar{C} & 0\end{bmatrix}}_{C}\begin{bmatrix}\bar{x}_k(t)\\ u_k(t-1)\end{bmatrix}\end{aligned} \tag{2.2}$$

The characteristic polynomial equation of the augmented model is

$$\rho(\lambda) = \det\begin{bmatrix}\lambda I - \bar{A} & -\bar{B}\\ 0 & I - \lambda I\end{bmatrix} = (\lambda - 1)^{n_u}\det(\lambda I - \bar{A}) = 0 \tag{2.3}$$

This means that there are n_u integrators are embedded in the augmented model. Defining a new state vector as

$$x_k(t) \triangleq \begin{bmatrix}\bar{x}_k(t)\\ u_k(t-1)\end{bmatrix} \tag{2.4}$$

the incremental model takes the following general form:

$$\begin{aligned}x_k(t+1) &= Ax_k(t) + B\delta u_k(t)\\ y_k &= Cx_k\end{aligned} \tag{2.5}$$

It is called an incremental state-space model [82] or a state-space model with embedded integrator [83].

Proposition 2.1. *[84] The augmented system (2.2) is observable if*

and only if $(\bar{C},\ \bar{A})$ *is observable and*

$$\begin{bmatrix} \bar{A}-I & \bar{B} \\ \bar{C} & 0 \end{bmatrix} \text{ has full column rank.} \tag{2.6}$$

Proof From the Hautus observability condition, system (2.2) is observable if and only if

$$\begin{bmatrix} \bar{A}-\lambda I & \bar{B} \\ 0 & I-\lambda I \\ \bar{C} & 0 \end{bmatrix} \text{ has full column rank } \forall\lambda. \tag{2.7}$$

From the Hautus condition, the first set of columns is linearly independent if and only if $(\bar{C},\ \bar{A})$ is observable. The second set of rows is linearly independent because of the identity matrix except possibly for $\lambda = 1$. Thus, for the augmented system, the Hautus condition needs to be checked for $\lambda = 1$ only. ■

Remark 2.1. *Observability of the augmented system (2.2) is lost if the number of inputs is more than the number of outputs or* $\bar{B}$ *has not full column rank.*

2.2.1.2 Velocity form

The delta input formulation has the disadvantage of losing observability if the number of inputs is more than the number of outputs. The alternative formulation, velocity form [83], is always observable if the original state-space model is observable. First, taking a difference operation on both sides of the state equation of the original state

space model (2.1), we obtain that

$$\bar{x}_k(t+1) - \bar{x}_k(t) = \bar{A}\left(\bar{x}_k(t) - \bar{x}_k(t-1)\right) + \bar{B}\left(u_k(t) - u_k(t-1)\right) \tag{2.8}$$

Let us denote the difference of the state variable by

$$\delta\bar{x}_k(t+1) = \bar{x}_k(t+1) - \bar{x}_k(t) \tag{2.9}$$

and the difference of the control variable by

$$\delta u_k(t) = u_k(t) - u_k(t-1) \tag{2.10}$$

The difference of the state-space model is as follows.

$$\delta\bar{x}_k(t+1) = \bar{A}\delta\bar{x}_k(t) + \bar{B}\delta u_k(t) \tag{2.11}$$

The next step is to connect $\delta\bar{x}_k(t)$ to the output $y_k(t)$. A new state vector is chosen to be

$$x_k(t) = \begin{bmatrix} \delta\bar{x}_k(t) \\ y_k(t) \end{bmatrix} \tag{2.12}$$

Note that

$$\begin{aligned} y_k(t+1) - y_k(t) &= \bar{C}\delta x_k(t+1) \\ &= \bar{C}\bar{A}\delta\bar{x}_k(t) + \bar{C}\bar{B}\delta u_k(t) \end{aligned} \tag{2.13}$$

These lead to the following augmented state-space model:

$$\begin{aligned}\begin{bmatrix}\delta\bar{x}_k(t+1)\\ y_k(t+1)\end{bmatrix} &= \overbrace{\begin{bmatrix}\bar{A} & 0\\ \bar{C}\bar{A} & I\end{bmatrix}}^{A}\overbrace{\begin{bmatrix}\delta\bar{x}_k(t)\\ y_k(t)\end{bmatrix}}^{x_k(t)} + \overbrace{\begin{bmatrix}\bar{B}\\ \bar{C}\bar{B}\end{bmatrix}}^{B}\delta u_k(t)\\ y_k(t) &= \underbrace{\begin{bmatrix}0 & I\end{bmatrix}}_{C}\begin{bmatrix}\delta\bar{x}_k(t)\\ y_k(t)\end{bmatrix}\end{aligned} \tag{2.14}$$

The characteristic polynomial equation of the augmented model is

$$\rho(\lambda) = \det\begin{bmatrix}\lambda I - \bar{A} & 0\\ -\bar{C}\bar{A} & I - \lambda I\end{bmatrix} = (\lambda-1)^{n_y}\det(\lambda I - \bar{A}) = 0 \tag{2.15}$$

This means that there are n_y integrators are embedded in the augmented model.

Proposition 2.2. *The augmented system (2.14) is observable if and only if* $(\bar{C},\ \bar{A})$ *is observable and*

$$\begin{bmatrix}\bar{A} - I & 0\\ \bar{C}\bar{A} & 0\\ 0 & I\end{bmatrix} \quad \text{has full column rank.} \tag{2.16}$$

Proof From the Hautus observability condition, system (2.14) is observable if and only if

$$\begin{bmatrix}\bar{A} - \lambda I & 0\\ \bar{C}\bar{A} & I - \lambda I\\ 0 & I\end{bmatrix} \quad \text{has full column rank } \forall\lambda. \tag{2.17}$$

From the Hautus condition, the first set of columns is linearly independent if and only if $(\bar{C},\ \bar{A})$ is observable. The second set of rows is linearly independent because of the identity matrix except possibly for $\lambda = 1$. Thus, for the augmented system, the Hautus condition needs to be checked for $\lambda = 1$ only. ■

2.2.2 Prediction Model

The system (2.5) can be rewritten as a lifted system because finite time intervals $[0, N]$ are considered in ILMPC:

$$\hat{\mathbf{y}}_k = \mathbf{G}_m \delta \mathbf{u}_k + \mathbf{F}_m x_k(0) \tag{2.18}$$

where $\mathbf{G}_m \in \mathbb{R}^{(n_y N)\times(n_u N)}$ and $\mathbf{F}_m \in \mathbb{R}^{(n_y N)\times n_x}$ are defined as

$$\mathbf{G}_m \triangleq \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2}B & \cdots & CB \end{bmatrix}, \quad \mathbf{F}_m \triangleq \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix} \tag{2.19}$$

and the vectors $\hat{\mathbf{y}}_k \in \mathbb{R}^{n_y N}$ and $\mathbf{u}_k \in \mathbb{R}^{n_u N}$ are defined as

$$\hat{\mathbf{y}}_k \triangleq \begin{bmatrix} \hat{y}_k(1)^T & \hat{y}_k(2)^T & \cdots & \hat{y}_k(N)^T \end{bmatrix}^T \tag{2.20}$$

$$\delta \mathbf{u}_k \triangleq \begin{bmatrix} \delta u_k(0)^T & \delta u_k(1)^T & \cdots & \delta u_k(N-1)^T \end{bmatrix}^T \tag{2.21}$$

The input-output relationship between two adjacent batches is

$$\hat{\mathbf{y}}_k = \mathbf{y}_{k-1} + \mathbf{G}_m \Delta \delta \mathbf{u}_k + \mathbf{F}_m \Delta x_k(0) \tag{2.22}$$

where $\hat{}$ (hat) means predicted value, δ is a time-increment operator and Δ is a batch-increment operator. That is,

$$\begin{aligned} \Delta\delta u_k(t) &= \{u_k(t) - u_k(t-1)\} - \{u_{k-1}(t) - u_{k-1}(t-1)\} \\ \Delta x_k(0) &= x_k(0) - x_{k-1}(0) \end{aligned} \tag{2.23}$$

Then, the following representation can be obtained using $\mathbf{e}_k = \mathbf{r} - \mathbf{y}_k$ where $\mathbf{r}$ is the reference trajectory.

$$\hat{\mathbf{e}}_k = \mathbf{e}_{k-1} - \mathbf{G}_m \Delta\delta \mathbf{u}_k - \mathbf{F}_m \Delta x_k(0) \tag{2.24}$$

The basic assumption of ILC is an identical initialization condition ($x_k(0) = x_{k-1}(0)$); thus, $\Delta x_k(0)$ is zero [3, 25]. However, we do not remove the initial state term because it is used for deriving free response term including estimated current states which are necessary for real-time feedback. At time t of the k^{th} batch, future predictions up to a prediction horizon p are formulated in terms of future control movements up to a control horizon m, previous control movements, and initial state. In the ILMPC algorithm, both horizons should not exceed remaining time points. Therefore, the concept of shrinking horizons [85] is used. Both horizons are updated as

$$\begin{aligned} p &= \begin{cases} p_0 & \text{, if } p_0 \leq N - t \\ N - t & \text{, otherwise} \end{cases} \\ m &= \begin{cases} m_0 & \text{, if } m_0 \leq N - t \\ N - t & \text{, otherwise} \end{cases} \end{aligned} \tag{2.25}$$

where p_0 is the initial prediction horizon and m_0 is the initial control horizon.

$$\begin{bmatrix} \hat{e}_k(t+1) \\ \hat{e}_k(t+2) \\ \vdots \\ \hat{e}_k(t+p) \end{bmatrix} = \begin{bmatrix} e_{k-1}(t+1) \\ e_{k-1}(t+2) \\ \vdots \\ e_{k-1}(t+p) \end{bmatrix}$$

$$\overbrace{\begin{array}{l} -\begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^{p-2}B & \cdots & CA^{p-m}B \end{bmatrix} \begin{bmatrix} \Delta\delta u_k(t) \\ \Delta\delta u_k(t+1) \\ \vdots \\ \Delta\delta u_k(t+m-1) \end{bmatrix} \\ \underbrace{-\begin{bmatrix} CA^{t}B & CA^{t-1}B & \cdots & CAB \\ CA^{t+1}B & CA^{t}B & \cdots & CA^{2}B \\ \vdots & \vdots & \ddots & \vdots \\ CA^{t+p-1}B & CA^{t+p-2}B & \cdots & CA^{p}B \end{bmatrix} \begin{bmatrix} \Delta\delta u_k(0) \\ \Delta\delta u_k(1) \\ \cdots \\ \Delta\delta u_k(t-1) \end{bmatrix}}_{\text{Free response}} \end{array}}^{\text{Forced response}}$$

$$\underbrace{-\begin{bmatrix} CA^{t+1} \\ CA^{t+2} \\ \vdots \\ CA^{t+p} \end{bmatrix} \Delta x_k(0)}_{\text{Free response}} \tag{2.26}$$

In the equation, free response term can be simplified and represented in terms of current states. The following is the example of the first

row of the free response vector.

$$
\begin{aligned}
&- CA^{t}B\Delta\delta u_k(0) - CA^{t-1}B\Delta\delta u_k(1)\cdots - CAB\Delta\delta u_k(t-1) \\
&- CA^{t+1}\Delta x_k(0) \\
=&- CA^{t}\left(A\Delta x_k(0) + B\Delta\delta u_k(0)\right) - CA^{t-1}B\Delta\delta u_k(1)\cdots \\
&- CAB\Delta\delta u_k(t-1) \\
=&- CA^{t}\Delta x_k(1) - CA^{t-1}B\Delta\delta u_k(1)\cdots - CAB\Delta\delta u_k(t-1) \\
&\vdots \\
=&- CA\Delta x_k(t)
\end{aligned}
\tag{2.27}
$$

As a result, prediction model is represented as follows:

$$
\hat{\mathbf{e}}_k^p(t+1|t) = \mathbf{e}_{k-1}^p(t+1) - \mathbf{G}\Delta\delta\mathbf{u}_k^m(t) - \mathbf{F}\Delta\hat{x}_k(t|t) \tag{2.28}
$$

$$
\mathbf{G} \triangleq \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^{p-2}B & \cdots & CA^{p-m}B \end{bmatrix}, \quad \mathbf{F} \triangleq \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^p \end{bmatrix} \tag{2.29}
$$

$$
\begin{aligned}
\hat{\mathbf{e}}_k^p(t+1|t) &\triangleq \begin{bmatrix} \hat{e}_k(t+1|t)^T & \hat{e}_k(t+2|t)^T & \cdots & \hat{e}_k(t+p|t)^T \end{bmatrix}^T \\
\mathbf{e}_{k-1}^p(t+1) &\triangleq \begin{bmatrix} e_{k-1}(t+1)^T & e_{k-1}(t+2)^T & \cdots & e_{k-1}(t+p)^T \end{bmatrix}^T \\
\Delta\delta\mathbf{u}_k^m(t) &\triangleq \begin{bmatrix} \Delta\delta u_k(t)^T & \Delta\delta u_k(t+1)^T & \cdots & \Delta\delta u_k(t+m-1)^T \end{bmatrix}^T
\end{aligned}
\tag{2.30}
$$

We assume that all the states are not measurable; hence, we use state estimates $\Delta\hat{x}_k(t|t)$ instead of $\Delta x_k(t)$.

2.3 Iterative Learning Model Predictive Controller

2.3.1 Unconstrained ILMPC

We can design the ILMPC controller with the following objective function.

$$\min_{\Delta\delta\mathbf{u}_k^m(t)} J = \frac{1}{2}\left\{\hat{\mathbf{e}}_k^p(t+1|t)^T\mathbf{Q}\hat{\mathbf{e}}_k^p(t+1|t) + \Delta\delta\mathbf{u}_k^m(t)^T\mathbf{R}\Delta\delta\mathbf{u}_k^m(t)\right\} \tag{2.31}$$

Substituting Eq. (2.28) into the objective function (2.31) with $\partial J/\partial\Delta\delta\mathbf{u}_k^m(t) = 0$ yields

$$\begin{aligned}\delta\mathbf{u}_k^m(t) &= \delta\mathbf{u}_{k-1}^m(t) + \bar{\mathbf{H}}\left(\mathbf{e}_{k-1}^p(t+1) - \mathbf{F}\Delta\hat{x}_k(t|t)\right)\\ \bar{\mathbf{H}} &= \left(\mathbf{G}^T\mathbf{Q}\mathbf{G} + \mathbf{R}\right)^{-1}\mathbf{G}^T\mathbf{Q}\end{aligned} \tag{2.32}$$

Among the optimal control actions, only the first control action is implemented as the current control law. The following is the ILMPC control law for unconstrained processes.

$$\begin{aligned}\delta u_k(t) &= \delta u_{k-1}(t) + \mathbf{H}\left(\mathbf{e}_{k-1}^p(t+1) - \mathbf{F}\Delta\hat{x}_k(t|t)\right)\\ \mathbf{H} &= \overbrace{\begin{bmatrix} I & 0 & 0 & \cdots & 0\end{bmatrix}}^{m \text{ block matrices}}\bar{\mathbf{H}}\end{aligned} \tag{2.33}$$

where $\mathbf{H}$ is called the learning gain matrix and $\Delta\hat{x}_k(t|t) = \hat{x}_k(t|t) - x_{k-1}(t)$ can be estimated using Kalman filter.

$$\begin{aligned}\Delta\hat{x}_k(t|t-1) &= A\Delta\hat{x}_k(t-1|t-1) + B\Delta\delta u_k(t-1)\\ \Delta\hat{x}_k(t|t) &= \Delta\hat{x}_k(t|t-1) + K\left(\Delta y_k(t) - C\Delta\hat{x}_k(t|t-1)\right)\end{aligned} \tag{2.34}$$

where K is a steady-state Kalman gain.

2.3.2 Constrained ILMPC

In many control applications, constraints are imposed on processes for safety and smooth operations. In ILMPC controller, constraints are composed of upper and lower limits on the input values ($\mathbf{u}^m_{min} \leq \mathbf{u}^m_k(t) \leq \mathbf{u}^m_{max}$), the rate of input change with respect to the time index ($\delta\mathbf{u}^m_{min} \leq \delta\mathbf{u}_k(t) \leq \delta\mathbf{u}^m_{max}$), the rate of input change with respect to the batch index ($\Delta\mathbf{u}^m_{min} \leq \Delta\mathbf{u}^m_k(t) \leq \Delta\mathbf{u}^m_{max}$), and output values ($\mathbf{y}^p_{min} - \varepsilon^p_k(t+1) \leq \hat{\mathbf{y}}^p_k(t+1|t) \leq \mathbf{y}^p_{max} + \varepsilon^p_k(t+1)$). $\varepsilon^p_k(t+1)$, called slack variable, is defined such that it is non-zero only if a constraint is violated, and ensures that a feasible solution will always be found. This is referred to as constraint softening [86, 87]. Each constraint should be expressed with respect to $\Delta\delta\mathbf{u}^m_k(t)$ as follows:

- Input values

$$\mathbf{u}^m_k(t) = \mathbf{I}_m u_k(t-1) + \mathbf{I}_L \delta\mathbf{u}^m_{k-1}(t) + \mathbf{I}_L \Delta\delta\mathbf{u}^m_k(t) \tag{2.35}$$

- The rate of input change with respect to the time index

$$\delta\mathbf{u}^m_k(t) = \delta\mathbf{u}^m_{k-1}(t) + \Delta\delta\mathbf{u}^m_k(t) \tag{2.36}$$

- The rate of input change with respect to the batch index

$$\Delta\mathbf{u}^m_k(t) = \mathbf{I}_m \Delta u_k(t-1) + \mathbf{I}_L \Delta\delta\mathbf{u}^m_k(t) \tag{2.37}$$

- Output values

$$\hat{\mathbf{y}}_k^p(t+1|t) = \mathbf{y}_{k-1}^p(t+1) + \mathbf{G}\Delta\delta\mathbf{u}_k^m(t) + \mathbf{F}\Delta\hat{x}_k(t|t) \quad (2.38)$$

where

$$\mathbf{I}_m \triangleq \begin{bmatrix} I \\ I \\ \vdots \\ I \end{bmatrix}, \quad \mathbf{I}_L \triangleq \begin{bmatrix} I & 0 & \cdots & 0 \\ I & I & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ I & I & \cdots & I \end{bmatrix} \quad (2.39)$$

The following is a cost function incorporating the slack variable and omitting constant terms.

$$\begin{aligned} \min_{\Delta\delta\mathbf{u}_k^m(t),\, \varepsilon_k^p(t+1)} & \frac{1}{2}\Delta\delta\mathbf{u}_k^m(t)^T \left(\mathbf{G}^T\mathbf{Q}\mathbf{G} + \mathbf{R}\right)\Delta\delta\mathbf{u}_k^m(t) \\ & + \left(\mathbf{G}^T\mathbf{Q}\mathbf{F}\Delta\hat{x}_k(t|t) - \mathbf{G}^T\mathbf{Q}\mathbf{e}_{k-1}^p(t+1)\right)^T\Delta\delta\mathbf{u}_k^m(t) \\ & + \frac{1}{2}\varepsilon_k^p(t+1)^T\mathbf{S}\varepsilon_k^p(t+1) \end{aligned} \quad (2.40)$$

The cost function can be rewritten by combining $\Delta\delta\mathbf{u}_k^m(t)$ and $\varepsilon_k^p(t+1)$ into one vector. The following is a standard quadratic programming (QP) problem for constrained ILMPC.

$$\begin{aligned} \min_{\Delta\delta\mathbf{u}_k^m(t),\, \varepsilon_k^p(t+1)} & \frac{1}{2}\begin{bmatrix}\Delta\delta\mathbf{u}_k^m(t) \\ \varepsilon_k^p(t+1)\end{bmatrix}^T \begin{bmatrix}\mathbf{G}^T\mathbf{Q}\mathbf{G} + \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}\end{bmatrix}\begin{bmatrix}\Delta\delta\mathbf{u}_k^m(t) \\ \varepsilon_k^p(t+1)\end{bmatrix} \\ & + \begin{bmatrix}\mathbf{G}^T\mathbf{Q}\left(\mathbf{F}\Delta\hat{x}_k(t|t) - \mathbf{e}_{k-1}^p(t+1)\right) \\ \mathbf{0}\end{bmatrix}^T\begin{bmatrix}\Delta\delta\mathbf{u}_k^m(t) \\ \varepsilon_k^p(t+1)\end{bmatrix} \end{aligned} \quad (2.41)$$

subject to

$$\mathbf{E}\begin{bmatrix}\Delta\delta\mathbf{u}_k^m(t)\\ \varepsilon_k^p(t+1)\end{bmatrix} \leq \mathbf{M}_k(t) \tag{2.42}$$

where

$$\mathbf{E} \triangleq \begin{bmatrix} -\mathbf{I}_L & \mathbf{0}\\ \mathbf{I}_L & \mathbf{0}\\ -\mathbf{I} & \mathbf{0}\\ \mathbf{I} & \mathbf{0}\\ -\mathbf{I}_L & \mathbf{0}\\ \mathbf{I}_L & \mathbf{0}\\ -\mathbf{G} & -\mathbf{I}\\ \mathbf{G} & -\mathbf{I}\\ \mathbf{0} & -\mathbf{I}\end{bmatrix},\ \mathbf{M}_k(t) \triangleq \begin{bmatrix} -\mathbf{u}_{min}^m + \mathbf{I}_m u_k(t-1) + \mathbf{I}_L\delta\mathbf{u}_{k-1}^m(t)\\ \mathbf{u}_{max}^m - \mathbf{I}_m u_k(t-1) - \mathbf{I}_L\delta\mathbf{u}_{k-1}^m(t)\\ -\delta\mathbf{u}_{min}^m + \delta\mathbf{u}_{k-1}^m(t)\\ \delta\mathbf{u}_{max}^m - \delta\mathbf{u}_{k-1}^m(t)\\ -\Delta\mathbf{u}_{min}^m + \mathbf{I}_m\Delta u_k(t-1)\\ \Delta\mathbf{u}_{max}^m - \mathbf{I}_m\Delta u_k(t-1)\\ -\mathbf{y}_{min}^p + \mathbf{y}_{k-1}^p(t+1) + \mathbf{F}\Delta\hat{x}(t|t)\\ \mathbf{y}_{max}^p - \mathbf{y}_{k-1}^p(t+1) - \mathbf{F}\Delta\hat{x}(t|t)\\ \mathbf{0}\end{bmatrix} \tag{2.43}$$

The optimization problem can be solved by appropriate QP solver. The first input in the optimal sequence is then sent into the plant.

2.3.3 Convergence Property

We will provide sufficient conditions for asymptotic stability and monotonic convergence of errors along the batch direction. First, we derive the estimated state in terms of input and output using the

steady-state Kalman gain.

$$\begin{aligned}\Delta\hat{x}_k(t|t) =& \Delta\hat{x}_k(t|t-1) + K\left(\Delta y_k(t) - \Delta\hat{x}_k(t|t-1)\right)\\ =& A\Delta\hat{x}_k(t-1|t-1) + B\Delta\delta u_k(t-1)\\ &+ K\Big(\Delta y_k(t) - CA\Delta\hat{x}_k(t-1|t-1)\\ &- CB\Delta\delta u_k(t-1)\Big)\\ =& A_K\Delta\hat{x}_k(t-1|t-1) + B_K\Delta\delta u_k(t-1) + K\Delta y_k(t)\end{aligned} \tag{2.44}$$

where $A_K \triangleq A - KCA$ and $B_K \triangleq B - KCB$. The recursive formulation can be represented in terms of $\Delta\delta u_k(t)$ and $\Delta y_k(t)$ as follows:

$$\Delta\hat{x}(t|t) = \sum_{i=0}^{t-1} A_K^i B_K \Delta\delta u_k(t-1-i) + \sum_{i=0}^{t-1} A_K^i K \Delta y_k(t-i) \tag{2.45}$$

Then, we should derive $\Delta\delta u_k(t)$ using input update law (2.33) and $\Delta\hat{x}_k(t|t)$ of Eq. (2.45). We assume that tracking errors after the terminal point N are zero, $e_k(N+1) = e_k(N+2) = \cdots = e_k(N+p-1) = 0$, in order to use an identical learning gain matrix in all input updates. The input update law (2.33) can be rearranged as follows:

$$\Delta\delta u_k(t) = \mathbf{H}\mathbf{e}_{k-1}^p(t+1) - \mathbf{H}\mathbf{F}\Delta\hat{x}_k(t|t) \tag{2.46}$$

First, we define $\mathbf{H} \triangleq \begin{bmatrix} H_1 & H_2 & \cdots & H_p \end{bmatrix}$, where $H_i,\ i = 1,\ \cdots,\ p$ is the i^{th} block column of the learning gain matrix. In Eq. (2.45), $\Delta y_k(t) = -\Delta e_k(t)$ because definition of tracking error is $e_k(t) \triangleq r(t) - y_k(t)$. The input sequences can be represented in terms of track-

ing errors using the above relationships and definitions.

$$\begin{aligned}
\Delta\delta u_k(0) &= \mathbf{He}^p_{k-1}(1) - \mathbf{HF}\Delta\hat{x}(0|0) \\
&= H_1 e_{k-1}(1) + H_2 e_{k-1}(2) + \cdots + H_p e_{k-1}(p) \\
\Delta\delta u_k(1) &= \mathbf{He}^p_{k-1}(2) - \mathbf{HF}\Delta\hat{x}(1|1) \\
&= \mathbf{He}^p_{k-1}(2) - \mathbf{HF}B_K\Delta\delta u_k(0) + \mathbf{HF}K\Delta e_k(1) \\
&\vdots \\
\Delta\delta u_k(N-1) &= \mathbf{He}^p_{k-1}(N) - \mathbf{HF}\sum_{i=0}^{N-2} A_K^i B_K \Delta\delta u_k(N-2-i) \\
&\quad + \mathbf{HF}\sum_{i=0}^{N-2} A_K^i K \Delta e_k(N-1)
\end{aligned} \tag{2.47}$$

These equations for a sequence of time steps can be written as

$$\begin{aligned}
\begin{bmatrix} \Delta\delta u_k(0) \\ \Delta\delta u_k(1) \\ \vdots \\ \Delta\delta u_k(N-1) \end{bmatrix} &= \begin{bmatrix} H_1 & H_2 & \cdots & H_p & 0 & \cdots & 0 \\ 0 & H_1 & H_2 & \cdots & H_p & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & H_1 \end{bmatrix} \begin{bmatrix} e_{k-1}(1) \\ e_{k-1}(2) \\ \vdots \\ e_{k-1}(N) \end{bmatrix} \\
&- \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \mathbf{HF}B_K & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{HF}A_K^{N-2}B_K & \mathbf{HF}A_K^{N-3}B_K & \cdots & 0 \end{bmatrix} \begin{bmatrix} \Delta\delta u_k(0) \\ \Delta\delta u_k(1) \\ \vdots \\ \Delta\delta u_k(N-1) \end{bmatrix} \\
&+ \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \mathbf{HF}K & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{HF}A_K^{N-2}K & \mathbf{HF}A_K^{N-3}K & \cdots & 0 \end{bmatrix} \begin{bmatrix} \Delta e_k(1) \\ \Delta e_k(2) \\ \vdots \\ \Delta e_k(N) \end{bmatrix}
\end{aligned} \tag{2.48}$$

The equation can be rearranged as follows:

$$\mathbf{H}_u \Delta\delta\mathbf{u}_k = \mathbf{H}_f \mathbf{e}_{k-1} + \mathbf{H}_b \mathbf{e}_k \tag{2.49}$$

where

$$\begin{aligned}
\mathbf{H}_u &\triangleq \begin{bmatrix} I & 0 & \cdots & 0 \\ \mathbf{HF}B_K & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{HF}A_K^{N-2}B_K & \mathbf{HF}A_K^{N-3}B_K & \cdots & I \end{bmatrix} \\
\mathbf{H}_f &\triangleq \begin{bmatrix} H_1 & H_2 & \cdots & 0 \\ -\mathbf{HF}K & H_1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -\mathbf{HF}A_K^{N-2}K & -\mathbf{HF}A_K^{N-3}K & \cdots & H_1 \end{bmatrix} \\
\mathbf{H}_b &\triangleq \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \mathbf{HF}K & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{HF}A_K^{N-2}K & \mathbf{HF}A_K^{N-3}K & \cdots & 0 \end{bmatrix}
\end{aligned} \tag{2.50}$$

Remark 2.2. *[Properties of block triangular matrices]*

- *A triangular matrix is invertible if and only if its diagonal entries are all non zero.*
- *The product of lower triangular matrices is lower triangular.*
- *The inverse of an invertible lower triangular matrix is lower triangular.*
- *Eigenvalues of a block triangular matrix are identical to the eigenvalues of diagonal blocks.*

- *Let A, B, and C be $n \times m$, $m \times m$, and $m \times n$ block lower triangular matrices, respectively. If all the block diagonals of any of A, B, or C are zero block, then all the block diagonals of ABC are zero block. Thus, $I + ABC$ is invertible.*

According to Remark 2.2, $\mathbf{H}_u$ is invertible. Thus, the relationship between input and tracking errors can be written as

$$\Delta\delta\mathbf{u}_k = \mathbf{H}_u^{-1}\mathbf{H}_f\mathbf{e}_{k-1} + \mathbf{H}_u^{-1}\mathbf{H}_b\mathbf{e}_k \tag{2.51}$$

Then, we use plant dynamics equation to obtain a relationship between errors of two adjacent batches. The system has the same initial condition for all batches. Therefore, $x_k(0) = x_{k-1}(0)$, that is, $\Delta x_k(0) = 0$.

$$\mathbf{e}_k = \mathbf{e}_{k-1} - \mathbf{G}_p\Delta\delta\mathbf{u}_k + \cancelto{0}{\mathbf{F}\Delta x_k(0)} \tag{2.52}$$

where $\mathbf{G}_p$ is the plant matrix. Substituting Eq. (2.51) into Eq. (2.52) yields

$$\left(\mathbf{I} + \mathbf{G}_p\mathbf{H}_u^{-1}\mathbf{H}_b\right)\mathbf{e}_k = \left(\mathbf{I} - \mathbf{G}_p\mathbf{H}_u^{-1}\mathbf{H}_f\right)\mathbf{e}_{k-1} \tag{2.53}$$

where $\mathbf{G}_p$, $\mathbf{H}_u^{-1}$, and $\mathbf{H}_b$ are lower block triangular matrices; all the block diagonals of $\mathbf{H}_b$ are zero block. Thus, $\mathbf{I} + \mathbf{G}_p\mathbf{H}_u^{-1}\mathbf{H}_b$ is invertible according to Remark 2.2. From Eq. (2.53), the error propagation is expressed as

$$\mathbf{e}_k = \mathbf{\Phi}\mathbf{e}_{k-1} \tag{2.54}$$

where

$$\mathbf{\Phi} \triangleq \left(\mathbf{I} + \mathbf{G}_p\mathbf{H}_u^{-1}\mathbf{H}_b\right)^{-1}\left(\mathbf{I} - \mathbf{G}_p\mathbf{H}_u^{-1}\mathbf{H}_f\right) \tag{2.55}$$

Theorem 2.1. *[88] Consider the linear system (2.5) and the ILMPC controller (2.33). The system converges asymptotically to zero as* $k \to \infty$ *if* $\rho(\mathbf{\Phi}) < 1$*, where* $\rho(\cdot)$ *is a spectral radius.*

Theorem 2.2. *[88] Consider the linear system (2.5) and the ILMPC controller (2.33). The system converges monotonically to zero as* $k \to \infty$ *if* $\|\mathbf{\Phi}\|_i < 1$*, where* i *is norm topology (1, 2, or* ∞*).*

2.3.4 Extension for Disturbance Model

The proposed ILMPC is easy to extend to the general state-space model including measured disturbance model because its formulation is similar to MPC. In general, unmeasured disturbance model is included in the state-space model for offset-free control. The incremental state-space model which is used in the propose algorithm guarantees offset-free control [89]. Thus, unmeasured disturbance model is not considered in the proposed method. The following general model is considered.

$$\begin{aligned} \bar{x}_k(t+1) &= \bar{A}\bar{x}_k(t) + \bar{B}u_k(t) + \bar{B}_d d_k(t) + \bar{\Gamma} w_k^x(t) \\ u_k(t) &= u_k(t-1) + \delta u_k(t) + w_k^u(t) \\ y_k(t) &= \bar{C}\bar{x}_k(t) + n_k(t) \end{aligned} \tag{2.56}$$

where $d_k(t)$ is measured disturbance, $w_k^x(t)$ is state noise, $n_k(t)$ is measurement noise, and $w_k^u(t)$ is white noise which is added for solv-

ability of the Riccati equation with the following covariance matrices:

$$\begin{aligned} E\left[w_k^x(t)\left\{w_k^x(t)\right\}^T\right] &= Q_x \\ E\left[w_k^u(t)\left\{w_k^u(t)\right\}^T\right] &= Q_u \\ E\left[n_k(t)\left\{n_k(t)\right\}^T\right] &= R_n \end{aligned} \tag{2.57}$$

The general model can be written in the following augmented state-space:

$$\begin{aligned} \begin{bmatrix} \bar{x}_k(t+1) \\ u_k(t) \end{bmatrix} &= \begin{bmatrix} \bar{A} & \bar{B} \\ 0 & I \end{bmatrix} \begin{bmatrix} \bar{x}_k(t) \\ u_k(t-1) \end{bmatrix} + \begin{bmatrix} \bar{B} \\ I \end{bmatrix} \delta u_k(t) \\ &\quad + \begin{bmatrix} \bar{B}_d \\ 0 \end{bmatrix} d_k(t) + \begin{bmatrix} \bar{\Gamma} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} w_k^x(t) \\ w_k^u(t) \end{bmatrix} \\ y_k(t) &= \begin{bmatrix} \bar{C} & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_k(t) \\ u_k(t-1) \end{bmatrix} + n_k(t) \end{aligned} \tag{2.58}$$

$$\begin{aligned} x_k(t+1) &= Ax_k(t) + B\delta u_k(t) + B_d d_k(t) + \Gamma w_k(t) \\ y_k(t) &= Cx_k(t) + n_k(t) \end{aligned} \tag{2.59}$$

In the augmented state-space model, deterministic parts are used for prediction model and stochastic parts are used for state estimation. The following prediction model can be obtained by the same procedure as described in Section 2.2.

$$\hat{\mathbf{e}}_k^p(t+1|t) = \mathbf{e}_{k-1}^p(t+1) - \mathbf{G}\Delta\delta\mathbf{u}_k^m(t) - \mathbf{G}_d\Delta\mathbf{d}_k^{p_d}(t) - \mathbf{F}\Delta\hat{x}_k(t|t) \tag{2.60}$$

where

$$\mathbf{G}_d \triangleq \begin{bmatrix} CB_d & 0 & \cdots & 0 \\ CAB_d & CB_d & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B_d & CA^{p-2}B_d & \cdots & CA^{p-p_d}B_d \end{bmatrix} \tag{2.61}$$

$$\Delta \mathbf{d}_k^{p_d}(t) \triangleq \begin{bmatrix} \Delta d_k(t)^T & \Delta d_k(t+1)^T & \cdots & \Delta d_k(t+p_d-1)^T \end{bmatrix}^T \tag{2.62}$$

and p_d is a disturbance horizon. If future disturbance cannot be forecasted, p_d should be 1 or the current measured disturbance $d_k(t)$ is used for future disturbance, that is, $d_k(t) = d_k(t+1) = \cdots = d_k(t+p_d-1)$. The steady-state Kalman gain can be computed by the following algebraic Riccati equation.

$$\begin{aligned} P &= APA^T - APC\left(CPC^T + R_n\right)^{-1} CPA^T + \Gamma \begin{bmatrix} Q_x & 0 \\ 0 & Q_u \end{bmatrix} \Gamma^T \\ K &= PC^T\left(CPC^T + R_n\right)^{-1} \end{aligned} \tag{2.63}$$

2.4 Numerical Illustrations

We provide three cases (unconstrained and constrained linear SISO system, constrained linear MIMO system, and nonlinear batch reactor) to show the effectiveness of the proposed algorithm. It is assumed that all the disturbances are unmeasurable; furthermore, unmeasured disturbance models are not used to design the ILMPC controller for all the cases.

2.4.1 (Case 1) Unconstrained and Constrained Linear SISO System

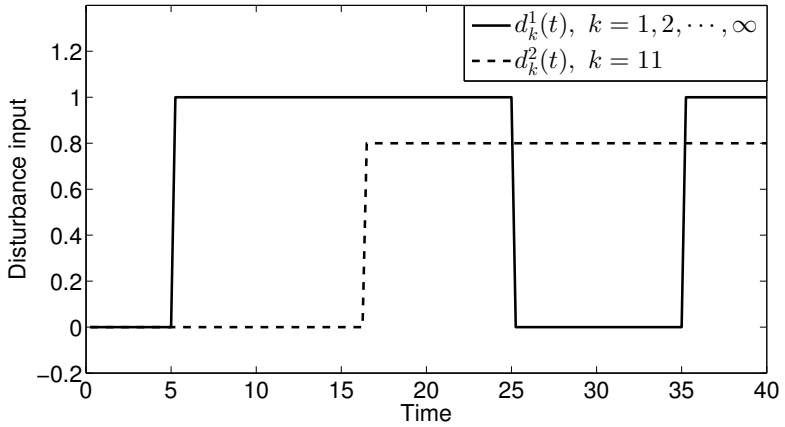


Figure 2.1: (Case 1) Repetitive disturbance input (d_k^1) for all batches and non-repetitive disturbance input (d_k^2) for the 11^{th} batch.

The proposed algorithm is evaluated using the following plant transfer function.

$$y_p(s) = \frac{0.8}{(5s+1)(3s+1)}u(s) + \frac{0.2}{2s+1}d^1(s) + \frac{0.2}{3s+1}d^2(s) \quad (2.64)$$

where $d^1(s)$ is repetitive disturbance input for all batches and $d^2(s)$ is non-repetitive disturbance input which is entered at the 11^{th} batch as shown in Fig. 2.1. The proposed ILMPC controller is designed using the following model transfer function.

$$y_m(s) = \frac{1.2}{(6s+1)(2s+1)}u(s) \quad (2.65)$$

Terminal time is 40 with sampling interval of 0.25. For designing the

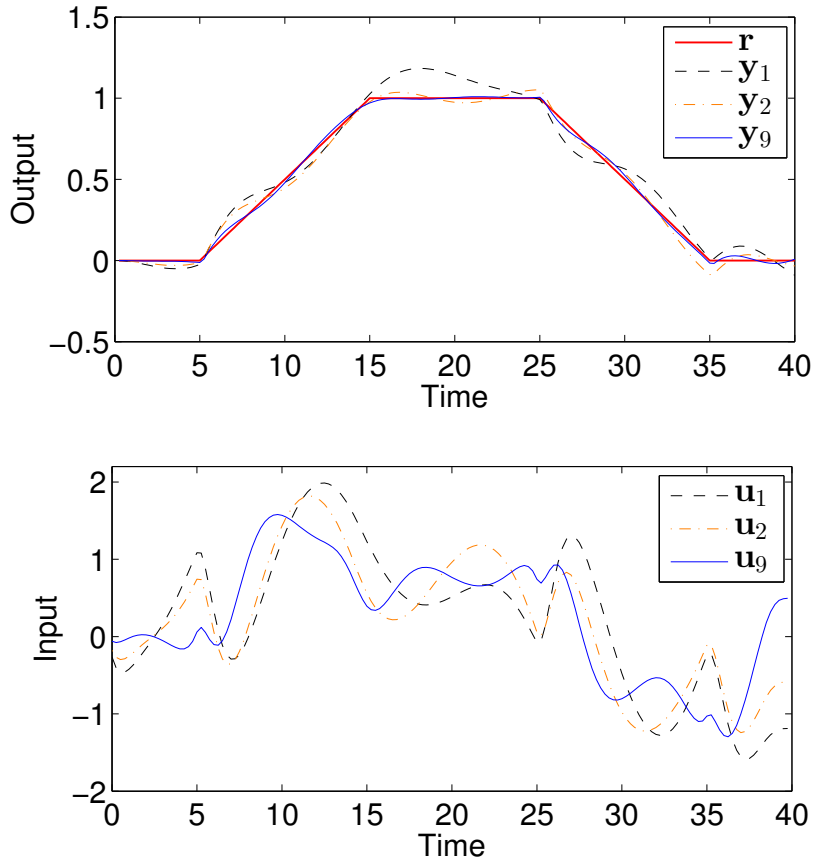


Figure 2.2: (Case 1) The results of the proposed ILMPC algorithm under model discrepancy and repetitive disturbance input.

controller, we used the following parameters.

$$\begin{aligned} &p = 80,\ m = 10, \\ &Q = I,\ R = 0.1I,\ S = 0, \\ &Q_x = 0.1I,\ Q_u = R_n = 0.0001. \end{aligned} \tag{2.66}$$

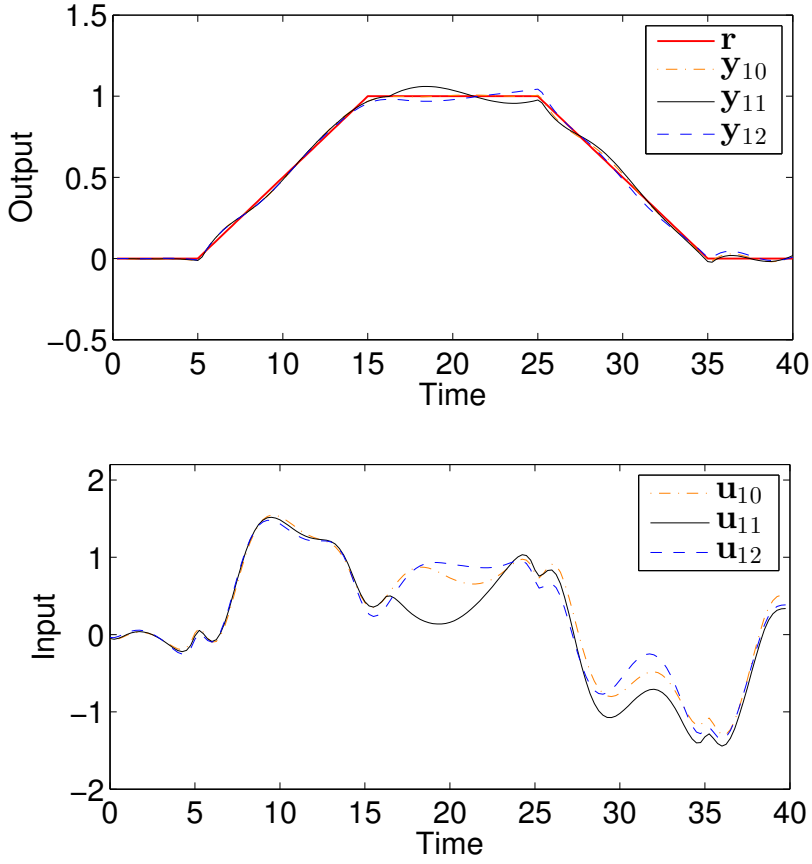


Figure 2.3: (Case 1) The performance of the proposed ILMPC algorithm for non-repetitive disturbance at the 11^{th} batch.

If the existing ILC technique combined with MPC where a state vector consists of the entire error sequences of a batch is used, the prediction horizon p is fixed as 100. The proposed technique can reduce the prediction horizon if the computational time is insufficient. In Fig. 2.2, the output of the 1^{st} batch shows oscillation because of repetitive disturbance $d_k^1(t)$. However, the output tracks the reference trajectory

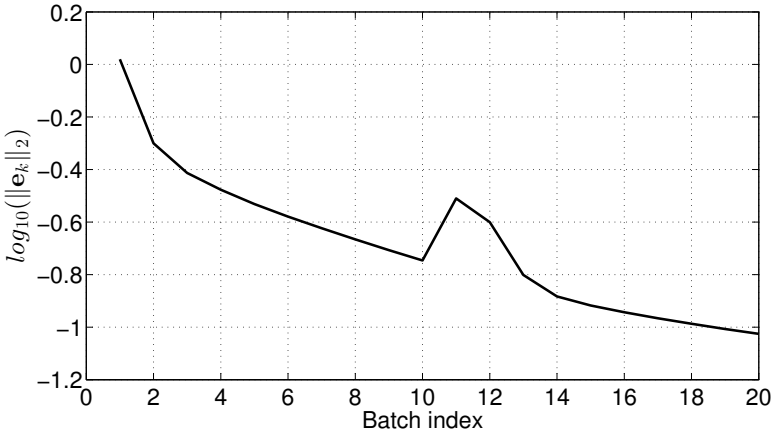


Figure 2.4: (Case 1) Log scale convergence performance for the linear SISO system.

while rejecting the repetitive disturbance at the 1^{st} batch because of the offset-free control. Although there are the effects of repetitive disturbance and plant-model mismatch, the tracking error is decreased as shown in Fig. 2.2 and Fig. 2.4. Non-repetitive disturbance input is entered at the 11^{th} batch. In Fig. 2.3, the disturbance effect is rejected in time horizon and the output converges to the reference trajectory in the batch direction. Fig. 2.4 shows that the proposed algorithm can reject the repetitive and non-repetitive disturbances in both time and batch horizons under model uncertainty.

Fig. 2.2 shows the large overshoot at the 1^{th} batch. Output constraint is used to eliminate the large overshoot. The following input and output constraints are applied to the ILMPC controller.

$$-1.5 \leq u_k(t) \leq 1.5, \quad -0.05 \leq y_k(t) \leq 1.05 \tag{2.67}$$

We used the same parameters as Eq. (2.66) and $S = 100$. The penalty,

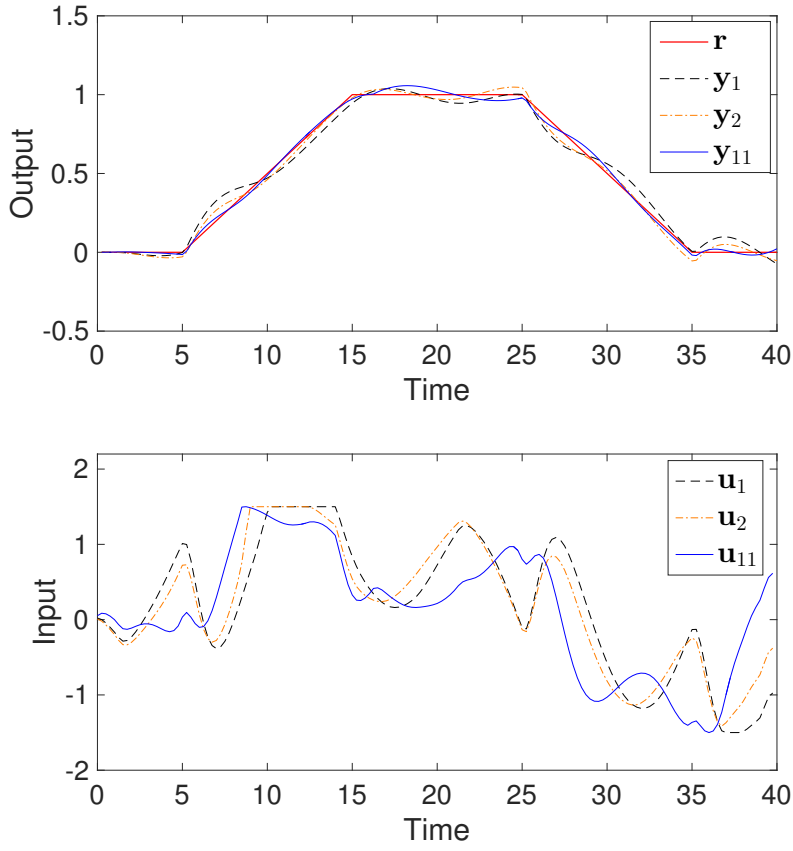


Figure 2.5: (Case 1) The results of the proposed ILMPC algorithm under the input and output constraints.

S, should be large enough for constraint violation. The large overshoot can be eliminated as shown in Fig. 2.5.

2.4.2 (Case 2) Constrained Linear MIMO System

The main advantage of MPC is that it is able to handle multivariable systems with constraints. In this case, we consider the following

constraints and MIMO system.

$$-15 \leq \mathbf{u}_{i,k} \leq 15, \text{ and } -5 \leq \delta\mathbf{u}_{i,k} \leq 5, \quad i = 1, 2 \tag{2.68}$$

$$y_p(s) = \begin{bmatrix} \frac{1.9}{240s^2+31s+1} & \frac{1.4}{130s^2+23s+1} \\ \frac{1.2}{180s^2+28s+1} & \frac{2.3}{96s^2+20s+1} \end{bmatrix} u(s) + \begin{bmatrix} \frac{3}{200s^2+30s+1} \\ \frac{3}{200s^2+30s+1} \end{bmatrix} d(s) \tag{2.69}$$

$$y_m(s) = \begin{bmatrix} \frac{2.3}{300s^2+35s+1} & \frac{1.7}{110s^2+21s+1} \\ \frac{1.7}{225s^2+30s+1} & \frac{2.8}{117s^2+22s+1} \end{bmatrix} \tag{2.70}$$

where $\mathbf{u}_{i,k}$ is the i^{th} input at the k^{th} batch. Terminal time is 100 with the sampling interval of 1. The following parameters are used.

$$\begin{aligned} &p = 60,\ m = 8, \\ &Q = I,\ R = 0.005I,\ S = 0, \\ &Q_x = Q_u = 0.01I,\ R_n = 0.0001I \end{aligned} \tag{2.71}$$

The non-repetitive disturbance in Fig. 2.6 is entered at the 11^{th} batch. Fig. 2.7 shows the efficacy of the proposed algorithm for the constrained MIMO system. The disturbance at the 11^{th} batch is rejected along the time direction. After two batches later, non-repetitive disturbance effect vanishes as shows in Fig. 2.8. In Fig. 2.7, the output cannot converge to the reference trajectory around time 5 to 10. This is because the upper bounds of the inputs. Except for this time periods, the outputs converge to the reference trajectories although there are constrains on the input movements.

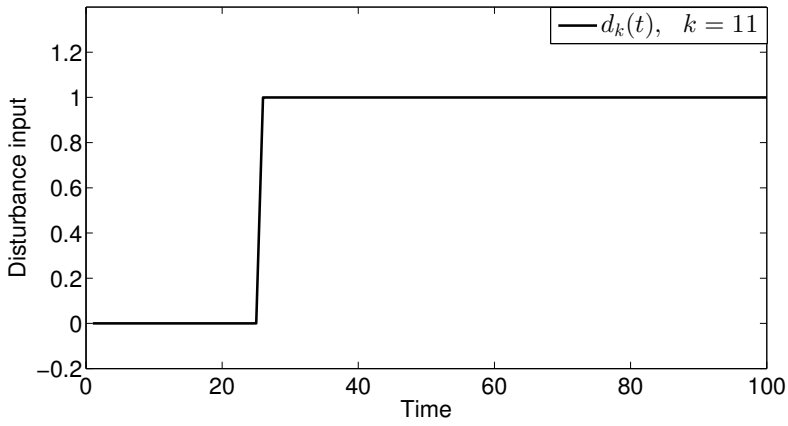


Figure 2.6: (Case 2) Non-repetitive disturbance input for the 11^{th} batch.

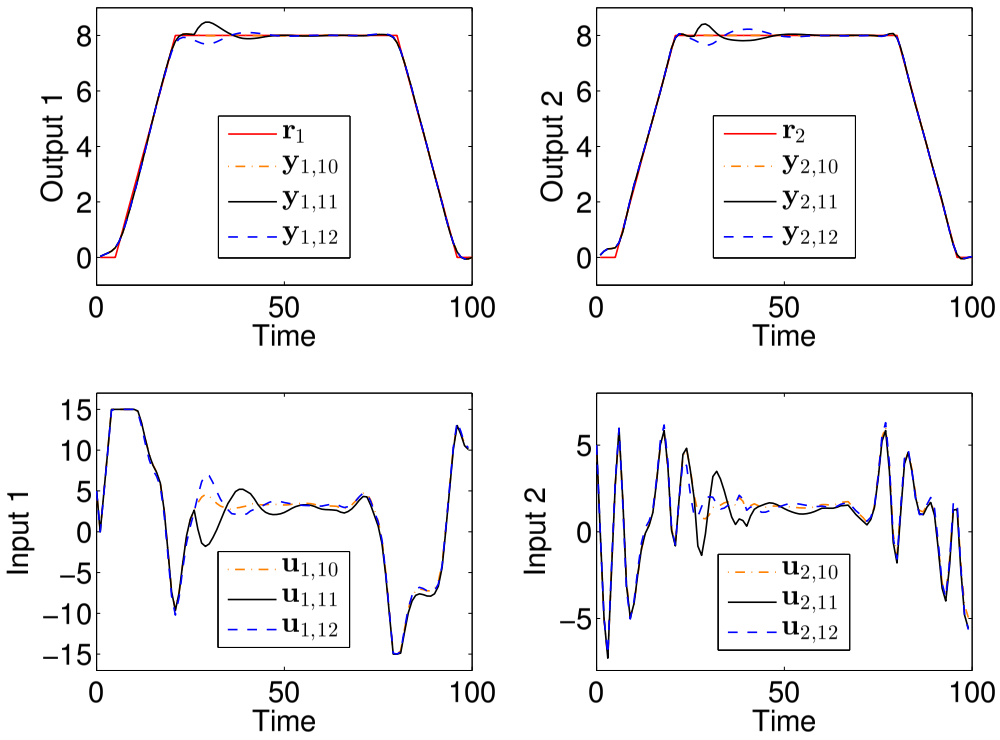


Figure 2.7: (Case 2) The performance of the proposed ILMPC algorithm for non-repetitive disturbance at the 11^{th} batch.

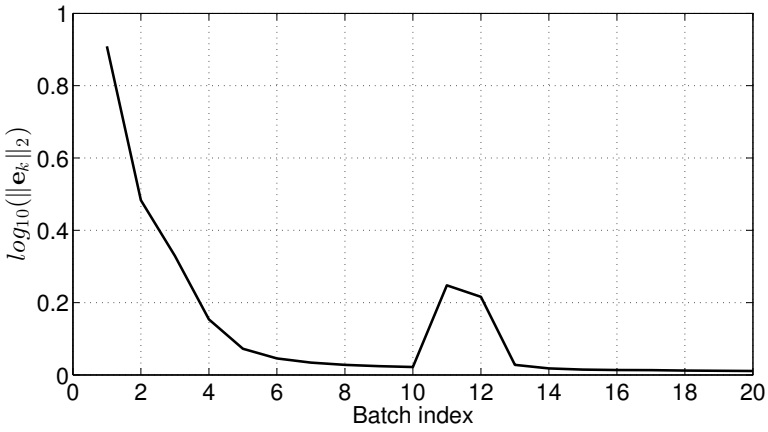


Figure 2.8: (Case 2) Log scale convergence performance for the constrained linear MIMO system.

2.4.3 (Case 3) Nonlinear Batch Reactor

We consider the temperature control of a nonlinear batch reactor where a second-order exothermic reaction $A \rightarrow B$ occurs [5]. It is assumed that the temperature of a cooling jacket (T_j) is directly manipulated.

$$\begin{aligned} \frac{dT}{dt} &= -\frac{UA}{MC_p}(T - T_j) - \frac{\Delta HV}{MC_p} k_0 e^{E/RT} C_A^2 \\ \frac{dC_A}{dt} &= -k_0 e^{E/RT} C_A^2 \end{aligned} \tag{2.72}$$

The following values were used for the parameters:

$$\begin{aligned}
\frac{UA}{MC_p} &= 0.09\ (l/min) \\
\frac{\Delta HV}{MC_p} &= -1.64\ (K\ l/mol) \\
k_0 &= 2.53 \times 10^{19}\ (l/mol\ min) \\
E/R &= 13,550\ (K) \\
T(0) &= 25\ (^\circ C) \\
C_A(0) &= 0.9\ (mol/l)
\end{aligned} \tag{2.73}$$

For system identification, a step test was performed with the size of 26 °C and the sampling interval of 1. We obtained the following discrete-time model using the least squares method with step input and step response, assuming that the system is second-order.

$$y_m(t) = \frac{0.0436z + 0.0425}{z^2 - 0.9153z + 0.0013}u(t) \tag{2.74}$$

We use the following parameters for controller design.

$$\begin{aligned}
&p = 80,\ m = 6, \\
&Q = I,\ R = 0.1I,\ S = 0, \\
&Q_x = Q_u = 0.01I,\ R_n = 0.0001
\end{aligned} \tag{2.75}$$

In this case, a repetitive disturbance is entered from the 8^{th} batch and a non-repetitive disturbance is entered at the 14^{th} batch. Fig. 2.9 shows two disturbance inputs. It is assumed that the disturbance input is filtered by $1/(2s+1)$, and then filtered signal enters at the plant output. Non-repetitive and repetitive disturbance are the same in this case.

In Fig. 2.10, the temperatures of the cooling jacket of the 8^{th} batch are decreasing to act against the repetitive disturbance. The output trajectory converges to the reference trajectory against the repetitive disturbance, as shown in Figs. 2.9 and 2.11. Fig. 2.11 shows the non-repetitive disturbance rejection effect. The non-repetitive disturbance effect at the 14^{th} batch remains at the output of the 15^{th} batch, although there is no additional disturbance at the 15^{th}. This response occurs because the controller learns from the previous batch. If the system parameters are changed or disturbance is entered at the k^{th} batch, the controller makes the output converge to the changed system or attempt to reject the disturbance at the next batch. Therefore, the input effect against the disturbance at the 14^{th} batch slightly carries over to the 15^{th}, and the output rapidly converges to the reference trajectory by the time-wise feedback. When the input of the 15^{th} batch is calculated, the controller cannot reflect the previous error by omitting $\mathbf{e}^{p}_{k-1}(t+1)$ of Eq. (2.41); alternatively, a weighting factor less than 1 can be used on $\mathbf{e}^{p}_{k-1}(t+1)$ to reject non-repetitive disturbance. Next, the input effect against the disturbance of the 14^{th} batch does not carry over to the 15^{th} batch, and the convergence performance is maintained at the level of the 13^{th} batch. However, if the previous error is neglected, then the controller cannot adapt to the repetitive disturbance. Many disturbances are unknown, unmeasured, and unpredicted. Furthermore, many batch processes are operated under model uncertainty. Thus, this type of tuning is not suitable. In Fig. 2.12, the proposed controller is shown to successfully monotonically reject the repetitive and non-repetitive disturbances.

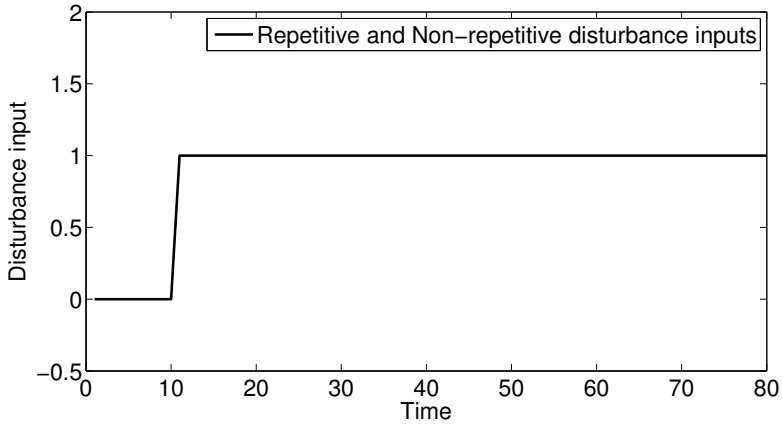


Figure 2.9: (Case 3) Disturbance input for $k = 8,\ 9,\ \cdots,\ \infty$ as a repetitive disturbance and for the 14^{th} batch as a non-repetitive disturbance.

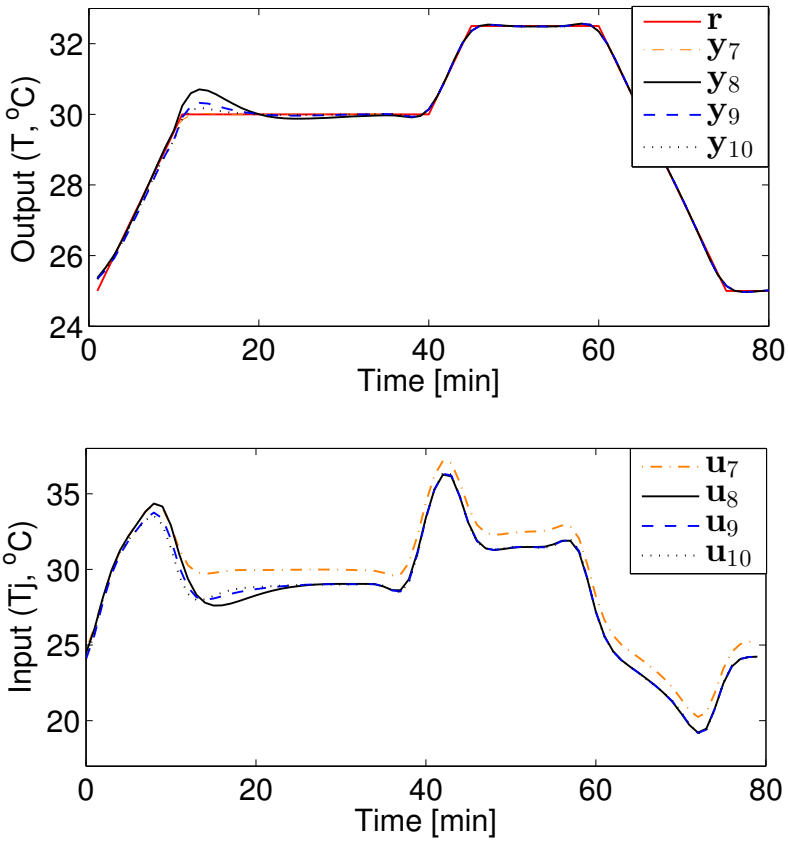


Figure 2.10: (Case 3) The performance of the proposed ILMPC algorithm against added repetitive disturbance at the 8^{th} batch.

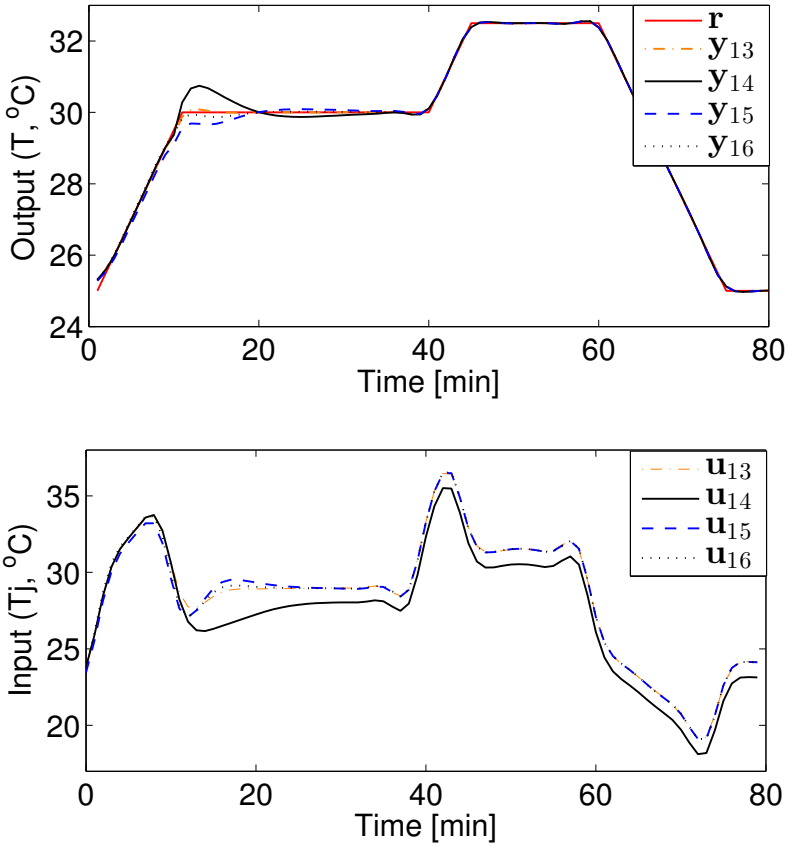


Figure 2.11: (Case 3) The performance of the proposed ILMPC algorithm for non-repetitive disturbance at the 14^{th} batch.

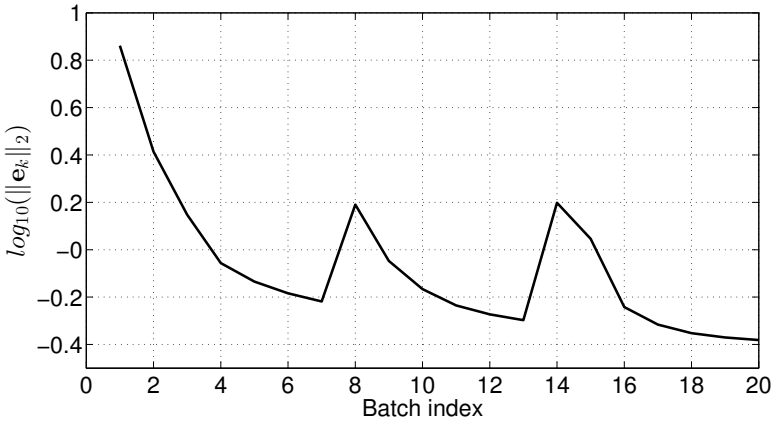


Figure 2.12: (Case 3) Log scale convergence performance for the nonlinear batch reactor.

2.5 Conclusion

In this paper, we have presented the constrained ILC algorithm combined with MPC. This algorithm can reject repetitive and non-repetitive disturbances along both the time and batch horizons under model uncertainty. We used the incremental state-space model to guarantee offset-free control. The slack variable is used to guarantee that a feasible solution will always be found. The disturbance model and stochastic characteristics can be easily considered because the proposed ILMPC is similar to the general MPC formulation. In this paper, we only considered a linear time-invariant system; however, ILMPC for a time-varying system can be derived using the same procedure. We presented three cases to show the effectiveness of the proposed ILMPC. Disturbance rejection and convergence were found to be successfully achieved in all the cases.

Chapter 3

Iterative Learning Control Combined with Model Predictive Control for Non-Zero Convergence

In this chapter, the case in which the output error converges to non-zero value is studied. The existing ILMPC techniques make the error converge to zero. However, if the error converges to zero, an impractical input trajectory may be calculated. We use a generalized objective function to independently tune weighting factors of manipulated variable change with respect to both the time index and batch horizons. If the generalized objective function is used, output error converges to non-zero values. We provide convergence analysis for both cases of zero convergence and non-zero convergence.

3.1 Iterative Learning Model Predictive Controller for Non-zero Convergence

We use the following objective function to design the ILMPC controller.

$$\min_{\Delta \mathbf{u}_k^m(t)} \frac{1}{2} \left\{ \|\mathbf{e}_k^p(t+1)\|_{\mathbf{Q}}^2 + \|\delta \mathbf{u}_k^m(t)\|_{\mathbf{R}}^2 + \|\Delta \mathbf{u}_k^m(t)\|_{\mathbf{S}}^2 \right\} \tag{3.1}$$

where

$$\begin{aligned}\mathbf{e}_k^p(t+1) &= \mathbf{r}^p(t+1) - \hat{\mathbf{y}}_k^p(t+1|t) \\ \mathbf{r}^p(t+1) &= \begin{bmatrix} r(t+1)^T & \cdots & r(t+p)^T \end{bmatrix}^T \end{aligned} \tag{3.2}$$

, $r(t)$ is the reference trajectory and $\|x\|_Q^2 = x^T Q x$. To obtain the solution, each input term of the prediction model (2.28) and the objective function (3.1) should be expressed with respect to $\Delta \mathbf{u}_k^m(t)$ as follows:

$$\begin{aligned}\delta \mathbf{u}_k^m(t) &= \mathbf{I}_L \Delta \mathbf{u}_k^m(t) + \mathbf{I}_L \mathbf{u}_{k-1}^m(t) - \mathbf{I}_a u_k(t-1) \\ \Delta \delta \mathbf{u}_k^m(t) &= \mathbf{I}_L \Delta \mathbf{u}_k^m(t) - \mathbf{I}_a \Delta u_k(t-1) \end{aligned} \tag{3.3}$$

where

$$\mathbf{I}_L = \begin{bmatrix} I & 0 & 0 & \cdots & 0 & 0 \\ -I & I & 0 & \cdots & 0 & 0 \\ 0 & -I & I & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & I & 0 \\ 0 & 0 & 0 & \cdots & -I & I \end{bmatrix}, \quad \mathbf{I}_a = \begin{bmatrix} I \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \tag{3.4}$$

The following analytical solution can be obtained using Eqs. (2.28), (3.1) and (3.3).

$$\Delta u_k(t) = \mathbf{I}_a^T \Delta \mathbf{u}_k^m(t) = -\mathbf{I}_a^T \mathcal{H}^{-1} \mathbf{f} \tag{3.5}$$

where

$$\mathcal{H} = \mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \mathbf{G} \mathbf{I}_L + \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L + \mathbf{S} \tag{3.6}$$

$$\begin{aligned}\mathbf{f} =& \mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \left(\mathbf{F} \Delta \hat{x}_k(t|t) - \mathbf{G} \mathbf{I}_a \Delta u_k(t-1) - \mathbf{e}_{k-1}^p(t+1) \right) \\ &+ \mathbf{I}_L^T \mathbf{R} \left(\mathbf{I}_L \mathbf{u}_{k-1}^m(t) - \mathbf{I}_a u_k(t-1) \right)\end{aligned} \tag{3.7}$$

Many control applications need to ensure safety and smooth operations. For this purpose, the following constraints are considered.

$$\begin{aligned}&\mathbf{u}_{\min}^m \leq \mathbf{u}_k^m(t) \leq \mathbf{u}_{\max}^m \\ &\delta \mathbf{u}_{\min}^m \leq \delta \mathbf{u}_k^m(t) \leq \delta \mathbf{u}_{\max}^m \\ &\Delta \mathbf{u}_{\min}^m \leq \Delta \mathbf{u}_k^m(t) \leq \Delta \mathbf{u}_{\max}^m \\ &\mathbf{y}_{\min}^p \leq \mathbf{y}_k^p(t+1|t) \leq \mathbf{y}_{\max}^p\end{aligned} \tag{3.8}$$

A standard quadratic programming (QP) problem for constrained ILMPC is as follows:

$$\min_{\Delta \mathbf{u}_k^m(t)} \frac{1}{2} \Delta \mathbf{u}_k^m(t) \mathcal{H} \Delta \mathbf{u}_k^m(t) + \mathbf{f}^T \Delta \mathbf{u}_k^m(t) \tag{3.9}$$

subject to

$$\mathbf{M} \Delta \mathbf{u}_k^m(t) \leq \mathbf{b}_k(t) \tag{3.10}$$

where

$$\mathbf{M} = \begin{bmatrix} -\mathbf{I} \\ \mathbf{I} \\ -\mathbf{I}_L \\ \mathbf{I}_L \\ -\mathbf{I}_L \\ \mathbf{I}_L \\ -\mathbf{G}\mathbf{I}_L \\ \mathbf{G}\mathbf{I}_L \end{bmatrix} \tag{3.11}$$

$$\mathbf{b}_k(t) = \begin{bmatrix} -\mathbf{u}^m_{\min} + \mathbf{u}^m_{k-1}(t) \\ \mathbf{u}^m_{\max} - \mathbf{u}^m_{k-1}(t) \\ -\delta\mathbf{u}^m_{\min} + \mathbf{I}_L\mathbf{u}^m_{k-1}(t) - \mathbf{I}_a u_k(t-1) \\ \delta\mathbf{u}^m_{\max} - \mathbf{I}_L\mathbf{u}^m_{k-1}(t) + \mathbf{I}_a u_k(t-1) \\ -\Delta\mathbf{u}^m_{\min} - \mathbf{I}_a\Delta u_{k-1}(t-1) \\ \Delta\mathbf{u}^m_{\max} + \mathbf{I}_a\Delta u_{k-1}(t-1) \\ -\mathbf{y}^p_{\min} + \mathbf{y}^p_{k-1}(t+1) - \mathbf{G}\mathbf{I}_a\Delta u_k(t-1) + \mathbf{F}\Delta\hat{x}_k(t|t) \\ \mathbf{y}^p_{\max} - \mathbf{y}^p_{k-1}(t+1) + \mathbf{G}\mathbf{I}_a\Delta u_k(t-1) - \mathbf{F}\Delta\hat{x}_k(t|t) \end{bmatrix} \tag{3.12}$$

The optimization problem can be solved by appropriate QP solver. The first input of the optimal solution is implemented on the plant. The formulation of the proposed ILMPC is similar to the conventional MPC formulation. Thus, various techniques applicable to MPC, such as disturbance model, time-varying model and advanced state estimation theory, can be applied without modifying the structure of the controller.

3.2 Convergence Analysis

3.2.1 Convergence Analysis for an Input Trajectory

First, we prove that $\Delta u_k(t)$ converges to zero for all t as $k \to \infty$ under the following assumptions.

1. There exists a feasible input trajectory such that $\mathbf{e}_\infty = 0$
2. All constraints (3.8) are satisfied when an input trajectory is converged.
3. A system has the same initial condition for all batches; the same

input trajectory and the same state trajectory lead to the same output trajectory.

4. $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{S}$ are symmetric positive definite.

Theorem 3.1. *Consider the assumptions and the QP problem. Then,* $\Delta u_k(t) \to 0 \; \forall t$ *as* $k \to \infty$.

Proof We consider the objective function and the minimizer of the optimization problem at time t of the k-th batch.

$$\Phi_k(t) = \frac{1}{2}\left\{\|\mathbf{e}_k^p(t+1)\|_{\mathbf{Q}}^2 + \|\delta\mathbf{u}_k^m(t)\|_{\mathbf{R}}^2 + \|\Delta\mathbf{u}_k^m(t)\|_{\mathbf{S}}^2\right\} \quad (3.13)$$

$$J_k(t) = \min_{\Delta\mathbf{u}_k^m(t)} \Phi_k(t) \geq 0 \quad (3.14)$$

subject to Eq. (3.8).

An optimal cost ($J_k(t)$) of an objective function is always less than or equal to a feasible cost ($\Phi_k(t)$), i.e., $J_k(t) \leq \Phi_k(t)$. Let ($\mathbf{e}_k^{*,p}(t+1),\ \mathbf{u}_k^{*,m}(t)$) be the optimal solution for the k-th batch. The optimal solution of the k-th batch until time t can be used for the $(k+1)$-th batch, then the optimal cost ($J_k(t)$) of the k-th batch becomes the feasible cost ($\Phi_{k+1}(t)$) of the $(k+1)$-th batch. Thus, $\mathbf{e}_{k+1}^p(t+1) = \mathbf{e}_k^{*,p}(t+1)$, $\mathbf{u}_{k+1}^m(t) = \mathbf{u}_k^{*,m}(t)$, $\delta\mathbf{u}_{k+1}^m = \delta\mathbf{u}_k^{*,m}$ and $\Delta\mathbf{u}_{k+1}^m(t) = \mathbf{u}_{k+1}^m(t) - \mathbf{u}_k^{*,m}(t) = 0$. As mentioned above, the optimal cost is always less than or equal to the feasible cost; therefore, $J_k(t) = \Phi_{k+1}(t) \geq J_{k+1}(t)$.

We have the following inequality.

$$J_{k+1}(t) \leq \frac{1}{2}\left\{\|\mathbf{e}_k^{*,p}(t+1)\|_{\mathbf{Q}}^2 + \|\delta\mathbf{u}_k^{*,m}(t)\|_{\mathbf{R}}^2\right\} \quad (3.15)$$

By adding and subtracting the same term, we have

$$\begin{aligned} J_{k+1}(t) &\leq \frac{1}{2}\left\{\|\mathbf{e}_k^{*,p}(t+1)\|_{\mathbf{Q}}^2 + \|\delta\mathbf{u}_k^{*,m}(t)\|_{\mathbf{R}}^2 + \|\Delta\mathbf{u}_k^{*,m}(t)\|_{\mathbf{S}}^2\right\} \\ &\quad - \frac{1}{2}\|\Delta\mathbf{u}_k^{*,m}(t)\|_{\mathbf{S}}^2 \\ &= J_k(t) - \frac{1}{2}\|\Delta\mathbf{u}_k^{*,m}(t)\|_{\mathbf{S}}^2 \end{aligned} \tag{3.16}$$

which yields

$$0 \leq J_{k+1}(t) + \frac{1}{2}\sum_{j=1}^{k}\|\Delta\mathbf{u}_j^{*,m}(t)\|_{\mathbf{S}}^2 \leq J_1(t) < \infty \tag{3.17}$$

Thus, $\Delta\mathbf{u}_k^{*,m}(t) \to 0,\ \forall t$ as $k \to \infty$. ■

3.2.2 Convergence Analysis for an Output Error

In this section, we show that the output error ($e_k(t)$) converges to a fixed value ($e^*(t)$) or 0 using the result of Section 3.2.1. If $k \to \infty$, all constraints are satisfied by the assumptions. Thus, the unconstrained solution (3.5) and the constrained solution (3.9, 3.10) are equal if $k \to \infty$. The purpose of this proof is to know the converged error for all time ($1 \sim N$), not prediction time horizon ($t \sim t+p$); therefore, we use the unconstrained solution and set $t = 0$ and $m = p = N$. In this case, $\mathbf{f}$ in Eq. (3.7) is simplified because $\Delta\hat{x}_k(0|0) = 0$ (the same initial condition for all batches), $\Delta u_k(-1) = 0$ and $u_k(-1) = 0$. Furthermore, $\mathbf{S}$ in Eq. (3.6) can be zero because $\Delta\mathbf{u}_\infty = 0$ and $\mathbf{S}$ does not affect the converged value. It affects the convergence rate. For the above reasons, $\mathcal{H}$ in Eq. (3.6)

and $\mathbf{f}$ in Eq. (3.7) are simplified as

$$\begin{aligned} \mathcal{H} &= \mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \mathbf{G} \mathbf{I}_L + \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \\ \mathbf{f} &= -\mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \mathbf{e}_{k-1} + \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \mathbf{u}_{k-1} \end{aligned} \tag{3.18}$$

Theorem 3.2. *Consider the proposed ILMPC controller,* $\mathbf{e}_k \to 0$ *as* $k \to \infty$ *if* $\mathbf{R} = 0$.

Proof The unconstrained solution with $t = 0$, $m = p = N$ and $\mathbf{R} = 0$ is as follows:

$$\Delta \mathbf{u}_k = \mathcal{H}^{-1} \mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \mathbf{e}_{k-1} \tag{3.19}$$

By Theorem 3.1, if $k \to \infty$,

$$\Delta \mathbf{u}_\infty = 0 = \mathcal{H}^{-1} \mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \mathbf{e}_\infty \tag{3.20}$$

This implies that $\mathbf{e}_\infty = 0$. ■

Theorem 3.3. *Consider the proposed ILMPC controller,* $\mathbf{e}_k \to \mathbf{e}^*$ *as* $k \to \infty$.

Proof The unconstrained solution with $t = 0$ and $m = p = N$ is as follows:

$$\Delta \mathbf{u}_k = \mathcal{H}^{-1} \left(\mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \mathbf{e}_{k-1} - \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \mathbf{u}_{k-1} \right) \tag{3.21}$$

The above equation can be expressed as follows:

$$\mathbf{u}_k = \left(\mathbf{I} - \mathcal{H}^{-1} \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \right) \mathbf{u}_{k-1} + \mathcal{H}^{-1} \mathbf{I}_L^T \mathbf{G}^T \mathbf{Q} \mathbf{e}_{k-1} \tag{3.22}$$

To simplify, we define Eq. (3.22) as follows:

$$\mathbf{u}_k = \mathbf{H}_1 \mathbf{u}_{k-1} + \mathbf{H}_2 \mathbf{e}_{k-1} \tag{3.23}$$

If $k \to \infty$,

$$\mathbf{u}_\infty = \mathbf{H}_1 \mathbf{u}_\infty + \mathbf{H}_2 \mathbf{e}_\infty = \mathbf{H}_1 \mathbf{u}_\infty + \mathbf{H}_2 \left(\mathbf{r} - \mathbf{y}_\infty \right) \tag{3.24}$$

The state-space model (2.5) can be expressed as the lifted vector form: $\mathbf{y}_\infty = \mathbf{G}_p \delta \mathbf{u}_\infty = \mathbf{G}_p \mathbf{I}_L \mathbf{u}_\infty$ where $\mathbf{G}_p$ is the plant matrix. Substituting the lifted vector form into Eq. (3.24) and rearranging, we have

$$\mathbf{u}_\infty = \left(\mathbf{I} - \mathbf{H}_1 + \mathbf{H}_2 \mathbf{G}_p \mathbf{I}_L \right)^{-1} \mathbf{H}_2 \mathbf{r} \tag{3.25}$$

We can obtain the final result by substituting Eq. (3.25) into $\mathbf{e}_\infty$ as follows:

$$\begin{aligned} \mathbf{e}^* &= \mathbf{r} - \mathbf{y}_\infty = \mathbf{r} - \mathbf{G}_p \mathbf{I}_L \mathbf{u}_\infty \\ &= \left\{ \mathbf{I} - \mathbf{G}_p \mathbf{I}_L \left(\mathbf{I} - \mathbf{H}_1 + \mathbf{H}_2 \mathbf{G}_p \mathbf{I}_L \right)^{-1} \mathbf{H}_2 \right\} \mathbf{r} \end{aligned} \tag{3.26}$$

■

To make analysis of Eq. (3.26) simple, we consider the scalar case of the equation. Assume that terminal time N is 1 and the system has single-input single-output (SISO), then $\mathbf{I} = \mathbf{I}_L = 1$; other parameters become scalars which are written in non-bold typeface. Eq. (3.26) is expressed as follows:

$$e^* = \left(1 - \frac{G_p H_2}{1 - H_1 + H_2 G_p} \right) r \tag{3.27}$$

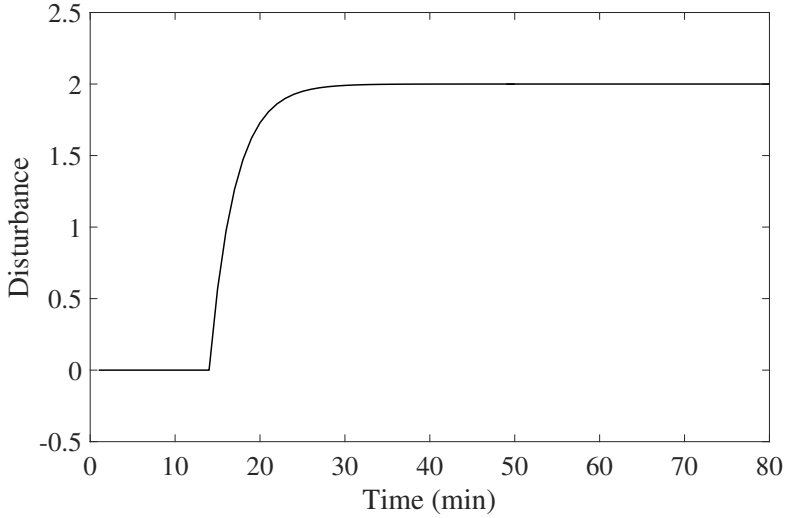


Figure 3.1: Disturbance for the 7th batch.

where

$$H_1 = 1 - \frac{R}{G^2Q + R}$$
$$H_2 = \frac{GQ}{G^2Q + R} \tag{3.28}$$

Eq. (3.27) is simplified as

$$e^* = \left(1 - \frac{GG_pQ}{GG_pQ + R}\right) r \tag{3.29}$$

The result indicates that bigger R increases the size of error. If $R = 0$, $e^* = 0$; it is the same result as Theorem 3.2. If $R \to \infty$, $e^* \to r$. The reason is that if $R \to \infty$, the input does not change from 0; thus, the output maintains zero value, i.e., $e^* = r - y = r$.

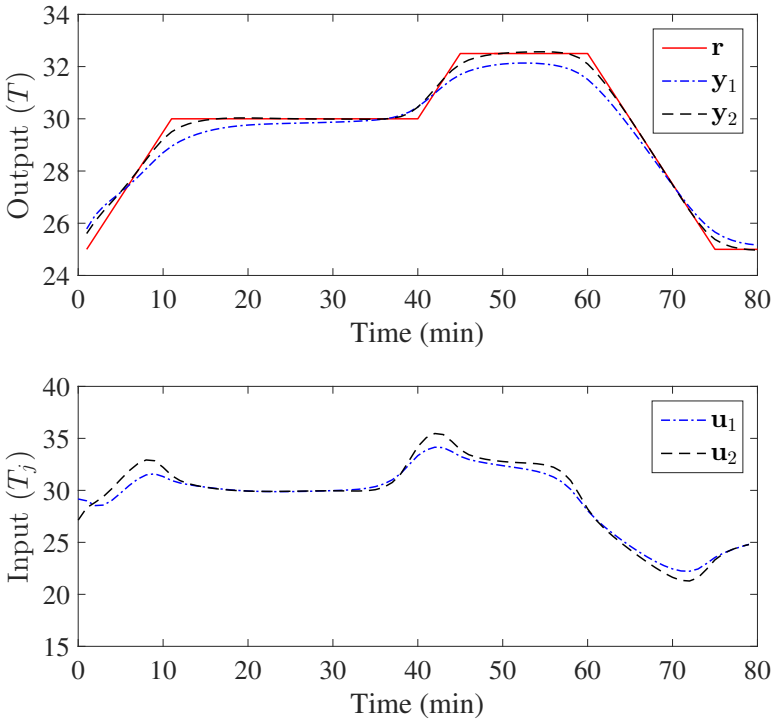


Figure 3.2: Result of the proposed ILMPC with $\mathbf{R} = 0.01I$ and $\mathbf{S} = 0.05I$.

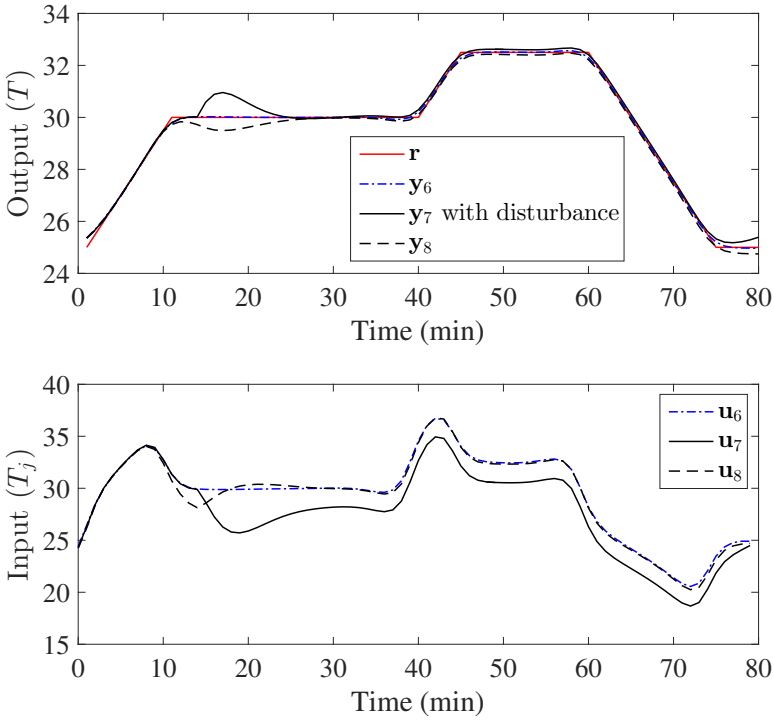


Figure 3.3: Disturbance rejection performance of the proposed ILMPC with $\mathbf{R} = 0.01I$ and $\mathbf{S} = 0.05I$.

3.3 Illustrative Example

We consider a cooling jacket temperature (T_j) control of a nonlinear batch reactor. A second-order exothermic reaction $A \rightarrow B$ occurs. It is assumed that T_j is directly manipulated.

$$\begin{aligned} \frac{dT}{dt} &= -\frac{UA}{MC_p}(T - T_j) - \frac{\Delta HV}{MC_p} k_0 \exp\left(\frac{E}{RT}\right) C_A^2 \\ \frac{dC_A}{dt} &= -k_0 \exp\left(\frac{E}{RT}\right) C_A^2 \end{aligned} \tag{3.30}$$

where T and C_A are the state variables, T is the output variable, and T_j is the input variable. The following parameters were used for plant:

$$\begin{aligned} \frac{UA}{MC_p} &= 0.09 \ (\mathrm{L/min}) \\ \frac{\Delta HV}{MC_p} &= -1.64 \ (\mathrm{K \cdot L/mol}) \\ k_0 &= 2.53 \times 10^{19} (\mathrm{L/mol \cdot min}) \\ \frac{E}{R} &= 13,500 \ (\mathrm{K}) \\ T(0) &= 25 \ (^\circ\mathrm{C}) \\ C_A(0) &= 0.9 \ (\mathrm{mol/L}) \end{aligned} \tag{3.31}$$

We obtained the following linear discrete-time model using the least squares method with a step input with an initial value of 25 °C and a size of 1 °C; the sampling interval of 1 min. We assumed that the

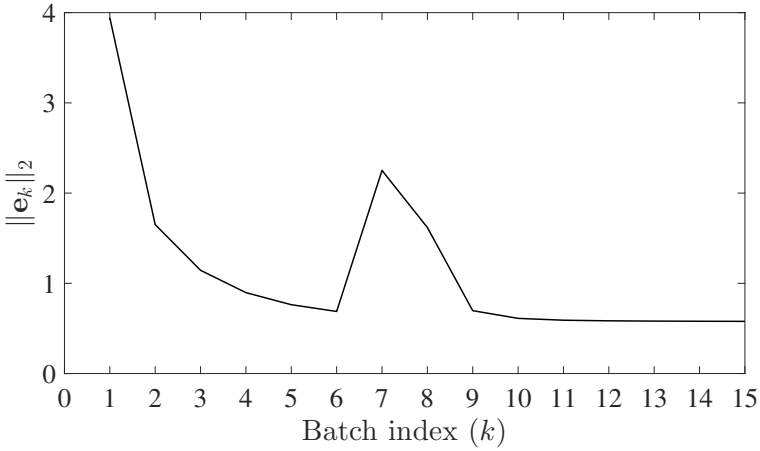


Figure 3.4: Convergence performance with $\mathbf{R} = 0.01I$ and $\mathbf{S} = 0.05I$ under the disturbance at the 7th batch.

system was second-order.

$$\begin{aligned} x(t+1) &= \begin{bmatrix} 0.9153 & -0.0416 \\ 0.0313 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 0.0436 & 1.3600 \end{bmatrix} x(t) \end{aligned} \tag{3.32}$$

We assume that there exists an unknown output disturbance at the 7th batch. Disturbance model is $1/(3s+1)$; and disturbance input is a step input with the size of 2 as shown in Fig. 3.1. The prediction horizon and the control horizon are 80 and 10, respectively. The weighting factors $\mathbf{Q}$ was fixed as the identity matrix for all simulations. For the first simulation which aimed to show the effectiveness of the disturbance rejection, we used $\mathbf{R} = 0.01I$ and $\mathbf{S} = 0.05I$. Fig. 3.2 shows the results of the 1st and the 2nd batches. The output of the 1st batch does not track the reference trajectory because

of the model uncertainty; the output of the 2nd batch converges to the reference trajectory. The unknown disturbance enters the system at the 7th batch as shown in Fig. 3.3. The disturbance is, however, rejected by the real-time feedback controller. The effect of the disturbance remains at the 8th batch because the ILMPC learns from the information of the previous batch. Therefore, the controller learns to reject the disturbance of the 7th batch. Because there is no disturbance at the 8th batch, the output rapidly converges to the reference trajectory again as shown in Fig. 3.4. In Section 3.2.2, we mentioned that the error cannot go to zero if $\mathbf{R}$ is not zero. The existing ILMPC techniques cannot tune the weighting factor for $\delta\mathbf{u}_k^m(t)$ independently. They can tune the weighting factor for $\Delta\mathbf{u}_k^m(t)$ or $\Delta\delta\mathbf{u}_k^m(t)$ where $\Delta\delta\mathbf{u}_k^m(t) = \delta\mathbf{u}_k^m(t) - \delta\mathbf{u}_{k-1}^m(t)$. The roles of the both weighting factors for $\Delta\mathbf{u}_k^m(t)$ and $\Delta\delta\mathbf{u}_k^m(t)$ are related to the convergence rate, not smooth input trajectory. Small or zero weighting factor for $\Delta\mathbf{u}_k^m(t)$ for fast convergence may show extreme sensitivity to high-frequency components of the output error [5]. If large weighting factor for $\Delta\mathbf{u}_k^m(t)$ and $\mathbf{R} = 0$ are used, a smooth input trajectory is obtained from the controller in the early batch; if the closed-loop error trajectory perfectly converges, the input trajectory is calculated from the controller for perfect tracking. The input trajectory for perfect tracking including the angular points generally shows non-smooth trajectory. Hence, the weighting factor for $\delta\mathbf{u}_k^m(t)$ is required to obtain a practical input trajectory. Fig. 3.5 shows the results with respect to three cases ($\mathbf{R} = 0.1I$, $\mathbf{R} = 0.01I$ and $\mathbf{R} = 0$) and Fig. 3.6 shows the convergence results. If we use $\mathbf{S} = 0.05I$, the output error cannot go to zero within 1000 batches. Thus, in this case, we used $\mathbf{S} = 0.01I$ for fast convergence.

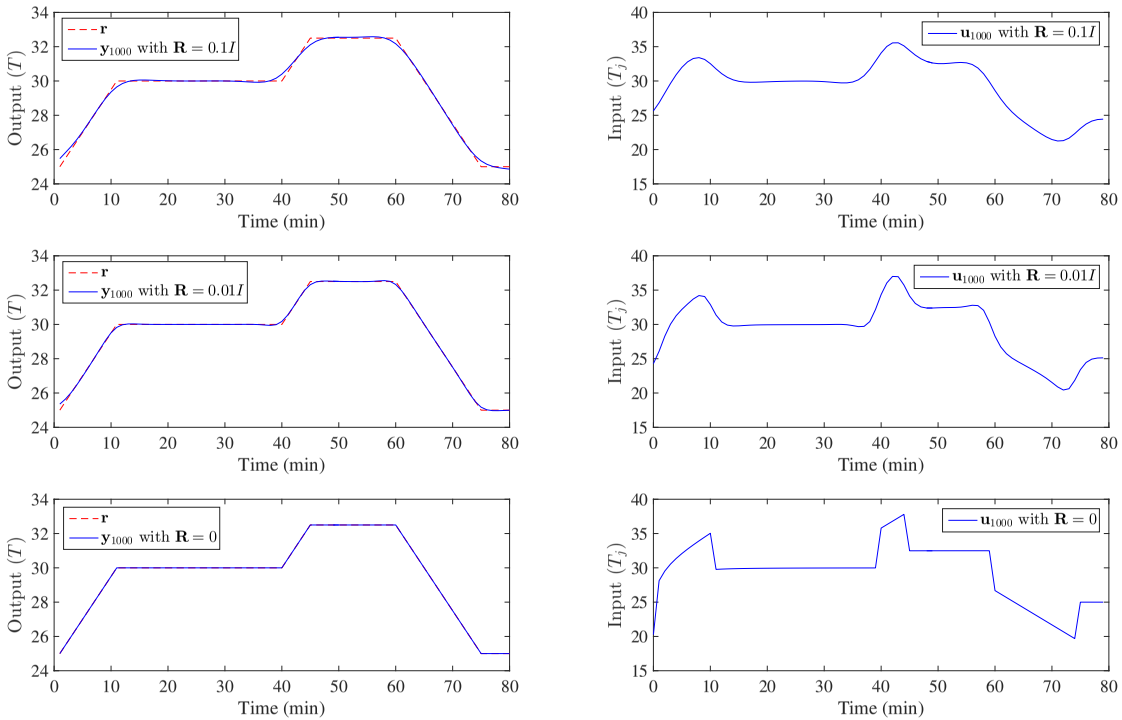


Figure 3.5: Results of the proposed ILMPC with respect to different sizes of $\mathbf{R}$ ($\mathbf{S} = 0.01I$).

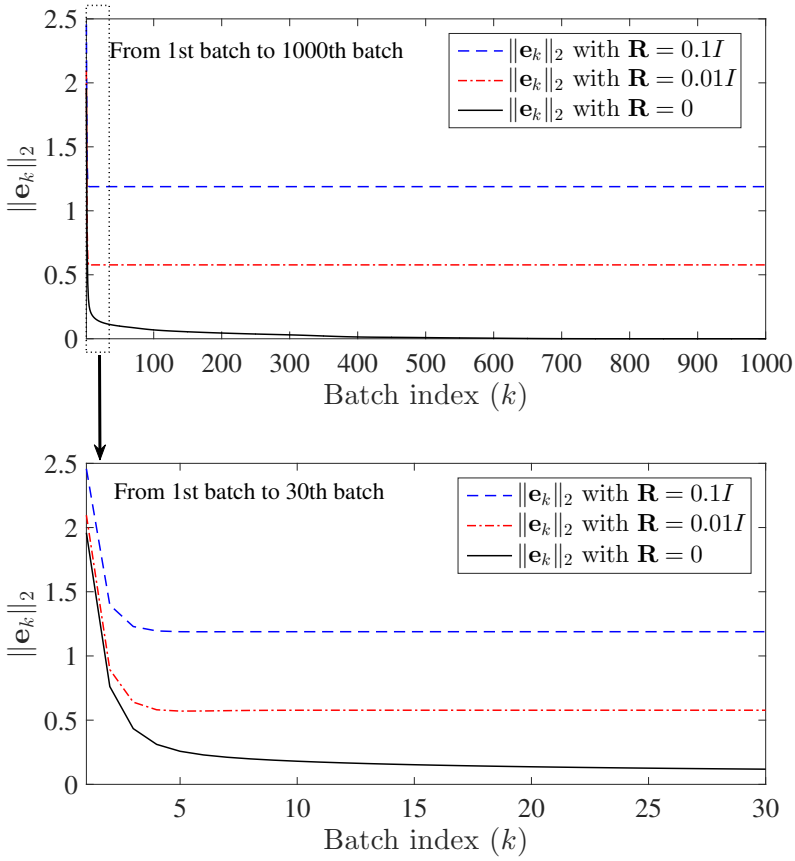


Figure 3.6: Convergence results with respect to different sizes of $\mathbf{R}$ ($\mathbf{S} = 0.01I$).

3.4 Conclusions

In this paper, we have proposed the iterative learning model predictive control technique for real-time disturbance rejection of batch processes. The proposed algorithm can independently tune the

weighting factor for the rate of input change with respect to the time index. We prove that the output error cannot go to zero if the weighting factor for $\delta \mathbf{u}_k^m(t)$ is not zero. The example is provided to show the effectiveness for disturbance rejection and iterative learning. Furthermore, the simulation shows that the weighting factor for $\delta \mathbf{u}_k^m(t)$ should be able to be tuned independently for obtaining a practical input trajectory.

Chapter 4

Iterative Learning Control Combined with Model Predictive Control for Tracking Specific Points

In this chapter, we propose a point-to-point ILMPC technique which can only consider the desired reference points, not an entire reference trajectory. It does not require to generate a reference trajectory which passes through the desired reference values. The existing ILMPC techniques aim to track a reference trajectory of repetitive process on a finite time interval while rejecting real-time disturbances. In many repetitive processes, however, the output does not need to track all points of a reference trajectory.

4.1 Introduction

Tracking an entire reference trajectory is not always necessary in many applications such as a robotic "pick and place" task, crane control, rapid thermal process, and chemical batch reactor [12, 13, 14]. Many systems only need to track the desired reference points, not the entire reference trajectory which is generated to pass through the reference points. An ILC technique that considers only the desired

reference points is called point-to-point ILC (PTP ILC) and has been studied recently [15, 16, 17]. Terminal ILC (TILC) has been also studied for tracking terminal point only [12, 18, 19]. It is a special case of the PTP ILC problem. These types of PTP ILC algorithms are open-loop control within an iteration; thus, they cannot reject real-time disturbances. If real-time disturbances should be rejected, the PTP ILC algorithm needs integrating with a real-time feedback controller. To overcome a similar issue, ILC combined with model predictive control (MPC), called iterative learning model predictive control (ILMPC), has been studied to reject real-time disturbances in iteration systems [4, 55, 9, 75]. However, the existing ILMPC algorithms can be used when a reference trajectory on the entire time sequences is prescribed.

In this chapter, we propose a PTP ILC algorithm combined with MPC, called point-to-point iterative learning model predictive control (PTP ILMPC). The proposed PTP ILMPC algorithm can be applied only using the desired reference points without generating an arbitrary reference trajectory passing through the desired reference points. Furthermore, unlike the existing PTP ILC algorithms, it is based on MPC which is a real-time feedback controller; hence it can handle real-time disturbances. The proposed PTP ILMPC algorithm involves input and output constraints. For output constraints softening, we introduce slack variable; thus, this algorithm ensures a feasible solution in the optimization step will always be found. In order to guarantee the convergence of tracking error, the suggested approach requires the error between measured and estimated outputs go to zero for all time as the number of iterations goes to infinity. However, neither classical observer nor Kalman filter guarantees the estimation

error converge to zero for all time points. To overcome this issue, iterative learning observer (ILO) is applied to the algorithm and it can ensure that the estimation error go to zero for all time as the number of iterations goes to infinity [90].

4.2 Point-to-Point Iterative Learning Model Predictive Control

4.2.1 Extraction Matrix Formulation

In the PTP ILMPC framework, the outputs need to track the desired reference points only. The reference time instants of the i-th output and the reference values of the ith output are defined by the set and the vector:

$$\begin{aligned} \psi^i &= \{t_1^i,\ t_2^i,\ \cdots,\ t_{N^i}^i\} \\ \tilde{\mathbf{r}}^i &= \begin{bmatrix} r^i(t_1^i) & r^i(t_2^i) & \cdots & r^i(t_{N^i}^i) \end{bmatrix}^T \end{aligned} \tag{4.1}$$

where $0 < t_1^i < t_2^i < \cdots < t_{N^i}^i \leq N$, and N^i is the number of reference points of i-th output and $r^i(t_j^i)$ is the j-th reference value of the i-th output. The vector of the reference values can be compactly represented as

$$\mathbf{r}_j^i = \begin{cases} r^i\,(j) & \text{, if } j \in \psi^i,\ j = 1,\ 2,\ \cdots,\ N \\ 0 & \text{, otherwise} \end{cases} \tag{4.2}$$

where $\mathbf{r}_j^i$ is the jth component of a column vector $\mathbf{r}^i$. The outputs should track the reference points. Thus, the outputs at the reference

time instants are only require to remain in the prediction model, that is, we only need to minimize the error of outputs at the reference time instants. We formulate the extraction matrix which can only extract rows corresponding to the reference points of the vectors and matrices of the prediction model. The following set, Ψ^i, is the set of the unique indices of the reference time instants for i-th output. It is based on the lifted vector formulation. In the lifted output vector, the i-th output of the j-th time point, $y^i(j)$, is the $(n_y j - (n_y - i))$-th component of the lifted output vector $\mathbf{y}$.

$$\Psi^i = \{n_y t_1^i - (n_y - i),\ n_y t_2^i - (n_y - i),\ \cdots,\ n_y t_{N^i}^i - (n_y - i)\} \quad (4.3)$$

Then, the union is defined as

$$\mathbf{\Psi} = \bigcup_{i=1}^{n_y} \Psi^i \quad (4.4)$$

The vector of reference values can be generated by the following rule.

$$\mathbf{r}_j = \begin{cases} \mathbf{r}^i \left(\frac{j+(n_y-i)}{n_y} \right) & \text{, if } j \in \Psi^i,\ j = 1,\ 2,\ \cdots,\ n_y N \\ 0 & \text{, otherwise} \end{cases} \quad (4.5)$$

where $\mathbf{r}_j$ is the jth component of a row vector $\mathbf{r}$ and the components of unspecified time instants are zero. For generating the extraction matrix, we first should generate the following row vector z.

$$z_i = \begin{cases} 1 & \text{, if } i \in \mathbf{\Psi},\ i = 1,\ 2,\ \cdots,\ n_y N \\ 0 & \text{, otherwise} \end{cases} \quad (4.6)$$

where z_i is the i-th component of a row vector z. The row vector z should be modified as the following form because the proposed algorithm is based on predictive control.

$$z^p(t+1) \triangleq \begin{bmatrix} z_{n_y t+1} & z_{n_y t+2} & \cdots & z_{n_y t+n_y p} \end{bmatrix} \tag{4.7}$$

where $z_{n_y t+1}$ is the $(n_y t+1)$-th component of the vector z (4.6). Finally, the extraction matrix is completed by the following rule.

$$Z^p_{i,\,j}(t+1) = \begin{cases} 1 & \text{, if } z^p_j(t+1) = 1,\ \sum_{l=1}^{j} z^p_l(t+1) = i \\ 0 & \text{, otherwise} \end{cases} \tag{4.8}$$

where $Z^p_{i,j}(t+1)$ is the $(i,\ j)$ element of a matrix $Z^p(t+1)$.

Example 4.1

Consider the MIMO system with two outputs, $N = 5$ and the following reference time instants.

$$\begin{aligned} \psi^1 &= \{2,\ 4\} \\ \psi^2 &= \{2,\ 5\} \end{aligned} \tag{4.9}$$

In this case, Ψ^i, $\mathbf{\Psi}$, z, and Z are as follows.

$$\Psi^1 = \{3,\ 7\} \tag{4.10}$$

$$\Psi^2 = \{4,\ 10\} \tag{4.11}$$

$$\mathbf{\Psi} = \{3,\ 4,\ 7,\ 10\} \tag{4.12}$$

$$z = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

$$Z = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.14}$$

If the current time is 2 and the prediction horizon is 2, $z^p(t+1)$ and $Z^p(t+1)$ are as follows.

$$z^p(t+1) = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.15}$$

$$Z^p(t+1) = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.16}$$

4.2.2 Constrained PTP ILMPC

We can generate the vectors, the matrices, and the prediction model for PTP ILMPC by using extraction matrix $Z^p(t+1)$. Tilde ($\tilde{}$) denotes the vectors or the matrices which only have components corresponding to the reference points. For example, $\mathbf{G}$ is used for ILMPC and $\tilde{\mathbf{G}}$ is used for PTP ILMPC. The prediction model for PTP ILMPC is given as

$$\hat{\tilde{\mathbf{y}}}_k^p(t+1|t) = \tilde{\mathbf{y}}_{k-1}^p(t+1) + \tilde{\mathbf{G}}\Delta\delta\mathbf{u}_k^m(t) + \tilde{\mathbf{F}}\Delta\hat{x}_k(t|t) \tag{4.17}$$

$$\tilde{\mathbf{G}} = Z^p(t+1)\mathbf{G},\ \tilde{\mathbf{F}} = Z^p(t+1)\mathbf{F} \tag{4.18}$$

$$\tilde{\mathbf{y}}_{k-1}^p(t+1) = Z^p(t+1)\mathbf{y}_{k-1}^p(t+1) \tag{4.19}$$

The vector of reference values is generated in the same manner. $\tilde{\mathbf{r}}^p(t+1) = Z^p(t+1)\mathbf{r}^p(t+1)$.

Then we can design the PTP ILMPC controller with the following objective function.

$$\min_{\Delta\mathbf{u}_k^m(t),\ \varepsilon_k^p(t+1)} \frac{1}{2}\Big\{ \|\hat{\tilde{\mathbf{e}}}_k^p(t+1|t)\|_{\tilde{\mathbf{Q}}}^2 + \|\mathbf{u}_k^m(t)\|_{\mathbf{S}}^2 + \|\delta\mathbf{u}_k^m(t)\|_{\mathbf{R}}^2 + \|\Delta\mathbf{u}_k^m(t)\|_{\mathbf{P}}^2 + \|\varepsilon_k^p(t+1)\|_{\mathbf{E}}^2 \Big\} \tag{4.20}$$

where $\hat{\tilde{\mathbf{e}}}_k^p(t+1|t) = \tilde{\mathbf{r}}^p(t+1) - \hat{\tilde{\mathbf{y}}}_k^p(t+1|t)$ and $\|x\|_Q^2 = x^T Q x$. In many control applications, input and output constraints are require to ensure safety, smooth operations. In ILMPC controller, we consider the following constraints for this purpose.

$$\begin{aligned}
&\mathbf{u}_{\text{min}}^m \le \mathbf{u}_k^m(t) \le \mathbf{u}_{\text{max}}^m \\
&\delta\mathbf{u}_{\text{min}}^m \le \delta\mathbf{u}_k^m(t) \le \delta\mathbf{u}_{\text{max}}^m \\
&\Delta\mathbf{u}_{\text{min}}^m \le \Delta\mathbf{u}_k^m(t) \le \Delta\mathbf{u}_{\text{max}}^m \\
&\mathbf{y}_{\text{min}}^p - \varepsilon_k^p(t+1) \le \hat{\mathbf{y}}_k^p(t+1|t) \le \mathbf{y}_{\text{max}}^p + \varepsilon_k^p(t+1) \\
&\varepsilon_k^p(t+1) \ge 0
\end{aligned} \tag{4.21}$$

where $\varepsilon_k^p(t+1)$, called slack variable, is defined such that it is non-zero only if a constraint is violated, and ensures that a feasible solution will always be found. It is referred to as constraints softening [86, 87]. $\mathbf{u}_k^m(t),\ \delta\mathbf{u}_k^m(t)$, and $\Delta\delta\mathbf{u}_k^m(t)$ in the objective function and

constraints can be expressed with respect to $\Delta \mathbf{u}_k^m(t)$ as follows:

$$\begin{aligned} \mathbf{u}_k^m(t) &= \mathbf{u}_{k-1}^m(t) + \Delta \mathbf{u}_k^m(t) \\ \delta \mathbf{u}_k^m(t) &= \mathbf{I}_L \mathbf{u}_{k-1}^m(t) - \mathbf{I}_a u_k(t-1) + \mathbf{I}_L \Delta \mathbf{u}_k^m(t) \\ \Delta \delta \mathbf{u}_k^m(t) &= -\mathbf{I}_a \Delta u_k(t-1) + \mathbf{I}_L \Delta \mathbf{u}_k^m(t) \end{aligned} \tag{4.22}$$

where

$$\mathbf{I}_L \in \mathbb{R}^{n_u m \times n_u m} = \begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ -I & I & 0 & \cdots & 0 \\ 0 & -I & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & I \end{bmatrix}, \ \mathbf{I}_a \in \mathbb{R}^{n_u m \times n_u} = \begin{bmatrix} I \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4.23}$$

where I is the identity matrix with the size of $n_u \times n_u$. The objective function can be rewritten by combining $\Delta \mathbf{u}_k^m(t)$ and $\varepsilon_k^p(t+1)$. The following is a standard quadratic programming (QP) problem for constrained PTP ILMPC.

$$\begin{aligned} \min_{\Delta \mathbf{u}_k^m(t),\, \varepsilon_k^p(t+1)} & \frac{1}{2} \begin{bmatrix} \Delta \mathbf{u}_k^m(t) \\ \varepsilon_k^p(t+1) \end{bmatrix}^T \begin{bmatrix} \mathcal{H} & 0 \\ 0 & \mathbf{E} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{u}_k^m(t) \\ \varepsilon_k^p(t+1) \end{bmatrix} \\ & + \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}^T \begin{bmatrix} \Delta \mathbf{u}_k^m(t) \\ \varepsilon_k^p(t+1) \end{bmatrix} \end{aligned} \tag{4.24}$$

subject to

$$\mathbf{M}^u \begin{bmatrix} \Delta \mathbf{u}_k^m(t) \\ \varepsilon_k^p(t+1) \end{bmatrix} \leq \mathbf{M}_k(t) \tag{4.25}$$

where

$$\mathcal{H} \triangleq \mathbf{I}_L^T \tilde{\mathbf{G}}^T \tilde{\mathbf{Q}} \tilde{\mathbf{G}} \mathbf{I}_L + \mathbf{S} + \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L + \mathbf{P} \tag{4.26}$$

$$\begin{aligned} \mathbf{f} \triangleq & \mathbf{I}_L^T \tilde{\mathbf{G}}^T \tilde{\mathbf{Q}} \left[\tilde{\mathbf{F}} \Delta \hat{x}_k(t|t) - \tilde{\mathbf{G}} \mathbf{I}_a \Delta u_k(t-1) - \tilde{\mathbf{e}}_{k-1}^p(t+1) \right] \\ & + \mathbf{S} \mathbf{u}_{k-1}^m(t) + \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \mathbf{u}_{k-1}^m(t) - \mathbf{I}_L^T \mathbf{R} \mathbf{I}_a u_k(t-1) \end{aligned} \tag{4.27}$$

$$\mathbf{M}^u \triangleq \begin{bmatrix} -\mathbf{I} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} \\ -\mathbf{I}_L & \mathbf{0} \\ \mathbf{I}_L & \mathbf{0} \\ -\mathbf{I}_L & \mathbf{0} \\ \mathbf{I}_L & \mathbf{0} \\ -\mathbf{G}\mathbf{I}_L & -\mathbf{I} \\ \mathbf{G}\mathbf{I}_L & -\mathbf{I} \\ \mathbf{0} & -\mathbf{I} \end{bmatrix} \tag{4.28}$$

$$\mathbf{M}_k(t) \triangleq \begin{bmatrix} -\mathbf{u}_{\text{min}}^m + \mathbf{u}_{k-1}^m(t) \\ \mathbf{u}_{\text{max}}^m - \mathbf{u}_{k-1}^m(t) \\ -\delta \mathbf{u}_{\text{min}}^m + \mathbf{I}_L \mathbf{u}_{k-1}^m(t) - \mathbf{I}_a u_k(t-1) \\ \delta \mathbf{u}_{\text{max}}^m - \mathbf{I}_L \mathbf{u}_{k-1}^m(t) + \mathbf{I}_a u_k(t-1) \\ -\Delta \mathbf{u}_{\text{min}}^m - \mathbf{I}_a \Delta u_{k-1}(t-1) \\ \Delta \mathbf{u}_{\text{max}}^m + \mathbf{I}_a \Delta u_{k-1}(t-1) \\ -\mathbf{y}_{\text{min}}^p + \mathbf{y}_{k-1}^p(t+1) - \mathbf{G}\mathbf{I}_a \Delta u_k(t-1) + \mathbf{F} \Delta \hat{x}_k(t|t) \\ \mathbf{y}_{\text{max}}^p - \mathbf{y}_{k-1}^p(t+1) + \mathbf{G}\mathbf{I}_a \Delta u_k(t-1) - \mathbf{F} \Delta \hat{x}_k(t|t) \\ \mathbf{0} \end{bmatrix} \tag{4.29}$$

where $\mathbf{I}$ is the identity matrix with the size of $n_u m \times n_u m$. The optimization problem can be solved by appropriate QP solver. The first input in the optimal sequence is sent into the plant.

4.2.3 Iterative Learning Observer

We applied the ILO [90] to the PTP ILMPC, so that the convergence of the PTP ILMPC can be guaranteed. We use the ILO to estimate output, not state. For convergence, $y_k^e(t) \triangleq y_k(t) - \hat{y}_k(t|t)$ should be zero for all time t as $k \to \infty$. General observer, however, can guarantee $y_k^e(t) \to 0$ as $t \to \infty$. Hence the error of $y_k(t) - \hat{y}_k(t|t)$ always exists at the early time points. First, the state is estimated along the time direction as follows:

$$\begin{aligned} \hat{x}_k(t|t-1) &= A\hat{x}_k(t-1|t-1) + B\delta u_k(t-1) \\ \hat{x}_k(t|t) &= \hat{x}_k(t|t-1) + K\left\{y_k(t) - C\hat{x}(t|t-1)\right\} \end{aligned} \tag{4.30}$$

where K is the time-wise observer gain for the system (2.5). The additional input $v_k(t)$ is added in the measurement update equation for further correction in the direction of iteration.

$$\hat{x}_k(t|t) = \hat{x}_k(t|t-1) + K\left\{y_k(t) - C\hat{x}_k(t|t-1)\right\} - v_k(t-1) \tag{4.31}$$

Using the observer error $x_k^e(t) \triangleq x_k(t) - \hat{x}_k(t|t)$ and the state space model (2.5), the following state space model can be obtained.

$$\begin{aligned} x_k^e(t+1) &= A^e x_k^e(t) + v_k(t) \\ y_k^e(t) &= C x_k^e(t) \end{aligned} \tag{4.32}$$

where $A^e \triangleq A - KCA$. Now, it becomes a problem to find input $v_k(t)$ to satisfy $y_k^e(t) \to 0 \ \forall t$ as $k \to \infty$. It can be solved by general ILC algorithm. We applied Arimoto-type ILC algorithm, so that an ILO gain can be determined regardless of time-wise observer gain K

because Arimoto-type ILC algorithm uses input and output matrices only. In this case, the input matrix is the identity matrix and the output matrix is C.

Theorem 4.1. *Consider the linear system (2.5) with $x_k^e(0) = x_0^e$ and the following ILO input update law.*

$$v_k(t) = v_{k-1}(t) - Ly_{k-1}^e(t+1) \tag{4.33}$$

The $y_k^e(t)$ converges asymptotically to zero $\forall t$ as $k \to \infty$ if L is chosen such that $\rho(I - CL) < 1$, where $\rho(\cdot)$ is a spectral radius.

Proof First, the state space model (4.32) and the ILO input update law (4.33) can be recast as the lifted form.

$$\mathbf{y}_k^e = \mathbf{G}^e\mathbf{v}_k + \mathbf{F}^e x_0^e \tag{4.34}$$

$$\mathbf{v}_k = \mathbf{v}_{k-1} - \mathbf{L}\mathbf{y}_{k-1}^e \tag{4.35}$$

where

$$
\begin{aligned}
\mathbf{G}^e &\triangleq \begin{bmatrix} C & 0 & \cdots & 0 \\ CA^e & C & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{eN-1} & CA^{eN-2} & \cdots & C \end{bmatrix}, \ \mathbf{F}^e \triangleq \begin{bmatrix} CA^e \\ CA^{e2} \\ \vdots \\ CA^{eN} \end{bmatrix} \\
\mathbf{L} &\triangleq \begin{bmatrix} L & 0 & \cdots & 0 \\ 0 & L & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & L \end{bmatrix} \\
\mathbf{y}_k^e &\triangleq \begin{bmatrix} y_k^e(1) & y_k^e(2) & \cdots & y_k^e(N) \end{bmatrix} \\
\mathbf{v}_k &\triangleq \begin{bmatrix} v_k(0) & v_k(1) & \cdots & v_k(N-1) \end{bmatrix}
\end{aligned}
\tag{4.36}
$$

Substitution of Eq.(4.35) into Eq.(4.34) yields

$$
\mathbf{y}_k^e = \mathbf{G}^e \mathbf{v}_{k-1} - \mathbf{G}^e \mathbf{L} \mathbf{y}_{k-1}^e + \mathbf{F}^e x_0^e \tag{4.37}
$$

Using $\mathbf{y}_{k-1}^e = \mathbf{G}^e \mathbf{v}_{k-1} + \mathbf{F}^e x_0^e$, the following can be derived:

$$
\mathbf{y}_k^e = \left(\mathbf{I} - \mathbf{G}^e \mathbf{L}\right) \mathbf{y}_{k-1}^e \tag{4.38}
$$

Thus, $\mathbf{y}_k^e \to 0$ as $k \to \infty$ if $\mathbf{L}$ is chosen such that $\rho(\mathbf{I} - \mathbf{G}^e \mathbf{L}) < 1$.
where

$$
\mathbf{I} - \mathbf{G}^e \mathbf{L} = \begin{bmatrix} I - CL & 0 & \cdots & 0 \\ -CA^e L & I - CL & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -CA^{eN-1} L & -CA^{eN-2} L & \cdots & I - CL \end{bmatrix} \tag{4.39}
$$

Eigenvalues of the block triangular matrix $(\mathbf{I} - \mathbf{G}^e\mathbf{L})$ are identical to the eigenvalues of the diagonal block $(I - CL)$, that is, $\rho(\mathbf{I} - \mathbf{G}^e\mathbf{L}) = \rho(I - CL)$. Thus, $\mathbf{y}_k^e \to 0$ as $k \to \infty$ if L is chosen such that $\rho(I - CL) < 1$. ■

4.3 Convergence Analysis

4.3.1 Convergence of Input Trajectory

This section presents the convergence proof for the constrained PTP ILMPC under the following assumptions.

A1. There exists a feasible input trajectory $\mathbf{u}_\infty$ such that $\mathbf{e}_\infty = 0$, $\mathbf{u}_{\min} \leq \mathbf{u}_\infty \leq \mathbf{u}_{\max}$ and $\delta\mathbf{u}_{\min} \leq \delta\mathbf{u}_\infty \leq \delta\mathbf{u}_{\max}$.

A2. Output constraints are satisfied when input trajectory is converged. Thus, $\varepsilon_\infty(t)$ become zero $\forall t$.

A3. The system has the same initial condition for all iteration and shows the same output trajectory and the state trajectory under the same input trajectory.

A4. $\tilde{\mathbf{Q}},\ \mathbf{P},\ \mathbf{E},\ \mathbf{S}$ and $\mathbf{R}$ are symmetric positive definite.

Lemma 4.1. *If $Q,\ S,\ R$ and E are symmetric positive definite, then*

$$\|a+b\|_Q^2 + \|c+d\|_S^2 + \|e\|_R^2 + \|f\|_E^2 \leq \left(\sqrt{\|a\|_Q^2 + \|c\|_S^2 + \|e\|_R^2 + \|f\|_E^2} + \sqrt{\|b\|_Q^2} + \sqrt{\|d\|_S^2}\right)^2 \quad (4.40)$$

Proof

$$
\begin{aligned}
&\|a+b\|_Q^2+\|c+d\|_S^2+\|e\|_R^2+\|f\|_E^2= \\
&(a+b)^TQ(a+b)+(c+d)^TS(c+d)+e^TRe+f^TEf= \\
&a^TQa+b^TQb+c^TSc+d^TSd+e^TRe+f^TEf+2a^TQb+2c^TSd\leq \\
&a^TQa+b^TQb+c^TSc+d^TSd+e^TRe+f^TEf \\
&+2\sqrt{a^TQa}\sqrt{b^TQb}+2\sqrt{c^TSc}\sqrt{d^TSd}\leq \\
&a^TQa+b^TQb+c^TSc+d^TSd+e^TRe+f^TEf \\
&+2\sqrt{a^TQa+c^TSc+e^TRe+f^TEf}\sqrt{b^TQb} \\
&+2\sqrt{a^TQa+c^TSc+e^TRe+f^TEf}\sqrt{d^TSd}+2\sqrt{b^TQb}\sqrt{d^TSd}= \\
&\left(\sqrt{a^TQa+c^TSc+e^TRe+f^TEf}+\sqrt{b^TQb}+\sqrt{d^TSd}\right)^2= \\
&\left(\sqrt{\|a\|_Q^2+\|c\|_S^2+\|e\|_R^2+\|f\|_E^2}+\sqrt{\|b\|_Q^2}+\sqrt{\|d\|_S^2}\right)^2
\end{aligned}
\tag{4.41}
$$

■

Theorem 4.2. *Consider the assumptions A1-A4, the system (2.1), the ILO (4.31) and the constrained optimization problem (4.20, 4.21). Then,* $\Delta u_k(t)\to 0\ \forall t\in\{0,\ 1,\ \cdots,\ N-1\}$ *as* $k\to\infty$.

Proof We consider the objective function at time t of the k-th iteration

$$
\begin{aligned}
\Phi_k(t)=&\frac{1}{2}\Big\{\|\hat{\tilde{\mathbf{e}}}_k^p(t+1|t)\|_{\tilde{\mathbf{Q}}}^2+\|\mathbf{u}_k^m(t)\|_{\mathbf{S}}^2+\|\delta\mathbf{u}_k^m(t)\|_{\mathbf{R}}^2 \\
&+\|\Delta\mathbf{u}_k^m(t)\|_{\mathbf{P}}^2+\|\varepsilon_k^p(t+1)\|_{\mathbf{E}}^2\Big\}
\end{aligned}
\tag{4.42}
$$

and the minimizer of the optimization problem

$$J_k(t) = \min_{\Delta \mathbf{u}_k^m(t),\ \varepsilon_k^p(t+1)} \Phi_k(t) \geq 0 \tag{4.43}$$

subject to

$$\begin{aligned}
&\mathbf{u}_{\text{min}}^m \leq \mathbf{u}_k^m(t) \leq \mathbf{u}_{\text{max}}^m \\
&\delta \mathbf{u}_{\text{min}}^m \leq \delta \mathbf{u}_k^m(t) \leq \delta \mathbf{u}_{\text{max}}^m \\
&\Delta \mathbf{u}_{\text{min}}^m \leq \Delta \mathbf{u}_k^m(t) \leq \Delta \mathbf{u}_{\text{max}}^m \\
&\mathbf{y}_{\text{min}}^p - \varepsilon_k^p(t+1) \leq \hat{\mathbf{y}}_k^p(t+1|t) \leq \mathbf{y}_{\text{max}}^p + \varepsilon_k^p(t+1) \\
&\varepsilon_k^p(t+1) \geq 0
\end{aligned} \tag{4.44}$$

We will use the fact that an optimal solution of an objective function is always less than or equal to a feasible solution, then we will use the k-th optimal solution for the $(k+1)$-th feasible solution. First, we define the following estimation errors

$$\begin{aligned}
\tilde{\mathbf{e}}_k^{e,p}(t+1|t) &\triangleq \tilde{\mathbf{e}}_k^p(t+1) - \hat{\tilde{\mathbf{e}}}_k^p(t+1|t) \\
\mathbf{y}_k^{e,p}(t+1|t) &\triangleq \mathbf{y}_k^p(t+1) - \hat{\mathbf{y}}_k^p(t+1|t)
\end{aligned} \tag{4.45}$$

We can derive the $(k+1)$-th prediction error and slack variable using (2.28), (4.17), (4.44), (4.45) and the above assumptions. The following is obtained using (2.28) and the definition, $\hat{\tilde{\mathbf{e}}}_{k+1}^p(t+1|t) = \tilde{\mathbf{r}}^p(t+1) - \hat{\tilde{\mathbf{y}}}_{k+1}^p(t+1|t)$.

$$\hat{\tilde{\mathbf{e}}}_{k+1}^p(t+1|t) = \tilde{\mathbf{e}}_k^p(t+1) - \tilde{\mathbf{G}} \Delta \delta \mathbf{u}_{k+1}^m(t) - \tilde{\mathbf{F}} \Delta \hat{x}_{k+1}(t|t) \tag{4.46}$$

Let $(\hat{\tilde{\mathbf{e}}}_k^{*,p}(t+1|t),\ \mathbf{u}_k^{*,m}(t),\ \varepsilon_k^{*,p}(t+1))$ be the optimal solution for

the k-th iteration. At the $(k+1)$-th iteration, the optimal solution of the k-th iteration until time t is used for the $(k+1)$-th iteration. Thus, $\Delta\delta\mathbf{u}^m_{k+1}(t)$ and $\Delta\hat{x}_{k+1}(t|t)$ become zero because of the assumption (A3). From the assumption (A3) and (4.45), we have

$$\hat{\tilde{\mathbf{e}}}^p_{k+1}(t+1|t) = \hat{\tilde{\mathbf{e}}}^p_k(t+1|t) + \tilde{\mathbf{e}}^{e,p}_k(t+1|t) \tag{4.47}$$

Similarly, using (2.28), (4.44) and (4.45), we also have

$$\varepsilon^p_{k+1}(t+1) = \varepsilon^p_k(t+1) + \mathbf{y}^{e,p}_k(t+1|t) \tag{4.48}$$

The following is a feasible solution but may not be optimal.

$$\begin{aligned}
\hat{\tilde{\mathbf{e}}}_{k+1}(t+1|t) &= \hat{\tilde{\mathbf{e}}}^{*,p}_k(t+1|t) + \tilde{\mathbf{e}}^{e,p}_k(t+1|t) \\
\mathbf{u}^m_{k+1}(t) &= \mathbf{u}^{*,m}_k(t) \\
\delta\mathbf{u}^m_{k+1}(t) &= \delta\mathbf{u}^{*,m}_k(t) \\
\Delta\mathbf{u}^m_{k+1}(t) &= 0 \\
\varepsilon^p_{k+1}(t+1) &= \varepsilon^{*,p}_k(t+1) + \mathbf{y}^{e,p}_k(t+1|t)
\end{aligned} \tag{4.49}$$

We have

$$\begin{aligned}
J_{k+1}(t) \leq \frac{1}{2}\Big\{ &\|\hat{\tilde{\mathbf{e}}}^{*,p}_k(t+1|t) + \tilde{\mathbf{e}}^{e,p}_k(t+1|t)\|^2_{\tilde{\mathbf{Q}}} + \|\mathbf{u}^{*,m}_k(t)\|^2_{\mathbf{S}} \\
&+ \|\delta\mathbf{u}^{*,m}_k(t)\|^2_{\mathbf{R}} + \|\varepsilon^{*,p}_k(t+1) + \mathbf{y}^{e,p}_k(t+1|t)\|^2_{\mathbf{E}} \Big\}
\end{aligned} \tag{4.50}$$

By Lemma 4.1, we have

$$\begin{aligned} J_{k+1}(t) \leq & \left(\sqrt{J_k(t) - \frac{1}{2}\|\Delta \mathbf{u}_k^{*,m}(t)\|_{\mathbf{P}}^2} \right. \\ & \left. + \sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_k^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_k^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right)^2 \\ \leq & \left(\sqrt{J_k(t)} + \sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_k^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} \right. \\ & \left. + \sqrt{\frac{1}{2}\|\mathbf{y}_k^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right)^2 \end{aligned} \tag{4.51}$$

which yields

$$\begin{aligned} \sqrt{J_{k+1}(t)} \leq \sqrt{J_1(t)} + \sum_{j=1}^{k} \Bigg\{ & \sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_j^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} \\ & + \sqrt{\frac{1}{2}\|\mathbf{y}_j^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \Bigg\} \end{aligned} \tag{4.52}$$

By Theorem 4.1, $\mathbf{y}_k^{e,p}(t+1|t) \to 0$ as $k \to \infty$. Furthermore, $\tilde{\mathbf{e}}_k^{e,p}(t+1|t) \to 0$ as $k \to \infty$ because $\tilde{\mathbf{e}}_k(t+1|t) = \tilde{\mathbf{e}}_k^p(t+1) - \hat{\tilde{\mathbf{e}}}_k^p(t+1|t) = \tilde{\mathbf{r}}^p(t+1) - Z^p(t+1)\mathbf{y}_k^p(t+1) - \tilde{\mathbf{r}}^p(t+1) + Z^p(t+1)\hat{\mathbf{y}}_k^p(t+1|t) = -Z^p(t+1)\mathbf{y}_k^{e,p}(t+1|t)$. The second term on the right-hand side of (4.52) is bounded for all k. Thus, we have

$$J_k(t) \leq J^{max}(t) < \infty \ \ \forall k < 0 \tag{4.53}$$

From (4.51), we obtain

$$
\begin{aligned}
J_{k+1}(t) \leq & \left(\sqrt{J_k(t) - \frac{1}{2}\|\Delta \mathbf{u}_k^{*,m}(t)\|_{\mathbf{P}}^2} \right. \\
& \left. + \sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_k^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_k^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right)^2 \\
= & J_k(t) - \frac{1}{2}\|\Delta \mathbf{u}_k^{*,m}(t)\|_{\mathbf{P}}^2 \\
& + \left(\sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_k^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_k^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right)^2 \\
& + 2\sqrt{J_k(t) - \frac{1}{2}\|\Delta \mathbf{u}_k^{*,m}(t)\|_{\mathbf{P}}^2} \\
& \times \left(\sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_k^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_k^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right) \\
\leq & J_k(t) - \frac{1}{2}\|\Delta \mathbf{u}_k^{*,m}(t)\|_{\mathbf{P}}^2 \\
& + \left(\sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_k^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_k^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right)^2 \\
& + 2\sqrt{J^{max}(t)} \\
& \times \left(\sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_k^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_k^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right)
\end{aligned}
\tag{4.54}
$$

Then, it leads to

$$
\begin{aligned}
&J_{k+1}(t) + \frac{1}{2}\sum_{j=1}^{k} \|\Delta \mathbf{u}_j^{*,m}(t)\|_{\mathbf{P}}^2 \leq J_1(t) \\
&+ \sum_{j=1}^{k} \left\{ \left(\sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_j^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_j^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right)^2 \right. \\
&+ 2\sqrt{J^{max}(t)} \\
&\left. \times \left(\sqrt{\frac{1}{2}\|\tilde{\mathbf{e}}_j^{e,p}(t+1|t)\|_{\tilde{\mathbf{Q}}}^2} + \sqrt{\frac{1}{2}\|\mathbf{y}_j^{e,p}(t+1|t)\|_{\mathbf{E}}^2} \right) \right\}
\end{aligned}
\tag{4.55}
$$

where $J^{max}(t)$, $\tilde{\mathbf{e}}_k^{e,p}(t+1|t)$, and $\mathbf{y}_k^{e,p}(t+1|t)$ are bounded for all k and t. Thus,

$$
J_{k+1}(t) + \frac{1}{2}\sum_{j=1}^{k} \|\Delta \mathbf{u}_j^{*,m}(t)\|_{\mathbf{P}}^2 < \infty \tag{4.56}
$$

Hence, $\Delta \mathbf{u}_k^{*,m}(t) \to 0,\ \forall t$ as $k \to \infty$. ■

4.3.2 Convergence of Error

So far we, we showed that $u_k(t)$ converges to $u^*(t),\ \forall t$ as $k \to \infty$. Now, we show that $e_k(t)$ converges to $e^*(t)$ or $0,\ \forall t$ as $k \to \infty$. By the assumptions of A1 and A2, input constraints and output constraints are respected as $k \to \infty$. Thus, constrained solution and unconstrained solution become equal as $k \to \infty$. Convergence is proved using unconstrained solution. We consider full time-sequence without real-time feedback because we are concerned with the error of all reference points when the convergence is achieved. We define $\mathbf{u}_k,\ \Delta \mathbf{u}_k,\ \tilde{\mathbf{r}},\ \tilde{\mathbf{y}}_k$, and $\tilde{\mathbf{e}}_k$ as the vectors of full time-sequence. It can

be derived by setting $t = 0$ and $m = p = N$. Unconstrained solution of the proposed PTP ILMPC is $\Delta \mathbf{u}_k^m(t) = -\mathcal{H}^{-1}\mathbf{f}$ where $\mathcal{H}$ and $\mathbf{f}$ are (4.26) and (4.27), respectively. In this section, we consider full time-sequence at time 0 and non-active output constraints (output constraints are satisfied), and thus we can set $\mathbf{P} = 0$ of $\mathcal{H}$ and $\mathbf{f}$ can be simplified as follows.

$$\begin{aligned} \mathcal{H} &= \mathbf{I}_L^T \tilde{\mathbf{G}}^T \tilde{\mathbf{Q}} \tilde{\mathbf{G}} \mathbf{I}_L + \mathbf{S} + \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \\ \mathbf{f} &= -\mathbf{I}_L^T \tilde{\mathbf{G}}^T \tilde{\mathbf{Q}} \tilde{\mathbf{e}}_{k-1} + \mathbf{S}\mathbf{u}_{k-1} + \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \mathbf{u}_{k-1} \end{aligned} \tag{4.57}$$

Theorem 4.3. *Consider the linear system (2.1) and the proposed PTP ILMPC controller.* $\tilde{e}_k(t) \to 0$ *for all reference points as* $k \to \infty$ *if* $\mathbf{S} = \mathbf{R} = 0$ *where* $\mathbf{S}$ *and* $\mathbf{R}$ *are the weighting matrices of (4.20).*

Proof In case of assuming full time-sequence, the unconstrained solution of the proposed PTP ILMPC is as follows

$$\Delta \mathbf{u}_k = \mathcal{H}^{-1} \left(\mathbf{I}_L^T \tilde{\mathbf{G}}^T \tilde{\mathbf{Q}} \tilde{\mathbf{e}}_{k-1} - \mathbf{S}\mathbf{u}_{k-1} - \mathbf{I}_L^T \mathbf{R} \mathbf{I}_L \mathbf{u}_{k-1} \right) \tag{4.58}$$

where $\mathbf{S},\ \mathbf{R} = 0$; thus,

$$\Delta \mathbf{u}_k = \mathcal{H}^{-1} \mathbf{I}_L^T \tilde{\mathbf{G}}^T \tilde{\mathbf{Q}} \tilde{\mathbf{e}}_{k-1} \tag{4.59}$$

By Theorem 4.2, if $k \to \infty$,

$$\Delta \mathbf{u}_\infty = 0 = \mathcal{H}^{-1} \mathbf{I}_L^T \tilde{\mathbf{G}}^T \tilde{\mathbf{Q}} \tilde{\mathbf{e}}_\infty \tag{4.60}$$

This implies that the final error satisfies $\tilde{\mathbf{e}}_\infty = 0$. ■

Theorem 4.4. *Consider the linear system (2.1) and the proposed PTP*

ILMPC controller. $\tilde{e}_k(t) \to e^*(t)$ *for all reference points as* $k \to \infty$.

Proof (4.58) can be rearranged as

$$\mathbf{u}_k = \left(\mathbf{I} - \mathcal{H}^{-1}\mathbf{S} - \mathcal{H}^{-1}\mathbf{I}_L^T\mathbf{R}\mathbf{I}_L\right)\mathbf{u}_{k-1} + \mathcal{H}^{-1}\mathbf{I}_L^T\tilde{\mathbf{G}}^T\tilde{\mathbf{Q}}\tilde{\mathbf{e}}_{k-1} \quad (4.61)$$

By Theorem 4.2, if $k \to \infty$,

$$\begin{aligned} \mathbf{u}_\infty &= \mathbf{H}_1\mathbf{u}_\infty + \mathbf{H}_2\tilde{\mathbf{e}}_\infty \\ &= \mathbf{H}_1\mathbf{u}_\infty + \mathbf{H}_2\left(\tilde{\mathbf{r}} - \tilde{\mathbf{y}}_\infty\right) \end{aligned} \quad (4.62)$$

where

$$\begin{aligned} \mathbf{H}_1 &\triangleq \mathbf{I} - \mathcal{H}^{-1}\mathbf{S} - \mathcal{H}^{-1}\mathbf{I}_L^T\mathbf{R}\mathbf{I}_L \\ \mathbf{H}_2 &\triangleq \mathcal{H}^{-1}\mathbf{I}_L^T\tilde{\mathbf{G}}^T\tilde{\mathbf{Q}} \end{aligned} \quad (4.63)$$

(2.28) can be expressed as the lifted form: $\tilde{\mathbf{y}}_\infty = \tilde{\mathbf{G}}_p\delta\mathbf{u}_\infty = \tilde{\mathbf{G}}_p\mathbf{I}_L\mathbf{u}_\infty$ with $x_k(0) = 0$ where $\tilde{\mathbf{G}}_p$ is the plant matrix. Substitution of $\tilde{\mathbf{y}}_\infty$ into (4.62) yields

$$\mathbf{u}_\infty = \mathbf{H}_1\mathbf{u}_\infty + \mathbf{H}_2\left(\tilde{\mathbf{r}} - \tilde{\mathbf{G}}_p\mathbf{I}_L\mathbf{u}_\infty\right) \quad (4.64)$$

, which can be rearranged to

$$\left(\mathbf{I} - \mathbf{H}_1 + \mathbf{H}_2\tilde{\mathbf{G}}_p\mathbf{I}_L\right)\mathbf{u}_\infty = \mathbf{H}_2\tilde{\mathbf{r}} \quad (4.65)$$

Substituting (4.63) into the matrix in the left-hand side of (4.65) yields

$$\mathbf{I} - \mathbf{H}_1 + \mathbf{H}_2\tilde{\mathbf{G}}_p\mathbf{I}_L = \mathcal{H}^{-1}\left(\mathbf{I}_L^T\tilde{\mathbf{G}}^T\tilde{\mathbf{Q}}\tilde{\mathbf{G}}_p\mathbf{I}_L + \mathbf{S} + \mathbf{I}_L\mathbf{R}\mathbf{I}_L\right) \quad (4.66)$$

$\mathcal{H}$ of (4.57) and the second matrix of the right-hand side of (4.66) have the same structure, and thus the matrix in the left-hand side of (4.65) is invertible because the hessian $(\mathcal{H})$ of the QP problem is invertible. Thus,

$$\mathbf{u}_\infty = \left(\mathbf{I} - \mathbf{H}_1 + \mathbf{H}_2\tilde{\mathbf{G}}_p\mathbf{I}_L\right)^{-1}\mathbf{H}_2\tilde{\mathbf{r}} \tag{4.67}$$

Finally, we can obtain the converged value of error ($\tilde{\mathbf{e}}^*$) by substituting (4.67) into $\tilde{\mathbf{e}}_\infty$ as follows

$$\begin{aligned}\tilde{\mathbf{e}}^* &= \tilde{\mathbf{e}}_\infty = \tilde{\mathbf{r}} - \tilde{\mathbf{y}}_\infty = \tilde{\mathbf{r}} - \tilde{\mathbf{G}}_p\mathbf{I}_L\mathbf{u}_\infty \\ &= \left(\mathbf{I} - \tilde{\mathbf{G}}_p\mathbf{I}_L\left(\mathbf{I} - \mathbf{H}_1 + \mathbf{H}_2\tilde{\mathbf{G}}_p\mathbf{I}_L\right)^{-1}\mathbf{H}_2\right)\tilde{\mathbf{r}}\end{aligned} \tag{4.68}$$

■

4.4 Numerical Examples

4.4.1 Example 1 (Linear SISO System with Disturbance)

The proposed algorithm is illustrated by the following plant transfer function.

$$y_p(s) = \frac{2.5}{(20s+1)(15s+1)}u(s) + \frac{1}{5s+1}d(s) \tag{4.69}$$

where $d(s)$ is disturbance input which enters the system at the 14th and 15th iterations. The controller is designed based on the following

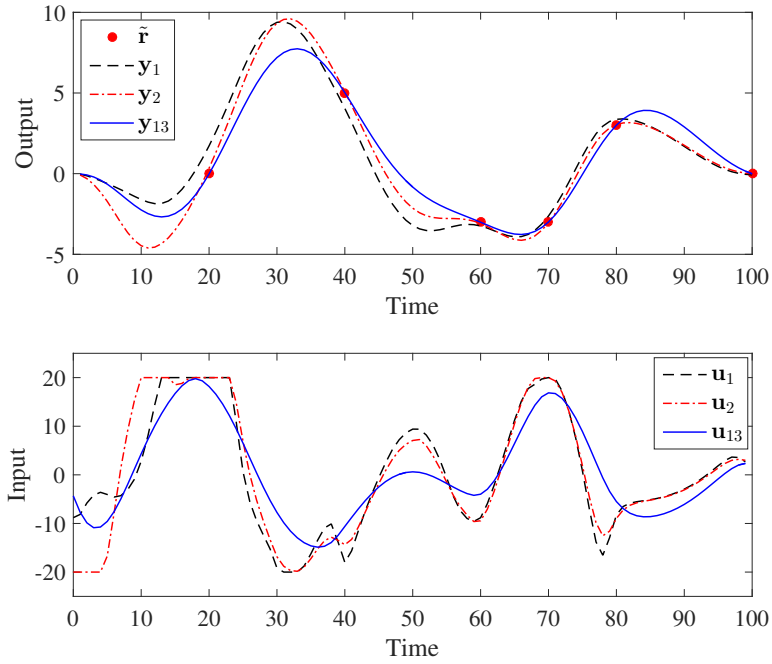


Figure 4.1: (Example 1) The performance of the proposed PTP ILMPC algorithm.

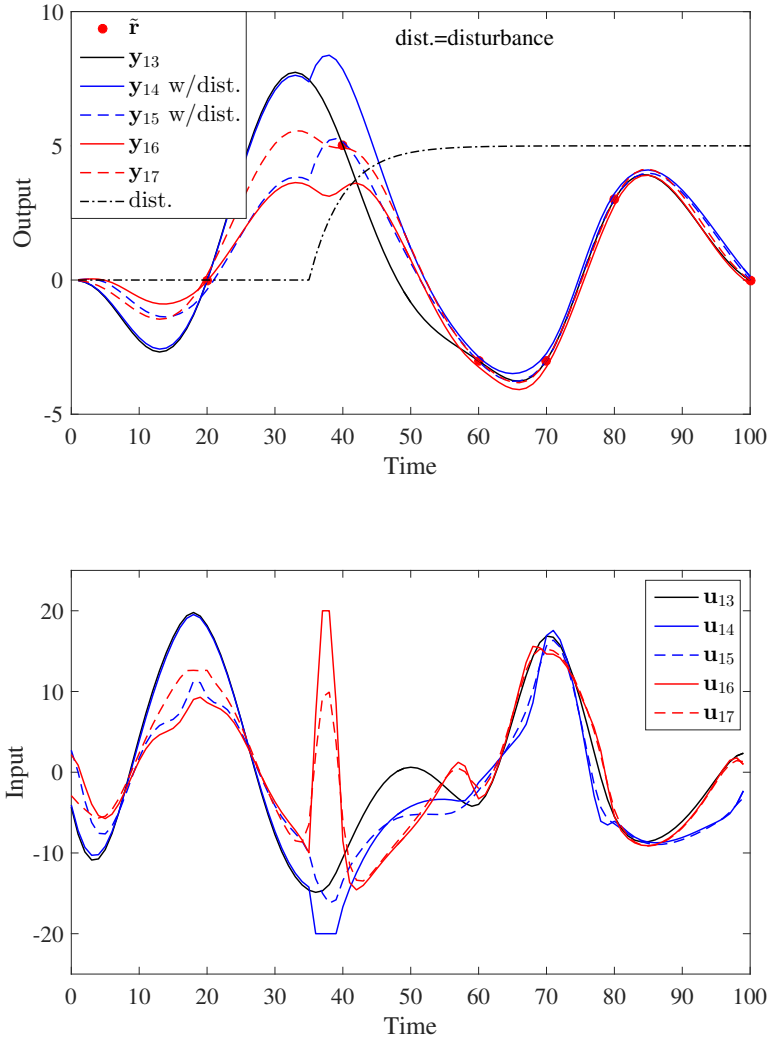


Figure 4.2: (Example 1) The disturbance rejection performance of the proposed PTP ILMPC algorithm.

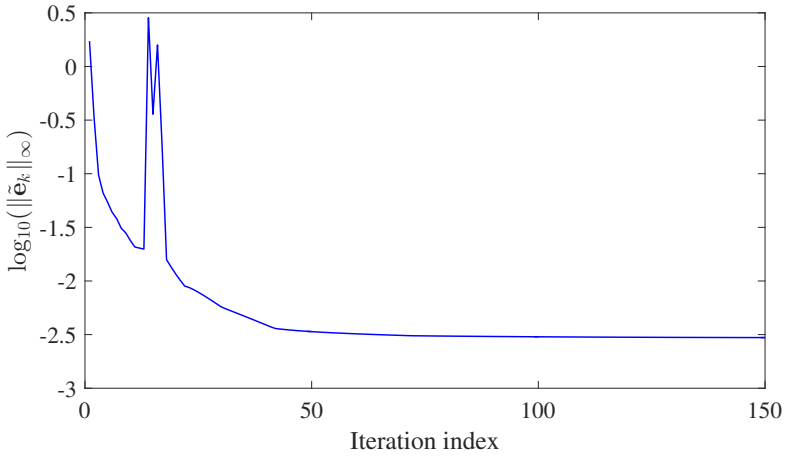


Figure 4.3: (Example 1) Log scale convergence performance for the constrained linear SISO system.

model transfer function.

$$y_m(s) = \frac{1.5}{(18s+1)(15s+1)}u(s) \tag{4.70}$$

Terminal time is 100 with the sampling interval of 1. We used the following parameters for designing the controller: $p_0 = 100,\ m_0 = 20,\ \tilde{\mathbf{Q}} = I,\ \mathbf{S} = 10^{-6}I,\ \mathbf{R} = \mathbf{P} = 10^{-4}I,\ \mathbf{E} = 10^4 I$. The reference time instants are 20, 40, 60, 70, 80, and 100. The vector of reference values is $\tilde{\mathbf{r}} = \begin{bmatrix} 0 & 5 & -3 & -3 & 3 & 0 \end{bmatrix}^T$. The following input constraint is applied to the PTP ILMPC controller.

$$-20 \leq u_k(t) \leq 20 \tag{4.71}$$

First, we need to determine the time-wise observer gain (K) and the iteration-wise observer gain (L). K was obtained using Kalman filter

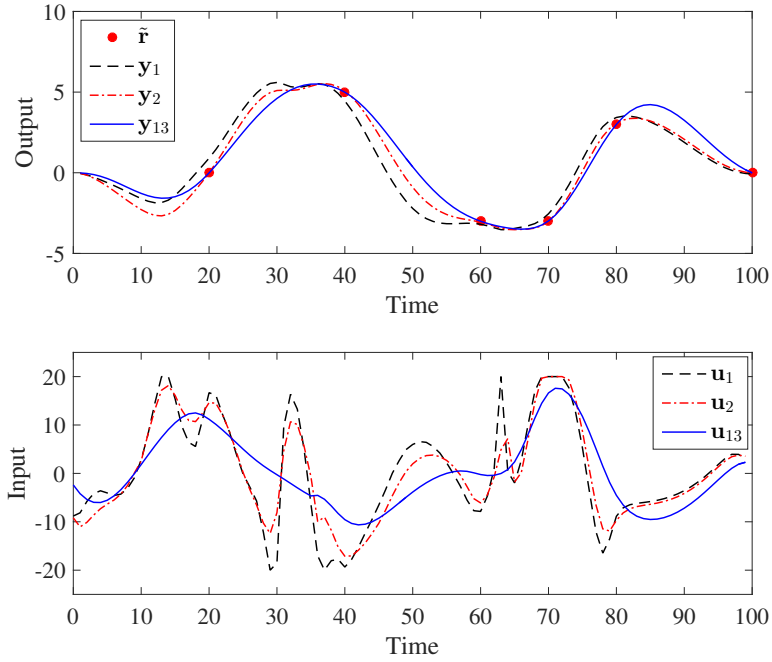


Figure 4.4: (Example 1) The performance of the proposed PTP ILMPC algorithm under output constraint.

in which the state noise covariance matrix was **diag**(0.1, 0.1, 0.001) and the measurement noise variance was 0.001. $C^T(CC^T)^{-1}$ was used for L such that $\rho(I - CL) < 1$.

Fig. 4.1 shows the performance of the proposed PTP ILMPC algorithm in the early iteration. Although there is plant-model mismatch, the tracking error is decreased as k increases. Fig. 4.2 shows the performance under the unknown real-time disturbance. The output of the 13th iteration is converged to the reference points. Step disturbance input with the size of 5 is entered at the time 35 of the 14th and 15th iterations. At the 14th iteration, the disturbance is rejected in time horizon. It is the main advantage of the proposed PTP ILMPC. The existing PTP ILC algorithms cannot reject real-time disturbance because they do not include real-time feedback controller. At the 15th iteration, the output is converging to the reference points by learning to reject previous disturbance. After the 16th iteration, the disturbance does not enter to the system. Although there is no disturbance, the tracking performance of the 16th iteration is decreased because the controller learns to reject the previous disturbance from the previous iteration; however, the output quickly converges to the reference points with the real-time feedback. At the 17th iteration, the output starts to converge to the all reference points again. Fig. 4.3 shows the maximum absolute error. Fig. 4.1 shows the large overshoot although there is no disturbance; thus, the following input and output constraints are used to reduce the large overshoot.

$$-20 \leq u_k(t) \leq 20, \ -3.5 \leq y_k(t) \leq 5.5 \tag{4.72}$$

The overshoot is reduced by applying output constraint as shown in

Fig. 4.4.

4.4.2 Example 2 (Linear SISO System)

This example shows that different output trajectories are created by the controller according to different weighting factor $\mathbf{R}$. The proposed algorithm is illustrated by the following plant transfer function.

$$y_p(s) = \frac{2.5}{(4s+1)(2s+1)}u(s) \tag{4.73}$$

The controller is designed based on the following model transfer function.

$$y_m(s) = \frac{1.5}{(5s+1)(3s+1)}u(s) \tag{4.74}$$

Terminal time is 100 with the sampling interval of 1. We used the following parameters for designing the controller.

$$\begin{aligned} p_0 = 100,\ m_0 = 50 \\ \tilde{\mathbf{Q}} = I,\ \mathbf{S} = \mathbf{R} = 0,\ \mathbf{P} = 0.05I,\ \mathbf{E} = 0 \end{aligned} \tag{4.75}$$

The reference time instants are 20, 40, 60, 70, 80, 100. The vector of reference values is as follows.

$$\tilde{\mathbf{r}} = \begin{bmatrix} 0 & 5 & -3 & -3 & 3 & 0 \end{bmatrix}^T \tag{4.76}$$

Kalman gain and iterative learning observer gain were calculated in the same manner as example 1. Fig. 4.5 shows the result with $\mathbf{R} = 0$ at 1st, 2nd and 30th batch.

In some case, the input trajectory may need to have a softer ap-

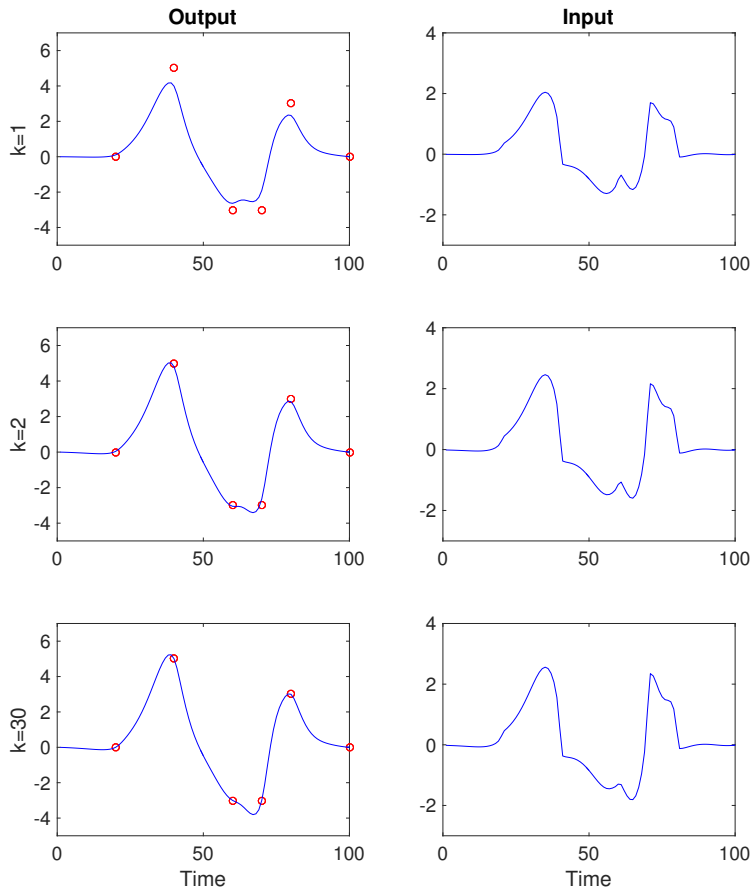


Figure 4.5: (Example 2) The performance of the proposed PTP ILMPC algorithm with $\mathbf{R} = 0$.

pearance. Thus, we use $\mathbf{R} = 0.1$ instead of $\mathbf{R} = 0$ for the next simulation. Fig. 4.6 is the second simulation with $\mathbf{R} = 0.1$ at 1st, 2nd and 30th batch. In Fig. 4.6, the output trajectory becomes a new trajectory for passing through the reference points. If conventional controller is used, a new reference trajectory through the points with a smooth

input trajectory should be calculated.

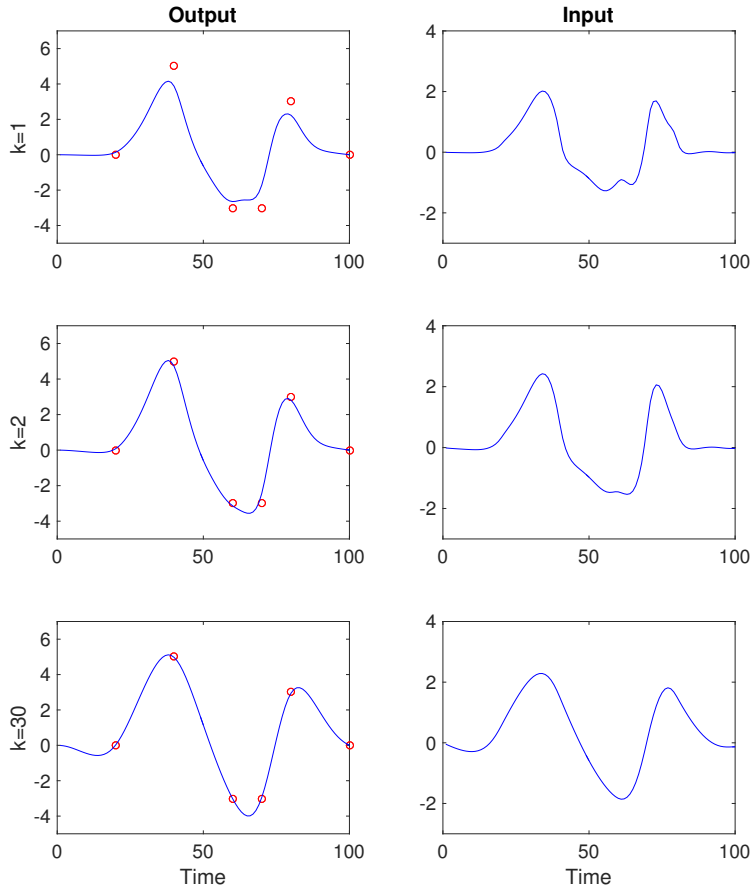


Figure 4.6: (Example 2) The performance of the proposed PTP ILMPC algorithm with $\mathbf{R} = 0.1$.

4.4.3 Example 3 (Comparison between the Proposed PTP ILMPC and PTP ILC)

In this section, we compare the proposed PTP ILMPC method with the existing PTP ILC method [16]. The plant, model and parameters except for $\mathbf{R}$ are identical to those in Example 1. To compare the two techniques, we set $\mathbf{R} = 0$ and the reason is that the PTP ILC method cannot tune a weighting factor $\mathbf{R}$. Fig. (4.7) shows the input and output trajectories of the two techniques in the first and second iterations. The PTP ILMPC method converges faster because of the real-time feedback. Fig. (4.8) shows the results under the disturbance which occurs at the 5th iteration. The PTP ILC method does not respond to the disturbance because it does not have real-time feedback function. At the 6th iteration, PTP ILC learns to reject the disturbance which occurred at the 5th iteration. However, the output trajectory is farther from the reference points because there is no disturbance at the 6th iteration. That is, PTP ILC tried to reject the disturbance which did not exist. At the 5th iteration, the proposed PTP ILMPC successfully rejects the disturbance. At the 6th iteration, PTP ILMPC also learns to reject the disturbance which does not exist. However, the output trajectory quickly converges to the reference points by real-time feedback. In Fig. (4.9), the proposed PTP ILMPC converges faster and is robust to the disturbance. Finally, we compare the performance under output constraints. The PTP ILC method cannot use output constraints. Thus, the output constraints technique we used was applied to the PTP ILC and then compared two techniques. The input trajectory which satisfies the output constraints depends entirely on the accuracy of the model because the PTP ILC method

is an open-loop control. The proposed PTP ILMPC method calculates the input value satisfying the output constraints in real time and shows excellent performance. The proposed PTP ILMPC method is superior to PTP ILC in output constraint, convergence rate and disturbance rejection performance. However, because PTP ILMPC needs to perform optimization (quadratic programming) every step, it takes longer calculation time than PTP ILC. If the sampling interval is sufficient to solve the QP problem, the proposed method shows better performance.

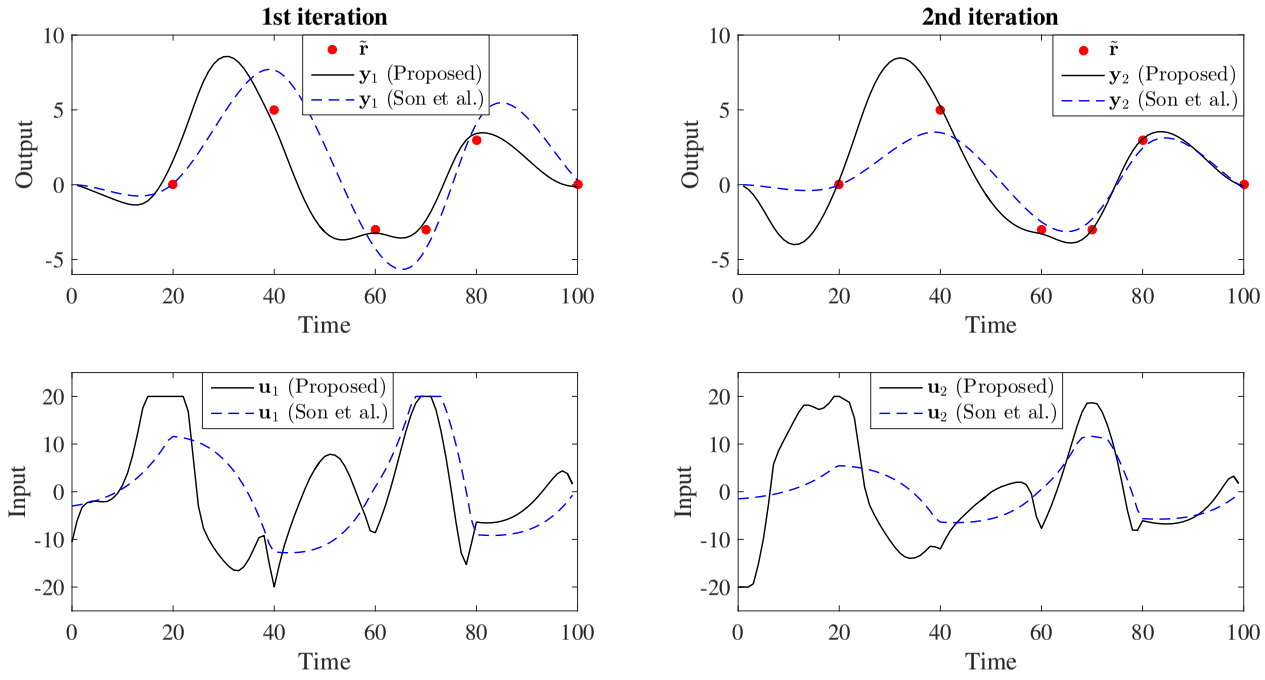


Figure 4.7: (Example 3) The performance of the proposed PTP ILMPC and the PTP ILC at the 1st and the 2nd iteration.

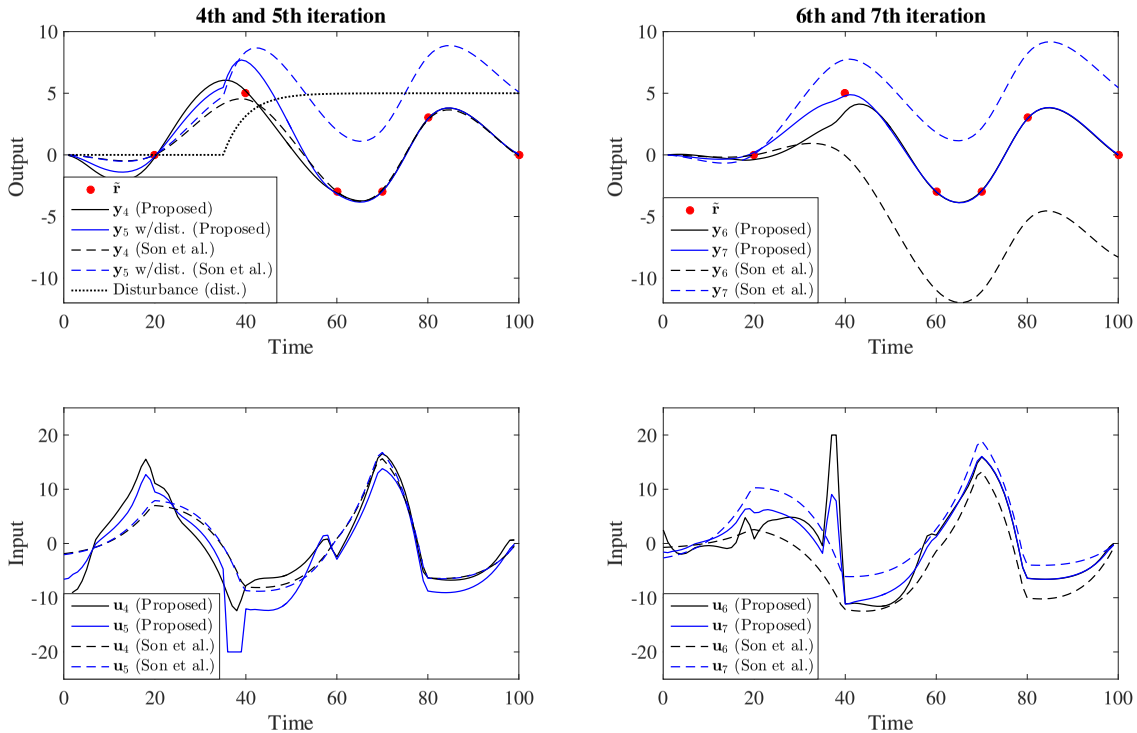


Figure 4.8: (Example 3) The performance of the proposed PTP ILMPC and the PTP ILC under the disturbance from the 4th to the 7th iteration.

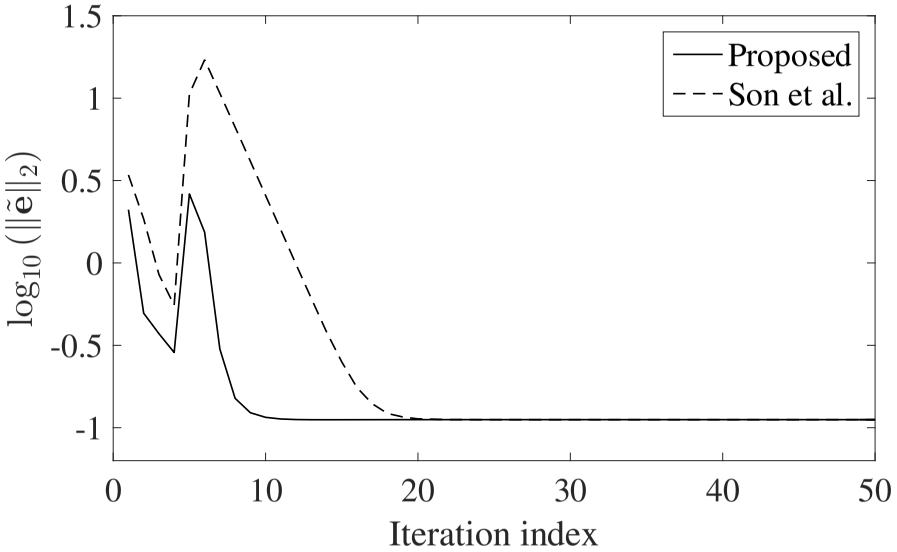


Figure 4.9: (Example 3) Log scale convergence performance of the proposed PTP ILMPC and the PTP ILC

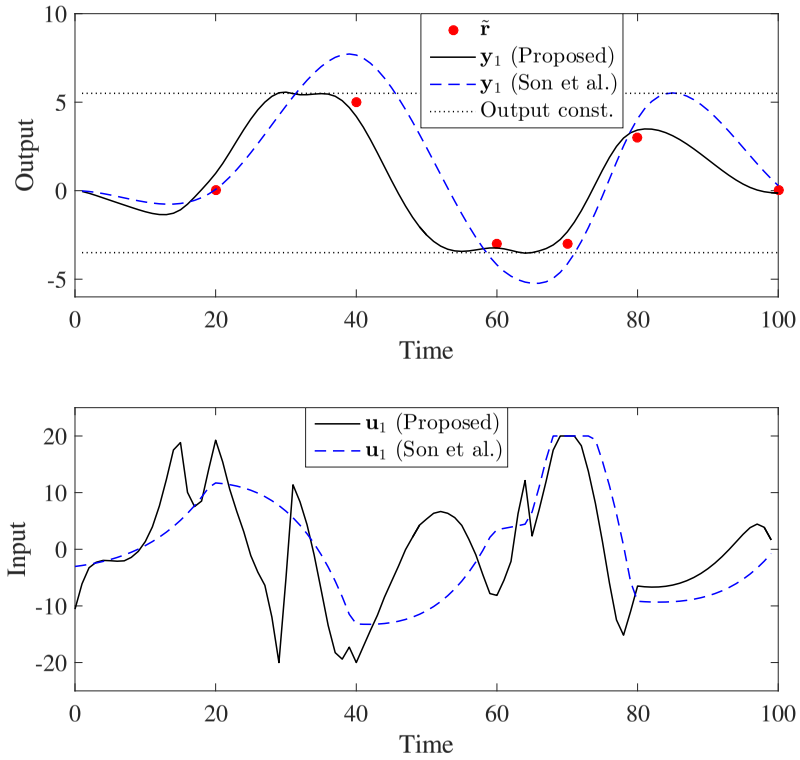


Figure 4.10: (Example 3) The performance of the proposed PTP ILMPC and the PTP ILC under output constraints

4.4.4 Example 4 (Nonlinear Semi-Batch Reactor)

We consider a jacketed semi-batch reactor where exothermic series-parallel first-order reactions occur [91].

$$\begin{aligned} A + B &\xrightarrow{k_1} C, \quad r_1 = k_1 C_A C_B \\ B + C &\xrightarrow{k_2} D, \quad r_2 = k_2 C_B C_C \end{aligned} \tag{4.77}$$

The reactor system is

$$\begin{aligned} \frac{dT}{dt} =& \frac{Q_{feed}}{V}(T_{feed} - T) - \frac{UA}{V\rho C_p}(T - T_j) - \\ & \frac{\Delta H_1}{\rho C_p} k_{10} e^{-\frac{E_1}{RT}} C_A C_B - \frac{\Delta H_2}{\rho C_p} k_{20} e^{-\frac{E_2}{RT}} C_B C_C \\ \frac{dC_A}{dt} =& -\frac{Q_{feed}}{V} C_A - k_{10} e^{-\frac{E_1}{RT}} C_A C_B \\ \frac{dC_B}{dt} =& \frac{Q_{feed}}{V}(C_{B,feed} - C_B) - k_{10} e^{-\frac{E_1}{RT}} C_A C_B - \\ & k_{20} e^{-\frac{E_2}{RT}} C_B C_C \\ \frac{dC_C}{dt} =& -\frac{Q_{feed}}{V} C_C + k_{10} e^{-\frac{E_1}{RT}} C_A C_B - k_{20} e^{-\frac{E_2}{RT}} C_B C_C \\ \frac{dV}{dt} =& Q_{feed}, \quad Q_{feed}(t) = \begin{cases} 0, \text{ if } t < 31 \text{ min} \\ Q_{feed}(t), \text{ if } t \geq 31 \end{cases} \end{aligned} \tag{4.78}$$

with the initial conditions of $T(0) = 298$ K, $C_A(0) = 1$ mol/L, $C_B = C_C = 0$ mol/L, and $V(0) = 50$ L. The parameters are specified as follows: $T_{feed} = 308$ K, $C_{B,feed} = 0.9$ mol/L, $UA/(\rho C_p) = 3.75$L/min, $k_{10} = 5.0969 \times 10^{16}$ L/mol $\cdot$ min, $k_{20} = 2.2391 \times 10^{17}$ L/mol $\cdot$ min, $E_1/R = 12305$ K, $E_2/R = 13450$ K, $\Delta H_1/(\rho C_p) = -28.5$ K $\cdot$ L/mol, and $\Delta H_2/(\rho C_p) = -20.5$ K $\cdot$ L/mol.

The manipulated variables are the feed flow rate of the reactant B ($Q_{feed}(t)$) and the jacket temperature ($T_j(t)$). The controlled variables are the reactor temperature ($T(t)$) and the yield of the desired product ($V(t)C_C(t)$). The first control objective is to maintain the reactor temperature at 308.15 K during the reaction period ($t = 30 \sim 80$ min) and to terminate the reactor operation at 303.15 K. The second control objective is to achieve the yield of 42 mol for the desired product at t =100 min. Terminal time is 100 min with the sampling interval of 1 min.

We used the following linear model to control the above system where the four states and the second output are nonlinear.

$$\begin{aligned} x_k(t+1) &= \begin{bmatrix} 2.367 & -1.300 & -0.002 & 0.088 & 0.025 \\ 1.631 & -0.548 & -0.003 & 0.076 & 0.027 \\ 0.003 & -0.003 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} x_k(t) \\ &+ \begin{bmatrix} 0.088 & 0.025 \\ 0.076 & 0.027 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \delta u_k(t) \\ y_k(t) &= \begin{bmatrix} 0.680 & 0.734 & 0.001 & 0 & 0 \\ -49.855 & 46.189 & 0.911 & 0 & 0 \end{bmatrix} x_k(t) \end{aligned} \tag{4.79}$$

A linearized model was obtained at $x_s = \begin{bmatrix} 298.15 & 1 & 0 & 1 & 100 \end{bmatrix}^T$ and $u_s = \begin{bmatrix} 0 & 298.15 \end{bmatrix}^T$. Then, two states were removed by minimal realization. Finally, the above state-space model was obtained using

the state-space augmentation. The following input and output constraints are applied to the controller.

$$\begin{aligned} 0.5 \leq & Q_{feed}(t) \leq 1.5 \text{ L/min} \\ & |\delta Q_{feed}(t)| \leq 0.5 \text{ L/min} \\ 293.15 \leq & T_j(t) \leq 318.15 \text{ K} \\ & |\delta T_j(t)| \leq 3 \text{ K} \\ & T(t) \leq 311.15 \text{ K} \end{aligned} \tag{4.80}$$

We used the following parameters for designing the controller: $p_0 = 100,\ m_0 = 40,\ \tilde{\mathbf{Q}} = I,\ \mathbf{S} = 0,\ \mathbf{R} = 0.01,\ \mathbf{P} = 0.02$, and $\mathbf{E} = 10^5 I$. Kalman gain was calculated using the state noise covariance matrix **diag**(0.1, 0.1, 0.1, 0.01, 0.01) and the measurement noise covariance matrix **diag**(0.001, 0.001). Iterative learning observer gain L was calculated by $C^T(CC^T)^{-1}$ in the same manner as example 1.

In Fig. 4.11, the reactor temperature shows large overshoot when the output constraint is not applied. Thus, we used the output constraint to reduce the large overshoot; however, the reactor temperature reaches 311.9 K although the upper limit of the constraint is 311.15 K. The reason is that the output constraint is not hard constraint. If there is no feasible solution which satisfies both input and output constraints, the output constraint is softened because of the slack variable. The second reason is plant-model mismatch. The input trajectories satisfying the output constraint are calculated using the model. Notwithstanding the reasons, the output constraint is effective to suppress large overshoot. Fig. 4.12 shows the results of the 1st, 3rd and 50th iteration. The yields at terminal times of 1st, 3rd and 50th iterations are 38.366, 41.107 and 42.000, respectively. The reac-

tor temperature is also successfully converged to the reference points as shown in Fig. 4.12.

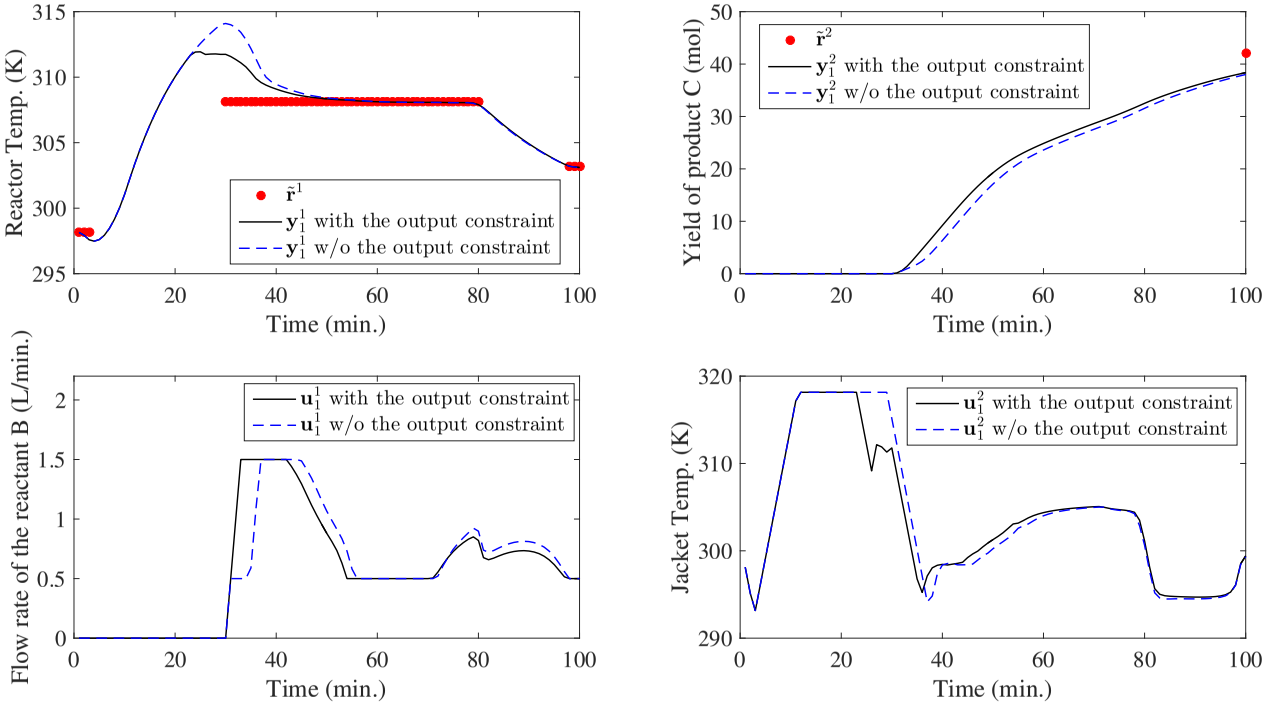


Figure 4.11: (Example 4) Comparison between the results with output constraint and the results without output constraint.

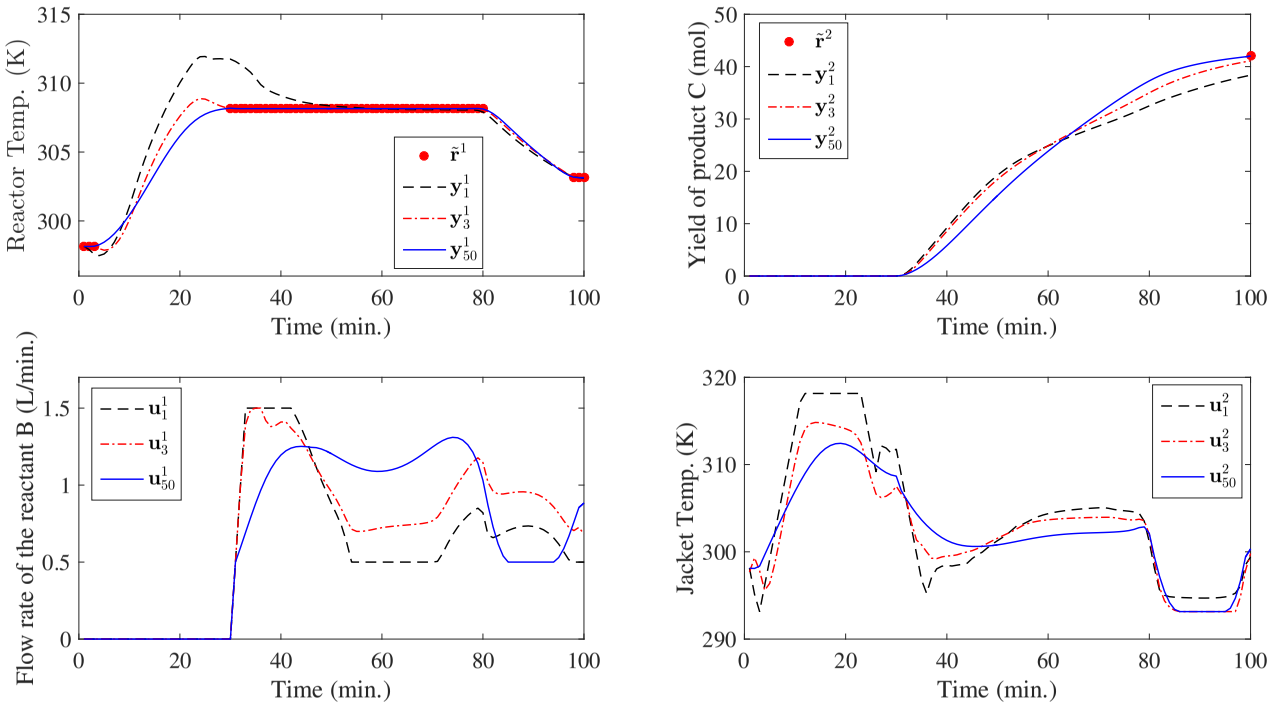


Figure 4.12: (Example 4) The performance of the proposed PTP ILMPC algorithm at 1st, 3rd and 50th iteration.

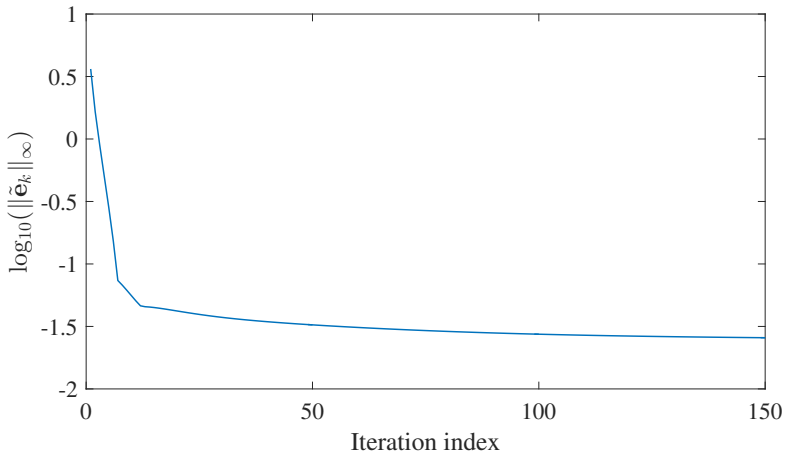


Figure 4.13: (Example 4) Log scale convergence performance for the nonlinear MIMO system.

4.5 Conclusion

This paper has proposed the novel control technique combining ILC, MPC and PTP tracking problem, called PTP ILMPC. This algorithm can track the reference points without generating reference trajectory while learning by using the information of previous iteration. Furthermore, the proposed algorithm can reject real-time disturbance because it is combined with MPC controller. In this paper, we have provided the algorithm for a linear time-invariant system. However, the proposed algorithm for a time-varying system can be derived by the same procedure. Two examples, linear SISO system and nonlinear MIMO system, are provided to show the effectiveness of the PTP ILMPC. Future work will consider time-delayed multi-rate sampled data system.

Chapter 5

Stochastic Iterative Learning Control for Batch-varying Reference Trajectory

In this chapter, we present adaptive ILC schemes for discrete LTI stochastic system with BVRT. In this case, if the state noise and measurement noise exist,convergence rate and tracking performance are degraded because the controller considers the difference arising from the noise as tracking error. To deal with such a problem, we propose two approaches. The first is based on a batch-domain Kalman filter, which uses the difference between the current output trajectory and the next reference trajectory as a state vector, while the second is based on a time-domain Kalman filter. In the second approach, the system is identified at the end of each batch in an iterative fashion using the observer/Kalman filter identification (OKID). Then, the stochastic problem is handled using Kalman filter with a steady-state Kalman gain obtained from the identification. Therefore, the second approach can track the reference trajectories of discrete LTI stochastic system using only the input–output information. Simulation examples are provided to show the effectiveness of the proposed schemes.

5.1 Introduction

Iterative learning control (ILC) is an effective control scheme in handling a system repeating the same task on a finite interval. Iterative learning controller controls a system in batch or iteration domain, while general controller, PID, LQR or MPC, controls a system in time domain. In the ILC, the input values for the entire time of the next batch operation are computed using input and output values of the current batch. ILC was first introduced for robot manipulators; in addition, it has been implemented in many industrial processes such as semiconductor manufacturing and chemical processes [3, 49, 13, 12, 62, 52]. Most of the ILC schemes focus on tracking batch-invariant reference trajectory. Recently, several ILC schemes have been studied for tracking batch-varying references [92, 93, 94], and they use a recursive least squares algorithm to update the parameters iteratively along the batch index. Our previous work [95] also handles a system with batch-varying references using lifted system framework and iterative learning identification. However, these studies present methods for deterministic system only.

In this paper, we present adaptive ILC schemes for discrete linear time-invariant (LTI) stochastic system with batch-varying reference trajectories (BVRT). In batch processes (polymerization reactor or rapid thermal process), reference trajectory can be changed in case feed conditions, start up speed or shut down speed needs to be varied. New reference trajectory can be calculated from off-line optimization. In addition, products with various specifications can be produced from the same system. For example, one etching system in semiconductor manufacturing can produce wafers with vari-

ous critical dimensions if the system can track BVRT. If the system has BVRT, convergence property of ILC differs from traditional ILC which aims at tracking an identical reference trajectory [95]. In this case, we should identify precise Markov parameters of system dynamics. Hence, we introduce iterative learning identification to satisfy convergence condition. In case of stochastic system, the presence of noises decreases the convergence rate and performance. This is because the controller considers noise as tracking error. To handle these issues, we propose two Kalman filter-based approaches. In case of batch-to-batch control problem, Kalman filter can be used in either time-domain or batch-domain. We apply Kalman filter in both the domains, and then compare the rate and tracking performance of the two approaches. In the first approach, we use Kalman filter in the batch-domain. Ahn et al. [88] proposed Kalman filter-augmented iterative learning control. This method can be applied only if a system has an identical reference trajectory and a fixed learning gain matrix. Hence, we extend the method to handle BVRT and batch-varying learning gain matrix. In the second approach, system Markov parameters are identified using the observer/Kalman filter identification (OKID) [96] in an iterative learning manner. The OKID algorithm is numerically efficient and robust with respect to measurement noise if the output residual error is zero-mean and Gaussian noise [97]. It also provides steady-state Kalman gain and system Markov parameters. With the steady-state Kalman gain, we can use the general Kalman filter in the time-domain for handling stochastic issue without covariance information of state and measurement noises. Therefore, the second approach uses only input-output information. The comparative results of the two approaches are provided in Section 5.4.

The rest of this paper is organized as follows: In Section 5.2, the deterministic ILC scheme for BVRT and convergence property are presented. In Section 5.3, the two Kalman filter-based approaches are proposed for handling stochastic issue. Then, numerical illustrations are provided in Section 5.4. Section 5.5 provides concluding remarks.

5.2 ILC for Batch-Varying Reference Trajectories

5.2.1 Convergence Property for ILC with Batch-Varying Reference Trajectories

First, we consider the following linear discrete time-invariant system which operates on an interval $t \in [0, N]$:

$$\begin{aligned} x_k(t+1) &= Ax_k(t) + Bu_k(t) \\ y_k(t) &= Cx_k(t) \end{aligned} \tag{5.1}$$

where $x_k(t) \in \mathbb{R}^n$ is the state vector; $u_k(t) \in \mathbb{R}^m$ is the input vector; $y_k(t) \in \mathbb{R}^q$ is the output vector; t is the time index; k is the batch index; and the matrices A, B, and C are real matrices of appropriate dimensions and assumed to be time-invariant. Because finite time intervals $[0, N]$ are considered in ILC, this system can be rewritten as a lifted system:

$$\mathbf{y}_k = \mathbf{G}_p \mathbf{u}_k \tag{5.2}$$

with $x_k(0) = 0$ and the plant matrix $\mathbf{G}_p = \mathbb{R}^{(qN)\times(mN)}$ defined as

$$\mathbf{G}_p = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2}B & \cdots & CB \end{bmatrix} \tag{5.3}$$

and the vectors $\mathbf{y}_k \in \mathbb{R}^{qN}$, and $\mathbf{u}_k \in \mathbb{R}^{mN}$ are defined as

$$\mathbf{y}_k = \begin{bmatrix} y_k^T(1) & y_k^T(2) & \cdots & y_k^T(N) \end{bmatrix}^T \tag{5.4}$$

$$\mathbf{u}_k = \begin{bmatrix} u_k^T(0) & u_k^T(1) & \cdots & u_k^T(N-1) \end{bmatrix}^T \tag{5.5}$$

The system matrix $\mathbf{G}_p$ is a Markov matrix with a lower triangular Toeplitz structure.

The most general input update law of the conventional ILC with batch-invariant reference trajectory is represented by $\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{H}(\mathbf{r} - \mathbf{y}_k) = \mathbf{u}_k + \mathbf{H}\mathbf{e}_k$ where $\mathbf{H}$ is a learning gain matrix, and $\mathbf{r}$ is a reference trajectory. It is assumed that input trajectory for next batch is calculated when the current batch operation is finished. Thus, $\mathbf{u}_{k+1}$ is calculated using available information $\mathbf{u}_k$ and $\mathbf{y}_k$. In this case, it is well known that $\mathbf{e}_k \to 0$ as $k \to \infty$ if $\|\mathbf{I} - \mathbf{G}_p\mathbf{H}\|_\infty < 1$ where $\mathbf{I}$ is the identity matrix [98]. In the conventional ILC formulation, $\mathbf{y}_k$ converges to the same reference $\mathbf{r}$ for all batches. Hence, it is possible to make the output converge as long as we know the values of the error and the model satisfying the convergence condition. If the reference trajectories are varied in batches, we should know not only the values of the error but also the input variation necessary to move the output

from the current reference $\mathbf{r}_k$ to the next reference $\mathbf{r}_{k+1}$. The desired input of $(k+1)$-th batch can be expressed as the following form:

$$\mathbf{u}_{k+1}^d = \mathbf{u}_k + (\mathbf{u}_{k+1}^d - \mathbf{u}_k) \tag{5.6}$$

where $\mathbf{u}_{k+1}^d$ is the desired input for next reference $\mathbf{r}_{k+1}$. With the plant description of $\mathbf{y}_k = \mathbf{G}_p\mathbf{u}_k$ and $\mathbf{r}_{k+1} = \mathbf{G}_p\mathbf{u}_{k+1}^d$, Eq. (5.6) can be rewritten as:

$$\mathbf{u}_{k+1}^d = \mathbf{u}_k + \mathbf{G}_p^{-1}(\mathbf{r}_{k+1} - \mathbf{y}_k) \tag{5.7}$$

In the ILC problem, it is assumed that the plant matrix $\mathbf{G}_p$ is unknown or not invertible. Hence, we introduce batch-varying learning gain matrix to obtain input update law of the ILC for BVRT:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{H}_k(\mathbf{r}_{k+1} - \mathbf{y}_k) \tag{5.8}$$

Theorem 5.1. *Consider the linear system (5.1) and the ILC controller (5.8). The system is convergent if $\boldsymbol{H}_k$ is chosen such that $\boldsymbol{G}_p\boldsymbol{H}_k = \boldsymbol{I}$.*

Proof The error at the $(k+1)$-th batch is derived as

$$\begin{aligned}
\mathbf{r}_{k+1} - \mathbf{y}_{k+1} &= \mathbf{r}_{k+1} - \mathbf{G}_p\mathbf{u}_{k+1} \\
&= \mathbf{r}_{k+1} - \mathbf{G}_p\left[\mathbf{u}_k + \mathbf{H}_k(\mathbf{r}_{k+1} - \mathbf{y}_k)\right] \\
&= \mathbf{r}_{k+1} - \mathbf{y}_k - \mathbf{G}_p\mathbf{H}_k\mathbf{r}_{k+1} + \mathbf{G}_p\mathbf{H}_k\mathbf{y}_k
\end{aligned} \tag{5.9}$$

then, adding $((\mathbf{r}_k - \mathbf{r}_k) + (\mathbf{G}_p\mathbf{H}_k\mathbf{r}_k - \mathbf{G}_p\mathbf{H}_k\mathbf{r}_k))$ to Eq. (5.9) yields

$$\begin{aligned}\mathbf{e}_{k+1} &= \mathbf{e}_k - \mathbf{G}_p\mathbf{H}_k\mathbf{e}_k + \Delta\mathbf{r}_{k+1} - \mathbf{G}_p\mathbf{H}_k\Delta\mathbf{r}_{k+1} \\ &= (\mathbf{I} - \mathbf{G}_p\mathbf{H}_k)\mathbf{e}_k + (\mathbf{I} - \mathbf{G}_p\mathbf{H}_k)\Delta\mathbf{r}_{k+1} \\ &= (\mathbf{I} - \mathbf{G}_p\mathbf{H}_k)^k\mathbf{e}_1 + \sum_{j=1}^{k}(\mathbf{I} - \mathbf{G}_p\mathbf{H}_k)^j\Delta\mathbf{r}_{k+2-j}\end{aligned} \tag{5.10}$$

where $\Delta\mathbf{r}_{k+1} = \mathbf{r}_{k+1} - \mathbf{r}_k$. In this case, the system cannot be convergent under traditional ILC convergence property, i.e., $\|\mathbf{I}-\mathbf{G}_p\mathbf{H}_k\|_\infty < 1$, because of the accumulated $\Delta\mathbf{r}_k$ on the error, and $\Delta\mathbf{r}_{k+1}$ cannot be 0 since the reference trajectories vary for all batches. Therefore, the second term should be zero for $\mathbf{e}_k \to 0$. For this, the learning gain matrix $\mathbf{H}_k$ should be chosen such that $\mathbf{G}_p\mathbf{H}_k = \mathbf{I}$ leading to $(\mathbf{I} - \mathbf{G}_p\mathbf{H}_k)\Delta\mathbf{r}_{k+1} = 0$. ■

5.2.2 Iterative Learning Identification

To find the learning gain matrix $\mathbf{H}_k$ such that $\mathbf{G}_p\mathbf{H}_k = \mathbf{I}$, we should find the precise system Markov parameter matrix $\mathbf{G}_p$. We can represent the input-output description in the following matrix form:

$$\mathbf{Y}_k = \mathbf{g}_p\mathbf{U}_k \tag{5.11}$$

where

$$\mathbf{Y}_k = \begin{bmatrix} y_k(1) & y_k(2) & \cdots & y_k(N) \end{bmatrix} \tag{5.12}$$

$$\mathbf{g}_p = \begin{bmatrix} CB & CAB & \cdots & CA^{N-1}B \end{bmatrix} \tag{5.13}$$

and

$$\mathbf{U}_k = \begin{bmatrix} u_k(0) & u_k(1) & u_k(2) & \cdots & u_k(N-1) \\ & u_k(0) & u_k(1) & \cdots & u_k(N-2) \\ & & u_k(0) & \cdots & u_k(N-3) \\ & & & \ddots & \vdots \\ & & & & u_k(0) \end{bmatrix} \tag{5.14}$$

If we find $\mathbf{g}_p$, we can reconstruct the system Markov parameter matrix $\mathbf{G}_p$. To compute $\mathbf{g}_p$, we can use the following system Markov parameters update law similar to the input update law in Eq. (5.8).

$$\mathbf{g}_k = \mathbf{g}_{k-1} + (\mathbf{Y}_k - \hat{\mathbf{Y}}_k)\mathbf{H}_k^M \tag{5.15}$$

where $\hat{\mathbf{Y}}_k = \mathbf{g}_{k-1}\mathbf{U}_k$ and $\mathbf{H}_k^M$ is the learning gain matrix of the system Markov parameters.

Theorem 5.2. *The estimated system Markov parameters $\boldsymbol{g}_k$ is convergent to the real system Markov parameters $\boldsymbol{g}_p$, if $\boldsymbol{H}_k^M$ is chosen such that $\|\boldsymbol{I} - \boldsymbol{U}_k\boldsymbol{H}_k^M\| < 1$.*

Proof The model error can be written in the following form:

$$\mathbf{Y}_{k+1} - \hat{\mathbf{Y}}_{k+1} = \mathbf{g}_p\mathbf{U}_{k+1} - \mathbf{g}_k\mathbf{U}_{k+1} \tag{5.16}$$

Substitution of Eq. (5.15) into Eq. (5.16) yields

$$\mathbf{Y}_{k+1} - \hat{\mathbf{Y}}_{k+1} = \mathbf{g}_p\mathbf{U}_{k+1} - \left[\mathbf{g}_{k-1} + (\mathbf{Y}_k - \hat{\mathbf{Y}}_k)\mathbf{H}_k^M\right]\mathbf{U}_{k+1} \tag{5.17}$$

Using $\mathbf{Y}_{k+1} = \mathbf{g}_p\mathbf{U}_{k+1}$ and $\hat{\mathbf{Y}}_{k+1} = \mathbf{g}_k\mathbf{U}_{k+1}$, the following can be derived:

$$(\mathbf{g}_p - \mathbf{g}_k)\mathbf{U}_{k+1} = (\mathbf{g}_p - \mathbf{g}_{k-1} - \mathbf{g}_p\mathbf{U}_k\mathbf{H}_k^M + \mathbf{g}_{k-1}\mathbf{U}_k\mathbf{H}_k^M)\mathbf{U}_{k+1} \quad (5.18)$$

This equation can be rearranged to

$$(\mathbf{g}_p - \mathbf{g}_k)\mathbf{U}_{k+1} = (\mathbf{g}_p - \mathbf{g}_{k-1})\left(\mathbf{I} - \mathbf{U}_k\mathbf{H}_k^M\right)\mathbf{U}_{k+1} \quad (5.19)$$

Then, it leads to the inequality

$$\|\mathbf{g}_p - \mathbf{g}_k\|\|\mathbf{U}_{k+1}\| \leq \|\mathbf{g}_p - \mathbf{g}_{k-1}\|\left\|\mathbf{I} - \mathbf{U}_k\mathbf{H}_k^M\right\|\|\mathbf{U}_{k+1}\| \quad (5.20)$$

Therefore, $\|\mathbf{g}_p - \mathbf{g}_k\| \rightarrow 0$ as $k \rightarrow \infty$ if $\mathbf{H}_k^M$ is chosen such that $\left\|\mathbf{I} - \mathbf{U}_k\mathbf{H}_k^M\right\| < 1$. ■

In this case, we can compute $\mathbf{H}_k^M$ using $\mathbf{U}_k$ such that $\mathbf{U}_k\mathbf{H}_k^M = \mathbf{I}$.

Lemma 5.1. *[99] Let $\boldsymbol{U}$ is an $m \times n$ matrix. Then, $rank(\boldsymbol{U}^T\boldsymbol{U}) = rank(\boldsymbol{U})$.*

Corollary 5.1. *If $\boldsymbol{U}$ is an $m \times n$ matrix such that $rank(\boldsymbol{U}) = n$, then $\boldsymbol{U}^T\boldsymbol{U}$ is invertible. Therefore, there exists a least-squares solution if $\boldsymbol{U}$ has a full column rank.*

The upper triangular matrix $\mathbf{U}_k$ with $u_k(0) \neq 0$ has always a full column rank; however, initial input $u_k(0)$ can be zero or very small value. In this case, we cannot obtain least-squares solution or numerical problem can occur. Hence, we compute $\mathbf{H}_k^M$ using the pseudo inverse of $\mathbf{U}_k$, $\mathbf{U}_k^\dagger$, based on the singular value decomposition (SVD). Therefore, we use the pseudo inverse of $\mathbf{U}_k$ as the learning gain matrix

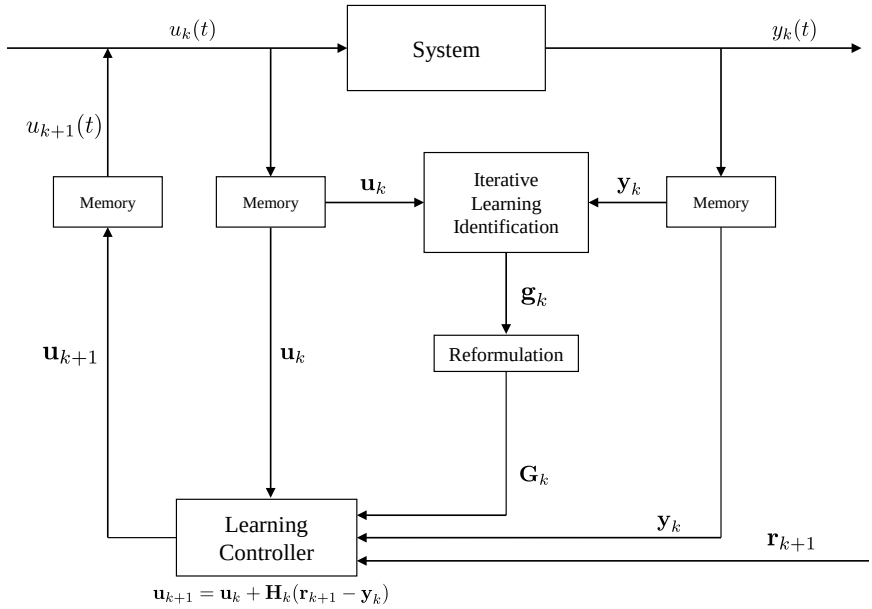


Figure 5.1: The scheme of the deterministic ILC for batch-varying reference trajectories.

$\mathbf{H}_k^M$ of the system Markov parameters. Finally, the following system Markov parameters update law is obtained.

$$\mathbf{g}_{k+1} = \mathbf{g}_k + \left(\mathbf{Y}_k - \hat{\mathbf{Y}}_k\right) \mathbf{U}_k^{\dagger} \tag{5.21}$$

5.2.3 Deterministic ILC Controller for Batch-Varying Reference Trajectories

We can design the ILC controller by the following objective function.

$$\min_{\Delta \mathbf{u}_{k+1}} J = \frac{1}{2} \left\{ \mathbf{e}_{k+1}^T \mathbf{e}_{k+1} \right\} \tag{5.22}$$

First, we need to derive $\mathbf{e}_{k+1}$ to use the objective function. The input-output relationship between two adjacent batches is

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{G}_p \Delta \mathbf{u}_{k+1} \tag{5.23}$$

then, adding $(r_{k+1} - r_k)$ to Eq. (5.23), the following error dynamics can be obtained.

$$\mathbf{e}_{k+1} = \mathbf{e}_k - \mathbf{G}_p \Delta \mathbf{u}_{k+1} + \mathbf{r}_{k+1} - \mathbf{r}_k \tag{5.24}$$

Substituting Eq. (5.24) for $\mathbf{e}_{k+1}$ in Eq. (5.22) with $\partial J / \partial \Delta \mathbf{u}_{k+1} = 0$ yields

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \left(\mathbf{G}_p^T \mathbf{G}_p\right)^{-1} \mathbf{G}_p^T \left(\mathbf{r}_{k+1} - \mathbf{y}_k\right) \tag{5.25}$$

Because the precise plant $\mathbf{G}_p$ is unknown, we use $\mathbf{G}_k$ instead of $\mathbf{G}_p$ because $\mathbf{G}_k \to \mathbf{G}_p$ if $k \to \infty$ from Theorem 5.2 where $\mathbf{G}_k$ is reconstructed using $\mathbf{g}_k$ (5.21). Then, we have the following input update law of the deterministic ILC for BVRT.

$$\begin{aligned} \mathbf{u}_{k+1} &= \mathbf{u}_k + \left(\mathbf{G}_k^T \mathbf{G}_k\right)^{-1} \mathbf{G}_k^T \left(\mathbf{r}_{k+1} - \mathbf{y}_k\right) \\ &= \mathbf{u}_k + \mathbf{H}_k \left(\mathbf{r}_{k+1} - \mathbf{y}_k\right) \end{aligned} \tag{5.26}$$

Otherwise, the learning gain matrix $\mathbf{H}_k$ can be obtained from the convergence property $\mathbf{G}_p \mathbf{H}_k = \mathbf{I}$ in Theorem 5.1. Using the least squares solution, we can obtain the learning gain matrix $\mathbf{H}_k = (\mathbf{G}_p^T \mathbf{G}_p)^{-1} \mathbf{G}_p$ directly. Then, $\mathbf{G}_p$ can be replaced by $\mathbf{G}_k$ from Theorem 5.2. The procedure of the deterministic ILC for BVRT is illustrated in Fig. 5.1.

5.2.3.1 Quadratic criterion-based ILC for batch-varying reference trajectories

In many ILC designs, the following quadratic objective function involving both regulation error and input change is applied [5].

$$\min_{\Delta \mathbf{u}_{k+1}} J = \frac{1}{2} \left\{ \mathbf{e}_{k+1}^T \mathbf{Q} \mathbf{e}_{k+1} + \Delta \mathbf{u}_{k+1}^T \mathbf{R} \Delta \mathbf{u}_{k+1} \right\} \tag{5.27}$$

By solving the quadratic objective function, we can obtain the following quadratic-criterion-based ILC (Q-ILC) input update law.

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \left(\mathbf{G}_k^T \mathbf{Q} \mathbf{G}_k + \mathbf{R} \right)^{-1} \mathbf{G}_k^T \mathbf{Q} \left(\mathbf{r}_{k+1} - \mathbf{y}_k \right) \tag{5.28}$$

Unlike typical ILC for batch-invariant reference trajectories, in the ILC for BVRT, input signal does not converge to a specific signal because the reference trajectories are changed along the batch index. Therefore, input penalty term on Q-ILC obstruct the convergence. That is, larger input weighting factor $\mathbf{R}$ shows worse convergence performance. According to Theorem 5.1 and Eq. (5.28), convergence property of the Q-ILC controller takes the following form and we can assume that convergence of the Markov parameter matrix $\mathbf{G}_k$ is complete if $k \to \infty$.

$$\mathbf{I} - \mathbf{G} \left(\mathbf{G}^T \mathbf{Q} \mathbf{G} + \mathbf{R} \right)^{-1} \mathbf{G}^T \mathbf{Q} = \mathbf{0} \tag{5.29}$$

where $\mathbf{0}$ denotes the zero matrix. To satisfy the above equation, all the eigenvalues of the left-hand side should be zero. The left-hand side can be expressed as the following form using matrix inversion lemma

[100].

$$\left(\mathbf{I} + \mathbf{G}\mathbf{R}^{-1}\mathbf{G}^{T}\mathbf{Q}\right)^{-1} = \mathbf{0} \tag{5.30}$$

In this case, the eigenvalues of the left-hand side cannot be zero.

$$\left|\lambda_i\left(\left(\mathbf{I} + \mathbf{G}\mathbf{R}^{-1}\mathbf{G}^{T}\mathbf{Q}\right)^{-1}\right)\right| = \left|\frac{1}{1 + \lambda_i\left(\mathbf{G}\mathbf{R}^{-1}\mathbf{G}^{T}\mathbf{Q}\right)}\right| \neq 0, \forall i \tag{5.31}$$

Therefore, convergence performance of the Q-ILC with BVRT decreases as the input weighting factor $\mathbf{R}$ increases. Comparison of convergence performance according to the size of $\mathbf{R}$ is illustrated in Fig. 5.8 of Section 5.4.

5.3 ILC for LTI Stochastic System with Batch-Varying Reference Trajectories

We consider the following linear discrete time-invariant stochastic system operating on an interval $t \in [0, N]$:

$$\begin{aligned} x_k(t+1) &= Ax_k(t) + Bu_k(t) + \Gamma w(t) \\ y_k(t) &= Cx_k(t) + v(t) \end{aligned} \tag{5.32}$$

where $w_k(t)$ is the state noise; $v_k(t)$ is the measurement noise and Γ is the state noise matrix. In the ILC controller for stochastic system, noises are included in the input update equation (5.26). Thus, convergence rate and performance are reduced since the controller considers the noise as tracking error.

In this paper, two Kalman filter-based approaches are proposed to deal with the stochastic issue. The first approach is based on the

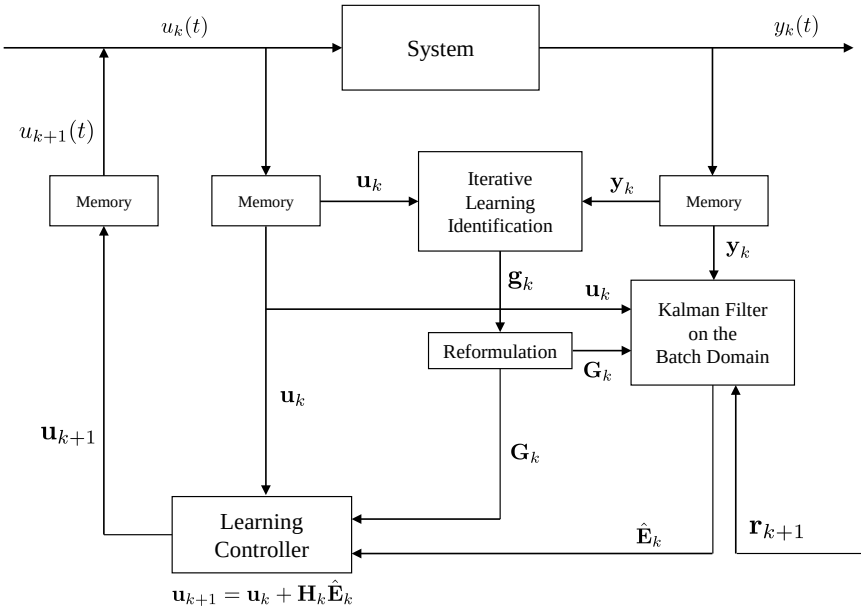


Figure 5.2: The scheme of the batch-domain Kalman filter-based stochastic ILC for batch-varying reference trajectories.

Kalman filtering on the batch-domain. The second one is based on the Kalman filtering on the time-domain using the steady-state Kalman gain obtained from the observer/Kalman filter identification (OKID).

5.3.1 Approach 1: Batch-Domain Kalman Filter-Based Approach

Suppose that there exist state noise, $\mathbf{w}_k \sim N(0, Q)$, and measurement noise, $\mathbf{v}_k \sim N(0, R)$, in input update law and input-output relationship, respectively [88]. By defining $\mathbf{E}_k = \mathbf{r}_{k+1} - \mathbf{y}_k$, we can obtain the following relationship:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{H}_k\mathbf{E}_k + \mathbf{w}_k \tag{5.33}$$

$$\tilde{\mathbf{y}}_k = \mathbf{y}_k + \mathbf{v}_k = \mathbf{G}_{k-1}\mathbf{u}_k + \mathbf{v}_k \tag{5.34}$$

$$\mathbf{E}_{k+1} = \mathbf{E}_k + \mathbf{r}_{k+2} - \mathbf{r}_{k+1} - \mathbf{G}_k\mathbf{u}_{k+1} + \mathbf{G}_{k-1}\mathbf{u}_k \tag{5.35}$$

$$\tilde{\mathbf{E}}_k = \mathbf{r}_{k+1} - \tilde{\mathbf{y}}_k = \mathbf{E}_k - \mathbf{v}_k \tag{5.36}$$

where $\mathbf{y}_k$ is a true output and $\tilde{\mathbf{y}}_k$ is a measured output. If the measured output is applied to the input update equation directly, the equation is affected by the state and measurement noises as follows.

$$\begin{aligned}\mathbf{u}_{k+1} =& \mathbf{u}_k + \mathbf{H}_k\tilde{\mathbf{E}}_k + \mathbf{w}_k = \mathbf{u}_k + \mathbf{H}_k\mathbf{E}_k - \mathbf{H}_k\mathbf{v}_k + \mathbf{w}_k \\ \mathbf{u}_{k+2} =& \mathbf{u}_k + \mathbf{H}_k\mathbf{E}_k - \mathbf{H}_k\mathbf{v}_k + \mathbf{w}_k + \mathbf{H}_{k+1}\mathbf{E}_{k+1} - \mathbf{H}_{k+1}\mathbf{v}_{k+1} + \mathbf{w}_{k+1}\end{aligned} \tag{5.37}$$

Input signal should be updated using $\mathbf{u}$ and $\mathbf{HE}$, not $\mathbf{Hv}$ and $\mathbf{w}$. Therefore, the input signal should be updated using filtered output $\mathbf{y}_k$ or filtered difference $\mathbf{E}_k$. Substitution of Eq. (5.33) into Eq. (5.35) yields

$$\mathbf{E}_{k+1} = (\mathbf{I} - \mathbf{G}_k\mathbf{H}_k)\mathbf{E}_k + \mathbf{r}_{k+2} - \mathbf{r}_{k+1} - \mathbf{G}_k\mathbf{u}_k + \mathbf{G}_{k-1}\mathbf{u}_k - \mathbf{G}_k\mathbf{w}_k \tag{5.38}$$

Because $\mathbf{G}_k\mathbf{H}_k$ can be assumed to be the identity matrix and $\mathbf{G}_{k-1}\mathbf{u}_k$ goes to $\mathbf{y}_k$ as $k \to \infty$ from the iterative learning identification, we can obtain the following state-space model in the batch-domain.

$$\begin{aligned}\mathbf{E}_{k+1} &= -\mathbf{E}_k + \begin{bmatrix}\mathbf{I} & -\mathbf{G}_k\end{bmatrix}\begin{bmatrix}\mathbf{r}_{k+2} \\ \mathbf{u}_k\end{bmatrix} - \mathbf{G}_k\mathbf{w}_k \\ \tilde{\mathbf{E}}_k &= \mathbf{E}_k - \mathbf{v}_k\end{aligned} \tag{5.39}$$

The Kalman filter equations are presented as follows.

- Batch update (Prediction)

$$\begin{aligned}\hat{\mathbf{E}}_{k+1}^{-} &= -\hat{\mathbf{E}}_k + \begin{bmatrix}\mathbf{I} & -\mathbf{G}_k\end{bmatrix}\begin{bmatrix}\mathbf{r}_{k+2}\\ \mathbf{u}_k\end{bmatrix} \\ \mathbf{P}_{k+1}^{-} &= \mathbf{P}_k + \mathbf{G}_k Q \mathbf{G}_k^T\end{aligned} \tag{5.40}$$

- Measurement update (Correction)

$$\begin{aligned}\mathbf{K}_k &= \mathbf{P}_k^{-}(\mathbf{P}_k^{-} + R)^{-1} \\ \hat{\mathbf{E}}_k &= \hat{\mathbf{E}}_k^{-} + \mathbf{K}_k(\tilde{\mathbf{E}}_k - \hat{\mathbf{E}}_k^{-}) \\ \mathbf{P}_k &= (\mathbf{I} - \mathbf{K}_k)\mathbf{P}_k^{-}\end{aligned} \tag{5.41}$$

where $\mathbf{P}_k$ and $\mathbf{K}_k$ are the error covariance matrix and Kalman gain, respectively. $\mathbf{G}_k$ in the Kalman filter equation is the same form as $\mathbf{G}_k$ in the deterministic process because $\mathbf{G}_k$ consists of impulse response coefficients which are deterministic parts in the LTI stochastic process. Therefore, $\mathbf{G}_k$ is updated using iterative learning identification (5.15) and estimated state $\hat{\mathbf{E}}_k$ is used for input update law as follows:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{H}_k \hat{\mathbf{E}}_k \tag{5.42}$$

The procedure of the approach 1 is summarized in Fig. 5.2.

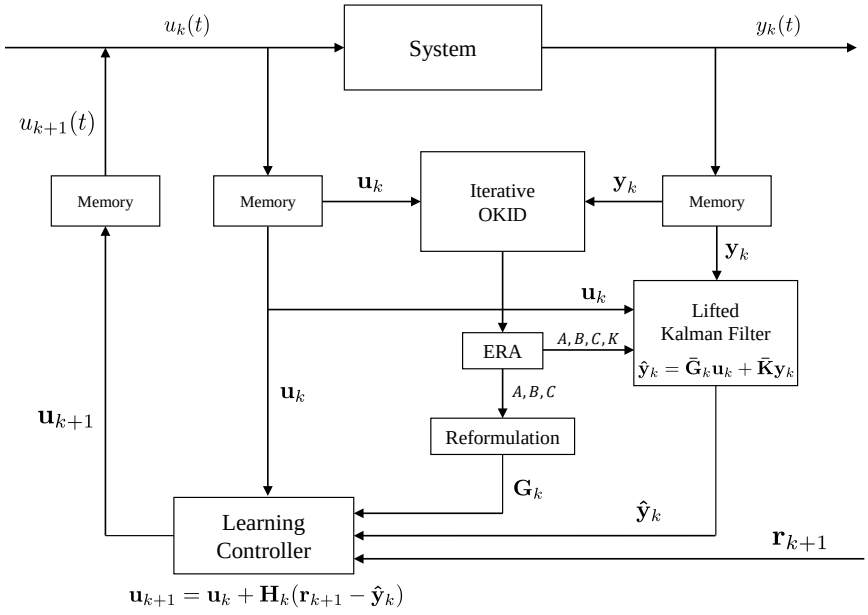


Figure 5.3: The scheme of the time-domain Kalman filter-based stochastic ILC for batch-varying reference trajectories.

5.3.2 Approach 2: Time-Domain Kalman Filter-Based Approach

5.3.2.1 Identification of Observer/Kalman filter Markov parameters (OKID)

This algorithm computes the Markov parameters of an observer or Kalman filter from experimental input and output data [96].

Add and subtract the term $My(t)$ to the right-hand side of the state equation, Eq. (5.1), to yield

$$\begin{aligned} x_k(t+1) &= Ax_k(t) + Bu_k(t) + My_k(t) - My_k(t) \\ &= (A+MC)x_k(t) + Bu_k(t) - My_k(t) \\ &= \bar{A}x_k(t) + \bar{B}\bar{u}_k(t) \end{aligned} \tag{5.43}$$

where

$$\begin{aligned} \bar{A} &= A + MC \\ \bar{B} &= \begin{bmatrix} B & -M \end{bmatrix} \\ \bar{u}_k &= \begin{bmatrix} u_k(t) \\ y_k(t) \end{bmatrix} \end{aligned} \tag{5.44}$$

The input-output description in matrix form becomes

$$\underset{q\times N}{\mathbf{Y}_k} = \underset{q\times(q+m)p}{\bar{\mathbf{g}}_k} \overset{(q+m)p\times N}{\bar{\mathbf{U}}_k} \tag{5.45}$$

where

$$\begin{aligned} \mathbf{Y}_k &= \begin{bmatrix} y_k(1) & y_k(2) & \cdots & y_k(p) & \cdots & y_k(N) \end{bmatrix} \\ \bar{\mathbf{g}}_k &= \begin{bmatrix} C\bar{B} & C\bar{A}\bar{B} & \cdots & C\bar{A}^{p-1}\bar{B} \end{bmatrix} \end{aligned} \tag{5.46}$$

and

$$\bar{\mathbf{U}}_k = \begin{bmatrix} \bar{u}_k(0) & \bar{u}_k(1) & \bar{u}_k(2) & \cdots & \bar{u}_k(p-1) & \cdots & \bar{u}_k(N-1) \\ & \bar{u}_k(0) & \bar{u}_k(1) & \cdots & \bar{u}_k(p-2) & \cdots & \bar{u}_k(N-2) \\ & & \bar{u}_k(0) & \cdots & \bar{u}_k(p-3) & \cdots & \bar{u}_k(N-3) \\ & & & \ddots & \vdots & \cdots & \vdots \\ & & & & \bar{u}_k(0) & \cdots & \bar{u}_k(N-p) \end{bmatrix} \tag{5.47}$$

The first p Markov parameters approximately satisfy $\bar{\mathbf{g}}_k = \mathbf{Y}_k \bar{\mathbf{U}}_k^{\dagger}$ and the approximation error decreases as p increases. To find $\bar{\mathbf{g}}_k$ iteratively, we can use the following system Markov parameters update law introduced in Section 5.2.2.

$$\bar{\mathbf{g}}_k = \bar{\mathbf{g}}_{k-1} + (\mathbf{Y}_k - \hat{\mathbf{Y}}_k)\bar{\mathbf{U}}_k^{\dagger} \tag{5.48}$$

To recover the system Markov parameters in $\mathbf{g}_k$ from the observer Markov parameters in $\bar{\mathbf{g}}_k$, the following notation is used.

$$\bar{\mathbf{g}}_k = \begin{bmatrix} C\bar{B} & C\bar{A}\bar{B} & \cdots & C\bar{A}^{p-1}\bar{B} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{g}}_{k,0} & \bar{\mathbf{g}}_{k,1} & \cdots & \bar{\mathbf{g}}_{k,p-1} \end{bmatrix} \tag{5.49}$$

where

$$\begin{aligned} \bar{\mathbf{g}}_{k,l} &= C\bar{A}^l\bar{B} \\ &= \begin{bmatrix} C(A+MC)^l B & -C(A+MC)^l M \end{bmatrix} \\ &\equiv \begin{bmatrix} \bar{\mathbf{g}}_{k,l}^{(1)} & \bar{\mathbf{g}}_{k,l}^{(2)} \end{bmatrix} \end{aligned} \tag{5.50}$$

The general relationship between the actual system Markov parame-

ters and observer Markov parameters is

$$\mathbf{g}_{k,l} = CA^lB = \bar{\mathbf{g}}^{(1)}_{k,l} + \sum_{i=0}^{l-1} \bar{\mathbf{g}}^{(2)}_{k,i} \bar{\mathbf{g}}_{k,l-i-1} \tag{5.51}$$

Eq. (5.51) can be rewritten in the following matrix form:

$$\begin{bmatrix} \mathbf{I} & & & & \\ -\bar{\mathbf{g}}^{(2)}_{k,0} & \mathbf{I} & & & \\ -\bar{\mathbf{g}}^{(2)}_{k,1} & -\bar{\mathbf{g}}^{(2)}_{k,0} & \mathbf{I} & & \\ \vdots & \vdots & \vdots & \ddots & \\ -\bar{\mathbf{g}}^{(2)}_{k,p-2} & -\bar{\mathbf{g}}^{(2)}_{k,p-3} & -\bar{\mathbf{g}}^{(2)}_{k,p-4} & \cdots & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{g}_{k,0} \\ \mathbf{g}_{k,1} \\ \mathbf{g}_{k,2} \\ \vdots \\ \mathbf{g}_{k,p-1} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{g}}^{(1)}_{k,0} \\ \bar{\mathbf{g}}^{(1)}_{k,1} \\ \bar{\mathbf{g}}^{(1)}_{k,2} \\ \vdots \\ \bar{\mathbf{g}}^{(1)}_{k,p-1} \end{bmatrix} \tag{5.52}$$

Note that identity matrix $\mathbf{I}$ and all $\bar{\mathbf{g}}^{(2)}_{k,i}$ are $q \times q$ square matrices; therefore, unique system Markov parameters can be computed by using inverse matrix. Then, the system Markov parameters, $\mathbf{g}_{k,i}$, are used in a Hankel matrix to identify A, B, and C by the eigensystem realization algorithm (ERA). The obtained A, C, and the observer Markov parameters, $\bar{\mathbf{g}}^{(2)}_{k,i}$, are used to calculate steady-state Kalman gain, K. For further details and proof, see the references [101, 102].

5.3.2.2 Input update law using filtered output

In section 5.3.2.1, we obtained the state space model (A, B, C) for computing the learning gain matrix $\mathbf{H}_k$ and steady-state Kalman gain (K) using iterative OKID and ERA, and therefore the filtered output can be computed without recursive calculation. Kalman filter equation for the time and measurement updates are presented as fol-

lows.

$$\begin{aligned}
\hat{x}_k^-(t) &= A\hat{x}_k(t-1) + Bu_k(t-1) && : \text{Time update} \\
\hat{x}_k(t) &= \hat{x}_k^-(t) + K[y_k(t) - C\hat{x}_k^-(t)] && : \text{Measurement update} \\
\hat{y}_k(t) &= C\hat{x}_k(t) && : \text{Filtered output}
\end{aligned} \tag{5.53}$$

where $\hat{x}_k^-(t)$ is predicted *a priori* state estimate, $\hat{x}_k(t)$ is updated *a posteriori* state estimate. By substituting the time update equation into measurement update equation, we can obtain the following equation:

$$\begin{aligned}
\hat{x}_k(t) =& A\hat{x}_k(t-1) + Bu_k(t-1) \\
&+ K\left\{y_k(t) - C[A\hat{x}_k(t-1) + Bu_k(t-1)]\right\} \\
=& (A - KCA)\hat{x}_k(t-1) + (B - KCB)u_k(t-1) + Ky_k(t) \\
=& A_K\hat{x}_k(t-1) + B_K u_k(t-1) + Ky_k(t) \\
\hat{y}_k(t) =& C\hat{x}_k(t)
\end{aligned} \tag{5.54}$$

where $A_K = A - KCA$ and $B_K = B - KCB$. These equations for a sequence of time steps can be written as

$$\begin{bmatrix} \hat{y}_k(1) \\ \hat{y}_k(2) \\ \vdots \\ \hat{y}_k(N) \end{bmatrix} = \begin{bmatrix} CB_K & & & \\ CA_KB_K & CB_K & & \\ \vdots & \vdots & \ddots & \\ CA_K^{N-1}B_K & CA_K^{N-2}B_K & \cdots & CB_K \end{bmatrix} \begin{bmatrix} u_k(0) \\ u_k(1) \\ \vdots \\ u_k(N-1) \end{bmatrix} + \begin{bmatrix} CK & & & \\ CA_KK & CK & & \\ \vdots & \vdots & \ddots & \\ CA_K^{N-1}K & CA_K^{N-2}K & \cdots & CK \end{bmatrix} \begin{bmatrix} y_k(1) \\ y_k(2) \\ y_k(3) \\ \vdots \\ y_k(N) \end{bmatrix} \tag{5.55}$$

$$\hat{\mathbf{y}}_k = \bar{\mathbf{G}}_k \mathbf{u}_k + \bar{\mathbf{K}} \mathbf{y}_k \tag{5.56}$$

We can use the following input update law using the filtered output $\hat{\mathbf{y}}_k$.

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{H}_k(\mathbf{r}_{k+1} - \hat{\mathbf{y}}_k) \tag{5.57}$$

The procedure of the approach 2 is illustrated in Fig. 5.3.

5.4 Numerical Examples

5.4.1 Example 1 (Random Reference Trajectories

The proposed algorithms are evaluated using the following linear discrete time-invariant system (5.58) converted from the transfer

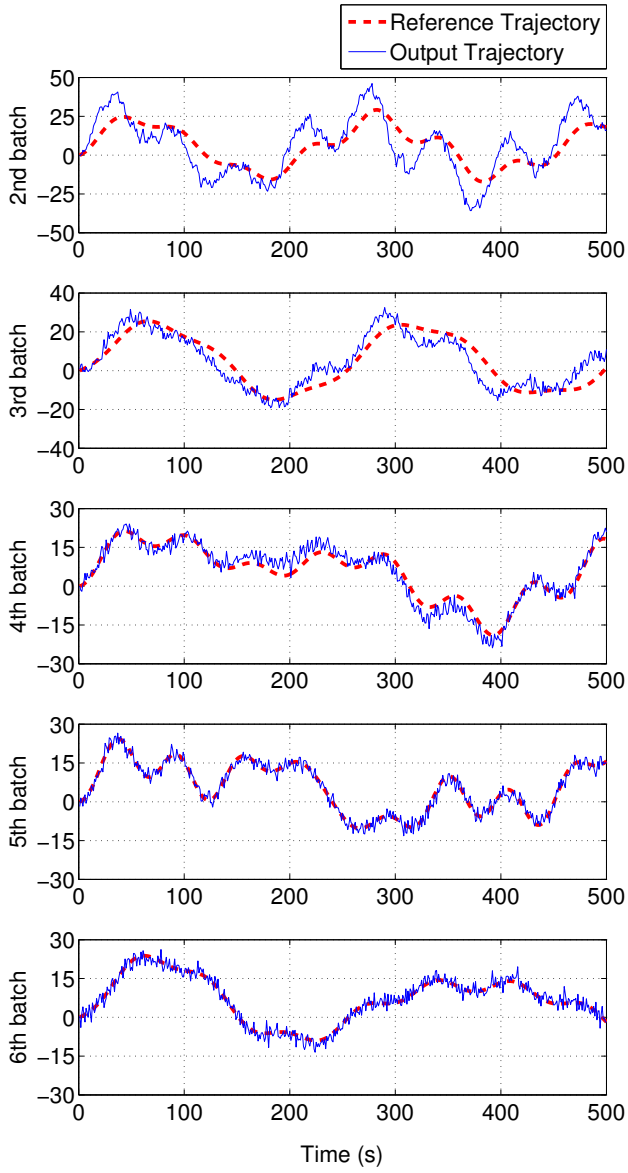


Figure 5.4: The tracking results of the deterministic ILC for batch-varying reference trajectories from second batch to sixth batch (Example 1).

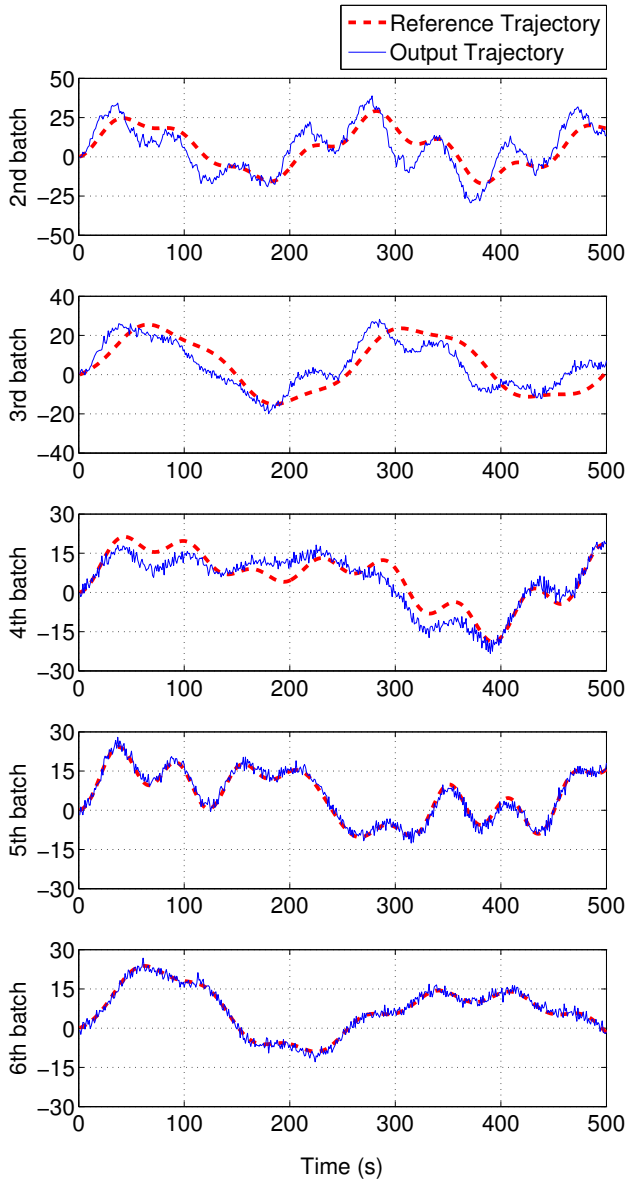


Figure 5.5: The tracking results of the batch-domain Kalman filter-based stochastic ILC for batch-varying reference trajectories from second batch to sixth batch (Example 1).

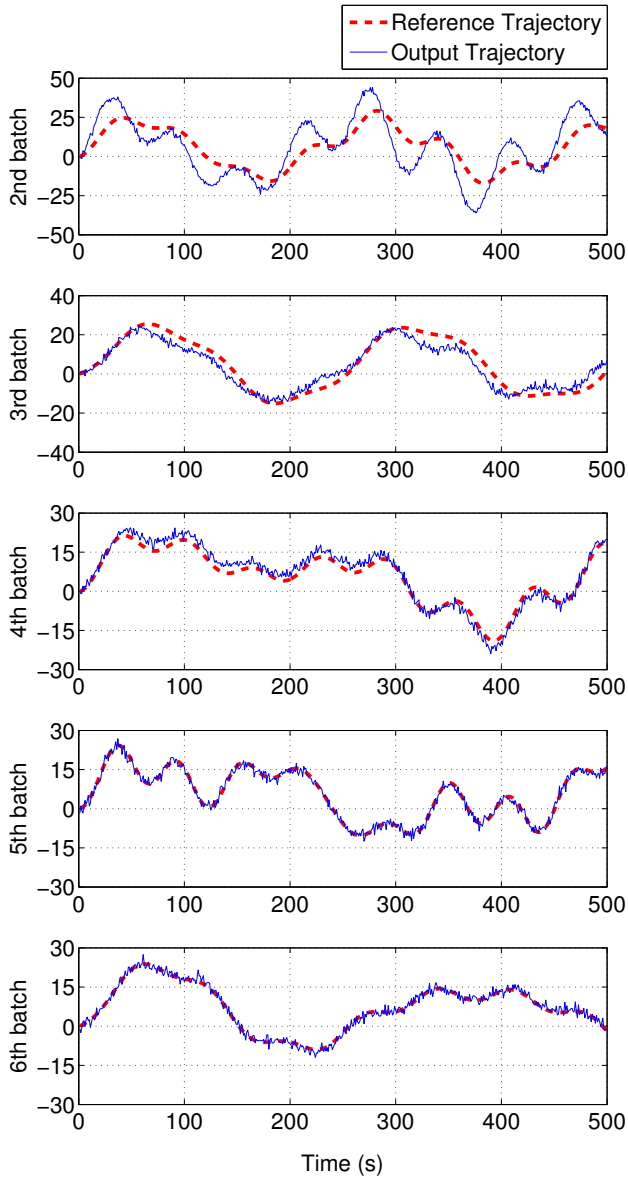


Figure 5.6: The tracking results of the time-domain Kalman filter-based stochastic ILC for batch-varying reference trajectories from second batch to sixth batch (Example 1).

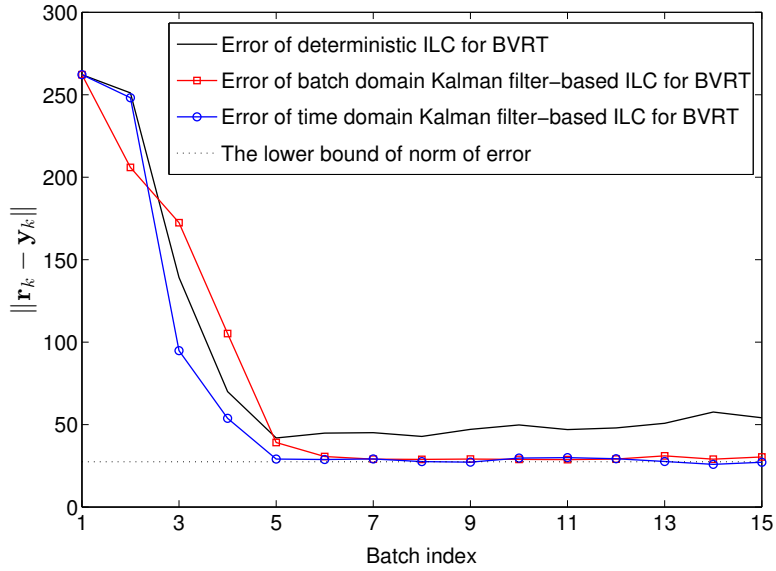


Figure 5.7: Comparison of the norm of error profiles of the proposed approaches (Example 1).

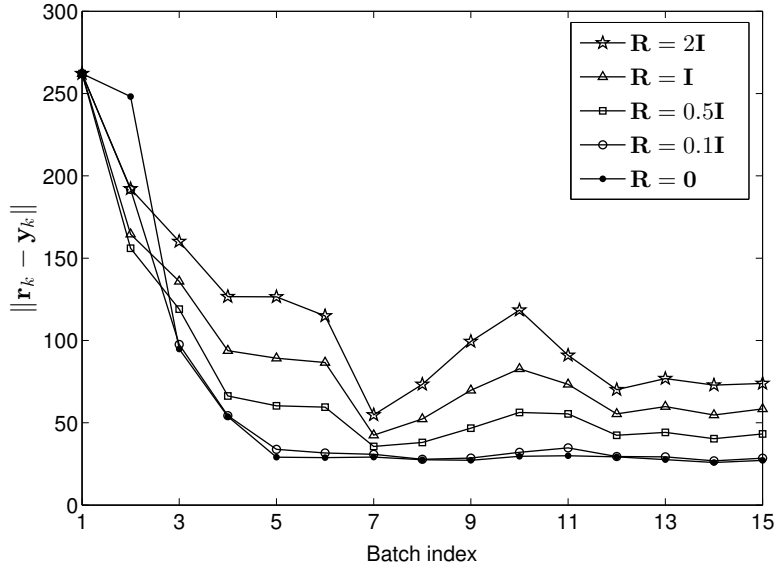


Figure 5.8: Convergence performance according to weighting factor ($\mathbf{R}$) of the Q-ILC controller (Example 1).

function $G(s) = 1.5/(16s^2 + 10s + 1)$.

$$x_k(t+1) = \begin{bmatrix} 0.5145 & -0.0460 \\ 0.7359 & 0.9745 \end{bmatrix} x_k(t) + \begin{bmatrix} 0.7359 \\ 0.4082 \end{bmatrix} u_k(t) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} w_k(t)$$
$$y_k(t) = \begin{bmatrix} 0 & 0.0938 \end{bmatrix} x_k(t) + v_k(t) \tag{5.58}$$

which operates on $t \in [0, 1, \cdots, 500]$ and $w_k(t) \sim N(0, 0.01)$, $v_k(t) \sim N(0, 1.5)$. Q and R are 0.01 and 1.5, respectively. BVRT are produced using the following random trajectory generator.

$$r_k(t) = 10 \left[sin\left(\frac{t}{d_{1,k}}\right) + sin^2\left(\frac{t}{d_{2,k}}\right) + sin^3\left(\frac{t}{d_{3,k}}\right) \right] \tag{5.59}$$

where $d_{1,k}, d_{2,k}$ and $d_{3,k}$ are random integers from the uniform distribution between 20 and 90, changed along the batch index.

First, deterministic ILC for BVRT presented in Section 5.2.3 was applied to the given stochastic system. A unit step input was used for the first batch. To use the input update law (5.26) for the second batch, learning gain matrix $\mathbf{H}_1 = (\mathbf{G}_1^T \mathbf{G}_1)^{-1} \mathbf{G}_1^T$ was computed using iterative learning identification (5.21). Then, the input update law (5.26) was applied to find the second input trajectory. Fig. 4 shows the simulation results of the deterministic ILC for BVRT from the second to sixth batches. The result shows good tracking performance even if reference trajectories vary with every batch. However, outputs have larger noise than inherent noise in the system because deterministic ILC for BVRT does not consider the noise effect.

Before simulating two Kalman filter-based ILC for BVRT proposed in Section 5.3, we should calculate the norm of inherent noise

(state and measurement noises) in the stochastic system. In this paper, convergence performance is evaluated using the norm of error, which cannot go to zero in the stochastic system because inherent noise cannot be eliminated. Thus, the norm of error in the stochastic system can converge up to the norm of inherent noise. In order to calculate the norm of inherent noise, we have to pre-calculate the covariance of output vector as in the following steps.

$$\begin{aligned}
\mathbb{E}[x_k(t+1)x_k(t+1)^T] &= A\mathbb{E}[x_k(t)x_k(t)^T]A^T + \Gamma Q\Gamma^T \\
\mathbb{E}[y_k(t)y_k(t)^T] &= C\mathbb{E}[x_k(t)x_k(t)^T]C^T + R \\
&= \sum_{i=0}^{t-1} CA^i\Gamma Q\Gamma^T(A^T)^iC^T + R \\
&\equiv R_y(t)
\end{aligned} \tag{5.60}$$

Defining $\mathbf{R}_y \equiv \text{diag}\{R_y(1), R_y(2), \cdots, R_y(500)\}$, the norm of stochastic vector $\mathbf{y}_k$ is, $\|\mathbf{y}_k\| = \sqrt{\mathbb{E}[\mathbf{y}_k^T\mathbf{y}_k]} = \sqrt{\text{trace}(\mathbf{R}_y)} = 27.49$. Hence, the lower bound of the norm of error is 27.49.

The simulation results of batch-domain Kalman filter-based ILC for BVRT proposed in Section 5.3.1 are shown in Fig. 5. In this case, the convergence rate is slower than that of the deterministic ILC for BVRT because several batches are required for convergence of Kalman filter in the batch-domain. However, it shows better convergence performance than deterministic ILC for BVRT. Finally, Fig. 6 shows the simulation results of the time-domain Kalman filter-based ILC for BVRT. Comparison results of the deterministic approach, batch-domain and time-domain Kalman filter-based approaches are are shown in Fig. 7. Both the Kalman filter-based approaches are convergent up to the lower bound of the norm of error unlike de-

terministic approach, but the time-domain Kalman filter-based approach shows faster convergence rate. This is because steady-state Kalman gain is used in the approach 2. In addition, the approach 2 can converge without using covariance information. Thus from numerical simulations, we can conclude that second approach is better than the first.

In Section 5.2.3.1, we mentioned that convergence performance decreases as input weighting factor $\mathbf{R}$ of Q-ILC controller increases. Learning gain matrix $(\mathbf{G}_k^T\mathbf{G}_k)^{-1}\mathbf{G}_k^T$ of the time-domain approach was replaced by the quadratic criterion-based learning gain matrix $(\mathbf{G}_k^T\mathbf{Q}\mathbf{G}_k+\mathbf{R})^{-1}\mathbf{G}_k^T\mathbf{Q}$, and the simulation was performed using various size of $\mathbf{R}$, fixing $\mathbf{Q}=\mathbf{I}$. As the size of $\mathbf{R}$ increases, convergence performance gradually diminishes as shown in Fig. 8.

5.4.2 Example 2 (Particular Types of Reference Trajectories

We consider temperature control of a linearized batch reactor where a second-order exothermic reaction $A \rightarrow B$ takes place [5]. It is assumed that temperature of a cooling jacket is directly manipulated.

$$\begin{aligned}\frac{dT(t)}{dt} =& \left[-k0\frac{\Delta HV}{MC_p}\frac{E}{R}\frac{C_{A0}^2}{T_0^2}e^{-E/RT_0} - \frac{UA}{MC_p}\right]T(t) \\ & - 2\frac{\Delta HV}{MC_p}k_0C_{A0}e^{-E/RT_0}C_A(t) + \frac{UA}{MC_p}T_j(t) + w_1(t) \\ \frac{dC_A(t)}{dt} =& - k_0\frac{E}{R}\frac{C_{A0}^2}{T_0^2}e^{-E/RT_0}T(t) - 2k_0C_{A0}e^{-E/RT_0}C_A(t) + w_2(t)\end{aligned} \tag{5.61}$$

The following values were used for the parameters:

$$\begin{aligned}
\frac{UA}{MC_p} &= 0.09\ (l/min) \\
\frac{\Delta HV}{MC_p} &= -1.64\ (K\ l/mol) \\
k_0 &= 2.53 \times 10^{19}\ (l/mol\ min) \\
\frac{E}{R} &= 13,550\ (K) \\
T_0 &= 25\ (^{\circ}C) \\
C_{A0} &= 0.9\ (mol/l)
\end{aligned} \tag{5.62}$$

which operates on $t \in [0,\ 0.2,\ \cdots,\ 100]$ and $w_1(t) \sim N(0,\ 0.0001)$, $w_2(t) \sim N(0,\ 0.001)$ and measurement noise $v(t) \sim N(0,\ 0.1)$. Reference trajectory of batch reactor can be changed for different feed concentration, operating condition or reducing operation time. In this case, we consider three types of reference trajectories, which are for normal operation (1st, 2nd and 3rd batches), faster shut down (4th and 5th batches) and faster start up (6th and 7th batches) then, trajectories are repeated in a same order. Application process is the same with the example 1, but it is assumed that we obtained a unit step input and a step response before computing a input signal of a first batch. For calculating the lower bound of the norm of error for the example 2, we should convert the state noise covariance for the continuous time system to the covariance for the discrete time system. Then, we can obtain the lower bound of the norm of error, $\|\mathbf{y}_k\| = 7.57$, using Eq. (5.60). Fig. 5.9, 5.10 and 5.11 show the tracking results of one deterministic approach and two stochastic approaches. Simulation results of example 1 and 2 show similar

convergence trends as shown in Fig. 5.12. As with the previous example, the time domain Kalman filter-based approach shows better convergence performance than the batch domain Kalman filter-based approach.

5.5 Conclusion

This paper has presented adaptive ILC schemes for discrete LTI stochastic system with BVRT. For handling stochastic issue, we have developed two approaches. In the first approach, we suppose that there exist state noise in input update law and measurement noise in input-output relationship for defining the stochastic state-space model. Using defined state-space model, updated term $\mathbf{r}_{k+1} - \mathbf{y}_k$ in the input update law (5.42) is estimated using batch-domain Kalman filter. In the second approach, pure output signal for input update is estimated using time-domain Kalman filter. Markov parameters and steady-state Kalman gain are calculated in the iterative OKID step. The calculated Markov parameters are used for generating learning gain matrix and the steady-state Kalman gain is used for time-domain Kalman filter. Because the steady-state Kalman gain is computed in the iterative OKID step, the Kalman filter can be applied without covariance information. Both the approaches show similar convergence performance but the second approach shows faster convergence rate. Future work will consider nonlinear systems to cover wider operation ranges.

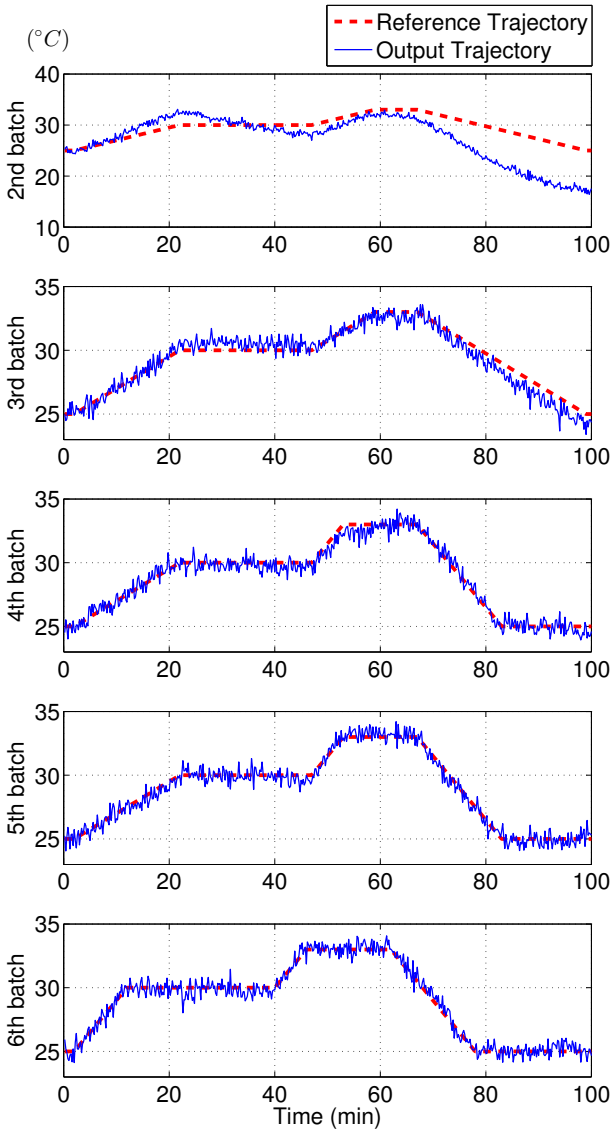


Figure 5.9: The tracking results of the deterministic ILC for batch-varying reference trajectories from second batch to sixth batch (Example 2).

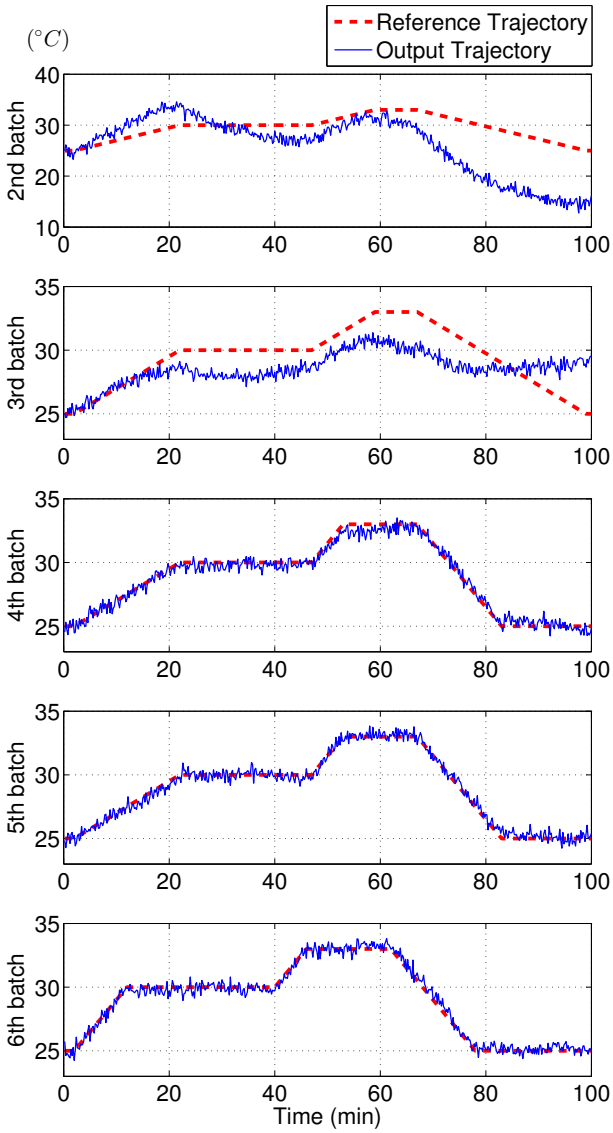


Figure 5.10: The tracking results of the batch-domain Kalman filter-based stochastic ILC for batch-varying reference trajectories from second batch to sixth batch (Example 2).

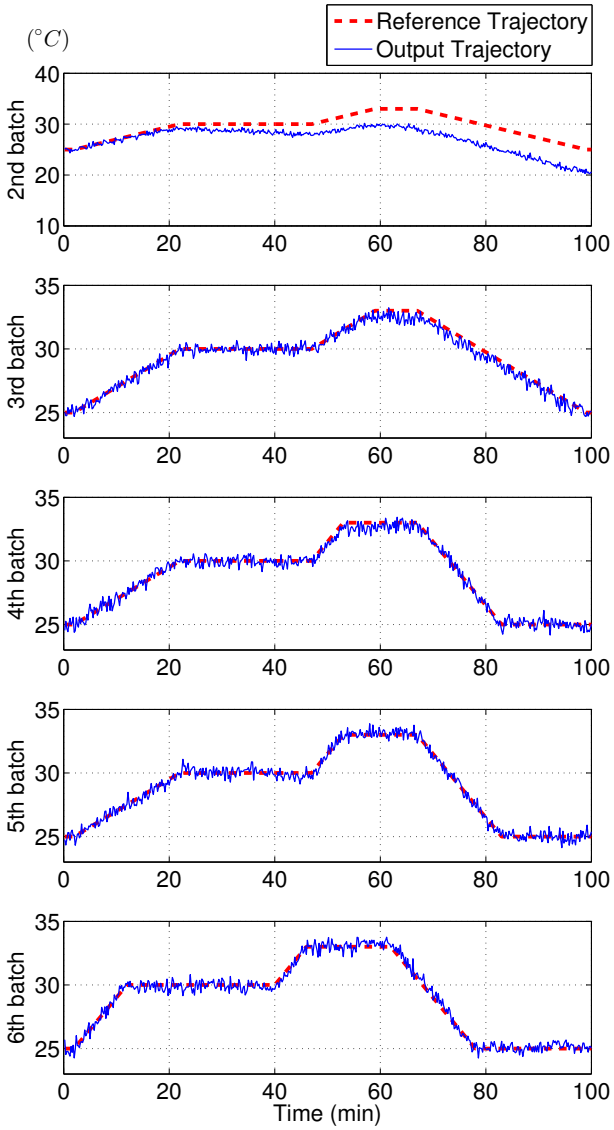


Figure 5.11: The tracking results of the time-domain Kalman filter-based stochastic ILC for batch-varying reference trajectories from second batch to sixth batch (Example 2).

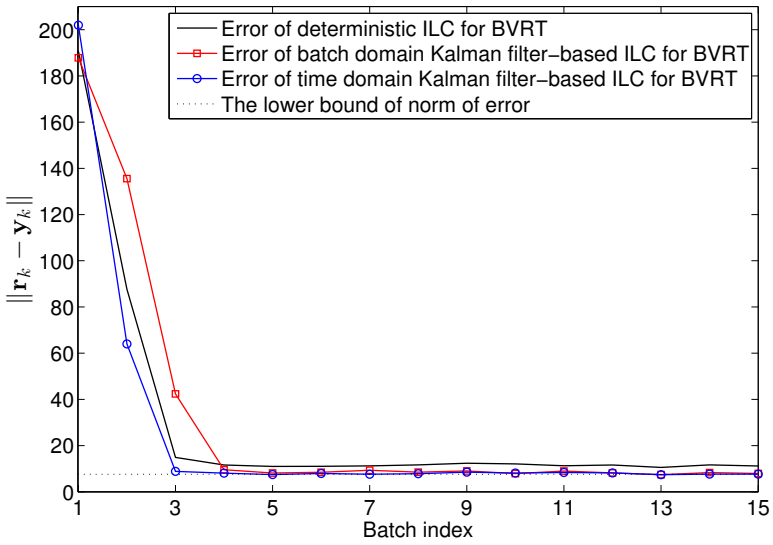


Figure 5.12: Comparison of the norm of error profiles of the proposed approaches (Example 2).

Chapter 6

Conclusions and Future Works

6.1 Conclusions

In a batch, cyclic, repetitive or iteration process, perfect tracking of a time-varying reference trajectory cannot be achieved with conventional control techniques because of highly nonlinear dynamics and model uncertainty. To address this issue, many ILC algorithms have been developed. In this thesis, we propose the standard form of ILMPC technique. The proposed ILMPC is similar to conventional MPC and includes all advantages of MPC and ILC; thus, many techniques for conventional control method can be applied without particular modification. However, it is not always desirable that the output error converge to zero. An input trajectory for perfect tracking including vertices of a reference trajectory may have a non-smooth trajectory. For convergence with non-zero error, we use a generalized objective function to independently tune weighting factors of manipulated variable change with respect to both the time index and batch horizons. The major contribution of this thesis is to propose a novel ILMPC for tracking specific desired points without generating a reference trajectory passing through the specific desired points. Tracking an entire reference trajectory is not always necessary in many

applications. To address this issue, we introduce an extraction matrix that extracts only the components related to specific points. Then, we design the PTP ILMPC algorithm using the extraction matrix. Tracking reference trajectories, disturbance rejection and convergence were found to be successfully achieved in all the cases.

6.2 Future work

There are several directions for further work based on the suggested framework in this thesis. They include:

- PTP ILMPC combined with economic MPC: If the output can converge to specific points, there is more degree of freedom because there is intervals where the output does not have reference trajectory to track. In the interval without reference trajectory, it may be possible to calculate the input trajectory that maximizes economic efficiency.
- PTP ILMPC combined with length-varying ILC: In many batch processes, shortening the operating time means improving the economy. The proposed PTP ILMPC method assumes the same operating time; thus, the assumption needs to be relaxed.

Bibliography

[1] G. Murray, Learning control of actuators in control systems, US Patent 3,555,252 (Jan. 12 1971).

[2] M. Uchiyama, Formation of high-speed motion pattern of a mechanical arm by trial (japanese title: 試行による人工の手の高速運動パタンの形成), Transactions of the Society of Instrument and Control Engineers 14 (1978) 706–712.

[3] S. Arimoto, S. Kawamura, F. Miyazaki, Bettering operation of robots by learning, Journal of Robotic systems 1 (2) (1984) 123–140.

[4] K. S. Lee, I. S. Chin, H. J. Lee, J. H. Lee, Model predictive control technique combined with iterative learning for batch processes, AIChE Journal 45 (10) (1999) 2175–2187.

[5] J. H. Lee, K. S. Lee, W. C. Kim, Model-based iterative learning control with a quadratic criterion for time-varying linear systems, Automatica 36 (5) (2000) 641–657.

[6] I. Chin, S. J. Qin, K. S. Lee, M. Cho, A two-stage iterative learning control technique combined with real-time feedback for independent disturbance rejection, Automatica 40 (11) (2004) 1913–1922.

[7] Z. H. Xiong, J. Zhang, X. Wang, Y. M. Xu, Tracking control for batch processes through integrating batch-to-batch iterative learning control and within-batch on-line control, Industrial & Engineering Chemistry Research 44 (11) (2005) 3983–3992.

[8] X. J. Liu, X. B. Kong, Nonlinear fuzzy model predictive iterative learning control for drum-type boiler-turbine system, Journal of Process Control 23 (8) (2013) 1023–1040.

[9] J. Y. Lu, Z. X. Cao, Z. Wang, F. R. Gao, A two-stage design of two-dimensional model predictive iterative learning control for nonrepetitive disturbance attenuation, Industrial & Engineering Chemistry Research 54 (21) (2015) 5683–5689.

[10] S. Y. Mo, L. M. Wang, Y. Yao, F. R. Gao, Two-time dimensional dynamic matrix control for batch processes with convergence analysis against the 2d interval uncertainty, Journal of Process Control 22 (5) (2012) 899–914.

[11] P. Lundstrom, J. H. Lee, M. Morari, S. Skogestad, Limitations of dynamic matrix control, Computers & Chemical Engineering 19 (4) (1995) 409–421.

[12] J. X. Xu, Y. Q. Chen, T. H. Lee, S. Yamamoto, Terminal iterative learning control with an application to rtpcvd thickness control, Automatica 35 (9) (1999) 1535–1542.

[13] K. S. Lee, J. H. Lee, Iterative learning control-based batch process control technique for integrated control of end product properties and transient profiles of process variables, Journal of Process Control 13 (7) (2003) 607–621.

[14] C. T. Freeman, Y. Tan, Iterative learning control with mixed constraints for point-to-point tracking, IEEE Transactions on Control Systems Technology 21 (3) (2013) 604–616.

[15] C. T. Freeman, Z. L. Cai, E. Rogers, P. L. Lewin, Iterative learning control for multiple point-to-point tracking application, IEEE Transactions on Control Systems Technology 19 (3) (2011) 590–600.

[16] T. D. Son, H. S. Ahn, K. L. Moore, Iterative learning control in optimal tracking problems with specified data points, Automatica 49 (5) (2013) 1465–1472.

[17] B. Chu, C. T. Freeman, D. H. Owens, A novel design framework for point-to-point ilc using successive projection, IEEE Transactions on Control Systems Technology 23 (3) (2015) 1156–1163.

[18] Z. S. Hou, Y. Wang, C. K. Yin, T. Tang, Terminal iterative learning control based station stop control of a train, International Journal of Control 84 (7) (2011) 1263–1274.

[19] R. H. Chi, D. W. Wang, Z. S. Hou, S. T. Jin, Data-driven optimal terminal iterative learning control, Journal of Process Control 22 (10) (2012) 2026–2037.

[20] S. Arimoto, S. Kawamura, F. Miyazaki, Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronics systems, The 23rd IEEE Conference on Decision and Control 23 (1984) 1–6.

[21] S. Arimoto, Mathematical theory of learning with applications to robot control, in: Adaptive and Learning Systems, Springer, 1986, pp. 379–388.

[22] M. Togai, O. Yamano, Analysis and design of an optimal learning control scheme for industrial robots: A discrete system approach, in: Decision and Control, 1985 24th IEEE Conference on, IEEE, 1985, pp. 1399–1404.

[23] S.-R. Oh, Z. Bien, I. H. Suh, An iterative learning control method with application to robot manipulators, IEEE Journal on Robotics and Automation 4 (5) (1988) 508–514.

[24] P. Lucibello, Learning control of linear systems, in: American Control Conference, no. 29, 1992, pp. 1888–1892.

[25] K. L. Moore, Iterative learning control for deterministic systems, Springer Science & Business Media, 1993.

[26] K. Lee, S. Bang, K. Chang, Feedback-assisted iterative learning control based on an inverse process model, Journal of Process Control 4 (2) (1994) 77–89.

[27] K. Yamada, K. Watanabe, M. Tsuchiya, T. Kaneko, Robust control design for repetitive control systems with filtered inverse, in: Proc. Asian Contr. Conf, 1994, pp. 243–246.

[28] K. M. Tao, R. L. Kosut, G. Aral, Learning feedforward control, in: American Control Conference, 1994, Vol. 3, IEEE, 1994, pp. 2575–2579.

[29] T. Sogo, N. Adachi, A gradient-type learning control algorithm for linear systems, in: Proceedings of ASCC, Vol. 3, 1994, pp. 227–230.

[30] N. Amann, D. H. Owens, E. Rogers, A. Wahl, An h∞ approach to linear iterative learning control design, International Journal of Adaptive Control and Signal Processing 10 (6) (1996) 767–781.

[31] N. Amann, D. H. Owens, E. Rogers, Iterative learning control using optimal feedback and feedforward actions, International Journal of Control 65 (2) (1996) 277–293.

[32] N. Amann, D. H. Owens, E. Rogers, Robustness of norm-optimal iterative learning control, in: Control'96, UKACC International Conference on (Conf. Publ. No. 427), Vol. 2, IET, 1996, pp. 1119–1124.

[33] K.-S. Lee, W.-C. Kim, J.-H. Lee, Model-based iterative learning control with quadratic criterion for linear batch processes, Journal of Institute of Control, Robotics and Systems 2 (3) (1996) 148–157.

[34] Z. Bien, K. M. Huh, Higher-order iterative learning control algorithm, in: IEE Proceedings D-Control Theory and Applications, Vol. 136, IET, 1989, pp. 105–112.

[35] Y. Chen, Z. Gong, C. Wen, Analysis of a high-order iterative learning control algorithm for uncertain nonlinear systems with state delays, Automatica 34 (3) (1998) 345–353.

[36] K. L. Moore, Y. Chen, On monotonic convergence of high order iterative learning update laws, IFAC Proceedings Volumes 35 (1) (2002) 19–24.

[37] S. Gunnarsson, M. Norrlöf, On the disturbance properties of high order iterative learning control algorithms, Automatica 42 (11) (2006) 2031–2034.

[38] J. Hätönen, D. H. Owens, K. Feng, Basis functions and parameter optimisation in high-order iterative learning control, Automatica 42 (2) (2006) 287–294.

[39] S. Arimoto, Robustness of learning control for robot manipulators, in: Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on, IEEE, 1990, pp. 1528–1533.

[40] S. Arimoto, T. Naniwa, H. Suzuki, Robustness of p-type learning control with a forgetting factor for robotic motions, in: Decision and Control, 1990., Proceedings of the 29th IEEE Conference on, IEEE, 1990, pp. 2640–2645.

[41] T.-y. Kuc, K. Nam, J. S. Lee, An iterative learning control of robot manipulators, IEEE Transactions on Robotics and Automation 7 (6) (1991) 835–842.

[42] R. Horowitz, Learning control of robot manipulators, Journal of Dynamic Systems, Measurement, and Control 115 (2B) (1993) 402–411.

[43] H. S. Ahn, Y. Chen, K. L. Moore, Iterative learning control: Brief survey and categorization, IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews 37 (6) (2007) 1099–1121.

[44] K. Lee, S. Bang, S. Yi, J. Son, S. Yoon, Iterative learning control of heat-up phase for a batch polymerization reactor, Journal of process control 6 (4) (1996) 255–262.

[45] J.-X. Xu, Q. Hu, T. H. Lee, S. Yamamoto, Iterative learning control with smith time delay compensator for batch processes, Journal of process Control 11 (3) (2001) 321–328.

[46] M. Mezghani, G. Roux, M. Cabassud, M.-V. Le Lann, B. Dahhou, G. Casamatta, Application of iterative learning control to an exothermic semibatch chemical reactor, IEEE Transactions on Control Systems Technology 10 (6) (2002) 822–834.

[47] Z. Xiong, J. Zhang, Product quality trajectory tracking in batch processes using iterative learning control based on time-varying perturbation models, Industrial & engineering chemistry research 42 (26) (2003) 6802–6814.

[48] J. Shi, F. Gao, T.-J. Wu, Robust design of integrated feedback and iterative learning control of a batch process based on a 2d roesser system, Journal of Process Control 15 (8) (2005) 907–924.

[49] J. H. Lee, K. S. Lee, Iterative learning control applied to batch processes: An overview, Control Engineering Practice 15 (10) (2007) 1306–1318.

[50] W. Cho, T. F. Edgar, J. Lee, Iterative learning dual-mode control of exothermic batch reactors, Control Engineering Practice 16 (10) (2008) 1244–1249.

[51] T. Ao, X. Dong, M. Zhizhong, Batch-to-batch iterative learning control of a batch polymerization process based on online sequential extreme learning machine, Industrial & Engineering Chemistry Research 48 (24) (2009) 11108–11114.

[52] H. Ahn, K. S. Lee, M. Kim, J. Lee, Control of a reactive batch distillation process using an iterative learning technique, Korean Journal of Chemical Engineering 31 (1) (2014) 6–11.

[53] R. Chi, Y. Liu, Z. Hou, S. Jin, Adaptive iterative learning control approach for ph neutralization of batch processes, in: Control Conference (ASCC), 2015 10th Asian, IEEE, 2015, pp. 1–5.

[54] F. Gao, Y. Yang, C. Shao, Robust iterative learning control with applications to injection molding process, Chemical Engineering Science 56 (24) (2001) 7025–7034.

[55] Y. Wang, D. Zhou, F. Gao, Iterative learning model predictive control for multi-phase batch processes, Journal of Process Control 18 (6) (2008) 543–557.

[56] L. Wang, X. He, D. Zhou, Average dwell time-based optimal iterative learning control for multi-phase batch processes, Journal of Process Control 40 (2016) 1–12.

[57] Y. Chen, J.-X. Xu, T. H. Lee, S. Yamamoto, An iterative learning control in rapid thermal processing, in: Proc. the IASTED Int. Conf. on Modeling, Simulation and Optimization (MSO'97), Citeseer, 1997, pp. 189–92.

[58] D. De Roover, O. H. Bosgra, Synthesis of robust multivariable iterative learning controllers with application to a wafer stage motion system, International Journal of Control 73 (10) (2000) 968–979.

[59] K. S. Lee, J. Lee, I. Chin, J. Choi, J. H. Lee, Control of wafer temperature uniformity in rapid thermal processing using an optimal iterative learning control technique, Industrial & engineering chemistry research 40 (7) (2001) 1661–1672.

[60] J. Y. Choi, H. M. Do, A learning approach of wafer temperature control in a rapid thermal processing system, IEEE Transactions on Semiconductor Manufacturing 14 (1) (2001) 1–10.

[61] B. G. Dijkstra, O. H. Bosgra, Exploiting iterative learning control for input shaping, with application to a wafer stage, in: American Control

Conference, 2003. Proceedings of the 2003, Vol. 6, IEEE, 2003, pp. 4811–4815.

[62] D. R. Yang, K. S. Lee, H. J. Ahn, J. H. Lee, Experimental application of a quadratic optimal iterative learning control method for control of wafer temperature uniformity in rapid thermal processing, IEEE Transactions on Semiconductor Manufacturing 16 (1) (2003) 36–44.

[63] M. Cho, Y. Lee, S. Joo, K. Lee, Sensor location, identification, and multivariable iterative learning control of an rtp process for maximum uniformity of wafer temperature distribution, in: Advanced Thermal Processing of Semiconductors, 2003. RTP 2003. 11th IEEE International Conference on, IEEE, 2003, pp. 177–184.

[64] C. L. van Oosten, O. H. Bosgra, B. G. Dijkstra, Reducing residual vibrations through iterative learning control, with application to a wafer stage, in: American Control Conference, 2004. Proceedings of the 2004, Vol. 6, IEEE, 2004, pp. 5150–5155.

[65] B. G. Dijkstra, Iterative learning control, with applications to a waferstage, TU Delft, Delft University of Technology, 2004.

[66] M. Cho, Y. Lee, S. Joo, K. S. Lee, Semi-empirical model-based multivariable iterative learning control of an rtp system, IEEE Transactions on Semiconductor Manufacturing 18 (3) (2005) 430–439.

[67] M. K. Cho, S. R. Joo, S. H. Won, K. S. Lee, Multivariable optimal learning control of wafer temperatures in a commercial rtp equipment, The Canadian Journal of Chemical Engineering 83 (2) (2005) 371–376.

[68] S. Mishra, W. Yeh, M. Tomizuka, Iterative learning control design for synchronization of wafer and reticle stages, in: 2008 American Control Conference, IEEE, 2008, pp. 3908–3913.

[69] S. Mishra, M. Tomizuka, Projection-based iterative learning control for wafer scanner systems, IEEE/ASME Transactions on Mechatronics 14 (3) (2009) 388–393.

[70] G. M. Bone, A novel iterative learning control formulation of generalized predictive control, Automatica 31 (10) (1995) 1483–1487.

[71] K. S. Lee, J. H. Lee, Convergence of constrained model-based predictive control for batch processes, IEEE Transactions on Automatic Control 45 (10) (2000) 1928–1932.

[72] J. Shi, F. Gao, T.-J. Wu, Single-cycle and multi-cycle generalized 2d model predictive iterative learning control (2D-GPILC) schemes for batch processes, Journal of Process Control 17 (9) (2007) 715–727.

[73] R. Zhang, A. Xue, J. Wang, S. Wang, Z. Ren, Neural network based iterative learning predictive control design for mechatronic systems with isolated nonlinearity, Journal of Process Control 19 (1) (2009) 68–74.

[74] Z. Mengfei, W. Shuqing, J. Xiaoming, Q. Zhang, Iterative learning model predictive control for a class of continuous/batch processes, Chinese Journal of Chemical Engineering 17 (6) (2009) 976–982.

[75] S.-K. Oh, J. M. Lee, Iterative learning model predictive control for constrained multivariable control of batch processes, Computers & Chemical Engineering 93 (2016) 284–292.

[76] P. Lucibello, Point to point polynomial control of linear systems by learning, in: Decision and Control, 1992., Proceedings of the 31st IEEE Conference on, IEEE, 1992, pp. 2531–2532.

[77] Y. Chen, J.-X. Xu, C. Wen, A high-order terminal iterative learning control scheme [rtp-cvd application], in: Decision and Control, 1997., Proceedings of the 36th IEEE Conference on, Vol. 4, IEEE, 1997, pp. 3771–3772.

[78] G. Gauthier, B. Boulet, Terminal iterative learning control design with singular value decomposition decoupling for thermoforming ovens, in: 2009 American Control Conference, IEEE, 2009, pp. 1640–1645.

[79] C. Freeman, Y. Tan, Point-to-point iterative learning control with mixed constraints, in: Proceedings of the 2011 American Control Conference, IEEE, 2011, pp. 3657–3662.

[80] T. D. Son, H.-S. Ahn, Terminal iterative learning control with multiple intermediate pass points, in: Proceedings of the 2011 American Control Conference, IEEE, 2011, pp. 3651–3656.

[81] C. T. Freeman, Constrained point-to-point iterative learning control with experimental verification, Control Engineering Practice 20 (5) (2012) 489–498.

[82] E. F. Camacho, C. B. Alba, Model predictive control, Springer Science & Business Media, 2013.

[83] L. Wang, Model predictive control system design and implementation using MATLAB, Springer Science & Business Media, 2009.

[84] U. Maeder, F. Borrelli, M. Morari, Linear offset-free model predictive control, Automatica 45 (10) (2009) 2214–2222.

[85] B. Joseph, F. Hanratty, Predictive control of quality in a batch manufacturing process using artificial neural-network models, Industrial & Engineering Chemistry Research 32 (1993) 1951–1961.

[86] E. Zafiriou, H.-W. Chiou, Output constraint softening for siso model predictive control, in: American Control Conference, IEEE, 1993, pp. 372–376.

[87] A. Zheng, M. Morari, Stability of model-predictive control with mixed constraints, IEEE Transactions on Automatic Control 40 (10) (1995) 1818–1823.

[88] H.-S. Ahn, K. L. Moore, Y. Chen, Iterative learning control: robustness and monotonic convergence for interval systems, Springer Science & Business Media, 2007.

[89] L. Wang, A tutorial on model predictive control: Using a linear velocity-form model, Developments in Chemical Engineering and Mineral Processing 12 (5-6) (2004) 573–614.

[90] J. Hätönen, K. Moore, A new arimoto-type algorithm to estimate states for repetitive processes: Iterative learning observer (ILO), in: Intelligent Control, 2007. ISIC 2007. IEEE 22nd International Symposium on, IEEE, 2007, pp. 232–236.

[91] J.-P. Corriou, Process control: theory and applications, Springer Science & Business Media, 2013.

[92] R. H. Chi, Z. S. Hou, J. X. Xu, Adaptive ILC for a class of discrete-time systems with iteration-varying trajectory and random initial condition, Automatica 44 (8) (2008) 2207–2213.

[93] X. D. Li, T. W. S. Chow, L. L. Cheng, Adaptive iterative learning control of non-linear MIMO continuous systems with iteration-varying initial error and reference trajectory, International Journal of Systems Science 44 (4) (2013) 786–794.

[94] J. X. Xu, J. Xu, On iterative learning from different tracking tasks in the presence of time-varying uncertainties, IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics 34 (1) (2004) 589–597.

[95] S.-K. Oh, J. M. Lee, Iterative learning control algorithm for a class of discrete lti system with batch-varying reference trajectories, in: Control, Automation and Systems (ICCAS), 14th International Conference on, IEEE, 2014, pp. 174–178.

[96] J. N. Juang, M. Phan, L. G. Horta, R. W. Longman, Identification of observer Kalman filter markov parameters - Theory and experiments, Journal of Guidance Control and Dynamics 16 (2) (1993) 320–329.

[97] A. Tiano, R. Sutton, A. Lozowicki, W. Naeem, Observer kalman filter identification of an autonomous underwater vehicle, Control Engineering Practice 15 (6) (2007) 727–739.

[98] K. L. Moore, M. Dahleh, S. P. Bhattacharyya, Iterative learning control - a survey and new results, Journal of Robotic Systems 9 (5) (1992) 563–594.

[99] S. H. Friedberg, A. J. Insel, L. E. Spence, Linear Algebra, Prentice Hall, 1997.

[100] T. Kailath, Linear systems, Vol. 1, Prentice-Hall Englewood Cliffs, NJ, 1980.

[101] J. N. Juang, R. S. Pappa, An eigensystem realization-algorithm for modal parameter-identification and model-reduction, Journal of Guidance Control and Dynamics 8 (5) (1985) 620–627.

[102] J.-N. Juang, M. Q. Phan, Identification and control of mechanical systems, Cambridge University Press, 2001.

초 록

본 논문에서는 제약조건이 있는 다변수 회분식 공정의 제어를 위해 반복학습제어(Iterative learning control, ILC)와 모델예측제어(Model predictive control, MPC)를 결합한 반복학습 모델예측제어(Iterative learning model predictive control, ILMPC)를 다룬다. 일반적인 ILC는 모델의 불확실성이 있더라도 이전 회분의 정보를 이용해 학습하기 때문에 출력을 기준궤적에 수렴시킬 수 있다. 하지만 기본적으로 개루프 제어이기 때문에 실시간 외란을 제거할 수 없다. MPC는 이전 회분의 정보를 이용하지 않기 때문에 모든 회분에서 동일한 성능을 보이며 모델의 정확도에 크게 의존한다. 본 논문에서 ILC와 MPC의 모든 장점을 포함하는 ILMPC를 제안한다. 많은 회분식 또는 반복 공정에서 출력은 모든 시간에서의 기준궤적을 추적할 필요가 없다. 따라서 본 논문에서는 원하는 점에만 수렴할 수 있는 새로운 ILMPC 기법을 제안한다. 제안한 기법을 사용할 경우 원하는 점을 지나는 기준궤적을 만드는 과정이 필요 없게 된다. 또한 본 논문은 점대점 추적, 반복 학습, 제약조건, 실시간 외란 제거 등의 성능을 보이기 위한 다양한 예제를 제공한다.

주요어 : 반복학습제어, 모델예측제어, 점대점 추적, 실시간 외란 제거

학번 : 2013-30282