creative commons
COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

공학박사 학위논문

Computationally Efficient Multilinear Model-based Control Combined with Data-driven Trajectory Optimization

데이터 기반 궤적 최적화를 결합한 계산 효율적인 다중선형 모델 기반 제어

2021년 2월

서울대학교 대학원
화학생물공학부
박 병 준

Computationally Efficient Multilinear Model-based Control Combined with Data-driven Trajectory Optimization

지도교수 이 종 민

이 논문을 공학박사 학위논문으로 제출함

2020년 12월

서울대학교 대학원

화학생물공학부

박 병 준

박 병 준의 공학박사 학위논문을 인준함

2020년 12월

위 원 장 이 원 보 (인)

부위원장 이 종 민

위 원 남 재 욱 (인)

위 원 정 동 휘

위 원 김 연 수

Abstract

Computationally Efficient Multilinear Model-based Control Combined with Data-driven Trajectory Optimization

Byungjun Park
School of Chemical and Biological Engineering
The Graduate School
Seoul National University

Model predictive control (MPC) is a widely used advanced control strategy applied in the process industry due to its capability to handle multivariate systems and constraints. When applied to nonlinear processes, linear MPC (LMPC) is limited to a relatively small operating region. On the other hand, nonlinear MPC (NMPC) is challenging due to the need for a nonlinear model with a large domain of validity and the computational load to solve nonlinear optimization problems. Multilinear MPC (MLMPC) or linear time-varying MPC (LTVMPC) complements the limitations, employing multiple linear models to predict dynamic behavior in a wide operating range. However, the main issue is obtaining the linear models, which is difficult to obtain without the nonlinear model and a trajectory from an initial condition to a set-point. Differential dynamic programming (DDP) can help to get the linear models and the suboptimal trajectory simultaneously. DDP iteratively improves the trajectory with the linear

models of the previous trajectory, which can be identified by excitation of input around the trajectory.

We propose four novel methodologies in the thesis. First, we propose a scheme to design MLMPC based on gap metric, which achieves convergence to LMPC and offset-free tracking. Second, we propose a switching strategy of MLMPC. It consists of a design of the subregions from an initial point to a set-point and LMPC for each subregion. Next, we develop a scheme that combines constrained differential dynamic programming (CDDP) and MLMPC, starting without any models. Finally, we developed an algorithm that combines LTVMPC and LMPC based on the models from CDDP. It exploits the suboptimal trajectory from CDDP and achieves offset-free tracking. We apply developed MPC algorithms to an illustrative example for validation. It also supports that multiple linear models are appropriate to control nonlinear processes with or without the nonlinear models.

Keywords: Optimal Control, Dynamic Optimization, Differential Dynamic Programming, Model Predictive Control
Student Number: 2015-21061

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and previous work

Model predictive control (MPC) is a kind of advanced process control (APC) whose control input is the solution of an optimization problem, considering system dynamics and constraints [1]. MPC has been widely used in many processes since the computational power was sufficient in the 1980s. There are two types of MPC, linear and nonlinear MPC. Nonlinear MPC (NMPC) solves optimization problems using the nonlinear model of a process. As many processes are nonlinear and have a wide operating range, NMPC is an appropriate controller. The disadvantage is that the computational time can be larger than the sampling time if the model is highly nonlinear and the number of decision variables and constraints are large. Linear MPC, which uses approximate linear models, can be a solution because the optimization problem is convex, which is solved much faster than nonlinear optimization problems. Unknown disturbance term or integral action is augmented to compensate model-plant mismatch and achieve offset-free tracking [1, 2]. However, a linear model cannot control nonlinear processes with a wide operating range because a linear model is only valid around a linearized point of the process,

i.e., the origin of the model.

To resolve this issue, there are two approaches of linear MPC using a set of linear models. The first approach is Multilinear model predictive control (MLMPC), where several linear models are combined to predict a nonlinear process [3, 4, 5]. MLMPC consists of a set of local linear MPC controllers, where each controller predicts the dynamic behavior using a distinct linear model in the set [6, 4, 7]. MLMPC receives the optimal input of each local MPC at each sampling time. Then it gives convex combinations of the inputs as the control action.

Two methods have been proposed to determine the weights of the local optimal inputs. The prediction-based method determines the weights based on the output prediction error of the local models at each sampling time [6, 8, 4]. Prediction-based MLMPC has been applied to various processes [9, 10, 11, 12, 13]. The second method calculates the weights based on the gap metric between the model and local dynamics of the process. A gap metric between two systems is a measure of the similarity in terms of the stability of the closed-loop systems where each system employs the same controller. If the controller stabilizes one of the two systems, the other is also stabilized by the controller if the gap metric between the systems is small [14]. Hence, the gap metric can help to choose a model for stabilization at an operating condition. Gap metric-based MLMPC has been proposed for nonlinear systems with wide operating ranges [7, 15]. One limitation of this method is that the dynamics at the state have to be known at each time, which is hard to satisfy in practical operation. In addition, MLMPC is vulnerable to an oscillation of inputs caused by the oscillation of the weights.

The other approach is linear time-vary MPC (LTVMPC), which predicts the dynamic behavior using the linear time-varying (LTV) model. Because it uses one linear model for each prediction step, it is free for oscillation compared to MLMPC, and its computational burden decreases compared to NMPC. Many application employs LTVMPC solving the above practical issues [16, 17, 18, 19, 20]. LTVMPC needs the reference trajectory and the LTV model that describes the dynamics around the trajectory. Thus, finding a suboptimal or optimal trajectory and identifying the LTV model around the trajectory are necessary, which is not easy without the nonlinear model of the process.

Trajectory optimization techniques can help to obtain suboptimal trajectories for a process with or without the nonlinear model. These are classified into shooting methods [21, 22], collocation methods [23, 24], and differential dynamic programming (DDP) [25, 26]. The first two methods require the nonlinear model, and the last method only exploits the first-order derivatives, i.e., local linear model, to optimize the trajectory. Thus, DDP is attractive if there is no knowledge about the nonlinear model of the process. DDP consists of the backward and forward passes. Backward pass reduces the optimization problem to convex one and solves it at each time step to get a new control sequence based on the nominal trajectory. The forward pass applies the control sequence to the plant and gets a new nominal trajectory. It iteratively improves the trajectory until it converges. and has convergence properties [27]. Constrained DDP (CDDP) has been proposed to consider constraints on the processes [28, 29]. Most recent works propose DDP with box input constraints [30] and nonlinear constraints [31]. Because a suboptimal trajectory and LTV model

can be obtained from DDP, LTVMPC can be easily applied if DDP precedes. However, both DDP and LTVMPC track the nominal trajectory instead of the set-point. It cannot track the set-point until the trajectory converges.

1.2 Statement of contributions

The main objective of this thesis is to develop the methods for tracking various steady-state set-points in a wide operating range of nonlinear processes. Multiple linear models are exploited to describe the dynamic behavior in the operating range. Trajectory optimization and identifying the linear models are simultaneously conducted without models. We validate the proposed methods through an illustrative example. The summary of the four chapters are below:

- A MLMPC design for tracking various set-points in a wide operating range of a nonlinear system.
- A switching MLMPC design for tracking various set-points in a wide operating range of a nonlinear system.
- A framework based on CDDP and MLMPC to track a set-point of a nonlinear system.
- A scheme combined with CDDP, LTVMPC, and LMPC for optimal control of a nonlinear system.

The first work is to design MLMPC to track various set-points in a wide operating range of a nonlinear system. To provide local models that approximate the nonlinear behavior in an operating range,

generalized gridding and K-medoids clustering algorithms are proposed based on gap metric. LMPC for each cluster is designed, each of which stabilizes the system at any equilibrium points in the corresponding cluster. We also propose to combine the prediction-based and gap metric-based approaches to determine the weights for the LMPC controllers. The overall method is applied to an example of a nonlinear continuous stirred tank reactor (CSTR). The proposed MLMPC shows superior to previous MLMPC methods in terms of set-point tracking, disturbance rejection, and stability.

The second part solves the same problem in the first part in a different way. We construct a graph of the equilibrium points adjacent to other clusters and find the optimal path from an initial point to a set-point by solving the shortest path problem. The cost includes the gap metric between any two equilibrium points in the graph. Exploiting the shortest path problem, a switching MLMPC is proposed, in which an LMPC controller steers from a node of the path to the next node. The overall method is applied to an example of a nonlinear CSTR. As a result, the proposed MLMPCs achieve offset-free tracking with and without disturbance for any pairs of initial points and set-points in the operating region.

The third part deals with tracking a set-point from an initial condition without models. It can happen if there are no models, or equilibrium points are not be obtained by model inversion for the nonlinear process. We propose a CDDP algorithm that considers input constraints when the nonlinear dynamics and nonzero steady-state input for a set-point are unknown. Obtaining the linear models based on conditional Gaussian distribution is included. To choose the models for MLMPC among the models obtained by CDDP, clustering

of the models based on sampling time and gap metric is proposed. Then a prediction weighting method for MLMPC is proposed, which converges to a linear model as the state is close to the set-point. A nonlinear CSTR is studied to demonstrate the effectiveness of the proposed scheme. Simulation studies show that the CDDP designed by the proposed algorithm improves the trajectory over iterations, and the resulting MLMPC achieves offset-free tracking property regardless of an initial point and a set-point in the operating region.

The final part solves the same problem in the third part in a different way. The suboptimal trajectories to track a set-point from CDDP are exploited to design an MPC controller. Concretely, dividing a suboptimal trajectory from CDDP in the transient and steady-state region is proposed. Selecting the linear model in the steady-state region based on gap metric is also proposed. Subsequently, LTVMPC is employed to track the suboptimal trajectory in the transient region. Then LMPC achieves offset-free tracking starting from the state close to the set-point. We prove that the feasibility of the proposed LTVMPC in the transient region around the nominal trajectory, and offset-free tracking property if the gap metric between the selected linear model and the dynamics at the set-point is small. Simulation studies through CSTR show that CDDP provides improved trajectories over iterations, and the proposed MPC achieves offset-free tracking and disturbance rejection regardless of an initial point and a set-point in the operating region.

In summary, All the works deal with the infinite-horizon regulatory tracking control problem, where the third and final part deal with the problem starting without the models, helped by solving the finite-horizon optimal control problem (FHOC).

1.3 Outline of the thesis

The remainder of the thesis is organized as follows. In Chapter 2, the background on the formulation of MPC and DDP is provided. The gap metric and related concepts are also introduced. In Chapter 3, MLMPC design based on gap metric for set-point tracking in a wide operating range of a nonlinear system is provided. The same problem is solved by designing a switching MLMPC followed by the path design based on gap metric in Chapter 4. Chapter 5 provides a data-driven MLMPC design using CDDP to solve the problem, starting without the models. A scheme combined with CDDP, LTVMPC, and LMPC for optimal control of the nonlinear system is given in Chapter 6. Finally, general concluding remarks and possible directions for further study are given in Chapter 7.

Chapter 2

Background and preliminaries

2.1 Offset-free linear model predictive control

Consider a discrete-time nonlinear system,

$$\begin{aligned} x_{k+1} =& f(x_k, u_k), \\ y_k =& g(x_k), \\ z_k =& H y_k, \end{aligned} \tag{2.1}$$

with constraints

$$\begin{aligned} u &\in \mathcal{U} \\ x &\in \mathcal{X} \end{aligned} \tag{2.2}$$

where f and g are continuously differentiable functions, $x_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$, and $y_k \in \mathbb{R}^p$ is the state, input, and measured output vector of the system at k^{th} time step. The controlled variables $z_k \in \mathbb{R}^r$ are a linear combination of the measured variables for which offset-free behavior is sought. $\mathcal{U}$ and $\mathcal{X}$ are constraint sets presented as compact polyhedral region. Suppose that a linear model which approximates

the process (2.1) around (x_o, u_o, y_o) is given.

$$\begin{aligned} x^o(k+1) &= Ax^o(k) + Bu^o(k), \\ y^o(k) &= Cx^o(k), \end{aligned} \tag{2.3}$$

where $x^o(k) := x_k - x_o$, $u^o(k) := u_k - u_o$, $y^o(k) := y_k - y_o$. In order to capture the mismatch between (2.1) and (2.3) in steady state, the disturbance model is augmented to the linear model [1].

$$\begin{aligned} \begin{bmatrix} x^o(k+1) \\ d^o(k+1) \end{bmatrix} &= \underbrace{\begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix}}_{A_a} \underbrace{\begin{bmatrix} x^o(k) \\ d^o(k) \end{bmatrix}}_{x_a^o(k)} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{B_a} u^o(k), \\ y^o(k) &= \underbrace{\begin{bmatrix} C & C_d \end{bmatrix}}_{C_a} \begin{bmatrix} x^o(k) \\ d^o(k) \end{bmatrix}, \end{aligned} \tag{2.4}$$

where $d^o \in \mathbb{R}^{n_d}$. B_d and C_d is design to satisfying the conditions for the observability of (2.4), which are given in the following theorem.

Theorem 2.1. *[32] The augmented system* (2.4) *is observable if and only if* (C, A) *is observable and*

$$\begin{bmatrix} A - I & B_d \\ C & C_d \end{bmatrix} \tag{2.5}$$

has full column rank.

The augmented state estimator is designed as follows:

$$\hat{x}_a^o(k+1) = A_a\hat{x}_a^o(k) + B_a u^o(k) + \begin{bmatrix} L_x \\ L_d \end{bmatrix} (C_a\hat{x}_a^o(k) - y^o(k)), \tag{2.6}$$

where $L_x \in \mathbb{R}^{n\times p}$ and $L_d \in \mathbb{R}^{n_d\times p}$ are estimator gains for state and disturbance, respectively, chosen to make the estimator stable. The following lemma is given for the observer (2.6) where $n_d = p$.

Lemma 2.1. *[32] Suppose the observer* (2.6) *is stable and* $n_d = p$. *Then the steady state of the observer* (2.6) *satisfies*

$$\begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} \hat{x}_\infty^o \\ u_\infty^o \end{bmatrix} = \begin{bmatrix} -B_d\hat{d}_\infty^o \\ y_\infty^o - C_d\hat{d}_\infty^o \end{bmatrix} \tag{2.7}$$

where $y_\infty^o := y_\infty - y_o$ *and* $u_\infty^o := u_\infty - u_o$, y_∞ *and* u_∞ *are the steady state measured output and input of the system* (2.1), $\hat{x}_\infty^o$ *and* $\hat{d}_\infty^o$ *are the state and disturbance estimates from the observer* (2.6) *at steady state, respectively.*

Thus, there is no offset between the measured output and the output of the augmented model if the system (2.1) and the observer (2.6) are in a steady state.

The objective of offset-free linear MPC is to make the controlled variables z track the reference signal r which is assumed to converge to a constant, i.e., $r_k \to r_\infty$ as $k \to \infty$, with the linear time-invariant model (2.3) and the observer (2.6). The observer condition (2.7) sug-

gests that at steady state the MPC should satisfy

$$\begin{bmatrix} A-I & B \\ HC & 0 \end{bmatrix} \begin{bmatrix} x_\infty^o \\ u_\infty^o \end{bmatrix} = \begin{bmatrix} -B_d \hat{d}_\infty^o \\ r_\infty^o - HC_d \hat{d}_\infty^o \end{bmatrix} \tag{2.8}$$

where x_∞^o is the MPC state at steady-state and $r_\infty^o := r_\infty - Hy_o$. For x_∞^o and u_∞^o to exist for any $\hat{d}_\infty^o$ and r_∞^o, the matrix in left hand side of (2.8) must has full row rank which implies $m \geq r$. The offset-free MPC controller is designed by

$$\begin{aligned} \min_{u_0,\cdots,u_{p-1}} & \sum_{j=0}^{p-1} \|x_j - \bar{x}(k)\|_Q^2 + \|u_j - \bar{u}(k)\|_R^2 + \|x_p - \bar{x}(k)\|_{Q_T}^2 \\ s.t.\ & x_{j+1} = Ax_j + Bu_j + B_d d_j \\ & d_{j+1} = d_j \\ & y_j = Cx_j \\ & u_j \in \mathcal{U}, x_j \in \mathcal{X}, x_p \in \mathcal{X}_f \\ & x_0 = \hat{x}^o(k),\ d_0 = \hat{d}^o(k) \\ & j = 0, ..., p \end{aligned} \tag{2.9}$$

where $Q \succeq 0, R \succ 0$, and $Q_T \succ 0$ are the weighting matrices for the state, the input, and the terminal state, respectively. Q_T is the solution of discrete-time algebraic Riccati equation (DARE), which makes MPC equivalent to LQR in the unconstrained region. $\mathcal{X}_f$ is the terminal constraint to satisfy the recursive feasibility and the stability of MPC [1]. The target state $\bar{x}(k)$ and the target input $\bar{u}(k)$ at k^{th} time

step are obtained by solving

$$\begin{bmatrix} A-I & B \\ HC & 0 \end{bmatrix} \begin{bmatrix} \bar{x}(k) \\ \bar{u}(k) \end{bmatrix} = \begin{bmatrix} -B_d \hat{d}^o(k) \\ r^o(k) - HC_d \hat{d}^o(k) \end{bmatrix}, \tag{2.10}$$

where $r^o(k) := r_k - Hy_o \in \mathbb{R}^r$. Let $U^*(k) = \{u_0^*, \cdots, u_{p-1}^*\}$ be the optimal solution of (2.9) and (2.10) at time k. Then the first sample of $U^*(k)$ is applied to the system (2.1)

$$u_k = u_0^* + u_o \tag{2.11}$$

Often in practice, one desires to track all measured outputs with zero offsets. Thus, we assume $n_d = p = r$ in the thesis. The following theorem is provided for offset-free control when $n_d = p$.

Theorem 2.2. *[1] Consider the case $n_d = p$. Assume that for $r_k \to r_\infty$ as $k \to \infty$, the MPC problem* (2.9) *and* (2.6) *is feasible for all $k \in \mathcal{N}_+$, unconstrained for $k \geq j$ with $j \in \mathcal{N}_+$ and the closed-loop system* (2.1)*,* (2.2)*,* (2.6)*,* (2.9)*, and* (2.10) *converges to $\hat{x}_\infty^o$, $\hat{d}_\infty^o$, y_∞^o. Then $z_k \to r_\infty$ as $k \to \infty$.*

2.2 Gap metric and stability margin

In the previous section, we have seen that a linear model can be exploited for tracking a steady-state set-point even if there is a mismatch between the plant and the model. In this section, the gap metric is introduced as being appropriate for the study of uncertainty in feedback systems, which can be applied to study the model-plant mismatch of the closed-loop system controlled by linear MPC.

Before we provide the concept of the gap metric, we introduce some concepts to define the gap metric.

Definition 2.1. *A Banach space is a real or complex normed vector space that is also a complete metric space with respect to the distance function induced by the norm.*

Definition 2.2. *A Hilbert space is a real or complex inner product space that is also a complete metric space with respect to the distance function induced by the inner product.*

If a vector space is a Hilbert space, it is a Banach space. One of the most familiar examples of a Hilbert space is the Euclidean vector space consisting of n-dimensional vectors, denoted by $\mathbb{R}^n$, and equipped with the dot product. More generally, $\mathbb{C}^{n\times m}$ with the inner product defined as

$$\Big\langle A, B \Big\rangle := \text{trace} A^* B = \sum_{i=1}^{n}\sum_{j=1}^{m} \bar{a}_{ij} b_{ij}, \forall A, B \in \mathbb{C}^{n\times m} \tag{2.12}$$

We define a Hilbert space as follows:

Definition 2.3. *$\mathcal{L}_2[a, b]$ is an infinite-dimensional Hilbert space that consists of all square-integrable and Lebesgue measurable functions defined on an interval $[a, b]$ with the inner product and the induced norm defined as*

$$\Big\langle f, g \Big\rangle := \int_a^b f(t)^* g(t) dt \tag{2.13}$$

$$\|f\|_2 := \sqrt{\Big\langle f, f \Big\rangle} < \infty \tag{2.14}$$

for $f, g \in \mathcal{L}_2[a, b]$.

The following Hilbert space is defined for the matrix-valued functions on $\mathbb{R}$.

Definition 2.4. $\mathcal{L}_2(\mathbb{R})$ *or* $\mathcal{L}_2(-\infty, \infty)$ *is an infinite-dimensional Hilbert space which consists of all square-integrable, Lebesgue measurable, and matrix-valued functions defined on an interval* $\mathbb{R}$ *with the inner product and the induced norm defined as*

$$\left\langle f, g \right\rangle := \int_{-\infty}^{\infty} \mathit{trace}[f(t)^* g(t)] dt \tag{2.15}$$

$$\|f\|_2 := \sqrt{\left\langle f, f \right\rangle} < \infty \tag{2.16}$$

for $f, g \in \mathcal{L}_2(\mathbb{R})$.

$\mathcal{L}_{2+} = \mathcal{L}_2[0, \infty)$ and $\mathcal{L}_{2-} = \mathcal{L}_2(-\infty, 0]$ are defined similarly.

$\mathcal{L}_{2+}$: subspace of $\mathcal{L}_2(-\infty, \infty)$ with functions zero for $t < 0$. $\mathcal{L}_{2-}$: subspace of $\mathcal{L}_2(-\infty, \infty)$ with functions zero for $t > 0$.

A Hilbert space of matrix-valued functions on $j\mathbb{R}$ is also defined.

Definition 2.5. $\mathcal{L}_2(j\mathbb{R})$ *or* $\mathcal{L}_2$ *is an infinite-dimensional Hilbert space that consists of all square-integrable, Lebesgue measurable, and matrix-valued functions defined on an interval* $\mathbb{R}$ *with the inner product and the induced norm defined as*

$$\left\langle f, g \right\rangle := \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathit{trace}[f(j\omega)^* g(j\omega)] d\omega \tag{2.17}$$

$$\|f\|_2 := \sqrt{\left\langle f, f \right\rangle} < \infty \tag{2.18}$$

for $f, g \in \mathcal{L}_2(j\mathbb{R})$.

Definition 2.6. *$\mathcal{H}_2$ is a closed subspace of $\mathcal{L}_2(j\mathbb{R})$ with analytic matrix-valued function $F(s)$ in $Re(s) > 0$ (open right-half plane). The corresponding norm is defined as*

$$\|F\|^2 := \sup_{\sigma>0} \frac{1}{2\pi} \int_{-\infty}^{\infty} \textit{trace}[F(\sigma + j\omega)^* F(\sigma + j\omega)] d\omega \qquad (2.19)$$

It can be shown [33] that

$$\|F\|^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} \text{trace}[F(j\omega)^* F(j\omega)] d\omega \qquad (2.20)$$

Hence, we can compute the norm for $\mathcal{H}_2$ just as we do for $\mathcal{L}_2(j\mathbb{R})$. The real rational subspace of $\mathcal{H}_2$, which consists of all strictly proper and real rational stable transfer matrices, is denoted by $\mathcal{RH}_2$. The state-space representation can be applied for the transfer matrices in $\mathcal{RH}_2$. The $\mathcal{L}_2(j\mathbb{R})$ in the frequency domain can be related to the $\mathcal{L}_2(\mathbb{R})$ defined in the time domain. It can be shown that Laplace and inverse transform yield an isometric isomorphism between $\mathcal{L}_2(\mathbb{R})$ and $\mathcal{L}_2(j\mathbb{R})$. In addition, there is an isometric isomorphism between $\mathcal{L}_2[0, \infty)$ and $\mathcal{H}_2$.

Other classes of important complex matrix functions used in this book are those bounded on the imaginary axis.

Definition 2.7. *$\mathcal{L}_\infty(j\mathbb{R})$ or $\mathcal{L}_\infty$ is Banach space of matrix-valued (or scalar-valued) functions that are (essentially) bounded on $j\mathbb{R}$, with norm*

$$\|F\|_\infty := \textit{ess} \sup_{\omega \in \mathbb{R}} \bar{\sigma}[F(j\omega)] \qquad (2.21)$$

The rational subspace of $\mathcal{L}_\infty$, denoted by $\mathcal{RL}_\infty(j\mathbb{R})$ or simply

$\mathcal{RL}_\infty$, consists of all proper and real rational transfer matrices with no poles on the imaginary axis.

Definition 2.8. *$\mathcal{H}_\infty$ is a (closed) subspace of $\mathcal{L}_\infty$ with functions that are analytic and bounded in the open right-half plane. The $\mathcal{H}_\infty$ norm is defined as*

$$\|F\|_\infty := \sup_{Re(s)>0} \bar{\sigma}[F(s)] = \sup_{\omega\in\mathbb{R}} \bar{\sigma}[F(j\omega)] \tag{2.22}$$

The real rational subspace of $\mathcal{H}_\infty$ is denoted by $\mathcal{RH}_\infty$, which consists of all proper and real rational stable transfer matrices, for which the state-space representation can be applied.

The concept of the coprimeness between two transer functions is given by the following definition

Definition 2.9. *Two matrices M and N in $\mathcal{RH}_\infty$ are right coprime over $\mathcal{RH}_\infty$ if they have the same number of columns and if there exist matrices X_r and Y_r in $\mathcal{RH}_\infty$ such that*

$$\begin{bmatrix} X_r & Y_r \end{bmatrix} \begin{bmatrix} M \\ N \end{bmatrix} = X_r M + Y_r N = I \tag{2.23}$$

Similarly, two matrices $\tilde{M}$ and $\tilde{N}$ in $\mathcal{RH}_\infty$ are left coprime over $\mathcal{RH}_\infty$ if they have the same number of rows and if there exist matrices X_l and Y_l in $\mathcal{RH}_\infty$ such that

$$\begin{bmatrix} \tilde{M} & \tilde{N} \end{bmatrix} \begin{bmatrix} X_l \\ Y_l \end{bmatrix} = \tilde{M} X_l + \tilde{N} Y_l = I \tag{2.24}$$

Now let P be a proper real rational matrix. A right coprime fac-

torization (rcf) of P is a factorization $P = NM^{-1}$, where N and M are right coprime over $\mathcal{RH}_\infty$. It is called a normalized right coprime factorization if $M^*M + N^*N = I$, where M^* denotes the complex conjugate of M. Similarly, a left coprime factorization (lcf) has the form $P = \tilde{M}^{-1}\tilde{N}$, where $\tilde{N}$ and $\tilde{M}$ are left-coprime over $\mathcal{RH}_\infty$. It is called a normalized left coprime factorization if $MM^* + NN^* = I$. The graph of P is defined as the subspace of $\mathcal{H}_2$ consisting of all pairs (u, y) such that $y = Pu$, and denoted by

$$\begin{bmatrix} M \\ N \end{bmatrix} \mathcal{H}_2 \tag{2.25}$$

Now we can define a gap between two systems.Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be $p \times m$ rational transfer matrices, where p and m are the dimension of input and output, respectively. Let $\mathcal{P}_1$ and $\mathcal{P}_2$ have the following normalized right coprime factorizations:

$$\mathcal{P}_1 = N_1 M_1^{-1}, M_1^* M_1 + N_1^* N_1 = I,$$
$$\mathcal{P}_2 = N_2 M_2^{-1}, M_2^* M_2 + N_2^* N_2 = I$$

The gap between $\mathcal{P}_1$ and $\mathcal{P}_2$ is defined by

$$\delta_g(\mathcal{P}_1, \mathcal{P}_2) = \left\| \Pi \begin{bmatrix} M_1 \\ N_1 \end{bmatrix}_{\mathcal{H}_2} - \Pi \begin{bmatrix} M_2 \\ N_2 \end{bmatrix}_{\mathcal{H}_2} \right\| \tag{2.26}$$

and can be computed as [34]

$$\delta_g(\mathcal{P}_1, \mathcal{P}_2) = \max\left\{ \overrightarrow{\delta}_g(\mathcal{P}_1, \mathcal{P}_2), \overrightarrow{\delta}_g(\mathcal{P}_2, \mathcal{P}_1) \right\}, \tag{2.27}$$

where $\overrightarrow{\delta}_g(\mathcal{P}_1, \mathcal{P}_2)$ is the directed gap, and it is obtained by

$$\overrightarrow{\delta}_g(\mathcal{P}_1, \mathcal{P}_2) = \inf_{Q \in \mathcal{H}_\infty} \left\| \begin{bmatrix} M_1 \\ N_1 \end{bmatrix} - \begin{bmatrix} M_2 \\ N_2 \end{bmatrix} Q \right\|_\infty. \tag{2.28}$$

The gap between two linear systems is the measure that determines if each system is stabilized by the same controller. The stability margin of a linear system $\mathcal{P}$ and its stabilizing controller K indicates robustness to unstructured perturbations of a closed-loop system. It is defined as [14]

$$b_{\mathcal{P},K} = \left\| \begin{bmatrix} I \\ K \end{bmatrix} (I + \mathcal{P}K)^{-1} \begin{bmatrix} I & \mathcal{P} \end{bmatrix} \right\|_\infty^{-1} \tag{2.29}$$

The following theorem can be exploited to design a feedback control system:

Theorem 2.3. *[35] Suppose the feedback system with the pair $(\mathcal{P}_0, K_0)$ is stable. Let $\mathbb{P} := \{\mathcal{P} : \delta_g(\mathcal{P}, \mathcal{P}_0) < r_1\}$ and $\mathbb{K} := \{K : \delta_g(K, K_0) < r_2\}$. Then*

(a) The feedback system with the pair $(\mathcal{P}, K)$ is also stable for all $\mathcal{P} \in \mathbb{P}$ and $K \in \mathbb{K}$ if and only if

$$\sin^{-1} b_{\mathcal{P}_0,K_0} \geq \sin^{-1} r_1 + \sin^{-1} r_2.$$

(b) The worst possible performance resulting from these sets of

plants and controllers is given by

$$\inf_{\mathcal{P}\in\mathbb{P},K\in\mathbb{K}} \sin^{-1} b_{\mathcal{P},K} = \sin^{-1} b_{\mathcal{P}_0,K_0} - \sin^{-1} r_1 - \sin^{-1} r_2$$

If the gap between two systems is close to zero, the controller K_0 can also stabilize $\mathcal{P}$. Thus, two linear systems whose gap is close to zero have one common feedback controller stabilizing both systems.

Remark 2.1. *The metric induced by the operator norm cannot measure the distance between two unstable systems that can be stabilized in the closed-loop. The gap metric can deal with the problem. Other metrics such as $\nu-$gap metric, graph metric, chordal metric can also resolve this problem [36, 37, 38].*

2.3 Multilinear model predictive control

If a linear model cannot describe the dynamic behavior of the process (2.1) in the wide operating range, a set of linear models can be exploited simultaneously to predict the behavior. Consider the system (2.1) and constraints (2.2). Suppose that a set of n_M linear models which approximate the process (2.1) in an operating range are given as

$$\begin{aligned} x_i(k+1) &= A_i x_i(k) + B_i u_i(k), \\ y_i(k) &= C_i x_i(k), \end{aligned} \tag{2.30}$$

where $x_i(k) = x_k - x_{oi}, u_i(k) = u_k - u_{oi},$ and $y_i(k) = y_k - y_{oi}$ are deviation variables for the state, input, and, measured output. $A_i, B_i,$ and C_i are the matrices for the i^{th} model that approximate

the dynamic behavior of (2.1) around (x_{oi}, u_{oi}). The i^{th} linear MPC controller at time k is designed by

$$\begin{aligned} \min_{u_0,\cdots,u_{p-1}} & \sum_{j=0}^{p-1} \|r_j - z_j\|^2_{Q_i} + \|u_j\|^2_{R_i} \\ s.t.\ & x_{j+1} = A_i x_j + B_i u_j \\ & y_j = C_i x_j \\ & z_j = H y_j \\ & u_j + u_{oi} \in \mathcal{U}, x_j + x_{oi} \in \mathcal{X} \\ & x_0 = \hat{x}_i(k),\ r_j = r_i(k+j) \\ & j = 0, ..., p \end{aligned} \tag{2.31}$$

where where Q_i and R_i are the weighting matrices of the i^{th} linear MPC for the state and input, respectively. $\hat{x}_i(k)$ is the estimated state of the i^{th} model at time k, and $r_i(k) = r_k - H y_{oi}$ is the deviation variable for the reference. Let $U_i^*(k) = \{u_{i,0}^*, \cdots, u_{i,p-1}^*\}$ be the solution of (2.31) of the i^{th} linear MPC at the time k, and let $u_i^*(k)$ be the first element of $U_i^*(k)$. Then the input at the time k is determined by

$$u_k = \sum_{i=1}^{n_M} \phi_i(k) u_i^*(k) \tag{2.32}$$

where ϕ_i is the normalized weights satisfying $\sum_{i=1}^{n_M} \phi_i(k) = 1$.

There are two approaches to combine the controllers, the switching and weighting methods [39]. In the switching method, one local linear MPC controller is chosen in the set of local MPCs by a user-defined criterion and applied to the plant at each sample time. The switching method can cause chattering in systems with strong non-

linearities [5]. Meanwhile, the weighting method gives a weighted sum of the inputs from several local linear MPC controllers. This approach can operate the system smoothly unlike the switching method due to the gradual changes of the weights [7].

Selecting the weights is a key step in the weighting method. There are two popular methods to select the weights. First, the gap metric-based weighting method calculates the weights of the local models based on the gap between a model and the linearized system at a point [7]. Generally, the state at the sampling time or the set-point is chosen. Let $s_k = (x_{ok}, u_{ok}, y_{ok})$ be an equilibrium point chosen at time k. The linearized model of (2.1) around (x_{ok}, u_{ok}, y_{ok}) is denoted by $P(k)$, and P_i denotes the i^{th} model (2.30). Then the gap-metric based weight of the i^{th} linear MPC controller at time k is defined as

$$\phi_i(k) = \frac{(1 - \gamma_i(\theta_k))^{k_e}}{\sum_{j=1}^{n_M}(1 - \gamma_j(\theta_k))^{k_e}}, \tag{2.33}$$

where $\gamma_i(\theta_k) = \delta_g(P_i, P(k))$ is the gap between P_i and $P(k)$, and $k_e \geq 1$ is a tuning parameter.

The second method computes the weights using the predicted output error of each model in the models [4]. Let the residual of the i^{th} local model at time instant k as

$$\epsilon_i(k) = y_i(k) - \hat{y}_i(k), \tag{2.34}$$

where $\hat{y}_i(k)$ is the estimated output from the i^{th} local model (2.30) and its estimator. The weights are defined based on Bayesian ap-

proach:

$$\phi_i(k) = \frac{\exp(-\frac{1}{2}\epsilon_i(k)^T \Lambda^i \epsilon_i(k))\phi_i(k-1)}{\sum_{j=1}^{n_m} \exp(-\frac{1}{2}\epsilon_j(k)^T \Lambda^i \epsilon_j(k))\phi_j(k-1)}, \tag{2.35}$$

where Λ^i is the diagonal scaling matrix for the residuals. For normalizing the output, Λ^i is a set inversely proportional to the output in general. Once the weight of a model reaches zero, it stays zero until the end of operation as the weight is proportional to the previous weight. In order to resolve this problem, an artificial lower limit on the probability is imposed.

$$\begin{aligned}
p_i(k) &= \frac{\exp(-\frac{1}{2}\epsilon_i(k)^T \Lambda^i \epsilon_i(k))\varphi_i(k-1)}{\sum_{j=1}^{n_m} \exp(-\frac{1}{2}\epsilon_j(k)^T \Lambda^i \epsilon_j(k))\varphi_j(k-1)}, \\
\varphi_i(k) &= \begin{cases} p_i(k), & p_i(k) > \mu, \\ \mu, & p_i(k) \leq \mu, \end{cases} \\
\phi_i(k) &= \begin{cases} p_i(k) / \left(\sum_{p_i(k)>\mu, 1 \leq i \leq n_m} p_i(k) \right), & p_i(k) > \mu, \\ 0, & p_i(k) \leq \mu, \end{cases}
\end{aligned} \tag{2.36}$$

where μ is the lower limit, $p_i(k)$ is the i^{th} weight by Bayesian approach, $\varphi_i(k)$ is the i^{th} weight that prevents the zero probability, and $\phi_i(k)$ is the weight applied to the i^{th} controller output.

2.4 Linear time-varying model predictive control

If a trajectory for the system (2.1) and the approximating linear model at each sampling time are provided, LTV model can be constructed as a set of the models, and LTVMPC can be exploited to track

the trajectory. Suppose that a state and input trajectory from time 0 to N denoted by $\bar{\mathbf{X}}_0 = \{\bar{x}_0, \cdots, \bar{x}_N\}$ and $\bar{\mathbf{U}}_0 = \{\bar{u}_0, \cdots, \bar{u}_{N-1}\}$ are given. Assume that a LTV models is obtained from linearizing the nonlinear model (2.1) or system identification and expressed by

$$\begin{aligned} x^v(k+1) &= A_k x^v(k) + B_k u^v(k), \\ y^v(k) &= C_k x^v(k), \end{aligned} \tag{2.37}$$

where $x^v(k) := x_k - \bar{x}_k$, $u^v(k) := u_k - \bar{u}_k$, and $y^v(k) := y_k - \bar{y}_k$ are the deviation variables for the state, input, and measured output. (A_k, B_k, C_k) is the LTV model at the k^{th} time step. Then the LTVMPC controller to track the trajectory at the k^{th} time step is designed by

$$\begin{aligned} \min_{u_0^v, \cdots, u_{p-1}^v} & \sum_{j=0}^{p-1} \left\| y_j^v \right\|_Q^2 + \left\| u_j^v \right\|_R^2 + \left\| x_p^v \right\|_{P_{k+p}}^2 \\ s.t.\ & x_{j+1}^v = A_{k+j} x_j^v + B_{k+j} u_j^v,\ j = 0, ..., N-1 \\ & y_j^v = C_{k+j} x_j^v,\ j = 0, ..., N \\ & u_j^v + \bar{u}_{k+j} \in \mathcal{U}, x_j^v + \bar{x}_{k+j} \in \mathcal{X}, \\ & x_p^v \in \mathcal{X}_{k+p}^f \\ & x_0 = \hat{x}^v(k), \end{aligned} \tag{2.38}$$

where $Q \succeq 0$ and $R \succ 0$ are the weighting matrices for the state and the input, respectively. P_k is computed by solving the following finite

horizon LQR problem for LTV systems.

$$
\begin{aligned}
(x_k^v)^T P_k x_k^v = \min_{u_k,\cdots,u_{N-1}} \sum_{j=k}^{N-1} \left\|y_j^v\right\|_Q^2 + \left\|u_j^v\right\|_R^2 + \|y_N^v\|_Q^2 \\
s.t.\ x_{j+1} = A_j x_j^v + B_j u_j^v, \\
j = k, ..., N-1
\end{aligned}
\tag{2.39}
$$

Induction backwards in time can be used to obtain P_k at each time.

$$
\begin{aligned}
P_N =& C_N^T Q C_N \\
K_i =& -\left(R + B_i^T C_{i+1}^T P_{i+1} C_{i+1} B_i\right)^{-1} B_i^T C_{i+1}^T P_{i+1} C_{i+1} A_i \\
P_i =& C_{i+1}^T Q C_{i+1} + K_i^T R K_i + (A_i + B_i K_i)^T C_{i+1}^T P_{i+1} C_{i+1} (A_i + B_i K_i)
\end{aligned}
\tag{2.40}
$$

Then the LTVMPC (2.38) is equivalent to LQR for LTV systems (2.39) in the unconstrained region.

2.5 Differential dynamic programming

Consider the discrete-time nonlinear system (2.1) whose measured output is the state, i.e., the system is fully measured. The system is expressed as

$$
\begin{aligned}
x_{k+1} =& f(x_k, u_k), \\
y_k =& x_k, \\
z_k =& H y_k
\end{aligned}
\tag{2.41}
$$

Assuming that the final time step is $N \in \mathbb{N}$, the cost at k^{th} time step is defined by a sequence of states, $\mathbf{X}_k = \{x_k, \cdots, x_N\}$, and a sequence

of inputs, $\mathbf{U}_k = \{u_k, \cdots, u_{N-1}\}$,

$$J_k(\mathbf{X}_k, \mathbf{U}_k) = \sum_{j=k}^{N-1} l(x_j, u_j) + l^f(x_N), \tag{2.42}$$

where $l(x, u) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is the running cost and $l^f(x) : \mathbb{R}^n \to \mathbb{R}$ is the final cost. The optimal value function at k^{th} time step is defined by the optimal cost-to-go starting at a given x_k

$$V_k(x_k) = \min_{\mathbf{U}_k} J_k(\mathbf{X}_k, \mathbf{U}_k), \tag{2.43}$$

The optimal value function has the following recursive nature by Bellman's principle of optimality

$$\begin{aligned} V_k(x_k) &= \min_{u_k} l(x_k, u_k) + V_{k+1}(f(x_k, u_k)), \\ V_N(x_N) &= l^f(x_N) \end{aligned} \tag{2.44}$$

DDP is an iterative method to solve the optimal control problem (2.44). At each iteration, DDP performs a backward pass on the nominal trajectory to generate a new control sequence, and then a forward-pass to compute and evaluate a new nominal trajectory. Let Q_k be the variation of $l(x_k, u_k) + V_{k+1}(f(x_k, u_k))$ around the (x_k, u_k)

$$Q_k(\delta x_k, \delta u_k) = l(x_k + \delta x_k, u_k + \delta u_k) + V_{k+1}(f(x_k + \delta x_k, u_k + \delta u_k)) \tag{2.45}$$

In the backward pass, DDP optimizes (2.45) with respect to δu_k at each time step by approximating $Q_k(\delta x_k, \delta u_k)$ to the second order

around the nominal trajectory $(\mathbf{X}_0, \mathbf{U}_0)$.

$$Q_k(\delta x_k, \delta u_k) \approx \frac{1}{2} \begin{bmatrix} 1 \\ \delta x_k \\ \delta u_k \end{bmatrix}^T \begin{bmatrix} 2Q_k & Q_{x,k}^T & Q_{u,k}^T \\ Q_{x,k} & Q_{xx,k} & Q_{xu,k} \\ Q_{u,k} & Q_{ux,k} & Q_{uu,k} \end{bmatrix} \begin{bmatrix} 1 \\ \delta x_k \\ \delta u_k \end{bmatrix} \tag{2.46}$$

where the subscripts of $Q_{\cdot,k}$ denotes differentiation at (x_k, u_k) of the nominal trajectory in denominator layout. Dropping k for readability and denoting V_{k+1} by V', the expansion coefficients are

$$\begin{aligned} Q_x &= l_x + f_x^T V'_x, \\ Q_u &= l_u + f_u^T V'_x, \\ Q_{xx} &= l_{xx} + f_x^T V'_{xx} f_x + V'_x \cdot f_{xx}, \\ Q_{uu} &= l_{uu} + f_u^T V'_{xx} f_u + V'_x \cdot f_{uu}, \\ Q_{ux} &= l_{ux} + f_u^T V'_{xx} f_x + V'_x \cdot f_{ux} \end{aligned} \tag{2.47}$$

Then the value function around the nominal trajectory at k^{th} time step is obtained by optimizing the approximated $Q(\delta x, \delta u)$ over δu.

$$\delta u^* = \arg\min_{\delta u} Q(\delta x, \delta u) = -Q_{uu}^{-1}(Q_u + Q_{ux}\delta x) := K_g \delta x + K_c \tag{2.48}$$

Substituting (2.48) to (2.46), the approximated value function is updated as

$$V_k(x) = V_k(x_k) + \frac{1}{2}(x - x_k)^T V_{xx,k}(x - x_k) + V_{x,k}^T(x - x_k) + \Delta V_k \tag{2.49}$$

where x_k is the state of the nominal trajectory at the k^{th} time step, and

$$\begin{aligned}
V_{xx,k} =& Q_{xx,k} + K_{g,k}^T Q_{uu,k} K_{g,k} + Q_{ux,k}^T K_{g,k} + K_{g,k}^T Q_{ux,k}, \\
=& Q_{xx,k} - Q_{xu,k} Q_{uu,k}^{-1} Q_{ux,k}, \\
V_{x,k} =& Q_{x,k} + K_{g,k}^T Q_{uu,k} K_{c,k} + Q_{ux,k}^T K_{c,k} + K_{g,k}^T Q_{u,k}, \\
=& Q_{x,k} - Q_{xu,k} Q_{uu,k}^{-1} Q_{u,k}, \\
\Delta V_k =& - Q_{u,k}^T Q_{uu,k}^{-1} Q_{u,k} + \frac{1}{2} K_{c,k}^T Q_{uu,k} K_{c,k}, \\
=& - \frac{1}{2} Q_{u,k}^T Q_{uu,k}^{-1} Q_{u,k}
\end{aligned} \tag{2.50}$$

The hessian $V_{xx,k}$ and the gradient $V_{x,k}$ are then passed to the $(k-1)^{th}$ time step and the backward pass continues until $k \geq 0$, starting with $V_{xx,N} = l_{xx}^f$ and $V_{x,N} = l_x^f$.

In the forward pass, the nominal state-control trajectory is updated using the optimal linear feedback (2.48).

$$\begin{aligned}
u_k^{new} =& u_k + K_{g,k}(x_k^{new} - x_k) + K_{c,k}, \\
x_{k+1}^{new} =& f(x_k^{new}, u_k^{new}), \;\; x_0^{new} = x_0
\end{aligned} \tag{2.51}$$

Then the new nominal trajectories $\mathbf{X}_0^{new} = \{x_0^{new}, \cdots, x_N^{new}\}$ and $\mathbf{U}_0^{new} = \{u_0^{new}, \cdots, u_N^{new}\}$ replace $\mathbf{X}_0$ and $\mathbf{U}_0$, and the backward pass at the next iteration starts with the updated state-control nominal trajectory if it does not converge.

Chapter 3

Offset-free multilinear model predictive control based on gap metric [1]

3.1 Introduction

MPC with multiple models is an alternative to track set-points in a wide operating range of nonlinear chemical processes, because multiple models can describe the dynamic behavior of the process better than a model, and related optimization problems can be solved faster than those formulated by a nonlinear model. There are three approaches to use multiple models: the first method constructs a global nonlinear model by interpolating multiple local models using weighting functions [6, 40, 41, 42]. Then an MPC controller is designed based on the global model. However, this method is hard to implement due to the computational load of the nonlinear optimization. The second approach designs a min-max optimization-based global MPC using the linear models in a model bank, which is a type of robust MPC [43, 44, 45]. In this approach, inaccurate process models are also considered for prediction, and conservative control law

[1] This chapter is an adapted version of B. Park, D. H. Jung, and J. M. Lee, "Offset-free Multilinear Model Predictive Control Based on Gap Metric," *Journal of Process Control*, Under review

is obtained. The last method is multilinear model predictive control (MLMPC), which designs a set of local linear MPC controllers based on the models in a model bank [6, 4, 7]. The resulting controller is called a scheduled controller, which prescribes control input as a convex combination of the local MPC actions. In summary, the input of the first controller is calculated from nonlinear optimization based on the combination of local nonlinear models. The input of the second controller is calculated from min-max optimization considering all local models. The input of the third controller is the combination of the inputs calculated from quadratic programming (QP) based local linear models. The last approach is preferred because it requires less amount of calculation time than the first approach, and yields a less conservative solution than the second approach.

There are two types of MLMPC to combine the local models: (1) prediction-based MLMPC and (2) gap metric-based MLMPC. The first method decides the weights of local MPCs based on the output prediction error and recursive Bayesian weighting. Dynamic matrix control (DMC) [6], state-space model based MPC [8], and multiple disturbance model [4] have been proposed for the prediction-based MLMPC. Because it is easy to implement in real-time, this approach has been applied to several applications [9, 10, 11, 12, 13]. The key step of prediction-based MLMPC is the construction of the local linear models. These models should describe the whole operating region, and it commonly depends on prior knowledge, experience, or trial and error [46]. Since, the model bank, the local MPCs, and the weighting function, need to be independently designed, it is difficult to tune the parameters of each component to meet the requirements for scheduled controllers. In addition, the prediction-based method

does not guarantee the stability of the closed-loop system.

The other approach exploits the gap metric, which is useful to compare the closed-loop behavior of two systems [47]. The gap metric between two linear systems is defined as the infimum of the infinity norm of the difference between one system and the other multiplied by a transfer function [14]. The gap metric can judge whether two linear systems can be stabilized by the same controller. It is possible to quantify the similarity of linear systems in a model bank and the operating ranges for which a local controller can be used. Thus, gap metric-based MLMPC can be designed to have the stability property compared to the first one. There are two kinds of gap metric-based weighting approaches: the current state-based approach [15] and the reference-based approach [7]. In the current state-based approach, the gap metrics of the current state and the models in the model bank are calculated, and the weights of the models depend on the gap metrics. In the reference-based approach, the gap metrics of the reference and the models are calculated instead of the current state. The existing gap metric-based MLMPCs have some limitations. First, the number of the local linear models grows exponentially as the number of the state and action increases. Thus, it is difficult to extend to a large scale system. Second, the existing gap metric-based weighting approaches are not suitable for controlling the systems with a wide operating range. In the current state-based approach, the gap metric of the current state and the models in the model bank is calculated at each sampling time, which is computationally heavy. On the other hand, the reference-based approach cannot predict the behavior of the current state if the gap metric between the current state and the setpoint is large because it only considers whether linear models can

approximate the dynamics at the set-point.

In this chapter, we propose a novel gap metric-based model bank selection that consists of a generalized gridding algorithm and K-medoids clustering for a system of an arbitrary dimension. In addition, we propose to design local controllers and prove that each local controller has the offset-free tracking property in the corresponding partition of the operating region. Lastly, a weighting method to design a scheduled controller is proposed to combine the prediction-based weighting and gap metric-based approach to compute the weights efficiently. We show the resulting scheduled controller improves the prediction performance, and verify the offset-free tracking property and the stability at any set-points in a designated operating region through simulations.

3.2 Local linear MPC design

Consider a nonlinear process

$$\begin{aligned} \dot{x} &= f(x,u), \\ y &= h(x), \\ u &\in \mathcal{U}, x \in \mathcal{X}, \end{aligned} \tag{3.1}$$

where f and h are continuously differentiable functions, $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control input, and $y \in \mathbb{R}^p$ is not only the measured output, but alos the controlled variables. $\mathcal{X}$ and $\mathcal{U}$ are the state and input constraints, respectively.

In order to obtain local linear models, the vector of the scheduling variables $\theta \in \Theta \subset \mathbb{R}^{N_\theta}$, so-called the scheduling vector, needs to

be defined. We assume the uniqueness and continuity of the equilibrium point corresponding to a vector of scheduling variables.

Assumption 3.1. *$\theta \in \Theta$ determines the unique equilibrium point and the equilibrium point satisfies the input and state constraints, where $\Theta \subset \mathbb{R}^{N_\theta}$ is a compact set.*

Assumption 3.2. *A function $F : \Theta \to \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$ that associates a scheduling vector with an equilibrium point of the system* (3.1) *is injective and continuous.*

The scheduling variables involve a subset of the states, inputs, outputs, disturbances, references, and model parameters. We also assume that the system is approximated by N_M local linear models. Each local linear model is obtained by linearizing the original system at the corresponding equilibrium point. Let the triplet of the steady state, input, and output at θ_i be $(x_{oi}, u_{oi}, y_{oi}) = (x_o(\theta_i), u_o(\theta_i), y_o(\theta_i))$, where θ_i is a representative component of the i^{th} subregion Θ_i. The i^{th} model is obtained by linearizing the nonlinear system at $s_i := (x_o(\theta_i), u_o(\theta_i), y_o(\theta_i)) = F(\theta_i) \in \mathcal{S}$, where $\mathcal{S}$ is the set of all equilibrium points corresponding to all scheduling variables in Θ, and s_i is the equilibrium point determined by θ_i. The set of the subregions $\{\Theta_i\}$ is assumed to be a partition of Θ:

Assumption 3.3. *A scheduling space Θ and its subregion Θ_i satisfy*

$$\emptyset \neq \Theta_i \subseteq \Theta, \bigcup_{i=1}^{N_M} \Theta_i = \Theta, \ \Theta_i \cap \Theta_j = \emptyset \text{ for } i \neq j. \tag{3.2}$$

The i^{th} local model P_i at the k^{th} sampling time is discretized:

$$\begin{aligned} x_i(k+1) &= A_i x_i(k) + B_i u_i(k), \\ y_i(k) &= C_i x_i(k), \end{aligned} \tag{3.3}$$

where $x_i(k) = x(k) - x_{oi}, u_i(k) = u(k) - u_{oi},$ and $y_i(k) = y(k) - y_{oi}$. $A_i, B_i,$ and C_i are the linearized and discretized matrices. By Assumption 1, each local model is uniquely determined because the equilibrium point is assumed to be unique for each θ_i. Linear offset-free MPC introduced in Chapter 2 is employed for the local controllers. The disturbance model is added to the linearized model:

$$\begin{aligned} \begin{bmatrix} x_i(k+1) \\ d_i(k+1) \end{bmatrix} &= \underbrace{\begin{bmatrix} A_i & B_{di} \\ 0 & I \end{bmatrix}}_{A_{ai}} \underbrace{\begin{bmatrix} x_i(k) \\ d_i(k) \end{bmatrix}}_{x_{ai}(k)} + \underbrace{\begin{bmatrix} B_i \\ 0 \end{bmatrix}}_{B_{ai}} u_i(k), \\ y_i(k) &= \underbrace{\begin{bmatrix} C_i & C_{di} \end{bmatrix}}_{C_{ai}} \begin{bmatrix} x_i(k) \\ d_i(k) \end{bmatrix}, \end{aligned} \tag{3.4}$$

where $d_i \in \mathbb{R}^p$ is unmeasured disturbance. Then the i^{th} linear MPC

controller at the k^{th} sampling time is designed as

$$\begin{aligned}
&\min_{u_0,\cdots,u_{N-1}} \sum_{j=0}^{N-1} \|x_j - \bar{x}_{k,i}\|^2_{Q_i} + \|u_j - \bar{u}_{k,i}\|^2_{R_i} + \|x_N - \bar{x}_{k,i}\|^2_{Q_{T,i}} \\
&\quad s.t.\ x_{j+1} = A_i x_j + B_i u_j + B_{di} d_j,\ j = 0, ..., N-1 \\
&\quad\quad d_{j+1} = d_j, \\
&\quad\quad y_j = C_i x_j,\ k = 0, ..., N \\
&\quad\quad u_j \in \mathcal{U}_i,\ x_j \in \mathcal{X}_i, \\
&\quad\quad x_0 = \hat{x}_i(k),\ d_0 = \hat{d}_i(k),
\end{aligned} \tag{3.5}$$

where $Q_i \succeq 0, R_i \succ 0$, and $Q_{T,i} \succ 0$ are the weighting matrices of the i^{th} local MPC for the state, input, and terminal state, respectively. $\mathcal{U}_i$ and $\mathcal{X}_i$ are the boundary conditions for the input and state for the i^{th} local MPC. Note that x_k is the predicted state, not the real state $x(k)$. The target state $\bar{x}_{k,i}$ and input $\bar{u}_{k,i}$ of the i^{th} linear model are

$$\begin{bmatrix} A_i - I & B_i \\ C_i & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_{k,i} \\ \bar{u}_{k,i} \end{bmatrix} = \begin{bmatrix} -B_{di}\hat{d}_i(k) \\ r_i(k) - C_{di}\hat{d}_i(k) \end{bmatrix}, \tag{3.6}$$

where $r_i(k) := r(k) - y_{oi} \in \mathbb{R}^p$ is the i^{th} transformed reference and $r(k)$ is the reference at the k^{th} sampling time. The observer is designed by

$$\begin{aligned}
\hat{x}_{ai}(k+1) =& A_{ai}\hat{x}_{ai}(k) + B_{ai}u_i(k) \\
&+ L_{ob}(C_{ai}\hat{x}_{ai}(k) - y_i(k)),
\end{aligned} \tag{3.7}$$

where L_{ob} is the observer gain. The observer is necessary for some reasons: (1) the number of the measurement smaller than that of the

state, (2) mismatch between the nonlinear process and the lienar model, and (3) unknown stochastic disturbance and noise. A typical choice is Kalman filter or Luenberger observer [48]. Kalman filter is preferred when the third reason is considered.

Remark 3.1. *The closed-loop system* (3.1)*,* (3.5)*,* (3.6)*, and* (3.7) *achieves the offset-free control if the closed-loop system converges [32, 49]. If $Q_{T,i}$ in* (3.5) *is the solution of discrete-time algebraic Riccati equation (DARE), MPC is equivalent to LQR in the unconstrained region. Then the gap metric stability margin of $b_{P_i,K_{LQR,i}}$ can be computed, where $K_{LQR,i}$ is the LQR gain obtained by the i^{th} local linear model, Q_i, and R_i.*

Remark 3.2. *The i^{th} local controller is designed to stabilize any equilibrium point in the i^{th} subregion Θ_i. The algorithms proposed in the next sections will provide the subregions of Θ and local MPCs. Any equilibrium point in each of the subregions can be stabilized by one of the local MPCs.*

3.3 Gap metric-based multilinear MPC

In this section, we propose a novel gap metric-based multilinear MPC algorithm which can improve the transient performance and guarantees the stability of the closed-loop system if the proposed MPC converges to a local MPC. It consists of the algorithms for gridding and clustering of local models, the design of the local MPC controllers, and calculating the weights of the local models.

3.3.1 Gap metric-based gridding algorithm

Let $\{\theta^{k,n_k}\}$ $(k = 1, ..., N_\theta, n_k = 1, ..., N_k)$ be a set for the k^{th} component of the scheduling vector θ. Then, the set of the initial scheduling vectors is denoted by $\Theta_{init} : \{\theta_n\} := \{\theta^{1,n_1}\} \times \cdots \times \{\theta^{N_\theta,n_{N_\theta}}\}$ $(n = 1, ..., \prod_{i=1}^{N_\theta} N_i)$. Each scheduling vector θ_i in the initial grid Θ_{init} has the corresponding equilibrium point s_i. The linearized system of Eq. (3.1) at s_i is obtained as (3.8) and denoted by P_i.

$$\begin{aligned}
\dot{x} &= \left.\frac{\partial f}{\partial x}\right|_{x_o(\theta_i),u_o(\theta_i)} (x - x_o(\theta_i)) + \left.\frac{\partial f}{\partial u}\right|_{x_o(\theta_i),u_o(\theta_i)} (u - u_o(\theta_i)), \\
y &= \left.\frac{\partial h}{\partial x}\right|_{x_o(\theta_i)} (x - x_o(\theta_i)) + y_o(\theta_i)
\end{aligned} \tag{3.8}$$

The following constraints are assumed for the scheduling space and grid points.

$$\begin{aligned}
\theta^{k,min} \leq \theta^{k,n_k} \leq \theta^{k,max}, \;\; k = 1, ..., N_\theta, n_k = 1, \cdots, N_k, \\
\theta^{k,1} = \theta^{k,min}, \;\; k = 1, ..., N_\theta, \\
\theta^{k,N_k} = \theta^{k,max}, \;\; k = 1, ..., N_\theta,
\end{aligned} \tag{3.9}$$

where $\theta^{k,min}$ and $\theta^{k,max}$ are the lower and upper bounds of the k^{th} component of the scheduling vector, respectively. In addition, we define a normalized scheduling vector as

$$\theta^n_{k,n_k} = \frac{\theta^{k,n_k} - \theta^{k,min}}{\theta^{k,max} - \theta^{k,min}}. \tag{3.10}$$

Thus, each component of a normalized vector has its upper and lower bounds as one and zero, respectively. The following Algorithm 1

guarantees that the gap metric between the linearized systems with two adjacent grid points is smaller than a designated threshold. $n_{bd,i}$ is defined as the number of the components of θ_i whose values are $\theta^{k,max}$ or $\theta^{k,min}$. Thus a large value of $n_{bd,i}$ means that the point is close to the boundaries of the grid. Because the proposed algorithm does not generate a point between the two points whose gap metric is less than the threshold, it is computationally efficient compared with the conventional gridding algorithm [46].

Algorithm 1: Gap metric-based gridding algorithm

Input: Bounds of the scheduling vector θ_{min}, θ_{max}, the components of the initial grid points $\{\theta_n\}$, a threshold of gap γ_{th}

Result: A set of the scheduling vector Θ_{grid}

Initialization :

$\Theta_{grid} = \emptyset, \Theta^n_{grid} = \emptyset, \mathcal{S}_{grid} = \emptyset, \mathcal{P}_{grid} = \emptyset, N_{grid} = 0$

$\Theta_{add} = \Theta_{init}, n_{add} = \prod_{k=1}^{N_\theta} N_k$, Calculate Θ^n_{add}

while $n_{add} > 0$ **do**

  $\Theta^n_{add,new} \leftarrow \emptyset, n_{add,new} \leftarrow 0$

  Restore Θ_{add} from Θ^n_{add}

  $\Theta_{grid} \leftarrow \Theta_{grid} \cup \Theta_{add}, \Theta^n_{grid} \leftarrow \Theta^n_{grid} \cup \Theta_{add}$

  for $i \leftarrow 0$ **to** n_{add} **by** 1 **do**

    Obtain the equilibrium point s_i corresponding to $\theta_i \in \Theta_{add}$

    Construct $P_i := P(\theta_i)$ from s_i

    $\mathcal{S}_{grid} \leftarrow \mathcal{S}_{grid} \cup \{s_i\}$

    $\mathcal{P}_{grid} \leftarrow \mathcal{P}_{grid} \cup \{P_i\}$

    Determine the number of the nearest points

$$n_{near,i} = 2N_\theta - n_{bd,i},$$

    Calculate the distance $d_n(i,j)$ between $\theta_i \in \Theta_{add}$ and $\theta_j \in \Theta_{grid}$

    $\{P_{k_i}\}$ corresponding to the nearest $n_{near,i}$ points are selected

    Calculate $\delta_g(P_i, P_{k_i}), k = 1, \cdots, n_{near,i}$

    if $\delta_g(P_i, P_{k_i}) > \gamma_{th}$ **then**

      $\theta^n_{new} \leftarrow \frac{1}{2}(\theta^n_i + \theta^n_{k_i})$

      $\Theta^n_{add,new} \leftarrow \Theta^n_{add,new} \cup \{\theta^n_{new}\}\ n_{add,new} \leftarrow n_{add,new} + 1$

    end

  end

  $\Theta_{add} \leftarrow \Theta^n_{grid,new}$

  $n_{add} \leftarrow n_{add,new}$

  $N_{grid} \leftarrow N_{grid} + n_{add}$

end

Save $\Theta_{grid}, \Theta^n_{grid}, \mathcal{S}_{grid}, \mathcal{P}_{grid}, N_{grid}$

The obtained grid is assumed to satisfy Assumption 3.4:

Assumption 3.4. *Let the convex hull of a set of grid points $\{\theta_i\}$ be Conv$(\{\theta_i\})$. For all $P_i := P(\theta_i)$ from $\{\theta_i\}$, any $P_j := P(\theta_j)$ from $\theta_j \in Conv(\{\theta_i\})$ satisfies*

$$\delta_g(P_i, P_j) < \max_{\theta_s, \theta_l \in \{\theta_i\}} \delta_g(P_s, P_l).$$

Assumption 3.4 is not restrictive because it can be satisfied by refining the grid. Suppose the gap metric between the linearized systems of (3.1) at θ_1 and θ_2 is the maximum of gap metrics between two points in $\{\theta_i\}$. Assumption 3.4 means that if a controller K_1 that stabilizes P_1 can stabilize P_2, K_1 can stabilize any P_k from $\theta_k \in Conv(\{\theta_i\})$. Thus, we can design a local controller that can stabilize the system (2.1) at any equilibrium point in $Conv(\{\theta_i\})$. $Conv(\{\theta_i\})$ is a convex hull of $\{\theta_i\}$. It should be checked that all the local models from the extreme points are stabilized by the controller to be used in $\{\theta_i\}$.

Remark 3.3. *The threshold value γ_{th} in Algorithm 1 needs to be small enough so that there exists a local controller K stabilizing an equilibrium point such that the gap stability margin of the closed-loop system is larger than γ_{th}, i.e., $b_{P,K} > \gamma_{th}$, where P is the linearized system at the equilibrium point.*

3.3.2 Gap metric-based K-medoids clustering

In order to construct a relevant model bank that represents the process over the entire operating region, K-medoids clustering algo-

rithm, one of the unsupervised machine learning algorithms, is chosen for the base algorithm to classify the models in the grid and to select the representative models [50]. Because K-medoids clustering is based on the most centrally located object in a cluster, it is less sensitive to outliers in comparison with the K-means clustering [51]. However, the conventional K-medoids clustering considers only one metric in evaluating the distance between two objects. In order to consider both Euclidean metric and gap metric, a gap metric-based K-medoids clustering algorithm is proposed in Algorithm 2. The inputs are the grids from Algorithm 1 and three parameters: k_{cl} is the number of clusters; ϵ_{tol} is the threshold for the change of clusters; n_{tol} is the maximum number of iterations.

Algorithm 2: Gap metric-based K-medoids clustering algorithm

Input: $\Theta_{grid}, \Theta^n_{grid}, \mathcal{S}_{grid}, \mathcal{P}_{grid}, N_{grid}, k_{cl}, \epsilon_{tol}, n_{tol}$

Result: Sets of clusters $\{\Theta_i\}, \{\Theta^n_i\}, \{\mathcal{S}_i\}, \{\mathcal{P}_i\}$
Sets of medoids $\{\theta_{med,i}\}, \{\theta^n_{med,i}\}, \{s_{med,i}\}, \{P_{med,i}\}$

Initialization : Set $flag = 1, it = 1$

$\Delta_{k,j} = \delta_g(P_k, P_j)$, $D^n_{k,j} = d(\theta^n_k, \theta^n_j)$, $1 \le k, j \le N_{grid}$

Select initial medoids:

Set v_j as $v_j = \sum_{k=1}^{N_{grid}} \frac{\Delta_{k,j}}{\sum_{l=1}^{N_{grid}} \Delta_{k,l}}$ and sort v_j's in ascending order

Select k_{cl} indices having the first k_{cl} smallest values for initial medoids and denote by $IDX_{med} = \{idx_{med,i}\}, \quad i = 1, \cdots, k_{cl}$

Calculate sets of medoids $\{\theta^{it}_{med,i}\}, \{\theta^{n,it}_{med,i}\}, \{s^{it}_{med,i}\}, \{P^{it}_{med,i}\}$

while $flag$ **do**

Clustering:

Obtain the sets of clusters by assigning each object to the nearest medoid in terms of gap metric and the Euclidean metric: $\{\Theta^{it}_i\}, \{\Theta^{n,it}_i\}, \{\mathcal{S}^{it}_i\}, \{\mathcal{P}^{it}_i\}$

Calculate the sum of the gap metric from all objects to their medoids: σ_{it}

Calculate medoids:

Find a new medoid of each cluster that minimizes the maximum of the gap metric to other objects in its cluster:
$\{\theta^{it+1}_{med,i}\}, \{\theta^{n,it+1}_{med,i}\}, \{s^{it+1}_{med,i}\}, \{P^{it+1}_{med,i}\}$, update IDX_{med}

if $it == 1$ **then**
$flag \leftarrow 1$

else if $it < n_{tol}$ *or* $|\sigma_{it} - \sigma_{it-1}|/\sigma_{it-1} > \epsilon_{tol}$ **then**
$flag \leftarrow 1$

else
$flag \leftarrow 0$

end

$it \leftarrow it + 1$

end

Save
$\{\Theta_i\}, \{\Theta^n_i\}, \{\mathcal{S}_i\}, \{\mathcal{P}_i\}, \{\theta_{med,i}\}, \{\theta^n_{med,i}\}, \{s_{med,i}\}, \{P_{med,i},\}, IDX_{med}$

In the proposed method, the weighted sum of the gap metric and Euclidean metric is used for clustering of linear systems. The gap metric makes a cluster in which two dynamics at two distinct equilibrium points have a small gap metric. The Euclidean metric prevents a cluster from being disconnected. In addition, the medoid is selected by the largest gap metric to design a controller that stabilizes the linearized system at a medoid. By Theorem 2.3, the gap stability margin of a closed-loop system, which consists of a controller and the linear model at the medoid, has to be larger than the gap metric between the medoid and any point in the cluster to stabilize all points in the cluster. If the number of clusters increases, the largest gap metric between the medoid and all the points in a cluster becomes smaller as each cluster is reduced. Thus, the feasible set of stabilizing controllers for a cluster becomes larger. However, the number of local controllers increases.

3.3.3 MLMPC design

There are no proofs that guarantee the stability at a set-point given weighting methods for existing MMPC algorithms. Based on the linear offset-free MPC and local models by the proposed gridding and clustering algorithms, we propose a scheduled controller to solve those problems. Theorem 3.1 shows that a linear offset-free MPC can stabilize an equilibrium point if the gap metric between the linearized dynamics at the equilibrium point and the local linear model used in MPC is smaller than the gap stability margin of the pair of the model and the controller in the MPC.

Theorem 3.1. *Suppose Assumption 3.1, 3.2, 3.3, and 3.4 hold and*

the MPC described in (3.7)*,* (3.5)*, and* (3.6) *is applied to the system* (3.1) *with a linear model* P_m *and a scheduling vector* θ *containing the set-point* r*. Let* s_θ *and* $P(\theta)$ *be the equilibrium point corresponding to* θ *and the linearized model at* s_θ*. Assume that* $Q_{T,\theta}$ *is the solution of DARE for* P_m *and the weights* Q_θ *and* R_θ *in* (3.5)*, and* K_{lqr} *is the corresponding LQR gain. If* $b_{P_m,K_{lqr}} > \delta_g(P_m, P(\theta))$*, then the equilibrium state* x_θ *corresponding to the set-point* r *in closed-loop system Eqs.* (3.1)*,* (3.5)*,* (3.6)*, and* (3.7) *is asymptotically stable.*

Proof *If the state* x_k *is in the neighborhood of* x_θ*, All the constraints in the MPC formulation of* (3.5) *are inactivated. Then, the solution of the MPC is equivalent to the solution of the LQR [1]:*

$$\begin{aligned} J^*_\infty(\delta x_0) = \min \sum_{k=0}^{\infty} \|\delta x_k\|^2_{Q_\theta} + \|\delta u_k\|^2_{R_\theta} = \|\delta x_0\|^2_{Q_{T,\theta}} \\ s.t.\ \delta x_{k+1} = A_\theta \delta x_k + B_\theta \delta u_k \\ \delta x_0 = \hat{x}(k) - \bar{x}(k),\ \delta u_0 = u(k) - \bar{u}(k). \end{aligned} \tag{3.11}$$

Then the closed-loop system between the k^{th} *and the* $(k+1)^{th}$ *sampling time is*

$$\begin{aligned} \frac{dx}{dt} &= f(x(t), -K_{lqr}\delta x_0 + \bar{u}(k)), \\ y &= h(x). \end{aligned} \tag{3.12}$$

The linearized and discretized system of (3.12) *at* x_θ *is*

$$\begin{aligned} \delta x(k+1) &= A_\theta \delta x(k) + B_\theta \delta u(k) \\ &= (A_\theta - B_\theta K_{lqr}) \delta x(k), \\ \delta y(k) &= C_\theta \delta x(k), \end{aligned} \tag{3.13}$$

where $\delta x(k) = x(k) - x_\theta$*;* $\delta u(k) = u(k) - u_\theta$*;* $\delta y(k) = y(k) - r$*;* A_θ, B_θ*, and* C_θ *are the matrices corresponding to* $P(\theta)$*;* u_θ *is the steady-state input at* x_θ*. Because LQR gain* K_{lqr} *satisfies Theorem 2.3, the autonomous system* (3.13) *is asymptotic stable and all the absolute eigenvalues of* $(A_\theta - B_\theta K_{lqr})$ *are less than one. Because the continuous matrix transformed from* $(A_\theta - B_\theta K_{lqr})$ *is Hurwitz matrix,* (x_θ, u_θ) *in the closed-loop system is asymptotically stable as the linearized dynamics of the system* (3.1) *is stabilized.* ■

Thus, we can guarantee the stability at all the equilibrium points in the operating region if Theorem 3.1 is satisfied in each cluster. In order to find the number of the clusters, Algorithm 3 is proposed. Note that $b_{P(\theta_{med,i}),K_{lqr,i}}$ should be larger than the threshold γ_{th} so that Algorithm 3 can be terminated. Q_i and R_i affect $K_{lqr,i}$, so these parameters have to be tuned. Finding Q_i, R_i can be time-consuming because it relies on heuristics. An alternative is to replace $b_{P(\theta_{med,i}),K_{lqr,i}}$ in Algorithm 3 by $\min(b_{opt}(P(\theta_{med,i})), \gamma_{max})$, where γ_{max} is a threshold for the gap metric between two systems in a cluster. Then we tune Q_i and R_i and check if the corresponding LQR gain can stabilize all the grid points in the cluster.

Algorithm 3: Determining the number of clusters

Initialization :

Set the number of cluster : k_{cl}

$flag \leftarrow 1$

while $flag$ **do**

Run the Algorithm 2

Calculate $\Delta_{max,i}$ as

$$\Delta_{max,i} = \max_{\theta \in \Theta_i} \delta_g(P(\theta), P(\theta_{med,i})), \quad i = 1, ..., k_{cl}$$

if $b_{P(\theta_{med,i}),K_{lqr,i}} \leq \Delta_{max,i}$ *for any* i **then**

$k_{cl} \leftarrow k_{cl} + 1$

$flag \leftarrow 1$

else

$flag \leftarrow 0$

end

end

Save $\{\Theta_i\}, \{\Theta_i^n\}, \{\mathcal{S}_i\}, \{\mathcal{P}_i\}$
$\{\theta_{med,i}\}, \{\theta_{med,i}^n\}, \{s_{med,i}\}, \{P_{med,i},\}, IDX_{med}$

The existence of a controller that can track any equilibrium point in a cluster is guaranteed by Theorem 3.2.

Theorem 3.2. *Consider a cluster Θ_i and its medoid $\theta_{med,i}$ of a system* (3.1) *obtained by the modified Algorithm 3 in which $b_{P(\theta_{med,i}),K_{lqr,i}}$ is replaced by* $\min(b_{opt}(P(\theta_{med,i})), \gamma_{max}))$. *For all s_i, s_j corresponding to $\theta_i, \theta_j \in \Theta$, there exists a controller that can steer s_i to s_j.*

Proof *Let a controller K_i^* satisfy the inequality:*

$$b_{P(\theta_{med,i}),K^*} > \delta_g(P(\theta_{med,i}), P(\theta_i)) \tag{3.14}$$

The existence of K_i^* is guaranteed by Algorithm 3, as the LQR gain $K_{lqr,i}$ satisfies the inequality (3.14). *By Theorem 2.3, the equilibrium point s_i of any pairs $(P(\theta_i), K_i^*)$ is asymptotically stable at the equilibrium point s_i. Thus, s_i of the nonlinear system* (3.1) *is asymptotically stable. Let the nonlinear system* (3.1) *be P_{nl}. (P_{nl}, K_i^*) has the region of attraction (ROA) at any s_i. There is a manifold of equilibrium points between arbitrary s_i and s_j by Assumption 3.2. Let a line on the manifold be $\bar{s}_{ij}$. Let s_1 and K_1^* be a point on $\bar{s}_{ij}$ and a controller, respectively such that the ROA of the pair $(P(\theta_1), K_1)$ includes s_i. Then K_1^* can steer s_i to s_1. Considering s_1 as the initial point s_i, the same procedure produces s_2 and K_2^* such that K_2^* can steer s_1 to s_2. Because $\bar{s}_{2j}$ is shorter than $\bar{s}_{1j}$, $K_i^*(i = 1, 2, \cdots)$ can steer s_i to s_j.* ■

Note that there is no guarantee that a local MPC satisfies Theorem 3.2 because MPC formulation involves constraints like (3.5). Hence, the stability is evaluated by the closed-loop control performance of several simulations. In addition, global stability is not guar-

anteed, which is a limitation of multilinear model control approaches applied to nonlinear systems [44].

In order to determines the weights of the local MPCs, we considered both gap metric-based method (2.33) and prediction based method (2.36). The gap metric-based method is based on the equilibrium point corresponding to the scheduling vector. It is advantageous in that the closed-loop behavior at a set-point can be guaranteed to be stable with a large gap stability margin of a local controller. However, there is no guarantee that the controller steers any states to the set-point. Although the gap metric between the set-point and the local model is small, it does not mean that the current state is well predicted and steered to the set-point because the dynamics of the current state can be completely different from that of the set-point. On the other hand, the output prediction-based method chooses the weights based on the current state. However, it does not guarantee stability at an equilibrium point. Combining both methods, the gap metric-based multilinear MPC algorithm is proposed in Algorithm 4.

Algorithm 4: Gap metric-based multilinear MPC

Initialization :

Obtain $\{\Theta_i\}, \{\Theta_i^n\}, \{\mathcal{S}_i\}, \{\mathcal{P}_i\}$, $\{\theta_{med,i}\}, \{\theta_{med,i}^n\}, \{s_{med,i}\}, \{P_{med,i},\}, IDX_{med}$ from Algorithm 1,2,3

$flag \leftarrow 1, t \leftarrow 0, w_i \leftarrow 1/k_{cl}$

for $k \leftarrow 0$ **to** N_T **by** 1 **do**

Check the reference $r(k)$ corresponding to $\theta(k)$

Measure $y(k)$ and calculate $e(k) = r(k) - y(k)$

Calculate $i^* = \arg\max \gamma_i(\theta(k))$

Calculate $\phi_i^1(k)$

if $i == i^*$ **then**

$\phi_i^1(k) \leftarrow 1$

else

$\phi_i^1(k) \leftarrow 0$

end

Calculate $\phi_i^2(k)$ from (2.36)

Calculate a_1 and a_2

$$a_1 = \exp(-e(k)^T \Lambda_e e(k))$$
$$a_2 = 1 - a_1$$

Calculate $w_i(k)$

$$w_i = a_1 \phi_i^1(k) + a_2 \phi_i^2(k)$$

Solve k_{cl} local MPC problems based on $\{P_{med,i}\}$ and obtain $\{u_i(k)\}$

Calculate $u(k)$ to the plant

$$u(k) = \sum_{i=1}^{k_{cl}} w_i u_i(k)$$

Apply $u(k)$ to the plant

end

Save $\{u(k)\}, \{y(k)\}$

The weights of the proposed multilinear MPC are calculated by three criteria, the gap metric $\{\gamma_i(\theta_k)\}$, prediction error $\{\epsilon_i(k)\}$, and error $e(k)$. Its criterion to give the weights is the combination of the prediction accuracy and the gap metric between models and the linearized system at the set-point. A dual weighting strategy is applied by the proposed algorithm: (1) when the state is out of the cluster and the target set-point exists in the cluster, the weights are dominated by the prediction accuracy; (2) when the output approaches the set-point, the weights are dominated by the gap metric. Unlike the weights calculated by the existing gap metric-based method, Eq. (2.33), only one local MPC is selected for the scheduled controller when the error becomes small. Thus, the stability property of the local controller can be exploited.

There is no guarantee of stability if the state is out of the ROA of the closed-loop system by a local MPC at the reference point. In this situation, the models whose prediction errors are small should have high weights. If the output is close to the reference, the current state is considered to be in the controllable set of a local controller that stabilizes the dynamics at the reference. Then the state is steered to the reference with asymptotic stability by Theorem 3.1 and 3.2.

Note that the parameter Λ_e can be determined by θ_k. Suppose the θ_k is on the cluster whose medoid is $\theta_{med,i}$. $\Delta y^t_{max,k}$ and Δy^t_k are defined as

$$\begin{aligned} \Delta y^t_{max,k} &= \max_{\theta \in \Theta_i} |y_{\theta_{med,i},k} - y_{\theta,j}|, \\ \Delta y^t_k &= |y_{\theta_{med,i},k} - y_{\theta_k,k}| \end{aligned} \tag{3.15}$$

Let the maximum distance between i^{th} component of the steady-state output in the cluster and that of the output at the medoid be $\Delta y_{max,i}$.

Then we can make the i^{th} diagonal component of Λ_e have a scale of $\Delta y_{max,i}^{-2}$, so that each component of the error gives a similar effect to determine the weights.

3.4 Results and discussions

3.4.1 Example 1 (SISO CSTR)

Consider a dimensionless exothermal continuous stirred tank reactor (CSTR) [52],

$$\begin{aligned}
\dot{x}_1 &= -x_1 + D_a(1-x_1)\text{exp}(-\frac{x_2}{1+x_2/\gamma}),\\
\dot{x}_2 &= -x_2 + BD_a(1-x_1)\text{exp}(-\frac{x_2}{1+x_2/\gamma}) + \eta(u-x_2),\\
y &= x_2,
\end{aligned} \tag{3.16}$$

where x_1, x_2 and u are the dimensionless conversion, temperature, and coolant temperature, respectively. D_a, γ, B and η are the dimensionless parameters related to the flow rate of the feed, activation energy of the reaction, heat capacity of the solution, and the heat transfer coefficient of the tank, respectively. The parameters for the plant and the controller are shown in Table 3.1. The control objective is to track a set-point of x_2 and to reject the disturbance by manipulating u. The output is the scheduling vector as it determines the equilibrium point uniquely.

Table 3.1: Parameters of Example 1

CSTR		**Local MPC**		**Global MPC**	
D_a	0.072	Q_i	1	k_e	2
γ	20	R_i	0.01	μ	0.05
B	8	N	10	Λ^i	100
η	0.3			Λ_e	1

The proposed gridding and clustering algorithms are applied with the parameters $\gamma_{th} = 0.1$ and $k_{cl} = 4$. The clusters and medoids of the steady-state input-output pair are obtained as shown in Figure 3.1 and Table 1. The system has the output multiplicity when the value of the steady-state input is between -0.519 and 0.527. Each cluster is connected and the maximum gap metric of each medoid does not exceed 0.51. The maximum spectral radius of the closed-loop systems controlled by the LQR controller in a cluster is less than 1 as shown in Table 3.1, which means the four local MPCs are set without stability issue. The controllers are combined into a multilinear MPC controller (MLMPC I) using the proposed weighting method. In order to check offset-free tracking, the state and measurement noise are not added. Luenberger observer is chosen for the filter as it is convenient to control the response time by changing the poles of the observer. Set-point tracking control and disturbance rejection are tested for MLMPC I. Gap metric-based MLMPC (MPMPC II) and prediction-based MLMPC (MPMPC III) are also tested for comparison and the results are shown in Figure 3.2, 3.3, 3.4,3.5, and Table 3.2. Subscripts c, g, and p represent MLMPC I, II, and III, respectively.

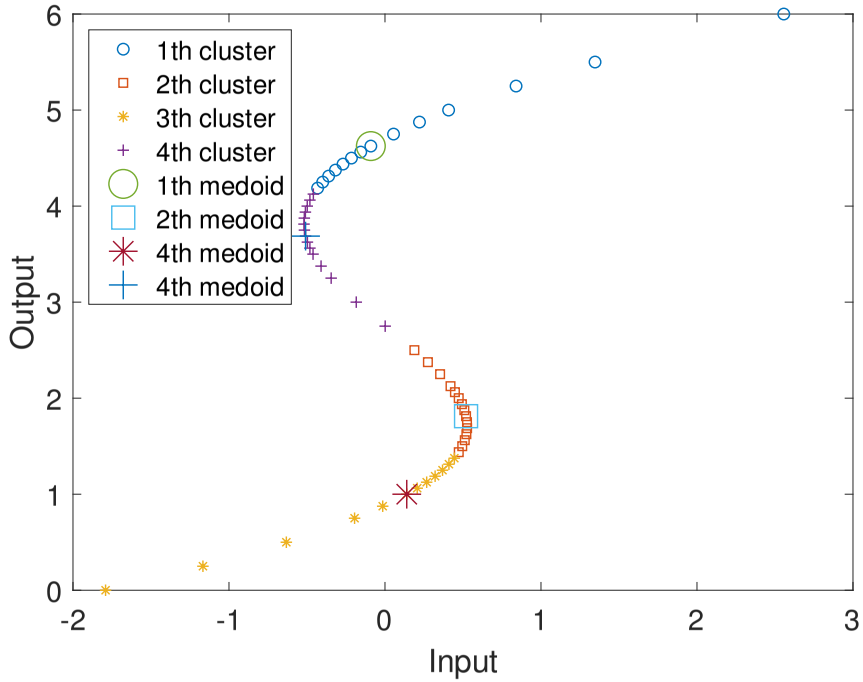

Figure 3.1: Clusters of steady-state input-output map for SISO CSTR

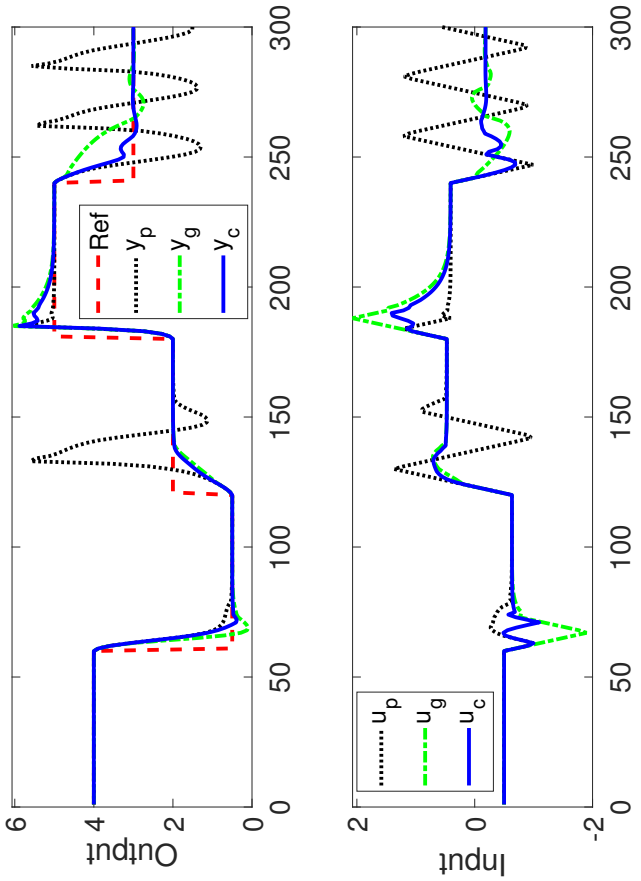

Figure 3.2: Set-point tracking control for SISO CSTR

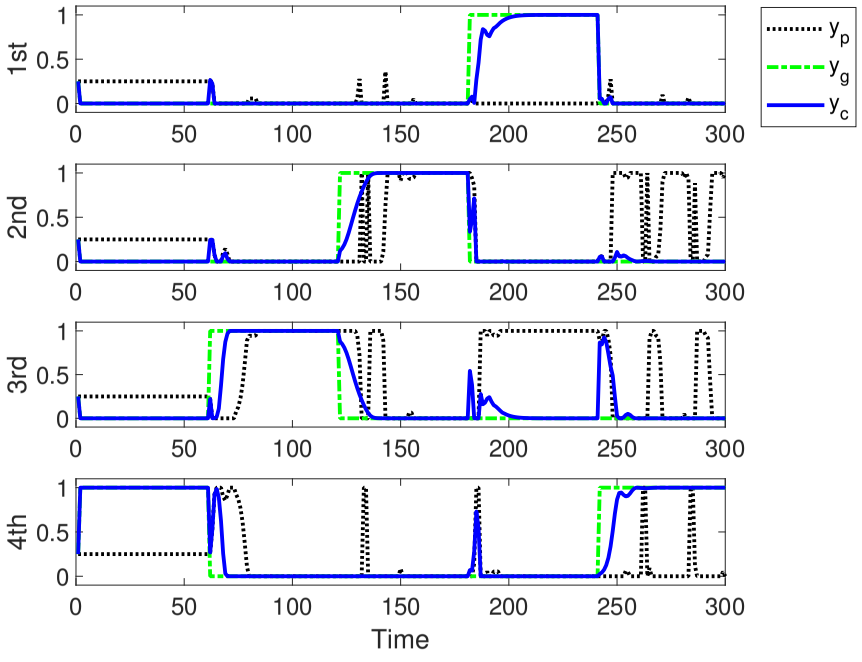

Figure 3.3: Weights of local controllers for tracking of SISO CSTR

The performance of MLMPC III is the worst for set-point tracking, as y_p has a high overshoot in the third reference step and oscillates in the last reference step. The weights of MLMPC III does not change at the first 10 sample times at the third reference step, which means the third local MPC is mainly used for the scheduled controller. The gap metric between the third medoid and the third reference is 0.93, which means the MPC using the third model is not adequate for control the neighborhood of the third reference. In addition, the gap metric between the third medoid and the last reference is 0.99, and it causes oscillation. On the other hand, y_g and y_c track the reference without oscillation in the whole operating range. In addition, the performance of a controller is measured by the sum of absolute errors (SAE). The SAE of an output is defined as $\sum_{k=1}^{N} |e(k)|$, where $e(k)$ is the error of the output at k^{th} sampling time and N is the number of the samples. Without disturbance, I has the lower SAE than II. y_c tracks the reference faster than y_g with the help of the prediction-based weight.

For disturbance rejection, we inject step input disturbances whose sizes are 1, -1, and -0.5 at time 30, 150, and 260, respectively. MLMPC I and MLMPC II show improved performances compared to MLMPC III in terms of SAE. As shown in Figure 3.4, the integrated MLMPC controller has a lower SAE than the others for disturbance rejection control. All controllers have the same weights for their local controllers after the disturbances are injected.

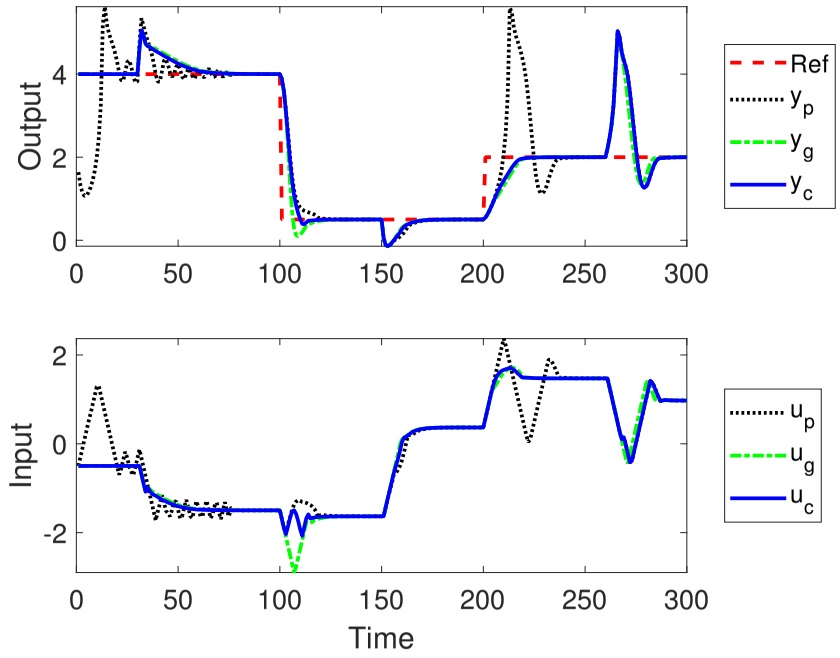

Figure 3.4: Disturbance rejection for SISO CSTR

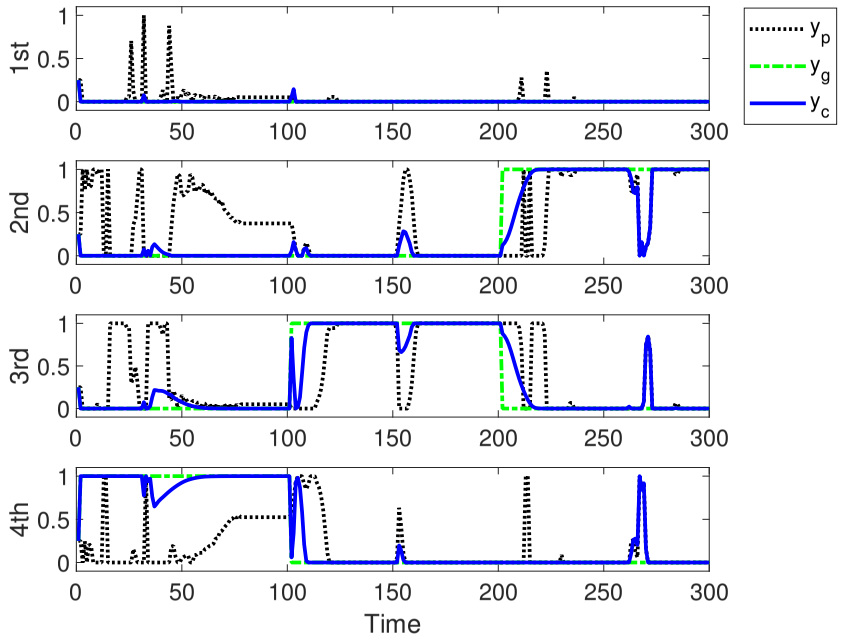

Figure 3.5: Weights of local controllers for disturbance rejection of SISO CSTR

Table 3.2: Integral absolute error for SISO CSTR

	prediction based	**Gap based**	**Proposed**
SAE (normal)	145.6	77.3	57.1
SAE (disturbance rejection)	144.0	71.2	73.3

We also check that the number of the clusters affects the stability and performance of the proposed controller as shown in Table 3.3. The second column represents the minimum gap metric between the last set-point and the models in the clusters. The third column represents the computation time during the closed-loop simulation. The last column represents the sum of output errors during the simulation. The table shows the minimum gap metric decreases as the number of clusters increases. It implies the stability margin increases with the number of the clusters. Indeed, we have observed the output oscillates if the number of clusters is smaller than four, and achieves offset-free control otherwise. On the other hand, the computation time linearly increases with respect to the number of the clusters, because the number of optimization problems to be solved increases. The transient performance does not seem to be related to the number of clusters.

Table 3.3: Effects of the number of clusters

# of clusters	**Gap metric**	**Computation time (s)**	**Sum of errors**
1	0.79	24.5	132
2	0.79	38.0	134
3	0.79	53.0	135
4	0.30	66.0	86
5	0.20	77.7	101
6	0.11	91.0	98
7	0.11	109.8	99
8	0.11	120.7	99
9	0.11	135.9	104
10	0.11	148.8	108

3.4.2 Example 2 (MIMO CSTR)

Consider a multi-input multi-output continuous stirred tank reactor (CSTR) [53]. It consists of an irreversible and exothermic reaction, and the temperature is controlled by a coolant stream.

$$\begin{aligned}
\dot{C}_A(t) =& \frac{q}{V}[C_{A0} - C_A(t)] - k_0 C_A(t) \mathrm{e}^{-E/RT(t)}, \\
\dot{T}(t) =& \frac{q}{V}[T_0 - T(t)] - \frac{\Delta H k_0}{\rho C_p} C_A(t) \mathrm{e}^{-E/RT(t)} \\
&+ \frac{\rho_c C_{pc}}{\rho C_p V} q_c(t)[1 - \mathrm{e}^{-\frac{h_A}{\rho_c C_{pc} q_c(t)}}][T_{c0} - T(t)], \\
y(t) =& [C_A(t) \;\; T(t)]^T.
\end{aligned} \tag{3.17}$$

The parameters and initial values of the variables in the system are given in Table 3.4. The measured concentration C_A and the temperature T are controlled by manipulating the flow rate of A, q and the coolant flow rate, q_c for set-point tracking and disturbance rejection. The input constraints are : $q \in [95, 150]$ and $q_c \in [60, 110]$.

Table 3.4: MIMO CSTR Parameters and Initial Values

Product concentration	C_A	0.1 mol/L
Coolant flow rate	q_c	103.41 L min^{-1}
Feed concentration	C_{A0}	1 mol/L
Inlet coolant temperature	T_{C0}	350 K
Heat transfer term	h_A	7×10^5 cal/min K
Activation energy term	E/R	1×10^4 K
Liquid densities	ρ, ρ_c	1×10^3 g/L
Reactor temperature	T	438.51 K
Process flow rate	q	100 L min^{-1}
Feed temperature	T_0	350 K
CSTR volume	V	100 L
Heat of reaction	ΔH	-2×10^5 cal/mol
Specific heats	C_p, C_{pc}	1 cal g^{-1} K^{-1}
Reaction rate constant 1	k_0	7.2×10^{10} min^{-1}

The steady-state input-output relationship is shown in Figure 3.6. Since the degrees of freedom in (3.17) are two, any two variables among two inputs and two outputs determine the equilibrium point uniquely. The manipulated variables are chosen as the scheduling variables in this example. The proposed gridding and clustering algorithm are applied with the parameters $\gamma_{th} = 0.1$ and $k_{cl} = 4$. The clusters and the medoids with 242 grid points are obtained as shown in Figure 3.7 and Table 3.5. Each cluster is connected and the maximum gap metric of each medoid does not exceed 0.36. The parameters for the controller are shown in Table 3.6. The maximum spectral radius of the closed-loop systems controlled by the LQR controller in a cluster is less than 1 as shown in Table 3.5. The controllers are combined into a global multilinear MPC controller (MLMPC I). The tuning parameters are: $k_e = 2, \Lambda^i = \begin{pmatrix} 1000 & 0 \\ 0 & 10 \end{pmatrix}, \mu = 0.05$, and $\Lambda_e = \begin{pmatrix} 400 & 0 \\ 0 & 0.0025 \end{pmatrix}$. Set-point tracking control and disturbance rejection are tested for MLMPC I, MLMPC II, and MLMPC III. The results are shown in Figures 3.8 and 3.9 and Table 3.7.

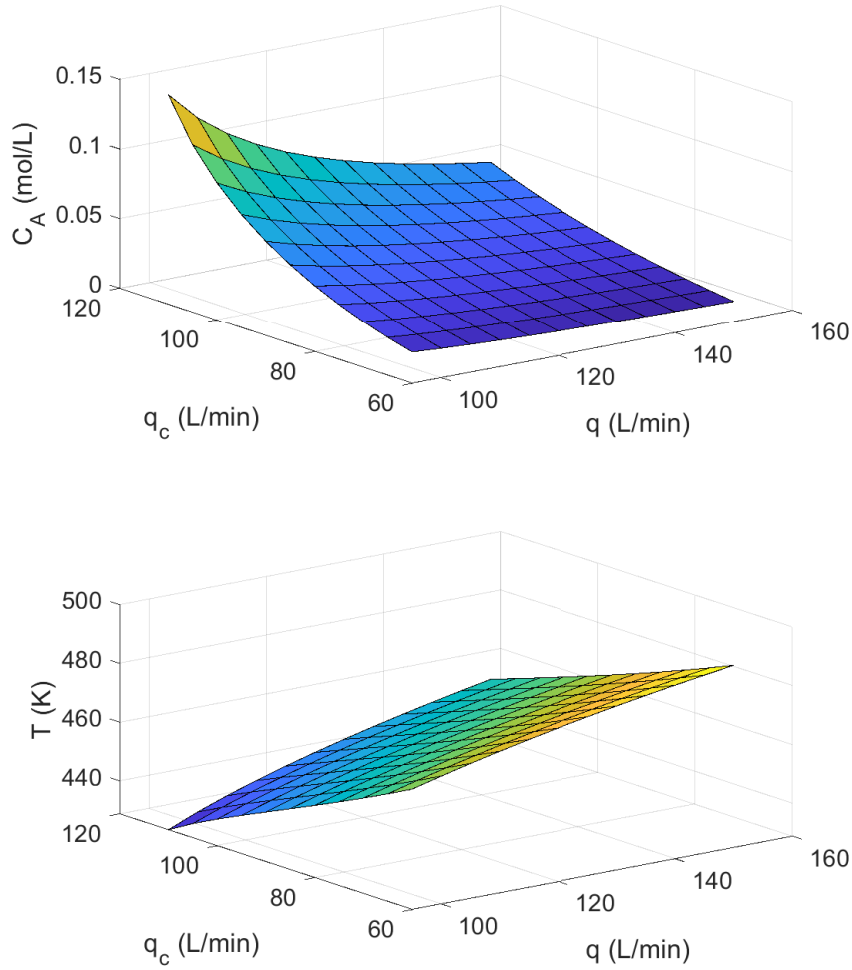

Figure 3.6: Steady-state input-output map for MIMO CSTR

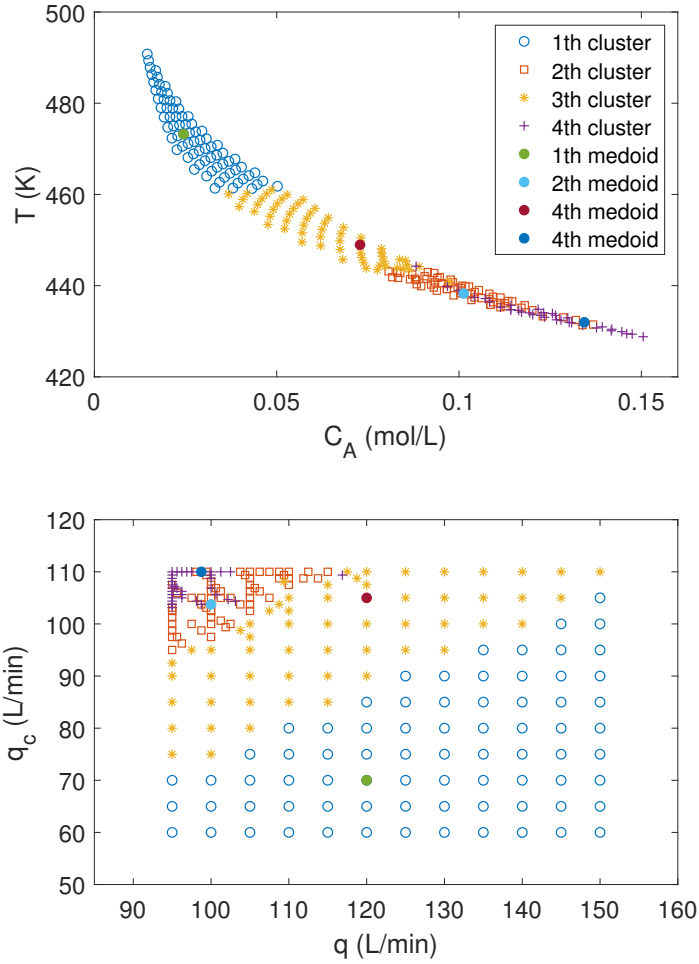

Figure 3.7: Clusters of steady-state input map and output map for MIMO CSTR

Table 3.5: Clustering result of MIMO CSTR

	1st	**2nd**	**3rd**	**4th**
θ_{med}	(120 ,70)	(95 ,100)	(120 ,105)	(98.75 ,110)
x_{med}	(0.025,473)	(0.099,438)	(0.073,449)	(0.013,432)
$\delta_{g,max}$	0.1053	0.2829	0.1747	0.3507
$\lambda_{max,med}$	0.21	0.51	0.42	0.49

Table 3.6: Parameters of Example 2

Local MPC		**Global MPC**	
Q_i	$\begin{pmatrix} 1000 & 0 \\ 0 & 1 \end{pmatrix}$	k_e	2
R_i	$\begin{pmatrix} 0.001 & 0 \\ 0 & 0.001 \end{pmatrix}$	μ	0.05
N	10	Λ^i	$\begin{pmatrix} 1000 & 0 \\ 0 & 10 \end{pmatrix}$
		Λ_e	$\begin{pmatrix} 400 & 0 \\ 0 & 0.0025 \end{pmatrix}$

y_g has the longest settling time in the second reference step. MLMPC II uses the last local MPC as the scheduled controller as shown in Fig 3.10. The gap metric between the last medoid and the first reference is 0.91. It can be interpreted that the large gap metric between the model and the state makes a slow transient response. Whereas, both y_p and y_c track the reference signal fast and accurately in the whole operating range. The overshoot of MLMPC I is lower than that of MLMPC III at the second reference step. As shown in Figure 3.10, MLMPC III is the combination of the second and the last local MPC. The gap metric between the second medoid and the second reference is 0.42, which is bigger than The gap metric between the last medoid and the second reference, 0.18. whereas, MLMPC I uses the last local MPC as the global MPC, which results in better transient response. For disturbance rejection, we inject step input disturbances whose sizes are (5,0), (5,5), and (5,-5) at 13 (min), 24 (min), and 32 (min), respectively. All controllers have similar performances as shown in Figure 3.9 and Table 3.7. However, MLMPC III has different weights between the second disturbance injection and the last disturbance injection. It shows the prediction accuracy of the third augmented model is similar to that of the first augmented model. The combination of the first and the third LQR controller stabilizes the system with the second input disturbance.

In summary, the proposed MLMPC controller outperforms the other two MLMPC controllers in terms of set-point tracking. The MLMPC I shows smooth transition compared to MLMPC II, and it converges to the local MPC that guarantees the stability compared to MLMPC III. Besides, the robust performance of the proposed controller is verified by the disturbance rejection test.

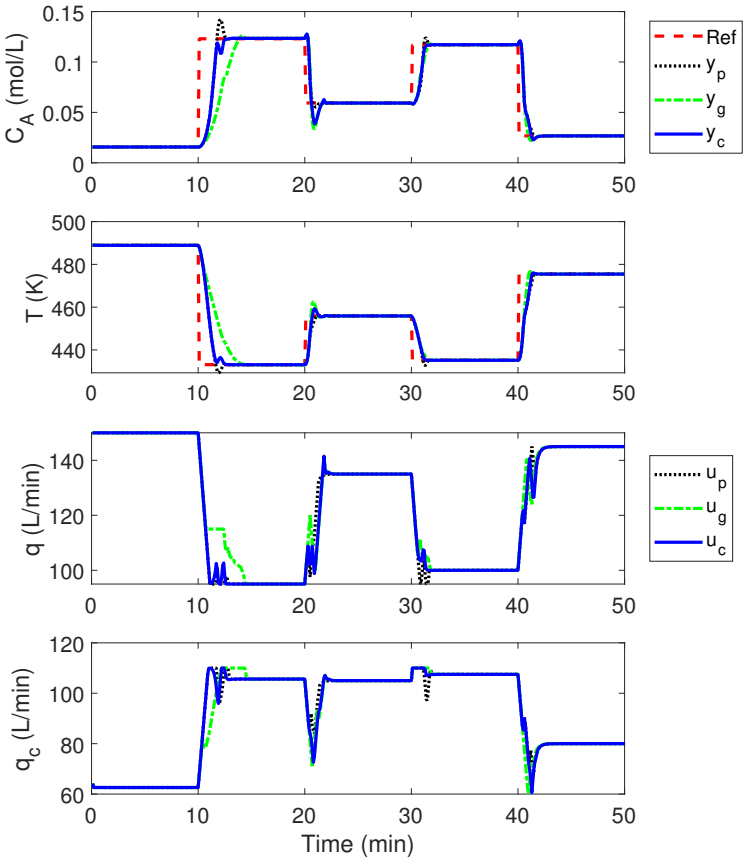

Figure 3.8: Set-point tracking for MIMO CSTR

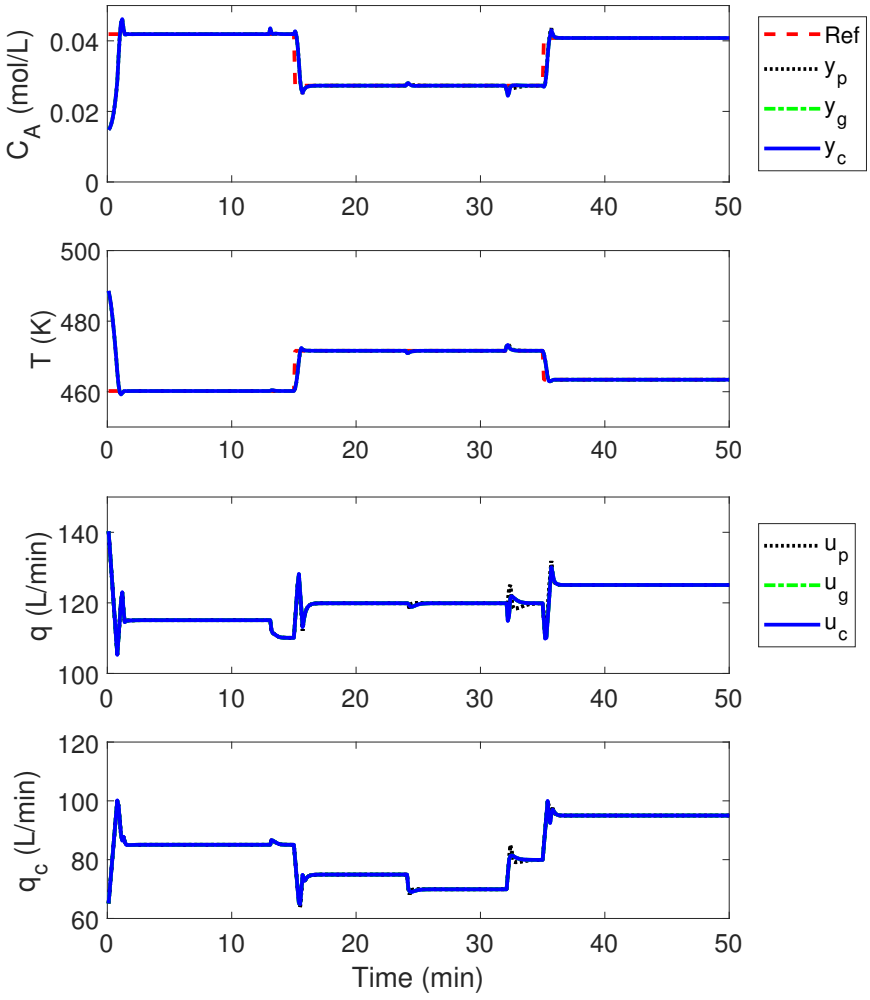

Figure 3.9: Disturbance rejection for MIMO CSTR

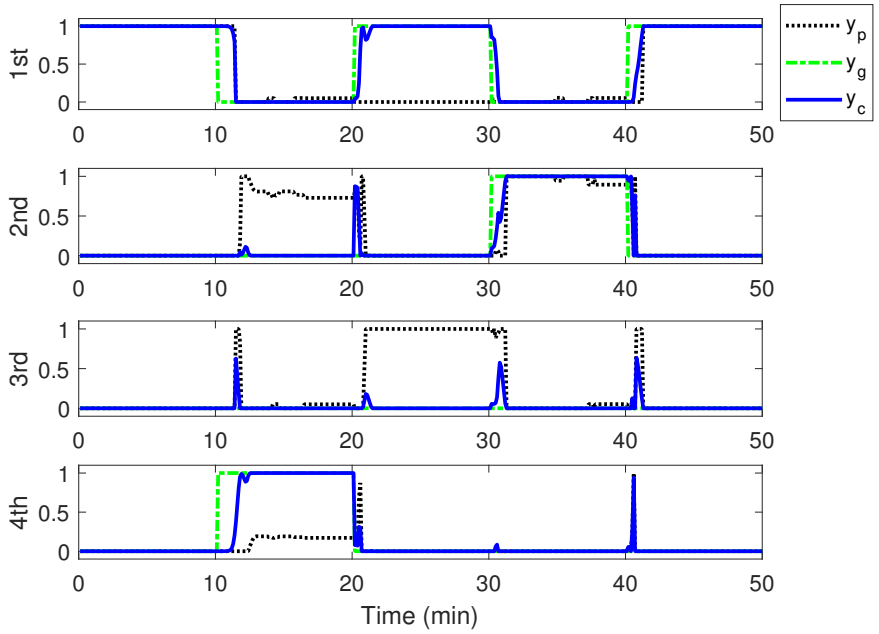

Figure 3.10: Weights of local controllers for tracking of MIMO CSTR

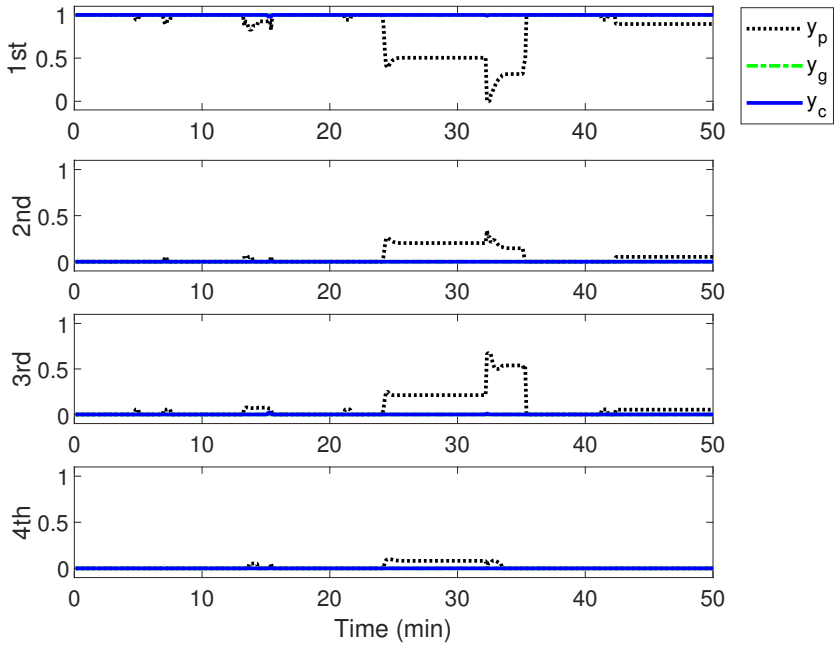

Figure 3.11: Weights of local controllers for disturbance rejection of MIMO CSTR

Table 3.7: Integral absolute error for MIMO CSTR

		Prediction based	**Gap based**	**Proposed**
SAE (normal)	$\mathbf{C_A}$	2.84	3.83	2.89
	$\mathbf{T}$	1035	1347	1023
SAE (disturbance rejection)	$\mathbf{C_A}$	0.3152	0.3150	0.3151
	$\mathbf{T}$	207.0	206.8	206.8

Chapter 4

Switching multilinear model predictive control based on gap metric [2]

4.1 Introduction

The prediction-based and gap metric-based approaches are combined to find the weights for local controllers in the previous chapter. It improves transient responses compared to the gap-metric based approach and guarantees the offset-free tracking property of transition in the subset of the operating region, called the subregion. However, this approach does not guarantee the reachability from one subregion to another subregion where the set-point is located. In this chapter, we propose to design local controllers and prove that each local controller has the offset-free tracking property if the behavior of the system is similar to the model. Second, we propose a method for determining the path along with the subregions from an initial point to a set-point in the operating region. Third, the switching strategy to design a global controller is proposed, which uses only one local controller at each sampling time. We show the resulting global controller

[2] This chapter is an adapted version of B. Park, Y. Kim, and J. M. Lee, "Design of switching multilinear model predictive control using gap metric," *Computers & Chemical Engineering*, Under review

guarantees the offset-free tracking property and the system stability at any set-point in a designated operating region.

4.2 Shortest path problem

Let V be a set of nodes and A be a set of directed edges between nodes in V. Given a directed graph $D = (V, A)$, two distinguished nodes $s, t \in V$, and nonnegative costs c_{ij} for $(i, j) \in A$, a minimum cost of the $s-t$ path is obtained by solving the shortest path problem. It is formulated as

$$
\begin{aligned}
z = &\min \sum_{(i,j)\in A} c_{ij}x_{ij} \\
&\sum_{k\in V^+(i)} x_{ik} - \sum_{k\in V^-(i)} x_{ki} = 1 \text{ for } i = s, \\
&\sum_{k\in V^+(i)} x_{ik} - \sum_{k\in V^-(i)} x_{ki} = 0 \text{ for } i \in V\backslash\{s,t\}, \\
&\sum_{k\in V^+(i)} x_{ik} - \sum_{k\in V^-(i)} x_{ki} = -1 \text{ for } i = t, \\
V^+(i) =& \{k|(i,k) \in A\}, \\
V^-(i) =& \{k|(k,i) \in A\}, \\
x_{ij} \in& \{0,1\} \text{ for } (i,j) \in A,
\end{aligned}
\tag{4.1}
$$

where $x_{ij} = 1$ if the directed edge (i, j) is in the shortest $s - t$ path. Figure 4.1 shows a instance of graph for shortest path problem. The shortest path problem is known as a well-solved problem in the sense that there exists an “efficient" algorithm for solving the problem. An algorithm on a graph $G = (V, A)$ with n nodes and m edges for $m \geq n$ is said “efficient" if, in the worst case, the algorithm requires $\mathcal{O}(m^p)$

elementary calculations for some integer p [54]. Some algorithms and their time complexities to solve this problem for directed graphs with nonnegative weights are shown in Table 4.1.

Remark 4.1. *A global controller can be designed using local controllers and the solution of the shortest path problem. Considering operating conditions as nodes, we can define a pair of two conditions as an edge if there exists a local controller such that any of two conditions can be moved to the other. Then, the shortest path problem can be exploited to determine the path from an initial operating condition to another condition. The plant is controlled by a local controller until the state is close to a node in the path. If the state reaches near the node, another controller is used to steer the state to another node constituting an edge in the path. Thus, the global controller is a set of local controllers based on a switching strategy.*

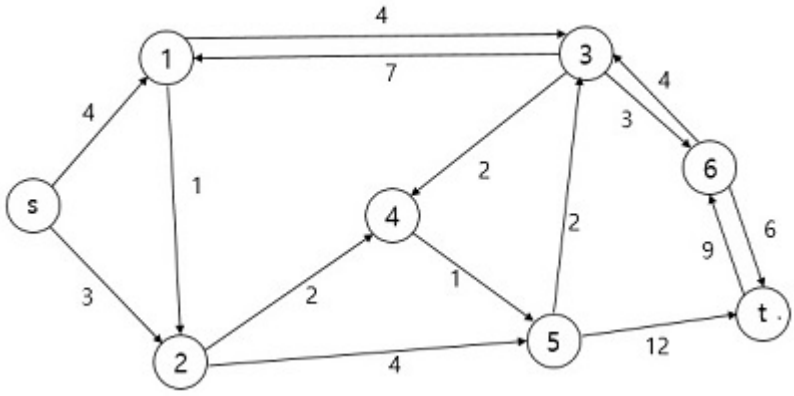

Figure 4.1: Shortest path instance

Table 4.1: Time complexity to solve the shortest path problem
(n, m: The number of nodes and edges)

Algorithm	**Time complexity**	**Author**
Bellman-–Ford algorithm	$\mathcal{O}(mn)$	Bellman [55], Shimbel [56], Moore [57]
Dijkstra's algorithm with list	$\mathcal{O}(n^2)$	Dijkstra [58]
Dijkstra's algorithm with binary heap	$\mathcal{O}((m+n)\log n)$	Johnson [59]
Dijkstra's algorithm with Fibonacci heap	$\mathcal{O}(m+n\log n)$	Fredman & Tarjan [60]
Thorup's algorithm	$\mathcal{O}(m+n\log\log n)$	Thorup [61]

4.3 Switching Multilinear Model Predictive Control

In this section, we propose a novel switching MLMPC algorithm which can guarantee the stability of the closed-loop system whose set-point is determined by a scheduling vector, regardless of the subregion to which it belongs. It consists of the design of the local MPC controllers, the path to the reference, and the switching strategy among the local MPC controllers based on gap metric.

4.3.1 Local MPC design

Before designing a local MPC that steers the state to a subregion, define a set of the scheduling vectors in the subregion Θ_j as a grid of Θ_j denoted by $\Theta^j_{grid} := \{\theta_i\} \subset \Theta_j$. Accordingly, the grid of Θ, Θ_{grid}, is defined by $\bigcup_{j=1}^{N_m} \Theta^j_{grid}$. Denoting the corresponding equilibrium point $F(\theta_i)$ by $s_i := (x_{s,i}, u_{s,i}, y_{s,i})$, the linearized system of Eq. (2.1) at s_i is obtained as and denoted by $P(\theta_i)$.

$$\begin{aligned} \dot{x} &= \left.\frac{\partial f}{\partial x}\right|_{x_{s,i},u_{s,i}} (x - x_{s,i}) + \left.\frac{\partial f}{\partial u}\right|_{x_{s,i},u_{s,i}} (u - u_{s,i}), \\ y &= \left.\frac{\partial h}{\partial x}\right|_{x_{s,i}} (x - x_{s,i}) + y_{s,i} \end{aligned} \tag{4.2}$$

Defining the representative scheduling vector in Θ^{grid}_j as the medoid of Θ_j and denoting it by $\theta^r_j \in \Theta^j_{grid}$, the grid Θ^j_{grid} is assumed to satisfy Assumption 4.1:

Assumption 4.1. *Let the convex hull of a grid Θ^j_{grid} be Conv(Θ^j_{grid})*

and contain Θ_j. *Any* $P(\theta_i)$ *from* $\theta_i \in Conv(\Theta^j_{grid})$ *satisfies*

$$\delta_g(P(\theta^r_j), P(\theta_i)) \leq \max_{\theta_s \in \Theta^j_{grid}} \delta_g(P(\theta^r_j), P(\theta_s)).$$

Suppose $\theta_1 \in \Theta^1_{grid}$ has the maximum gap metric with $\theta^r_1 \in \Theta^1_{grid}$. Assumption 4.1 means that if a controller K_1 that stabilizes $P(\theta^r_1)$ can stabilize $P(\theta_1)$, K_1 can stabilize any $P(\theta_k)$ from $\theta_k \in \Theta_1$. That is, we can design a local controller that can stabilize the system (2.1) at any equilibrium point in a subregion. Assumption 4.1 is not restrictive because it can be satisfied by refining the grid.

Remark 4.2. *To refine the grid, the threshold value* γ_{th} *can be defined so that the gap metric of any two adjacent equilibrium points is less than* γ_{th}. *If* γ_{th} *is small enough, there exists a subregion satisfying Assumption 4.1, because one of the scheduling vectors of its grid can approximate any the scheduling vector in the subregion.*

The following lemma is necessary to determine the stability of the closed-loop system at an equilibrium point.

Lemma 4.1. *[62] Let* $x = 0$ *be an equilibrium point for the nonlinear system*

$$\dot{x} = f(x),$$

where $f : D \to \mathbb{R}^n$ *is continuously differentiable and* D *is a neighborhood of the origin.*

$$A = \frac{\partial f}{\partial x}(x)\bigg|_{x=0}$$

Then,

1. The origin is asymptotically stable if $Re(\lambda_i) < 0$ *for all eigenvalues* λ_i *of* A*.*
2. The origin is unstable if $Re(\lambda_i) > 0$ *for one or more of the eigenvalues of* A*.*

The local linear MPC designed in Section 3 can be a local controller for each subregion under some conditions. Theorem 4.1 shows that (2.6), (2.9), and (2.10) can stabilize an equilibrium point if the gap metric between the linearized dynamics at the equilibrium point and the local linear model used in MPC is smaller than the gap stability margin of the pair of the model and the controller in the MPC.

Theorem 4.1. *Suppose the space of scheduling vector* Θ *and its grid* Θ_{grid} *of a nonlienar process* (2.1) *satisfy Assumptions 3.2, 3.3, and 4.1. For* $\theta_m \in \Theta_{grid}$ *and* $\theta \in \Theta$*, assume that the MPC described in* (2.6)*,* (2.9)*, and* (2.10) *is applied to the system* (2.1) *with a linear model* $P_m := P_{\theta_m}$ *to stabilize the system at an output* r *whose corresponding scheduling vector is* θ*. Denote* $F(\theta)$ *by* s_θ *and let* $P(\theta)$ *be the linearized model of* (2.1) *at* s_θ*. Assume that* $Q_{T,\theta}$ *is the matrix for the cost of LQR with* P_m *and the weights* Q_θ *and* R_θ *in* (2.9)*, and* K_{lqr} *is the gain of the corresponding LQR controller. If* $b_{P_m,K_{lqr}} > \delta_g(P_m, P(\theta))$*, then the closed-loop system Eqs.* (2.1)*,* (2.6)*,* (2.9)*, and* (2.10) *at the set-point* r *is asymptotically stable.*

Proof *Denote the state at steady-state by* x_θ*. If the state at the* k^{th} *sampling time* $x(k)$ *is in the neighborhood of* x_θ*, all the constraints in the MPC formulation of* (2.9) *are inactive. Then, the MPC* (2.9) *is*

reduced to the LQR [1]:

$$\begin{aligned} J^*_\infty(\delta x_0) = \min \sum_{k=0}^{\infty} \|\delta x_k\|^2_{Q_\theta} + \|\delta u_k\|^2_{R_\theta} = \|\delta x_0\|^2_{Q_{T,\theta}} \\ s.t.\ \delta x_{k+1} = A_\theta \delta x_k + B_\theta \delta u_k \\ \delta x_0 = \hat{x}(k) - \bar{x}(k),\ \delta u_0 = u(k) - \bar{u}(k). \end{aligned} \tag{4.3}$$

Then the closed-loop system between the k^{th} and the $(k+1)^{th}$ sampling time is

$$\begin{aligned} \frac{dx}{dt} &= f(x(t), -K_{lqr}\delta x_0 + \bar{u}(k)), \\ y &= h(x), \end{aligned} \tag{4.4}$$

where $K_{lqr} = (R_\theta + B_\theta^T Q_{T,\theta} B_\theta)^{-1} B_\theta^T Q_{T,\theta} A_\theta$. The linearized and discretized system of (4.4) *at the equilibrium point is*

$$\begin{aligned} \delta x(k+1) &= A_\theta \delta x(k) + B_\theta \delta u(k) \\ &= (A_\theta - B_\theta K_{lqr})\delta x(k), \\ \delta y(k) &= C_\theta \delta x(k), \end{aligned} \tag{4.5}$$

where $\delta x(k) = x(k) - x_\theta$; $\delta u(k) = u(k) - u_\theta$; $\delta y(k) = y(k) - r$; A_θ, B_θ, and C_θ are the matrices corresponding to $P(\theta)$; u_θ is the steady-state input at x_θ. Because LQR gain K_{lqr} satisfies Theorem 2.3, the autonomous system (4.5) *is asymptotically stable and all the absolute eigenvalues of $(A_\theta - B_\theta K_{lqr})$ are less than one. Because the continuous matrix transformed from $(A_\theta - B_\theta K_{lqr})$ is Hurwitz matrix, (x_θ, u_θ) in the closed-loop system is asymptotically stable by Lemma 4.1.* ■

Thus, we can guarantee the stability at all the equilibrium points in the operating region corresponding to Θ if Theorem 4.1 is satisfied. However, if the operating range is large, the MPC satisfying Theorem 4.1 may not exist. Thus, the operating region is partitioned into the subregions such that there exists a local MPC satisfying Theorem 4.1 for each subregion. Generating subregions based on gap metric have been proposed for multiple MPC [63, 64, 46]. In this study, gridding and K-medoids clustering algorithms based on gap metric are exploited to construct the grid and subregions of Θ [65]. Because it is not guaranteed that the number of linear models increases linearly, caution must be exercised when the method is applied to high dimensional systems.

Remark 4.3. *The switching method may have stability when moving between two adjacent subregions. Assume that the i^{th} subregion is adjacent to the j^{th} subregion. If there exists an equilibrium point in the i^{th} subregion such that the j^{th} local MPC can steer the point to the j^{th} subregion with stability, and vice versa, the switching method can construct a global controller that can steer the state to any equilibrium point corresponding to $\theta \in \Theta$.*

4.3.2 Path design based on gap metric

Even if we construct the subregions and the local MPCs, it is unclear how the state is steered from one subregion to another subregion by the local MPCs. If a local controller in a subregion can steer the state into other subregions, A global controller can be designed by the local MPCs as we mentioned in Remark 4.1 and 4.3. To find the equilibrium points in a subregion that the state in another subregion can

reach by the local MPC, we first propose to construct the boundary of a subregion Θ_i using the grid Θ^i_{grid} in Algorithm 5. The algorithm finds the vertices in the convex hull of the grid of a subregion and the points on the facets of the convex hull excepting the vertices.

Algorithm 5: Calculation of the boundaries of subregions

Input :
The set of the subregions of Θ: $\{\Theta^i_{grid}\}$
for $i \leftarrow 1$ **to** $|\{\Theta^i_{grid}\}|$ **by** 1 **do**
  $\Theta^i_{bd} \leftarrow \emptyset$
  $\{\theta^{temp}_k\} \leftarrow \Theta^i_{grid}$
  $\Theta^i_{bd} \leftarrow$ The set of the vertices of $Conv(\Theta^i_{grid})$
  Set $\{A^j_{facet}\}$ as the set of $n_\theta \times n_\theta$ matrices where the columns of A^j_{facet} constitute the j^{th} facet of $Conv(\Theta^i_{grid})$
  for $j \leftarrow 1$ **to** $|\{A^j_{facet}\}|$ **by** 1 **do**
    for $k \leftarrow 1$ **to** $|\{\theta^{temp}_k\}|$ **by** 1 **do**
      $b \leftarrow (A^j_{facet})^{-1}\theta^{temp}_k$
      if $0 \leq b_i < 1$ *and* $\sum_{i=1}^{n_\theta} b_i == 1$ **then**
        $\Theta^i_{bd} \leftarrow \Theta^i_{bd} \cup \{\theta^{temp}_k\}$
      end
    end
  end
end
Save $\{\Theta^i_{bd}\}$

The equilibrium points corresponding to a boundary Θ_{bd}^{i} are the candidates where the state from another subregion Θ_j can move by the local MPC for Θ_j. We then propose to construct the pairs between two subregions in Algorithm 6. Each pair of them consists of a point in the boundary of a subregion and the closest point in the boundary of another subregion. Because the components of the scheduling vector have different magnitudes, we define a normalized scheduling vector as

$$\theta_{k,n_k}^{n} = \frac{\theta^{k,n_k} - \theta^{k,min}}{\theta^{k,max} - \theta^{k,min}}, \tag{4.6}$$

where $\theta^{k,min}$ and $\theta^{k,max}$ are the lower and upper bounds of the k^{th} component of the scheduling vector, respectively. Thus, the components of the scheduling vector are scaled between zero and one. Then, we define a matrix D^e whose components are denoted by $d_{i,j}^{e}$, which is the Euclidean distance between the i^{th} and j^{th} normalized scheduling vector. Similarly, D^g is defined in terms of the gap metric.

Even if the pairs generated by Algorithm 6 can be channels between two subregions, it is unclear whether a local MPC steers the state from a point to the other point in a pair. Thus, we propose Algorithm 7 to choose the pairs where each point in a pair is close to the subregion the other point belongs to in terms of both the gap metric and Euclidean distance. In addition, we add the condition that each point in a pair is stabilized by the local MPC for the subregion the other point belongs to.

Algorithm 6: Calculation of the pairs between points from boundaries

Input :
The set of the boundaries of $\{\Theta_i\}$: $\{\Theta_{bd}^i\}$
The matrix of the Euclidean distance between scheduling vectors: D^e
The matrix of the gap metric between the linear systems corresponding to scheduling vectors: D^g

for $i \leftarrow 1$ **to** $|\{\Theta_{bd}^i\}| - 1$ **by** 1 **do**
  $\Theta_{bd}^{temp1} \leftarrow \Theta_{bd}^i$
  for $j \leftarrow i + 1$ **to** $|\{\Theta_{bd}^i\}|$ **by** 1 **do**
    $\Theta_{bd}^{temp2} \leftarrow \Theta_{bd}^j$
    $D_{temp}^e \leftarrow (d_{a,b}^{e,temp}), D_{temp}^g \leftarrow (d_{a,b}^{g,temp})$, where
    $d_{a,b}^{e,temp} := d_{a,b}^e, d_{a,b}^{g,temp} := d_{a,b}^g,\ \theta_a \in \Theta_{bd}^{temp1}, \theta_b \in \Theta_{bd}^{temp2}$
    $IDX_{pair1} \leftarrow \{\}, IDX_{pair2} \leftarrow \{\}$
    for $a \leftarrow 1$ **to** $|\Theta_{bd}^{temp1}|$ **by** 1 **do**
      $b_{temp} \leftarrow \mathrm{argmin}_{1 \le b \le |\Theta_{bd}^{temp2}|} d_{a,b}^{e,temp}$
      $IDX_{pair1} \leftarrow IDX_{pair1} \cup \{(a, b_{temp})\}$
      $IDX_{pair2} \leftarrow IDX_{pair2} \cup \{(b_{temp}, a)\}$
    end
    for $b \leftarrow 1$ **to** $|\Theta_{bd}^{temp2}|$ **by** 1 **do**
      $a_{temp} \leftarrow \mathrm{argmin}_{1 \le a \le |\Theta_{bd}^{temp1}|} d_{a,b}^{e,temp}$
      $IDX_{pair1} \leftarrow IDX_{pair1} \cup \{(a_{temp}, b)\}$
      $IDX_{pair2} \leftarrow IDX_{pair2} \cup \{(b, a_{temp})\}$
    end
    $idx_p^{i,j} \leftarrow IDX_{pair1}$
    $idx_p^{j,i} \leftarrow IDX_{pair2}$
  end
end
Save $IDX_p := \{idx_p^{i,j}\}$

Algorithm 7: Filtering of the pairs between points from boundaries

Input :

$\{\Theta^i_{bd}\}$, D^e,D^g, IDX_p

The threshold for the 2-norm and gap metric : $\gamma^e_{th}, \gamma^g_{th}$

The gain of local MPCs in unconstrained regions : $\{K^i_{lqr}\}$

for $i \leftarrow 1$ **to** $|\{\Theta^i_{bd}\}| - 1$ **by** 1 **do**

  for $j \leftarrow i + 1$ **to** $|\{\Theta^i_{bd}\}|$ **by** 1 **do**

    if $idx_p^{i,j} \neq \emptyset$ **then**

      $idx_p^{temp1} \leftarrow idx_p^{i,j}, idx_p^{temp2} \leftarrow idx_p^{j,i}$

      $D^e_{temp} \leftarrow (d^{e,temp}_{a,b}), D^g_{temp} \leftarrow (d^{g,temp}_{a,b})$, where

       $d^{e,temp}_{a,b} := d^e_{a,b}, d^{g,temp}_{a,b} := d^g_{a,b},\ \theta_a \in \Theta^i_{bd}, \theta_b \in \Theta^j_{bd}$

      for $k \leftarrow 1$ **to** $|idx_p^{temp1}|$ **by** 1 **do**

        Pick the k^{th} element (a, b) in idx_p^{temp1}

        $\alpha, \beta \leftarrow True$

        if $(P(\theta_a), K^j_{lqr})$ *is asymptotically stable* **then**

          $\alpha \leftarrow False$

        end

        if $(P(\theta_b), K^i_{lqr})$ *is asymptotically stable* **then**

          $\beta \leftarrow False$

        end

        if $d^{e,temp}_{a,b} > \gamma^e_{th}$ *or* $d^{g,temp}_{a,b} > \gamma^g_{th}$ *or* α *or* β **then**

          $idx_p^{temp1} \leftarrow idx_p^{temp1} - \{(a, b)\}$

          $idx_p^{temp2} \leftarrow idx_p^{temp2} - \{(b, a)\}$

        end

      end

    end

    $idx_p^{i,j} \leftarrow idx_p^{temp1}, idx_p^{j,i} \leftarrow idx_p^{temp2}$

  end

end

Save $IDX_p := \{idx_p^{i,j}\}$

According to the sets of the pairs generated by Algorithm 6 and 7, we define the adjacent subregions of a subregion.

Definition 4.1. *The adjacent subregions of a subregion Θ_i is defined and denoted by*

$$adj(\Theta_i) = \{\Theta_j | idx_p^{i,j} \neq \emptyset, \forall \Theta_j \subset \Theta\} \tag{4.7}$$

where $idx_p^{i,j}$ is the set of pairs constructed by Algorithms 6 and 7. If Θ_j is an adjacent subregion of Θ_i, there exists an equilibrium point in the i^{th} subregion such that the j^{th} local MPC can steer the point to the j^{th} subregion with stability, and vice versa. The following is assumed for the subregions $\{\Theta_i\}$:

Assumption 4.2. *Assume a set of subregions, the union of which is a proper subset of Θ. Then, the union of the adjacent subregions of the subregions is not the same as the union of the subregions.*

Assumption 4.2 means that there exists a path between any two subregions among the adjacent subregions. Thus, the switching method can construct a global controller that can steer the state to any equilibrium point corresponding to $\theta \in \Theta$.

4.3.3 Global MPC design

As we find the channels through which the state in a subregion can be steered to another subregion by a local MPC, the remained problem is how to decide the path between an initial state and a set-point. Here we propose to determine the path using the shortest path algorithm. To construct the graph for the shortest path problem, we propose two methods to decide the nodes, edges, and

edge costs of a graph and the intermediate equilibrium points between an initial state and a set-point in Algorithms 8 and 9. In Algorithm 8, a node is defined as each subregion in the operating region, denoting the set of the nodes by $V := \{i | \exists \Theta_i \subset \Theta \text{ for } i\}$. An edge is defined as a set of two subregions if one subregion is an adjacent subregion of the other, denoting the set of the edges by $A := \{\{i, j\} | \exists idx_p^{i,j} \subset IDX_p \text{ for } i, j \in V\}$. The intermediate equilibrium points Θ^{path} are chosen among the pairs generated by Algorithms 6 and 7 based on gap metric. It maximizes the gap metric stability margin of a local MPC at each intermediate equilibrium point that belongs to an adjacent subregion. In addition, it is independent of the initial state and set-point. It only depends on the subregions where they belong to. Thus, the graph can be used regardless of the initial state and set-point.

Algorithm 8: Graph construction 1

Input :
$\{\Theta_{bd}^i\}$, D^e,D^g, IDX_p
The medoids where local MPCs are constructed: $\{\theta_i^r\}$
The set of the nodes : $V = \{i|\Theta_i \subset \Theta\}$
The set of the edges : $A = \{\{i,j\}|idx_p^{i,j} \in IDX_p \text{ for } i,j \in V\}$.
for $i \leftarrow 1$ **to** $|\{\Theta_{bd}^i\}| - 1$ **by** 1 **do**
  for $j \leftarrow i+1$ **to** $|\{\Theta_{bd}^i\}|$ **by** 1 **do**
    if $idx_p^{i,j} \neq \emptyset$ **then**
      $idx_p^{temp} \leftarrow idx_p^{i,j}$
      for $k \leftarrow 1$ **to** $|idx_p^{temp}|$ **by** 1 **do**
        Pick the k^{th} element (a,b) in idx_p^{temp}
        Calculate for $\theta_a \in \Theta_{bd}^i, \theta_b \in \Theta_{bd}^j$
        $c_{ij}^k \leftarrow \delta_g(P(\theta_i^r), P(\theta_b)) + \delta_g(P(\theta_j^r), P(\theta_a))$
      end
    end
    $k^* \leftarrow \arg\min_k c_{ij}^k$, $c_{ij} \leftarrow \min_k c_{ij}^k$
    Set c_{ij} as the edge cost of $\{i,j\} \in A$
    $\theta_{ij}^{path} \leftarrow \theta_b, \theta_{ji}^{path} \leftarrow \theta_a$ for the k^{th} element (a,b) in idx_p^{temp}
  end
end
Save undirected graph $G = (V, A)$ with cost $C := \{c_{ij}\}$
Save $\Theta^{path} := \{\theta_{ij}^{path}\}$

Algorithm 9: Graph construction 2

(Offline)

Input : $\{\Theta^i_{bd}\}$, D^e,D^g, IDX_p

The set of the nodes : $V_0 = \{a|(a,b) \in idx_p^{i,j} \in IDX_p\}$

The set of the edges : $A_0 = \{(a_1,b_2)|(a_1,b_1) \in$
$\bigcup_{k\neq j} idx_p^{i,k}, (a_2,b_2) \in idx_p^{i,j}, |adj(\Theta_i)| \geq 2\}$.

for $(a,b) \in A$ **do**
  $c_{ab} \leftarrow \|\theta_a^n - \theta_b^n\|_2$
end

(Online)

The initial and terminal scheduling vectors : $\theta_s \in \Theta^i_{grid}, \theta_t \in \Theta^j_{grid}$

$V_s \leftarrow \{a|\theta_a \in \Theta^i_{grid}\}, V_t \leftarrow \{b|\theta_b \in \Theta^j_{grid}\}$

$A_s \leftarrow \{(a,b)|\theta_a \in \Theta^i_{grid}$ or $\theta_b \in \Theta^i_{grid}\}$

$A_t \leftarrow \{(a,b)|\theta_a \in \Theta^j_{grid}$ or $\theta_b \in \Theta^j_{grid}\}$

$V \leftarrow V_0 - (V_s \cup V_t), A \leftarrow A_0 - (A_s \cup A_t)$

$V \leftarrow V \cup \{s,t\}$

for $(a,b) \in \bigcup_k idx_p^{i,k}$ **do**
  $A \leftarrow A \cup (s,b), c_{sb} \leftarrow \|\theta_s^n - \theta_b^n\|_2$
end

for $(a,b) \in \bigcup_k idx_p^{k,j}$ **do**
  $A \leftarrow A \cup (a,t), c_{at} \leftarrow \|\theta_a^n - \theta_t^n\|_2$
end

Save directed graph $G = (V,A)$ with cost $C := \{c_{ij}\}$

Compared to Algorithm 8, Algorithm 9 can reduce the length of the path in terms of the Euclidean distance between an initial point and a set-point. Thus, it can be a better suboptimal solution if the cost function is quadratic with respect to the deviation of the state from the state at a set-point. The drawback is that the graph is reorganized whenever the set-point changes; however, the time complexity is $\mathcal{O}(m)$, which is negligible compared to that of the shortest path problem in Table 4.1, where n and m are the number of the nodes and edges, respectively.

Based on Algorithm 6 or 7, the global controller is proposed in Algorithm 10. The shortest path problem determines the intermediate state, which is the path from an initial operating condition to another. The plant is controlled by a local MPC until the state is close to an intermediate state in the path. Algorithm 10 considers the state is at steady-state if there remains 5% of the set-point for each output component [66]. If the state reaches the intermediate state, another local MPC is used to steer the state to another intermediate state constituting an edge in the path. Thus, the global controller is a set of local controllers based on the switching strategy.

Algorithm 10: Switching MMPC

Input : $\theta_0 \in \Theta_{n_0}, \theta_{n_{set}} \in \Theta_{set}, V, A, C, N_s, n_{set}$
Solve the shortest path problem (4.1) from θ_0 to θ_{set}
Set the intermediate vectors $\{\theta_1 \in \Theta_{n_1}, \cdots, \theta_t \in \Theta_{n_t}\}$ from the solution of (4.1)
$temp \leftarrow 0, count \leftarrow 0$
Set the reference output r as the output at θ_{temp+1}
for $k \leftarrow 0$ **to** N_s **by** 1 **do**
    Measure the output ate the k^{th} sampling time : $y(k)$
    Check $\{y_i^j(k)\}$, all the components of the output $y_i(k)$, are at steady-state:
    if $\left\| y_{ref,i}^j - y_i^j(k) \right\| / \left\| y_{ref,i}^j \right\| \leq 0.04$ **then**
        $count \leftarrow count + 1$
    end
    if $count \geq n_{set}$ *and* $temp < t$ **then**
        $temp \leftarrow temp + 1, count \leftarrow 0$
        if $temp == t$ **then**
            Set the reference output r as the output at θ_{set}
        else
            Set the reference output r as the output at θ_{temp+1}
        end
    end
    Solve the local MPC (2.6), (2.9), (2.10) for the subregion $\Theta_{n_{temp}}$
    Apply the first input of the solution to the plant (2.1)
    $u(k) \leftarrow u_0^{opt} + u_{s,n_{temp}}^r$
end

Theorem 4.2. *Suppose the space of scheduling vector Θ, its grid Θ_{grid}, and its subregions $\{\Theta_i\}$ of a nonlienar process* (2.1) *satisfy Assumptions 3.2, 3.3, 4.1, and 4.2. Assume that the intermediate points are determined by Algorithms 5, 6, and 7, and the shortest path of the graph obtained by Algorithm 8 or 9 is decided. Then, the global MPC of Algorithm 10 can steer the state of* (2.1) *between two arbitrary states in the steady-state if the scheduling vectors corresponding to the states are contained in Θ.*

Proof *Denote the scheduling vectors and the subregions corresponding to the initial point, the set-point, and the shortest path by $\theta_0 \in \Theta_0$, $\theta_n \in \Theta_n$, and $\{\theta_i | \theta_i \in \Theta_i, i = 1, \cdots, n-1\}$. The local MPC satisfies Theorem 4.1 for the subregion Θ_i by C_i. C_i can stabilize θ_{i+1} because Algorithm 7 checks the stability of $(P(\theta_{i+1}), C_i)$. Thus, θ_i can be steered to θ_{i+1} by C_i without offset. Thus, the global MPC can steer the state from θ_0 to θ_n.* ■

4.4 Results and discussions

Consider a multi-input multi-output stirred tank reactor (CSTR) [53].

$$\begin{aligned}
\dot{C}_A(t) =& \frac{q}{V}[C_{A0} - C_A(t)] - k_0 C_A(t) \mathrm{e}^{-E/RT(t)}, \\
\dot{T}(t) =& \frac{q}{V}[T_0 - T(t)] - \frac{\Delta H k_0}{\rho C_p} C_A(t) \mathrm{e}^{-E/RT(t)} \\
&+ \frac{\rho_c C_{pc}}{\rho C_p V} q_c(t)[1 - \mathrm{e}^{-\frac{h_A}{\rho_c C_{pc} q_c(t)}}][T_{c0} - T(t)], \\
y(t) =& [C_A(t) \;\; T(t)]^T.
\end{aligned} \tag{4.8}$$

It consists of an irreversible, exothermic reaction. The concentration C_A and the temperature T are controlled by manipulating the flow rate of A, q and the coolant flow rate, q_c. The parameters and initial values of the variables in the system are described in Table 4.2.

Table 4.2: MIMO CSTR Parameters and Initial Values

Product concentration	C_A	0.1 mol/L
Coolant flow rate	q_c	103.41 L min^{-1}
Feed concentration	C_{A0}	1 mol/L
Inlet coolant temperature	T_{C0}	350 K
Heat transfer term	h_A	7×10^5 cal/min K
Activation energy term	E/R	1×10^4 K
Liquid densities	ρ, ρ_c	1×10^3 g/L
Reactor temperature	T	438.51 K
Process flow rate	q	100 L min^{-1}
Feed temperature	T_0	350 K
CSTR volume	V	100 L
Heat of reaction	ΔH	-2 $\times$ 10 ^{5} cal/mol
Specific heats	C_p, C_{pc}	1 cal g $^{-1}$ K $^{-1}$
Reaction rate constant 1	k_0	7.2×10^{10} min $^{-1}$
Constraints on the flow rate	q_{min}, q_{max}	95, 150 L min^{-1}
Constraints on the coolant flow rate	$q_{c,min}, q_{c,max}$	60, 110 L min^{-1}

The steady-state input-output relationship is shown in Figure 4.2. The manipulated variables are chosen as the scheduling variables as they determine the equilibrium point uniquely. The operating region is divided into four subregions by the gridding and clustering method in [65]. The clusters and the medoids with 231 grid points are obtained as shown in Figure 4.3 and Table 4.3. Each cluster is connected and the maximum gap metric of each medoid does not exceed 0.36. The maximum spectral radius of the closed-loop systems controlled by the LQR controller in a cluster is less than 1 as shown in Table 4.3. The candidates for the intermediate equilibrium points are generated by Algorithms 5, 6, and 7 and are shown Figure 4.4. Setting both γ_{th}^{e} and γ_{th}^{g} as 0.1 in Algorithm 7, the candidates are close to the adjacent subregions in terms of the gap metric and Euclidean distance. Figure 4.5 shows the possible paths from an initial point and a set-point. Algorithm 8 leaves only a single path between two adjacent subregions so that the edges are reduced. On the other hand, Algorithm 9 exploits all the candidates for the possible paths. To check offset-free tracking, the state and measurement noises are not added. Luenberger observer is chosen for the filter as it is convenient to control the response time by changing the poles of the observer. Also, the performance of a controller is measured by the summation of absolute errors (SAE). The SAE of an output is defined as $\sum_{k=1}^{N} |e(k)|$, where $e(k)$ is the error of the output at k^{th} sampling time and N is the number of samples.

Table 4.3: Clustering result of MIMO CSTR

	1st	**2nd**	**3rd**	**4th**
θ_{med}	(120 ,70)	(95 ,100)	(120 ,105)	(98.75 ,110)
x_{med}	(0.025,473)	(0.099,438)	(0.073,449)	(0.013,432)
$\delta_{g,max}$	0.1053	0.2829	0.1747	0.3507
$\lambda_{max,med}$	0.21	0.51	0.42	0.49

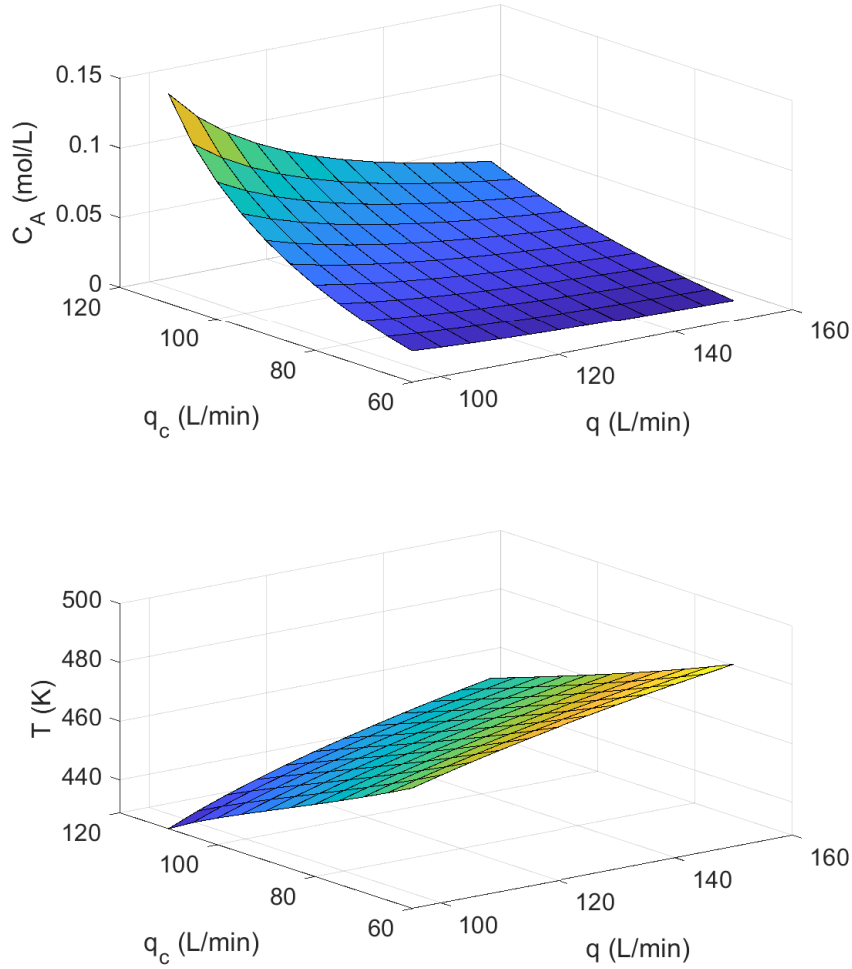

Figure 4.2: Steady-state input-output map for MIMO CSTR

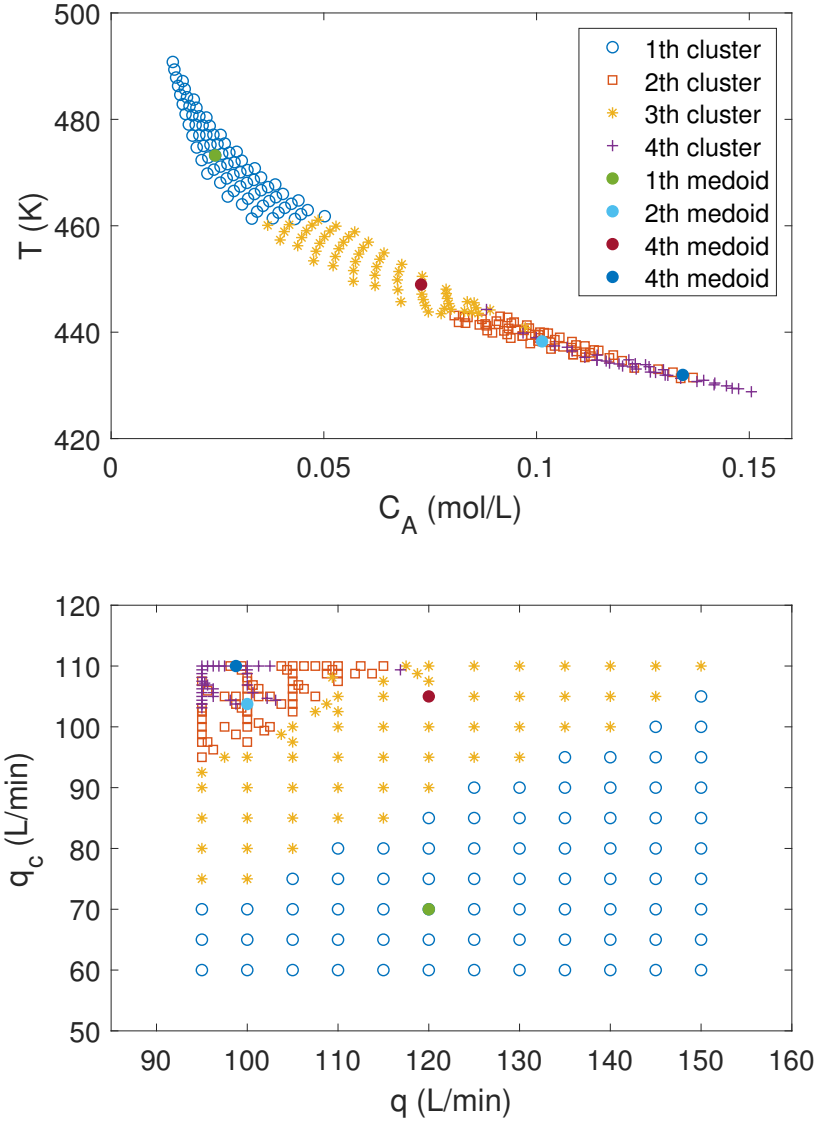

Figure 4.3: Subregions of steady-state input and output for MIMO CSTR

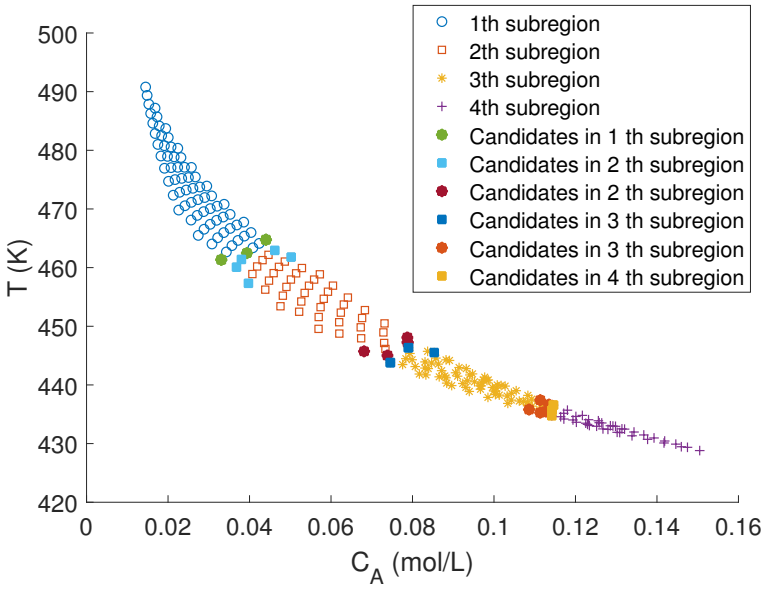

Figure 4.4: Candidates of the intermediate points for MIMO CSTR

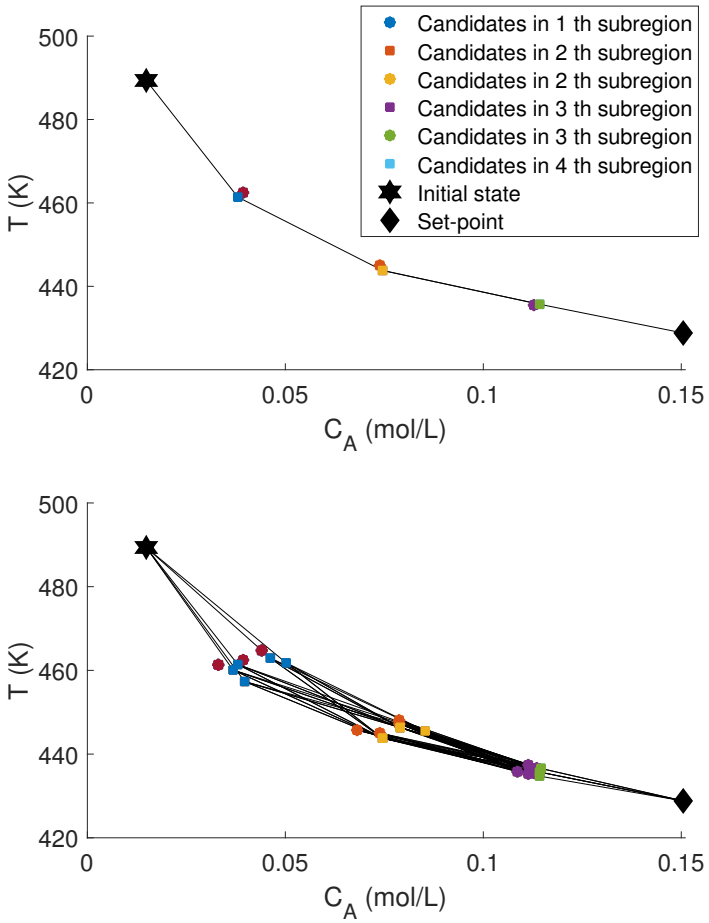

Figure 4.5: Possible paths from an initial point and the set-point for MIMO CSTR (Top: Algorithm 8, Bottom: Algorithm 9)

Set-point tracking control and disturbance rejection are tested for the controllers based on Algorithms 6 and 7. The parameters for MPC are shown in Table 4.4, and the results are shown in Table 4.5 and Figures 4.6, 4.7, and 4.8.

Denoting the global MPC designed by Algorithms 8 and 9 by MLMPC I and MLMPC II, respectively, MLMPC I and MLMPC II track all the references without offset. MLMPC I has the shorter settling time than MLMPC II in the first reference step in Figure 4.6. It seems that there is no difference after the first reference step. One reason can be small changes of the subregions during the changes of the references. In the first reference step, all the local MPCs are involved. However, three, two, and, three local MPCs are involved in the second, third, and fourth reference step, respectively. Table 4.5 shows the mean value of SAE when an initial condition and a set-point are chosen in the grid of a subregion and another subregion, respectively. The main difference comes from the transition between Θ_1 and Θ_2 and that between Θ_1 and Θ_4. Almost 10% of difference in the temperature is observed, and the mean values of the SAEs of all transitions in Table 4.5 are $(2.23, 970)$ and $(2.29, 1002)$ for MLMPC I and II, respectively. It shows that MLMPC I is better than MLMPC II in terms of SAE, especially when the initial state is far from the set-point in terms of Euclidean distance and gap metric. For disturbance rejection, we inject step input disturbances whose size are (20,0), (0,15), and (0,-20) at 10 (min), 24 (min), and 32 (min), respectively. All controllers reject the disturbances immediately.

In summary, MLMPC I shows smooth transition compared to MLMPC II, considering the gap metric between the medoid and the reference. In addition, the robust performance of the proposed con-

troller is verified by the disturbance rejection test. The relationship between the gap metric and the transient response should be investigated in future research.

Note that it is unclear why MLMPC I is better than MLMPC II in terms of SAE. One speculation is that MLMPC I can be better suboptimal than MLMPC II at the intermediate points. To design MLMPC I, the gap metric between a medoid and an intermediate point is minimized, which means that the LQR controller constructed by a medoid is close to that constructed by an intermediate point. Thus, the rate of convergence at an intermediate point can be fast when MLMPC I is used instead of MLMPC II. Although the distance between an initial point and a set-point is minimized in MLMPC II, the rate of convergence from an intermediate point to another intermediate point can be slow because of the suboptimal controller. In this case, the trajectory of the state does not seem to be a straight line.

Table 4.4: Parameters of MPC for MIMO CSTR

Local MPC		**Global MPC**	
Q_i	$\begin{pmatrix} 1000 & 0 \\ 0 & 1 \end{pmatrix}$	n_{set}	5
R_i	$\begin{pmatrix} 0.001 & 0 \\ 0 & 0.001 \end{pmatrix}$		
N	10		

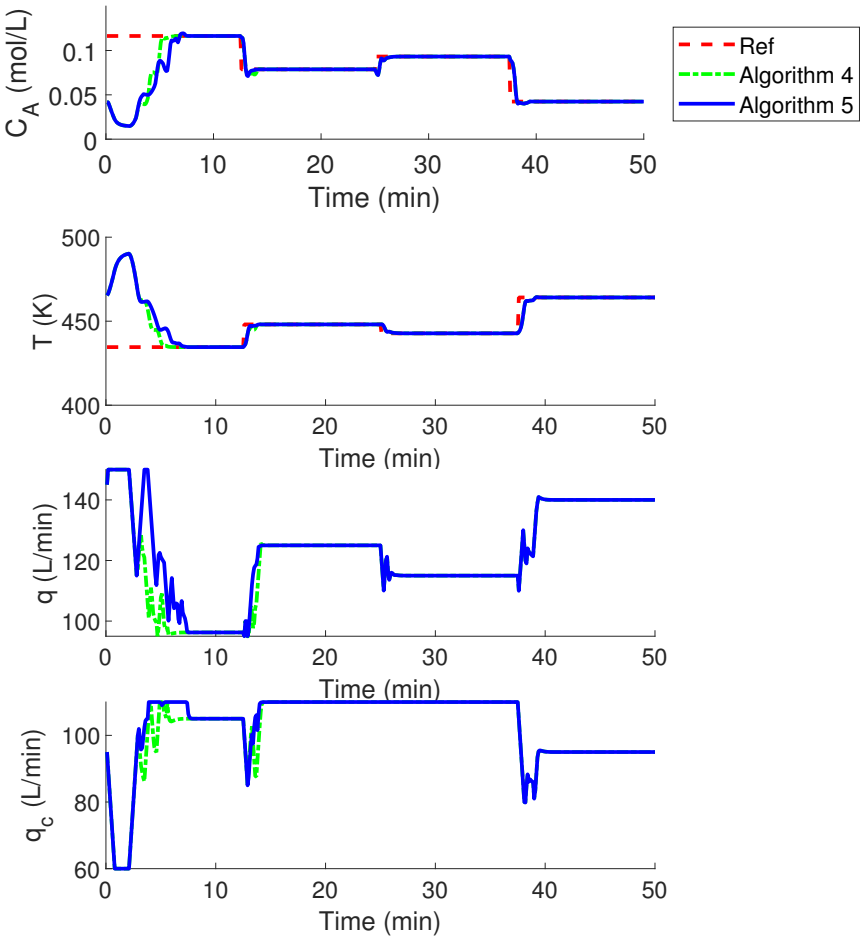

Figure 4.6: Set-point tracking for MIMO CSTR

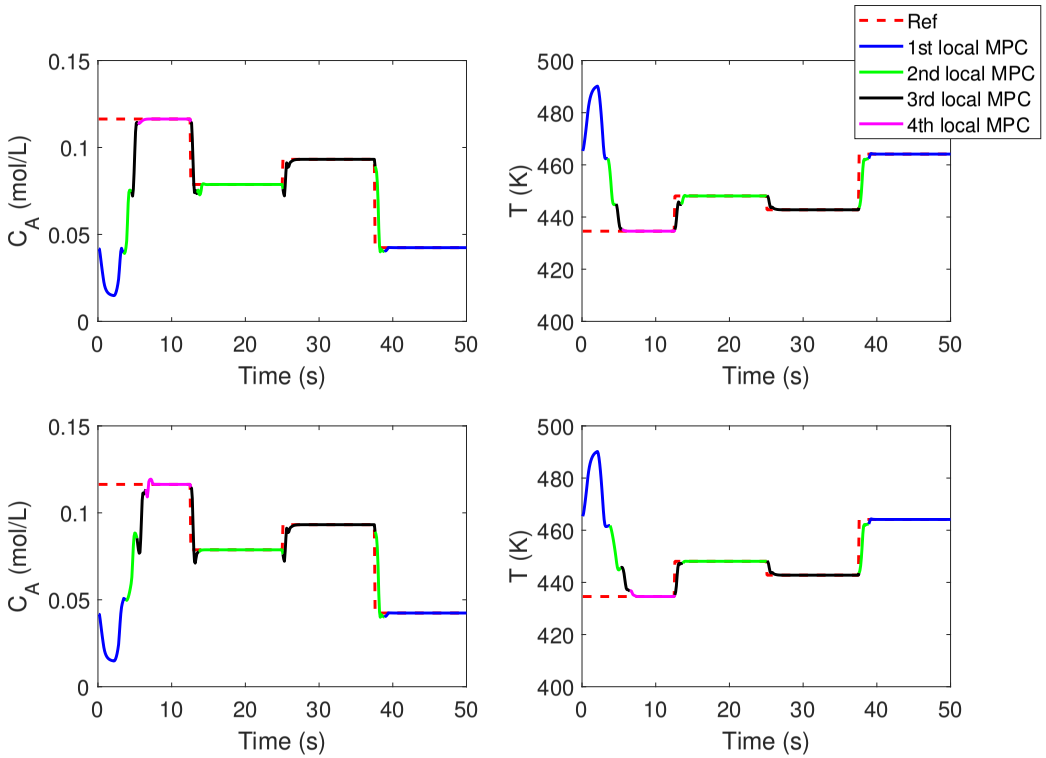

Figure 4.7: Set-point tracking for MIMO CSTR (Top: Algorithm 8, Bottom: Algorithm 9)

Table 4.5: Mean SAE for MIMO CSTR

	To		Θ_1		Θ_2		Θ_3		Θ_4	
From	**MLMPC**		**I**	**II**	**I**	**II**	**I**	**II**	**I**	**II**
Θ_1	**SAE**	C_A			1.27	1.27	3.04	3.16	4.84	5.30
		T			912	923	1523	1626	1928	2118
Θ_2	**SAE**	C_A	0.77	0.79			0.87	0.80	1.72	1.77
		T	471	469			228	224	388	431
Θ_3	**SAE**	C_A	1.62	1.64	1.15	1.12			2.49	2.52
		T	993	1037	599	573			878	884
Θ_4	**SAE**	C_A	3.83	4.01	2.80	2.79	2.32	2.29		
		T	1811	1845	1063	1044	844	839		

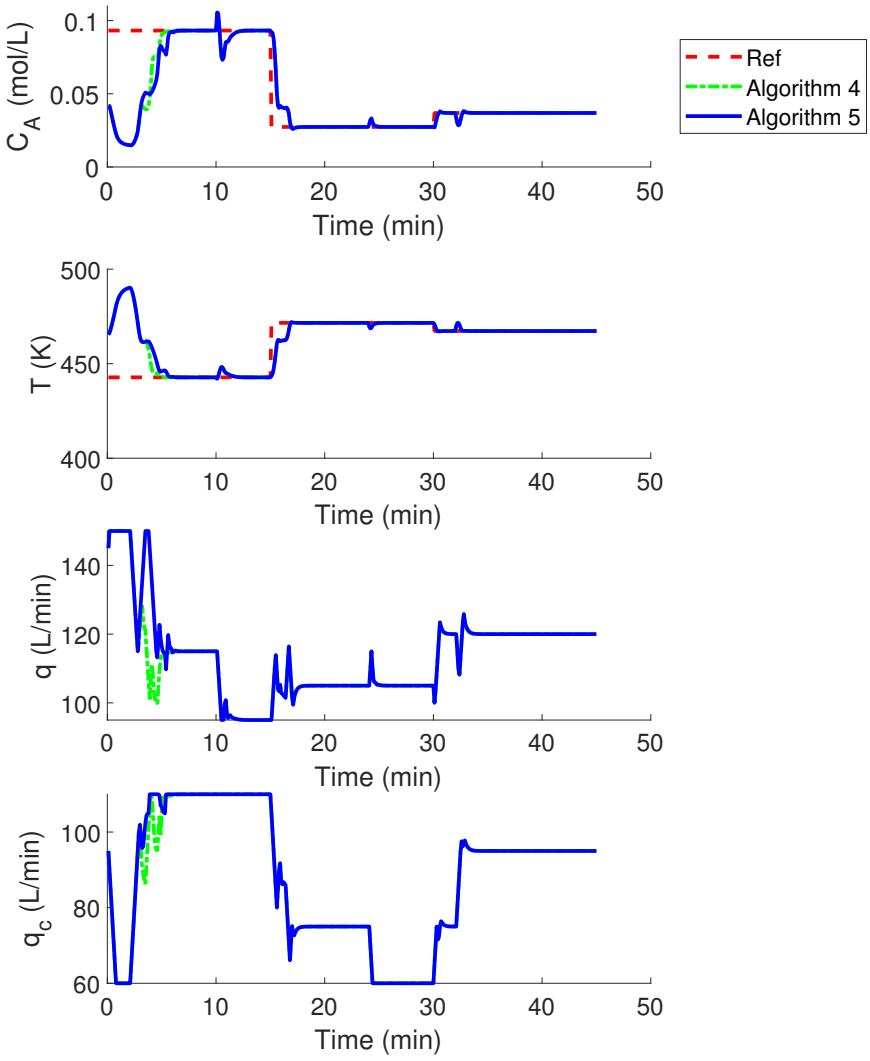

Figure 4.8: Disturbance rejection for MIMO CSTR

Chapter 5

Design of data-driven multilinear model predictive control [3]

5.1 Introduction

The methods to design MLMPC in the previous chapters need the nonlinear model for a process to obtain linear models for MLMPC and the equilibrium point in a set-point. However, a common issue for the two methods of MLMPC is that it is hard to design MLMPC without the nonlinear model of the process because the optimal or suboptimal trajectories are required to get linear models around the trajectories, which is difficult to be generated without the nonlinear model. Trajectory optimization techniques [21, 23, 25] can be exploited to generate the trajectories. In this chapter, DDP is exploited to obtain suboptimal trajectories to track a set-point, because DDP optimizes the trajectory by solving the optimization problem based on the first-order derivatives of the dynamics around the known feasible trajectory, i.e., local linear model, at each time step [26]. Thus, it needs only obtaining the local linear models around a known trajec-

[3] This chapter is an adapted version of B. Park, J. W. Kim, and J. M. Lee, "Data-driven offset-free multilinear model predictive control using constrained differential dynamic programming," *Journal of Process Control*, Under review

tory instead of the structure and parameters of the nonlinear model. DDP consists of backward and forward passes. Backward pass generates a new control sequence based on the nominal trajectory, and forward pass generates a new nominal trajectory based on the new control sequence. It iteratively tries to improve the trajectory until the nominal trajectory converges. It has been shown to possess convergence properties better than the Newton's method performed on the entire control sequence [27]. Because DDP does not consider any constraints, it is restrictive to apply the processes with constraints for which MPC is preferred. Constrained DDP (CDDP) has been developed to consider practical constraints on the processes [28, 29]. Most recent works on constrained DDP consider box input constraints [30] and nonlinear constraints [31]. Because DDP gives a set of linear time-varying (LTV) controllers to track the nominal trajectory instead of the set-point, it cannot achieve offset-free tracking unless the nominal trajectory achieves offset-free tracking. MPC can achieve offset-free tracking from the models obtained around a suboptimal trajectory that has offset, as it can remove the offset by including integral action or unknown disturbance to the model. Consequently, MLMPC and DDP complement each other if there is no nonlinear model for a process

In this work, we propose a framework for set-point tracking of nonlinear systems that considers input constraints when the nonlinear model and nonzero steady-state input for a set-point are unknown. It consists of a constrained DDP (CDDP), modeling for CDDP and MLMPC, and MLMPC. In the proposed scheme, CDDP includes the tuning parameters for improving the trajectory stably, and the convergence of the proposed CDDP is proved. In order to get the first

derivatives of the process at the nominal trajectory every iteration, a procedure to obtain the LTV model is proposed based on the conditional Gaussian process. Then the clustering of the models based on gap metric is proposed. It is designed to generate a cluster that consists of the models whose time steps are adjacent, which considers both gap metric and distance between the origins of a model and the representative model of a cluster. Last, the prediction-based MLMPC is proposed, because gap metric-based MLMPC requires the dynamics at the set-point, which is impossible without the nonlinear model. Our strategy makes the weight of a local MPC converge to one, if the output error and the prediction error of the MPC are less than designated thresholds. Hence, the proposed MLMPC is equivalent to linear offset-free MPC around the set-point. We prove that the converged MLMPC has offset-free tracking property if the gap metric between the converged linear model and the linearized model at the set-point is small.

5.2 Data-driven trajectory optimization

In this section, we propose a strategy to identify the model around a trajectory and improve the trajectory alternately without the nonlinear model. we consider input constraints, assume that the value of the input at a set-point is unknown.

5.2.1 Constrained differential dynamic programming

To solve the optimal control problem, dynamic programming (DP) requires solving the Bellman equation, which is impossible due to the curse of dimensionality (COD). Approximate dynamic pro-

gramming (ADP) or reinforcement learning (RL) try to solve the equation by approximating the value function from data. However, the structure of the value function is unknown, and the amount of the data increases as the sizes of the state and input constraints, and the complexity of the system increase. If the initial condition is fixed, DDP is a better choice to solve the Bellman equation in terms of the amount of data because it only solves the Bellman equation around a trajectory. However, DDP has limits to apply to the practical nonlinear processes directly. It does not consider constraints, and the feedback gain from the backward pass can be large so that the resulting trajectory changes rapidly and does not converge. In addition, there are no mathematical models for many processes. In this section, we propose a strategy to modify DDP for the process, the objective of which is steering the state to a set-point where the value of the input is unknown, and input constraints exist without any knowledge of models. We assume that the state is also the measured output and the controlled variable of the process (2.1). First, we augment the input to the dynamics of the system (2.1) to choose the increment of the input as the input of the augmented system.

$$\underbrace{\begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix}}_{z_{k+1}} = \underbrace{\begin{bmatrix} f(x_k, u_{k-1} + a_k) \\ u_{k-1} + a_k \end{bmatrix}}_{g(z_k, a_k)} \tag{5.1}$$

where $z_k := [x_k^T u_{k-1}^T]^T$ and $a_k := u_k - u_{k-1}$. Then the input of the augmented system is required to be zero, assuming the set-point is an equilibrium point. Quadratic cost is chosen for the running and

terminal cost.

$$\begin{aligned} J_k(\mathbf{Z}_k, \mathbf{A}_k) = \sum_{j=k}^{N-1} (h(z_j) - x^*)^T Q^{ddp} (h(z_j) - x^*) \\ + a_j^T R^{ddp} a_j + (x_j - x^*)^T P_N (x_j - x^*) \end{aligned} \tag{5.2}$$

where x^* is the set-point of the process, $Q^{ddp} \succeq 0$ and $R^{ddp} \succ 0$ are the weighting matrices for the augmented state and input, respectively. C_x is defined to satisfy $h(z_j) = C_x z_j = [I\ 0] z_j := x_j$. Similarly, C_u is defined to satisfy $C_u z_j = [0\ I] z_j := u_{j-1}$. The cost becomes zero when the state is at a set-point. The input constraints (2.2) are considered again.

$$\begin{aligned} u_{min} \leq u_j \leq u_{max} \\ \Delta u_{min} \leq a_j \leq \Delta u_{max} \end{aligned} \tag{5.3}$$

Then the constrained optimization is conducted in the backward pass.

$$\begin{aligned} &\min_{\delta a_k} Q_k(\delta z_k, \delta a_k) \\ &\text{subject to } u_{min} \leq\ u_{k-1} + a_k + \delta a_k \leq u_{max} \\ &\Delta u_{min} \leq\ a_k + \delta a_k \leq \Delta u_{max} \end{aligned} \tag{5.4}$$

where u_{k-1} is obtained from the trajectory of the previous forward pass. An active set method is exploited to solve the above quadratic program (QP) to consider the active constraints to update the optimal gains in the backward pass. Since δz_k is unknown during the backward pass, it is assumed to be zero in (5.3). In addition, box con-

straints are reduced.

$$\begin{aligned} &\min_{\delta a_k} \frac{1}{2}\delta a_k^T Q_{aa,k}\delta a_k + Q_{a,k}^T \delta a_k \\ &\text{subject to } \delta a_k \leq \min(u_{max} - u_{k-1} - a_k, \Delta u_{max} - a_k) \\ &\qquad\qquad \delta a_k \geq \max(u_{min} - u_{k-1} - a_k, \Delta u_{min} - a_k) \end{aligned} \tag{5.5}$$

After the QP is solved, the optimal feedback gain $K_{c,k}$ is assigned to the solution δa_k^*. To determine $K_{g,k}$, we exploit a projected Newton step using the reduced Hessian in the free sub-space used in [30]. First, the complimentary sets of clamped and free indices c and f are defined.

$$\begin{aligned} \mathsf{c}(\delta a_k) =& \left\{ j \in 1, \cdots, m \;\middle|\; \begin{array}{c} \delta a_{j,k} \text{ is active at minimum, } \lambda_{j,k}^{min} > 0, \text{ or} \\ \delta a_{j,k} \text{ is active at maximum, } \lambda_{j,k}^{max} > 0 \end{array} \right\} \\ \mathsf{f}(\delta a_k) =& \{ j \in 1, \cdots, m | j \notin \mathsf{c} \} \end{aligned} \tag{5.6}$$

where $\delta a_{j,k}$ is the j^{th} component of δa_k, and $\lambda_{j,k}^{min}$ and $\lambda_{j,k}^{max}$ are the Lagrange multiplier for the active constraint in (5.5) of $\delta a_{j,k}$ at minimum and maximum, respectively. Dropping k for readability, δa, Q_{aa}, and Q_{az} are rearranged according to the partition $\{\mathsf{c}, \mathsf{f}\}$.

$$\delta a \leftarrow \begin{bmatrix} \delta a_{\mathsf{f}} \\ \delta a_{\mathsf{c}} \end{bmatrix}, Q_{az} \leftarrow \begin{bmatrix} Q_{az,\mathsf{f}} \\ Q_{az,\mathsf{c}} \end{bmatrix}, Q_{aa} \leftarrow \begin{bmatrix} Q_{aa,\mathsf{ff}} & Q_{aa,\mathsf{fc}} \\ Q_{aa,\mathsf{cf}} & Q_{aa,\mathsf{cc}} \end{bmatrix}, \tag{5.7}$$

This decomposition is used to compute the optimal feedback gain for the free indices $K_{g,\mathsf{f}} = -Q_{aa,\mathsf{f}}^{-1} Q_{az,\mathsf{f}}$. The optimal gain for the clamped indices becomes zero. Thus, the components of the input in the clamped indices are feasible in the next forward pass regardless of the change of the state. However, the quadratic expansions at the

previous trajectory can be inaccurate at the updated trajectory after the forward step with the update gains $K_{g,k}$ and $K_{c,k}$, because the change of the state and input can be accumulated. In order to prevent a large change of the updated trajectory, we propose to modify the optimal gain as follows.

$$\delta a = \alpha_g K_g \delta z + \alpha_c K_c \quad (0 < \alpha_c \leq 1, 0 \leq \alpha_g \leq 1) \tag{5.8}$$

The approximated value function is updated as

$$\begin{aligned} V_{zz,k} =& Q_{zz,k} + \alpha_g^2 K_{g,k}^T Q_{aa,k} K_{g,k} + \alpha_g Q_{az,k}^T K_{g,k} + \alpha_g K_{g,k}^T Q_{az,k}, \\ V_{z,k} =& Q_{z,k} + \alpha_g \alpha_c K_{g,k}^T Q_{aa,k} K_{c,k} + \alpha_c Q_{az,k}^T K_{c,k} + \alpha_g K_{g,k}^T Q_{a,k}, \\ \Delta V_k =& -\alpha_c Q_{a,k}^T Q_{aa,k}^{-1} Q_{a,k} + \frac{1}{2} \alpha_c^2 K_{c,k}^T Q_{aa,k} K_{c,k}, \end{aligned} \tag{5.9}$$

In unconstrained case, it is expressed as

$$\begin{aligned} V_{zz,k} =& Q_{zz,k} + \alpha_g(\alpha_g - 2) K_{g,k}^T Q_{aa,k} K_{g,k}, \\ V_{z,k} =& Q_{z,k} + (\alpha_g \alpha_c - \alpha_g - \alpha_c) K_{g,k}^T Q_{aa,k} K_{c,k}, \\ \Delta V_k =& \frac{1}{2} \alpha_c (\alpha_c - 2) K_{c,k}^T Q_{aa,k} K_{c,k} \end{aligned} \tag{5.10}$$

α_g and α_c can change after each forward pass finishes. This modfication is different from the step size modification generally used in DDP [67]. It adds the multiplier to make the effect of K_g decrease. The proposed backward pass of CDDP is summarized in Algorithm 11.

Algorithm 11: CDDP Backward Pass

Input: $\mathbf{Z}_0$, $\mathbf{A}_0$

Result: $\{K_{g,k}\}$, $\{K_{c,k}\}$, $\{V_{zz,k}\}$, $\{V_{z,k}\}$, $\{\Delta V_k\}$

$V_{zz,N} \leftarrow l^f_{zz}, V_{z,N} \leftarrow l^f_z$

for $k = N-1, N-2, \cdots, 0$ **do**

$Q_z \leftarrow l_{z,k} + g_z^T V_{z,k+1}$

$Q_a \leftarrow l_{a,k} + g_a^T V_{z,k+1}$

$Q_{zz} \leftarrow l_{zz,k} + g_z^T V_{zz,k+1} g_z + V_{z,k+1}^T g_{zz}$

$Q_{aa} \leftarrow l_{aa,k} + g_a^T V_{zz,k+1} g_a + V_{z,k+1}^T g_{aa}$

$Q_{az} \leftarrow l_{az,k} + g_a^T V_{zz,k+1} g_z + V_{z,k+1}^T g_{az}$

Solve (5.5) and update $K_{c,k} \leftarrow \delta a_k^*$

Compute the clamped indices c and free indices f in (5.6)

Compute the rows of the optimal gain for f and c

$K_{g,\mathsf{f}} \leftarrow -Q_{aa,\mathsf{f}}^{-1} Q_{az,\mathsf{f}}, K_{g,\mathsf{c}} \leftarrow 0$

Compute $K_{g,k}$ by rearranging $K_{g,\mathsf{f}}, K_{g,\mathsf{c}}$

Update $V_{zz,k}$, $V_{z,k}$, ΔV_k using (5.9)

$K_{g,k} \leftarrow \alpha_g K_{g,k}, K_{c,k} \leftarrow \alpha_c K_{c,k}$

end

In the forward pass, the updated feedback gains are applied to compute the input. Although the feedback gains take into account the estimated active constraints, it cannot guarantee that the trajectory after the forward pass using the feedback gains from the backward pass, because the variation of the augmented state can cause the violation of the constraints. As the input constraints are affine, clamping the input from the feedback can make the input constraints during the forward pass feasible. After the forward pass is conducted, the cost of the new trajectory is computed and compared with that of the nominal trajectory. If the cost is reduced, the nominal trajectory is updated and α_g and α_c increase to change the trajectory aggressively. Otherwise, the new trajectory is discarded and the backward pass is performed with the same nominal trajectory, decreasing α_g and α_c. The proposed forward pass of CDDP is summarized in Algorithm 12.

Algorithm 12: CDDP Forward Pass

Input: $\mathbf{Z}_0$, $\mathbf{A}_0$, $\{K_{g,k}\}$, $\{K_{c,k}\}$
Result: $\mathbf{Z}_0$, $\mathbf{A}_0$
$J_{int} \leftarrow J_0(\mathbf{Z}_0, \mathbf{A}_0)$, $z \leftarrow z_0$
for $k = 0, 1, \cdots, N-1$ **do**
  $\delta z \leftarrow z - z_k$
  $z_{temp,k} \leftarrow z$, $u_{temp,k} \leftarrow C_u z$
  $a_{temp,k} \leftarrow a_k + K_{g,k}\delta z + K_{c,k}$
  Clamp $a_{temp,k}$ to satisfy
  $u_{min} \leq u_{k-1} + a_{temp,k} \leq u_{max}$
  $\Delta u_{min} \leq a_{temp,k} \leq \Delta u_{max}$
  $z \leftarrow g(z, a_{temp,k})$
end
$z_{temp,N} \leftarrow z$
$J_{temp} \leftarrow J_0(\mathbf{Z}_{temp,0}, \mathbf{A}_{temp,0})$
if $J_{temp} < J_{int}$ **then**
  $\mathbf{Z}_0 \leftarrow \mathbf{Z}_{temp,0}$, $\mathbf{A}_0 \leftarrow \mathbf{A}_{temp,0}$
  $\alpha_g \leftarrow \beta_1 \alpha_g$, $\alpha_c \leftarrow \beta_2 \alpha_c$, $\beta_1, \beta_2 > 1$
else
  $\alpha_g \leftarrow \alpha_1 \alpha_g$, $\alpha_c \leftarrow \alpha_2 \alpha_c$, $0 < \alpha_1, \alpha_2 < 1$
end

We give a Theorem for convergence of the modified CDDP.

Theorem 5.1. *For any feasible trajectory* $(\mathbf{Z}, \mathbf{A})$*, there exists* α_g *and* α_c *that improve the nominal trajectory if* Q *and* P_N *are positive semidefinite and* R *is positive definite in* (5.2) .

Proof *As* α_c *decreases,* ΔV_k *in* (5.9) *is close to the first term of RHS,* $-\alpha_c Q_{a,k}^T Q_{aa,k}^{-1} Q_{a,k}$ *as the second term of RHS is proportional to* α_c^2*.* $Q_{aa,k} = R + g_{a,k}^T V_{zz,k+1} g_{a,k}$ *is positive definite as long as* $V_{zz,k+1}$ *is positive semidefinite due to positive definite* R*.* $\Delta V_{zz,k+1}$ *in* (5.9) *is close to* $Q_{zz,k+1}$ *as* α_g *decreases.* $Q_{zz,k+1} = Q + g_{z,k+1}^T V_{zz,k+2} g_{z,k+1}$ *is positive semidefinite as long as* $V_{zz,k+2}$ *is positive semidefinite.*

$V_{zz,N} = P_N$ *is positive semidefinite. Hence,* $Q_{aa,N-1} = R + g_{a,N-1}^T V_{zz,N} g_{a,N-1}$ *is positive definite because* $V_{zz,N}$ *is positive semidefinite and* R *is positive definite. As* α_c *decreases,* ΔV_{N-1} *in* (5.9) *is close to the first term of RHS,* $-\alpha_c Q_{a,N-1}^T Q_{aa,N-1}^{-1} Q_{a,N-1}$ *as the second term of RHS is proportion to* α_c^2*. Consequently,* ΔV_{N-1} *is negative if* α_c *is small enough to satisfy*

$$0 < \alpha_c < \frac{2\lambda_{min}(Q_{a,N-1}^T Q_{aa,N-1}^{-1} Q_{a,N-1})}{\lambda_{max}(K_{c,N-1}^T Q_{aa,N-1} K_{c,N-1})} \tag{5.11}$$

where $\lambda_{max}(\cdot)$ *is the maximum eigenvalue of a matrix. On the other hand,* $Q_{zz,N-1} = Q + g_{z,N-1}^T V_{zz,N} g_{z,N-1}$ *is positive semidefinite due to positive semidefinite* Q *and* $V_{zz,N}$*. Because* $V_{zz,N-1}$ *in* (5.9) *is close to* $Q_{zz,N-1}$ *as* α_g *decreases, it can be positive semidefinite if* α_g *is*

small enough to satisfy

$$
\begin{aligned}
&k_1\alpha_g^2 + 2k_2\alpha_g^2 + k_3 \geq 0 \\
&k_1 = \lambda_{min}(K_{g,N-1}^T Q_{aa,N-1} K_{g,N-1}) \\
&k_2 = \lambda_{min}(K_{g,N-1}^T Q_{az,N-1}) \\
&k_3 = \lambda_{min}(Q_{zz,N-1})
\end{aligned}
\tag{5.12}
$$

0 *is the trivial solution of* (5.12)*. Thus,* $\Delta V_{N-1}, \cdots, \Delta V_0$ *can be negative with small* α_g *and* α_c*, and the updated trajectory is improved.* ■

5.2.2 Model identification around a trajectory

If the dynamics of the original system (2.1) is unknown, the derivatives of the augmented system (5.1), g_z and g_a, have to be obtained from data to proceed with the backward pass. We excite the input around the nominal trajectory and identify the local linear dynamics around the nominal trajectory, which is summarized in Algorithm 13. With the perturbation of the nominal input, the trajectories from n_{id} episodes are collected. Because the dynamics of u is known, i.e., $u_k = u_{k-1} + a_k$, only the dynamics of x is identified. Dropping k and Denoting x_{k+1} by x_+, the data at k^{th} time step are

$$
x_+^{epi} := \begin{bmatrix} x_+^1 \\ \vdots \\ x_+^{n_{id}} \end{bmatrix}, x^{epi} = \begin{bmatrix} x^1 \\ \vdots \\ x^{n_{id}} \end{bmatrix}, u^{epi} = \begin{bmatrix} u^1 \\ \vdots \\ u^{n_{id}} \end{bmatrix} \tag{5.13}
$$

where the superscript denotes the number of the experiments. Assuming that the augmented vector $(x^T, u^T, x_+^T)^T$ is multivariate Gaussian,

i.e., $(x^T, u^T, x_+^T)^T \sim \mathcal{N}(\mu, \Sigma)$, the distribution of state transition given a state and an input can be expressed as the condition Gaussian distribution

$$\begin{aligned} p(x_+|xu) &\sim \mathcal{N}(\bar{\mu}, \bar{\Sigma}) \\ \bar{\mu} &= \mu_{x_+} + \Sigma_{x+,xu}\Sigma_{xu,xu}^{-1}(xu - \mu_{xu}) \\ \bar{\Sigma} &= \Sigma_{x_+,x_+} - \Sigma_{x_+,xu}\Sigma_{xu,xu}\Sigma_{x_+,xu} \end{aligned} \tag{5.14}$$

where $xu := (x^T, u^T)^T$. Thus,

$$\begin{aligned} x_+ &\sim \mathcal{N}(f_x x + f_u + f_c, \bar{\Sigma}) \\ \begin{bmatrix} f_x \\ f_u \end{bmatrix} &= \Sigma_{x+,xu}\Sigma_{xu,xu}^{-1}, \quad f_c = -\Sigma_{x+,xu}\Sigma_{xu,xu}^{-1}\mu_{xu} \end{aligned} \tag{5.15}$$

Then g_z and g_a is obtained as

$$g_z = \begin{bmatrix} f_x & f_u \\ 0 & I \end{bmatrix}, g_a = \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{5.16}$$

Hence, the backward pass proceeds with a nominal trajectory and g_x and g_u obtained around the nominal trajectory. Algorithm 13 summarizes obtaining the local linear dynamics around the nominal trajectory. The proposed model-free CDDP proceeds in the order of identifying the local linear dynamics, the backward pass, and the forward pass.

Algorithm 13: Identifying g_z and g_a

Input: $\mathbf{Z}_0$, $\mathbf{A}_0$, $\{\Sigma_{a_k}\}$
Result: $\{g_{z,k}\}$, $\{g_{a,k}\}$
for $i = 1, 2, \cdots, n_{id}$ **do**
  $z \leftarrow z_0$
  Clamp u_0 to satisfy the constraints in Algorithm 12
  for $k = 0, 1, \cdots, N-1$ **do**
    $z_k^i \leftarrow z$
    Sample a from $\mathcal{N}(a_k, \Sigma_{a_k})$
    Clamp a to satisfy the constraints in Algorithm 12
    $z \leftarrow g(z, a)$ (from plant or simulation)
    $a_k^i \leftarrow a$
  end
  $z_N^i \leftarrow z$
end
for $k = 0, 1, \cdots, N-1$ **do**
  $x_+ \leftarrow x_{k+1}$, $x \leftarrow x_k$, $u \leftarrow u_k$
  Compute mean and covariance of $(x^T, u^T, x_+^T)^T$ and denote by μ, Σ
  Compute the conditional expectation and the augmented linear dynamics
  $\begin{bmatrix} f_x \\ f_u \end{bmatrix} \leftarrow \Sigma_{x+,xu}\Sigma_{xu,xu}^{-1}$
  $g_{z,k} \leftarrow \begin{bmatrix} f_x & f_u \\ 0 & I \end{bmatrix}, g_{a,k} \leftarrow \begin{bmatrix} 0 \\ I \end{bmatrix}$
end

Remark 5.1. *In practice, the convergence of the proposed CDDP highly depends on the choice of step sizes like α_g and α_c. The step sizes increase if the trajectory is improved. On the other hand, the step sizes decrease if the trajectory is not improved. The variations of line search methods can be applied [68, 69].*

5.3 Data-driven offset-free MLMPC

Although the proposed CDDP improves the trajectory, the number of the iteration to steer the state to a set-point is unknown. Also, it is vulnerable to disturbance as a proportional (P) controller is used at each time step. In this work, we proposed a novel prediction-based offset-free MLMPC algorithm that exploits the local linear models from CDDP. We will show the proposed controller achieves offset-free tracking before the CDDP converges, and rejects disturbance in the numerical example. It consists of the algorithms for clustering of local models, selection of the models for MLMPC, the design of the local MPC controllers, and calculating the weights of the local MPCs.

5.3.1 Gap metric-based clustering algorithm

To choose the clusters and the representative models from local models of CDDP, we proposed a gap metric-based clustering algorithm. Because CDDP uses a linear model at each time step, each linear model of CDDP has a time index. Hence, we propose to construct a cluster whose components are adjacent in terms of time index. This is a different approach to the clustering method in our previous work [65], where the models are clustered according to the distance between the states or inputs where the models are constructed. Com-

pared with clustering based on state or control input, it is free from the scaling of the state and input because the dimension of time is always one. Besides, there is a high possibility that the dynamic behaviors at adjacent time steps are similar to each other because the differences of the states and inputs at adjacent time steps are bounded by the continuous dynamics (2.1) and input constraints (5.3). A model whose mean of the gaps with other models in a cluster is minimum is chosen as the representative model, because the controller designed by the model is the most robust according to Theorem 2.3. The maximum gap with other models in a cluster can be another criterion. However, we observe that the maximum is one, which is the upper bound of the gap, in many cases. Thus, the chosen model is not guaranteed to be the best in terms of gap metric. Increasing the number of clusters until the maximum becomes smaller than one is another option, but it causes too many models. A gap metric-based clustering algorithm 14 is proposed as follows.

Remark 5.2. *The number of the constructed clusters n_{cl} can be arbitrarily chosen. If n_{cl} is too small, the dynamic behaviors in a cluster can be very different, and the representative model cannot describe the dynamic behavior in the cluster. Large n_{cl} means that several abrupt changes of the local dynamics of the system are expected during the operation, which is the case that the initial condition is far from the set-point. If CDDP gives a trajectory where the state is steered around the set-point, the last cluster will consist of the states around the set-point.*

Algorithm 14: Gap metric-based clustering algorithm

Input: $\{g_{x,k}\}$, $\mathbf{Z}_0$,n_{cl}

Result: Clusters of time indices $\{CL_i\}$
Time indices for representative models $\{med_i\}$
Liner models of the trajectory $\{P_i\}$

Obtain $\mathbf{X}_0$ and $\mathbf{U}_0$ from $\mathbf{Z}_0$

for $i = 0, \cdots, N-1$ **do**
  Obtain $f_{x,i}$ and $f_{u,i}$ from $g_{x,i}$
  $A_i \leftarrow f_{x,i}$
  $B_i \leftarrow f_{u,i}$
  $P_i \leftarrow$ state-space model with $(A_i, B_i, I, 0)$
end

Compute $\Delta_{i,j} \leftarrow \delta_g(P_i, P_j), \quad 0 \leq i, j \leq N-1$

$CL_j, CL_j^{ex} \leftarrow \{\}, med_i \leftarrow (2i+1)\lfloor N/(2n_{cl})\rfloor, \quad 1 \leq i \leq n_{cl}$

$flag \leftarrow 1, it \leftarrow 1$

while $flag$ **do**
  for $i = 0, \cdots, N-1$ **do**
    $i_{cl} = \Sigma_{j=1}^{n_{cl}}(i \geq med_j)$
    if $i_{cl} == 0$ **or** $i_{cl} == n_{cl}$ **then**
      $idx \leftarrow \max(i_{cl}, 1)$
    else
      $idx \leftarrow arg\min(\Delta_{i,i_{cl}}, \Delta_{i,i_{cl}+1})$
    end
    $CL_{idx} \leftarrow CL_{idx} \cup \{i\}$
  end
  for $j = 1, \cdots, n_{cl}$ **do**
    $med_j \leftarrow arg\min_{k \in CL_j}(\max_{i \in CL_j} \Delta_{i,k})$
  end
  if $it > 1$ **and** $CL_i^{ex} == CL_i$ **then**
    $flag \leftarrow 0$
  else
    $it \leftarrow it + 1, CL_i^{ex} \leftarrow CL_i$
  end
end

5.3.2 Prediction-based MLMPC

Based on linear offset-free MPC in Chapter 2 and local representative models of the clusters, we propose to design a prediction-based MLMPC algorithm that achieves offset-free control. Theorem 3.1 has shown that a linear offset-free MPC can track a set-point if the initial point is near the set-point. However, offset-free tracking is not guaranteed if the initial condition is far from the set-point. To resolve this issue, a prediction-based MLMPC algorithm is proposed in Algorithm 15. It tries to steer the state to the neighborhood of the set-point using models whose predictions are accurate. In order to determine the prediction accuracy of each model, prediction error is defined, i.e., $e_i(k) := y_k - \hat{y}_i(k|k-1)$, where y_k is the output at the k^{th} time step and $\hat{y}_i(k|k-1)$ is the predicted output of the i^{th} model at the $(k-1)^{\text{th}}$ time step. Then, local linear offset-free MPC is applied to track the set-point. The weights of the proposed MLMPC are calculated based on three criteria, the prediction error $\{e_i(k)\}$, error $e_k := r_k - y_k$, and index i^{ex}, where r_k and y_k is the reference and output at the k^{th} time step, and i^{ex} the indicator to determine what linear offset-free MPC is employed at the previous time step. First, it calculates base weights according to the output prediction based method (2.36). When the output error becomes smaller than the threshold α_{th} and there exist models whose prediction errors becomes smaller than the threshold β_{th}, the algorithm chooses the best model in terms of the prediction error and the corresponding local linear MPC as the controller. In addition, if a local MPC is only used at the previous time step whose models has a prediction error lower than β_{th} at the current time step, the algorithm chooses the local MPC as the controller, preventing os-

cillation caused by the change of the models whose prediction errors are low.

The designed weight in Algorithm 15 makes the MLMPC controller converge to a local MPC around the set-point, enjoying the offset-free tracking property by Theorem 3.1. Thus, there should exist a model among the model of the MLMPC whose gap metric with the linearized system at the set-point is small enough to satisfy Theorem 3.1. Because the trajectory from the proposed CDDP converges to the set-point as the number of iterations increases, we can obtain improved models in terms of the gap metric with the linearized system at the set-point after additional iterations of the proposed CDDP. The proposed framework for set-point tracking that considers input constraints without the knowledge of the model and the steady-state input at the set-point is summarized in Figure 5.1.

Remark 5.3. *If the system is stochastic, stochastic DDP can be employed to solve the stochastic optimal control problem[70]. However, obtaining the stochastic model is the main issue, because the effects of input and disturbance are hard to distinguish. Path integral control provides a way to find the optimal control input by simulating the uncontrolled system dynamics [71]. However, the number of the simulation is required to obtain the optimal control input. Thus, the efficiency of each sample is important. In addition, the probability of obtaining a low-cost trajectory depends on the variance of the Brownian motion. Thus, an importance sampling method in which both the mean of the control input and the variance of the Brownian motion can be adjusted is proposed [72]. Because path integral methods do not consider the stability during the simulations by uncontrolled system dynamics, it is required to develop the path integral methods that*

guarantee stability during all simulations.

Algorithm 15: Prediction-based MLMPC

Input: $\{med_i\}$, n_{cl}, $\{P_i\}$, $\mathbf{Z}_0$, $\{r_k\}$, α_{th}, β_{th}, p_{th}, Λ_e
Result: $\mathbf{X}_0$,$\mathbf{U}_0$
Obtain $\mathbf{X}_0$ and $\mathbf{U}_0$ from $\mathbf{Z}_0$, $\mathbf{X}_{lin} \leftarrow \mathbf{X}_0$, $\mathbf{U}_{lin} \leftarrow \mathbf{U}_0$
for $i = 1, \cdots, n_{cl}$ **do**
  $j \leftarrow med_i$, Construct local linear MPC using P_j
  $\hat{x}_i(0|-1) \leftarrow x_0 - x_{lin,j}$, $w_i(-1) \leftarrow 1/n_{cl}$, $i^{ex} \leftarrow 0$
end
for $k = 0, \cdots, N-1$ **do**
  Measure x_k and estimate $\hat{x}_i(k|k)$
  $e_k \leftarrow r_k - h(x_k)$, $md_g \leftarrow \{\}$
  for $i = 1, \cdots, n_{cl}$ **do**
    $j \leftarrow med_i$
    $e_i(k) \leftarrow y_k - h(\hat{x}_i(k|k-1) + x_{lin,j})$
    $w_i(k) \leftarrow \max(w_i(k-1)\exp(-e_i(k)^T \Lambda_e e_i(k)), p_{th})$
    if $\|e_i(k)\| < \beta_{th}$ **then**
      $md_g \leftarrow md_g \cup \{i\}$
    end
    Solve n_{cl} local MPC problems and obtain $u_i(k)$
  end
  $w_i(k) \leftarrow w_i(k) / \sum_{i=1}^{n_{cl}} w_i(k)$, $w_i^* \leftarrow 0$
  if $md_g \neq \emptyset$ **and** $\|e_k\| < \alpha_{th}$ **then**
    if $i^{ex} \notin md_g$ **then**
      $i^{ex} \leftarrow arg\min_i \|e_i(k)\|$
    $w_{i^{ex}}^* \leftarrow 1$
  else
    $w_i^* \leftarrow w_i(k)$, $i^{ex} \leftarrow 0$
  end
  Calculate u_k to the plant

$$u_k = \sum_{i=1}^{n_{cl}} w_i^*(u_i(k) + u_{lin,med_i})$$

  Apply u_k to plant and local models to get x_{k+1} and $\hat{x}_i(k+1|k)$
end

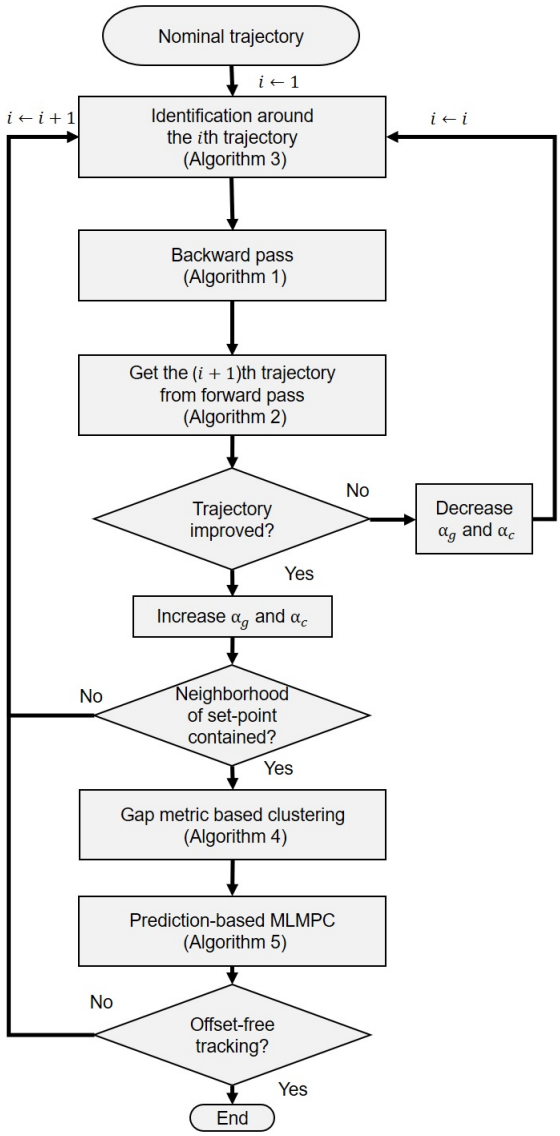

Figure 5.1: Overall algorithm

5.4 Results and discussions

Consider a multi-input multi-output (MIMO) continuous stirred tank reactor (CSTR) [53].

$$\begin{aligned}
\dot{C}_A(t) =& \frac{q}{V}[C_{A0} - C_A(t)] - k_0 C_A(t) \mathrm{e}^{-E/RT(t)}, \\
\dot{T}(t) =& \frac{q}{V}[T_0 - T(t)] - \frac{\Delta H k_0}{\rho C_p} C_A(t) \mathrm{e}^{-E/RT(t)} \\
&+ \frac{\rho_c C_{pc}}{\rho C_p V} q_c(t)[1 - \mathrm{e}^{-\frac{h_A}{\rho_c C_{pc} q_c(t)}}][T_{c0} - T(t)], \\
y(t) =& [C_A(t) \;\; T(t)]^T.
\end{aligned} \tag{5.17}$$

It consists of an irreversible, exothermic reaction. The concentration C_A and the temperature T are controlled by manipulating the flow rate of A, q and the coolant flow rate, q_c. The parameters and initial values of the variables in the system are shown in Table 5.1.

The equilibrium point of the process is determined uniquely if the values of the inputs are given [53]. The steady-state input-output relationship is shown in Figure 5.2. The goal of the controller is to drive the state from an equilibrium point to another equilibrium point within the operating condition depicted in Figure 5.2, assuming that the system dynamics and the input values at the set-point are unknown.

Table 5.1: MIMO CSTR Parameters and Initial Values

Product concentration	C_A	0.0245 mol/L
Coolant flow rate	q_c	70 L min^{-1}
Feed concentration	C_{A0}	1 mol/L
Inlet coolant temperature	T_{C0}	350 K
Heat transfer term	h_A	7×10^5 cal/min K
Activation energy term	E/R	1×10^4 K
Liquid densities	ρ, ρ_c	1×10^3 g/L
Reactor temperature	T	473.23 K
Process flow rate	q	120 L min^{-1}
Feed temperature	T_0	350 K
CSTR volume	V	100 L
Heat of reaction	ΔH	-2 $\times$ 10 ^{5} cal/mol
Specific heats	C_p, C_{pc}	1 cal g $^{-1}$ K $^{-1}$
Reaction rate constant	k_0	7.2×10^{10} min $^{-1}$
Constraints on the flow rate	q_{min}, q_{max}	95, 150 L min^{-1}
Constraints on the coolant flow rate	$q_{c,min}, q_{c,max}$	60, 110 L min^{-1}
Constraints on the flow rate	$\Delta q_{c,max}, \Delta q_{max}$	5 L min^{-1}
Constraints on the coolant flow rate	$\Delta q_{c,min}, \Delta q_{min}$	-5 L min^{-1}

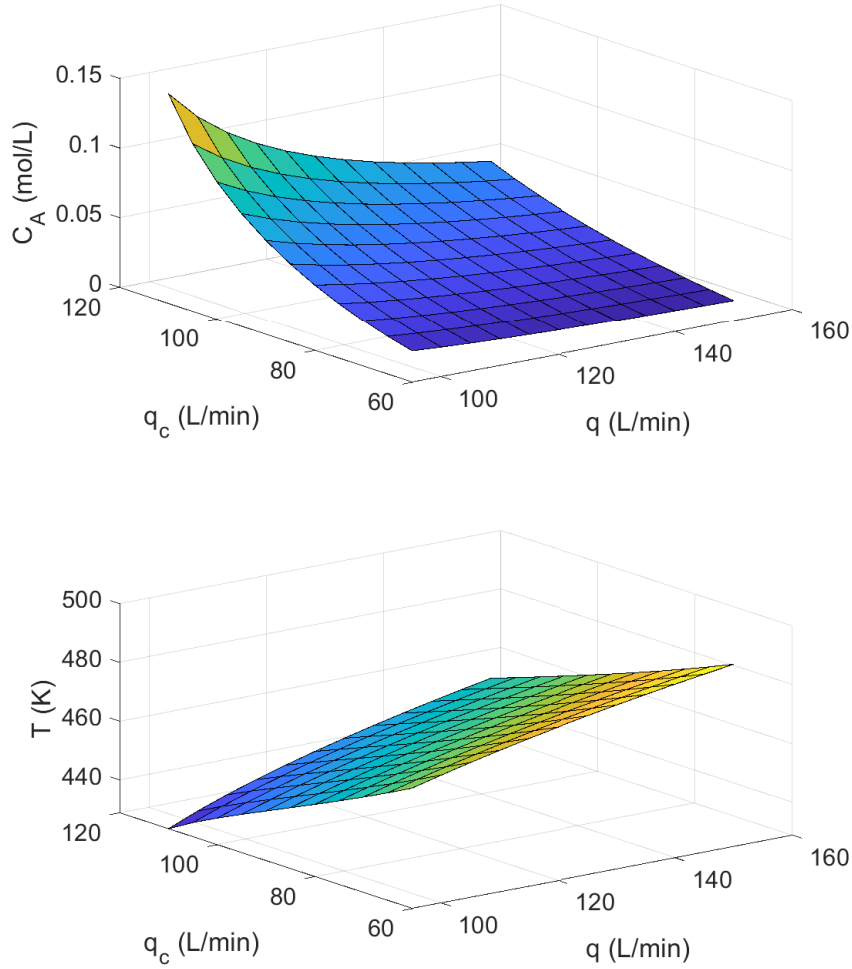

Figure 5.2: Steady-state input-output map for MIMO CSTR

First, the proposed CDDP is conducted to obtain suboptimal trajectories and models for MLMPC. We adjusted the initial values of α_g and α_c in Algorithm 11 and 12 to 0.02, since the feedback gains K_g and K_c alone were so high that had the trajectory were not improved in terms of the cost of CDDP at the initial time step, i.e., $J_0(\mathbf{Z}_0, \mathbf{A}_0)$. We choose $\beta_1, \beta_2 \sim \sqrt{n}$ and $\alpha_1, \alpha_2 \sim 1/\sqrt{n}$ in Algorithm 12, where n is the number of iterations of CDDP. In order to obtain the models for CDDP, the standard deviation $\Sigma_{a_k}^{0.5}$ to sample input in Algorithm 13 is chosen as 1% of the upper and lower boundaries of inputs, i.e., $\frac{(\Delta q_{max} - \Delta q_{min})}{100}$ and $\frac{(\Delta q_{c,max} - \Delta q_{c,min})}{100}$. The parameters of CDDP are shown in Table 5.2, and the result of performing the CDDP is shown in Figures 5.3 and 5.4. The cost of CDDP at the initial time step, i.e., $J_0(\mathbf{Z_0}, \mathbf{A_0})$, decreases and the states are steered to the set-point as the number of the iterations increases by adjusting α_g and α_c. However, the number of the iterations as well as the optimal hyperparameter to achieve offset-free tracking are unknown. In order to achieve offset-free tracking, the proposed MLMPC is applied every time CDDP generates an improved trajectory in terms of the cost of CDDP at the initial time, i.e., $J_0(\mathbf{Z_0}, \mathbf{A_0})$. In order to check offset-free tracking, the state and measurement noise are not added. Luenberger observer is chosen for the filter as it is convenient to control the response time by changing the poles of the observer.

Table 5.2: Parameters of CDDP for MIMO CSTR

Q	$5I$
R	$0.1I$
P_N	$5I$
α_g	0.02
α_c	0.02
Σ_{a_k}	$0.01I$

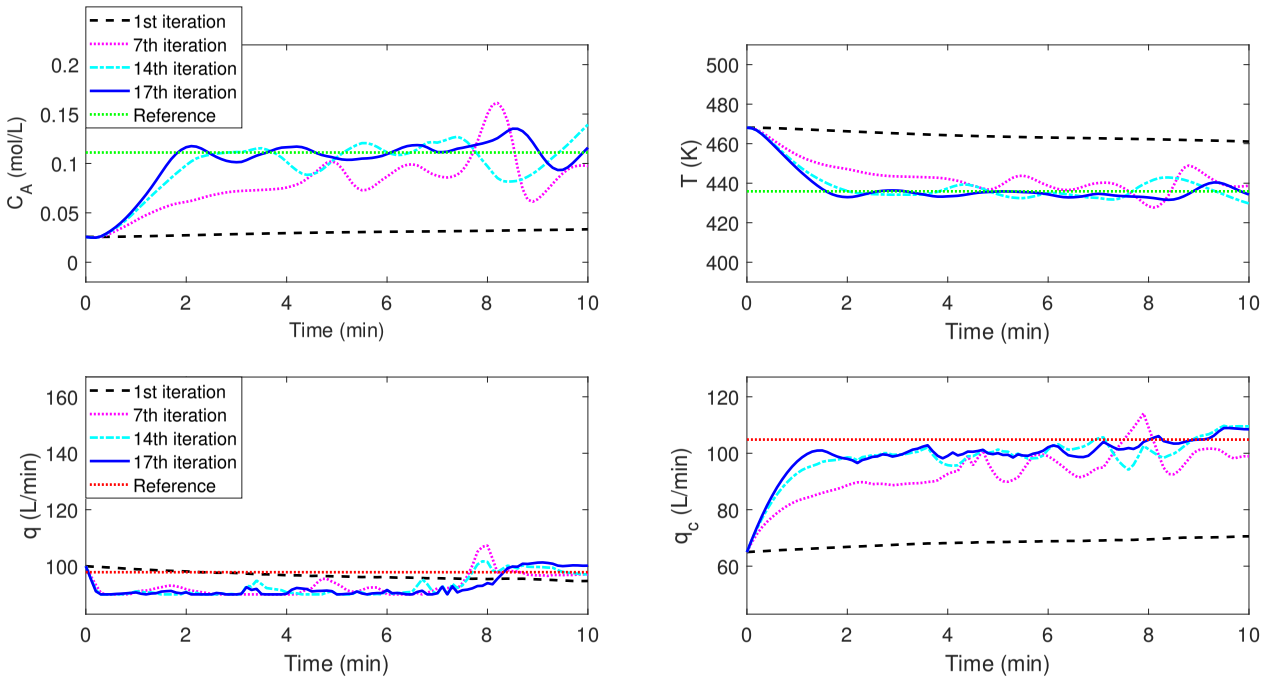

Figure 5.3: Trajectories from CDDP

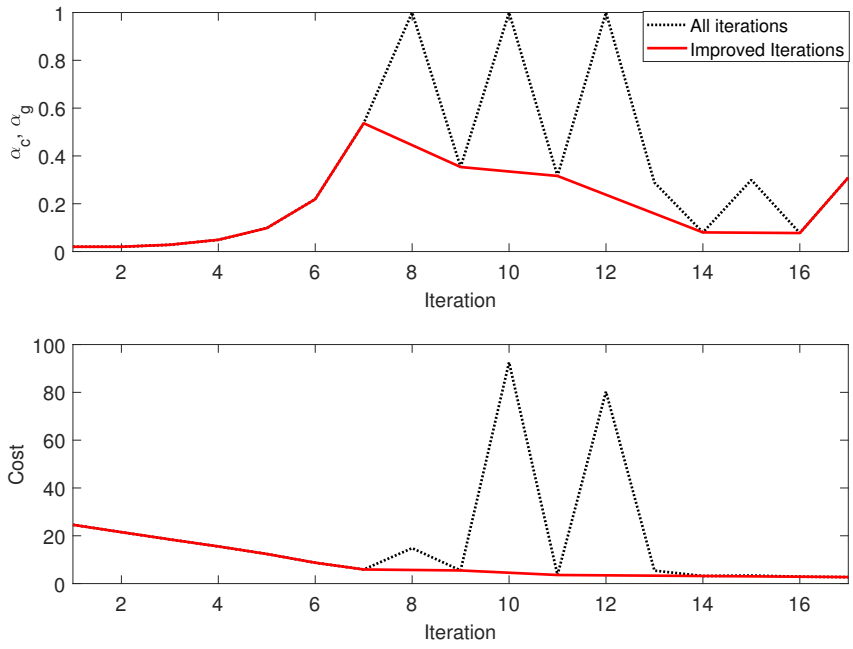

Figure 5.4: Cost of CDDP (red line: the iterations that improves the cost of all previous iterations, black dash-dot line: all iterations)

The parameters of MLMPC is shown in Table 5.3, and the result is shown in Table 5.4 and Figures 5.5, 5.6, and 5.7. The proposed MLMPC achieves offset-free tracking using the models from the 17^{th} trajectory of the CDDP shown in Figure 5.5. Figure 5.6 shows the locations where the clusters and the representative models are constructed at the final iteration. When the MLMPC achieves offset-free tracking, it converges to linear offset-free MPC as shown in Figure 5.7. Before achieving offset-free tracking, the trajectories from MLMPC has oscillation. The gap metrics between the representative model and the models in a cluster, and the distances between the linearized points, i.e, the origins, of the models and the set-point are shown in Table 5.4. It is checked that both the gap metric and the distance are important to track the set-point. If the distance between the linearized point of a model and the set-point is large, the model may not accurately predict the behavior around the set-point even if the gap metric between the representative model and the models in a cluster is small. The gap metric just indicates the stability at the origins of two closed-loop systems controlled by one controller. Thus, CDDP must generate the trajectory whose states and inputs are close to the equilibrium point at the set-point. We also verify that the MLMPC rejects disturbance by injecting step input disturbances whose magnitudes are 10 for the first input at 5 min and -5 for the second input at 7 min, respectively. Figure 5.8 shows the controller rejects all disturbances immediately.

Table 5.3: Parameters of MLMPC for MIMO CSTR

n_{cl}	4
α_{th}	0.05
β_{th}	0.1
p_{th}	10^{-6}
Λ_e	$100I$
Q	I
R	$0.1I$

Table 5.4: Gap metric and distance between models and set-point

	Gap metric	$\lvert\Delta C_A\rvert$(mol/L)	$\lvert\Delta T\rvert$(K)	$\lvert\Delta q\rvert$ (L/min)	$\lvert\Delta q_c\rvert$ (L/min)
7^{th} iteration					
1^{st} model	0.14	0.08	28.90	5.22	32.20
2^{nd} model	0.18	0.06	16.24	5.99	23.74
3^{rd} model	0.31	0.05	12.36	4.84	18.51
4^{th} model	0.45	0.03	6.24	7.83	13.77
14^{th} iteration					
1^{st} model	0.23	0.08	24.15	7.78	21.87
2^{nd} model	0.55	0.01	0.44	6.94	7.73
3^{rd} model	0.56	0.01	1.11	4.18	3.36
4^{th} model	0.59	0.04	9.47	3.58	10.49
17^{th} iteration					
1^{st} model	0.27	0.08	24.79	7.79	21.41
2^{nd} model	0.50	0.00	2.23	7.42	5.48
3^{rd} model	0.54	0.00	0.85	6.53	6.63
4^{th} model	0.49	0.01	0.42	7.58	4.07

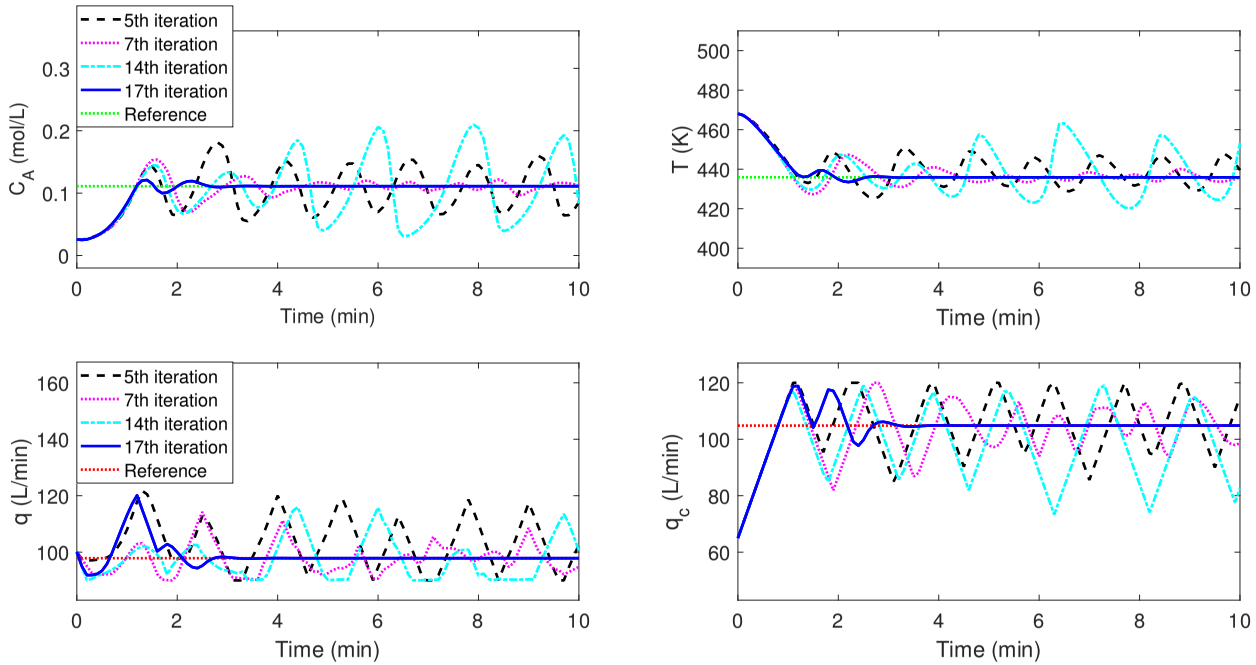

Figure 5.5: Trajectories from MLMPC

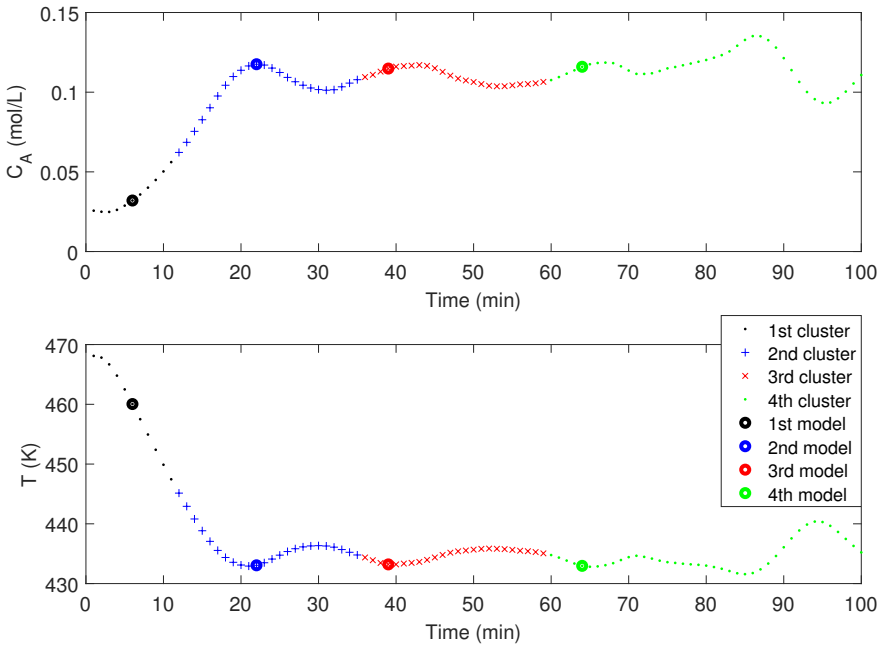

Figure 5.6: Clusters and multiple models for MLMPC from the trajectory of CDDP at the final iteration

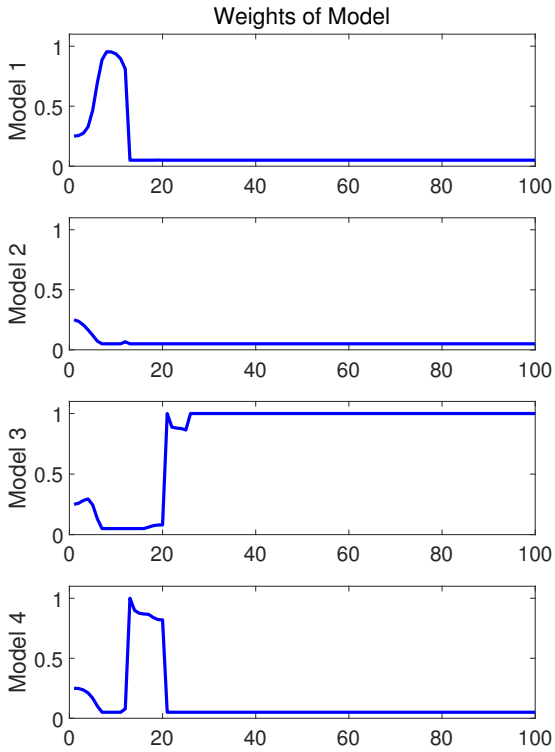

Figure 5.7: Weights of MLMPC at the final iteration

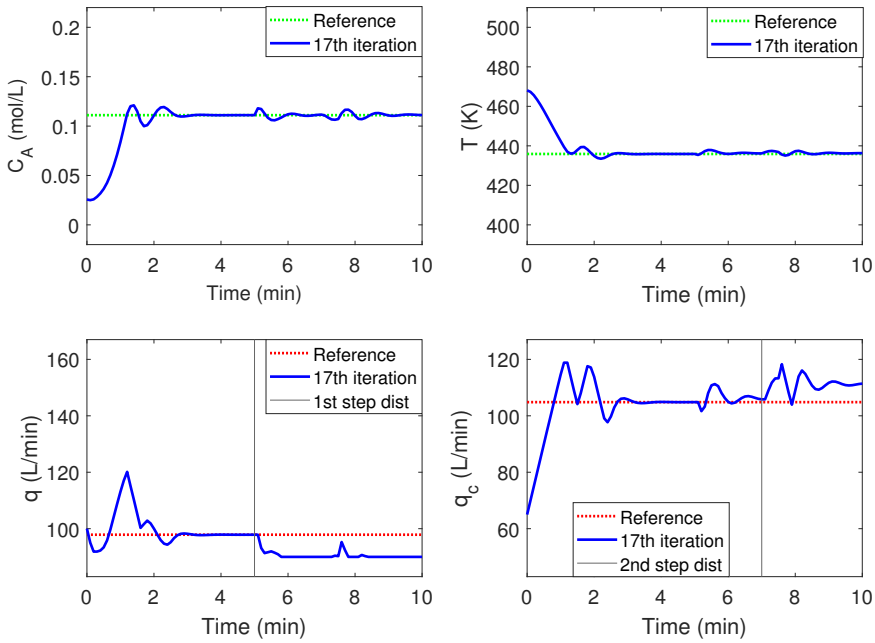

Figure 5.8: Disturbance rejection of MLMPC

In order to check the offset-free tracking property of the proposed scheme extensively, we applied the proposed CDDP and MLMPC for the wide range of initial conditions and set-points. We employed MLMPC with the improved trajectories of CDDP after 4^{th} iteration of CDDP, because initial trajectories are away from the set-point. Tables 5.5 and 5.6, and Figure 5.9 show the initial conditions, set-points, and the number of iterations of CDDP to achieve offset-tracking, respectively. The mean of the iterations for offset-free tracking is 6.13, which means most of the cases achieve offset-free tracking within six iterations. The pairs of the initial points and the set-points whose number of iterations to achieve offset-free tracking larger than 10 is shown in Figure 5.10. It seems the distance between state affects the number of iterations to achieve offset-free tracking. Dividing the difference between the initial points and the set-point by 100 and 0.1 for the temperature and the concentration, respectively, the correlation coefficient between the scaled distance between the set-points and the initial conditions and the number of iterations is calculated, and the value is 0.103. Hence, the large distance between an initial condition and a set-point does not guarantee the high number of iterations to achieve offset-free tracking.

Table 5.5: Initial conditions

	$C_A(mol/L)$	$T(K)$
x_1^{init}	0.026	468.10
x_2^{init}	0.018	484.15
x_3^{init}	0.114	435.69
x_4^{init}	0.086	444.57
x_5^{init}	0.043	461.27
x_6^{init}	0.056	457.96
x_7^{init}	0.041	458.90
x_8^{init}	0.081	443.13
x_9^{init}	0.024	473.22
x_{10}^{init}	0.030	472.22

Table 5.6: Set-points

	$C_A(mol/L)$	$T(K)$
x_1^{set}	0.111	435.89
x_2^{set}	0.041	465.91
x_3^{set}	0.094	441.22
x_4^{set}	0.022	477.63
x_5^{set}	0.020	482.18
x_6^{set}	0.025	469.73
x_7^{set}	0.038	462.66
x_8^{set}	0.053	457.98
x_9^{set}	0.051	452.79
x_{10}^{set}	0.075	448.19

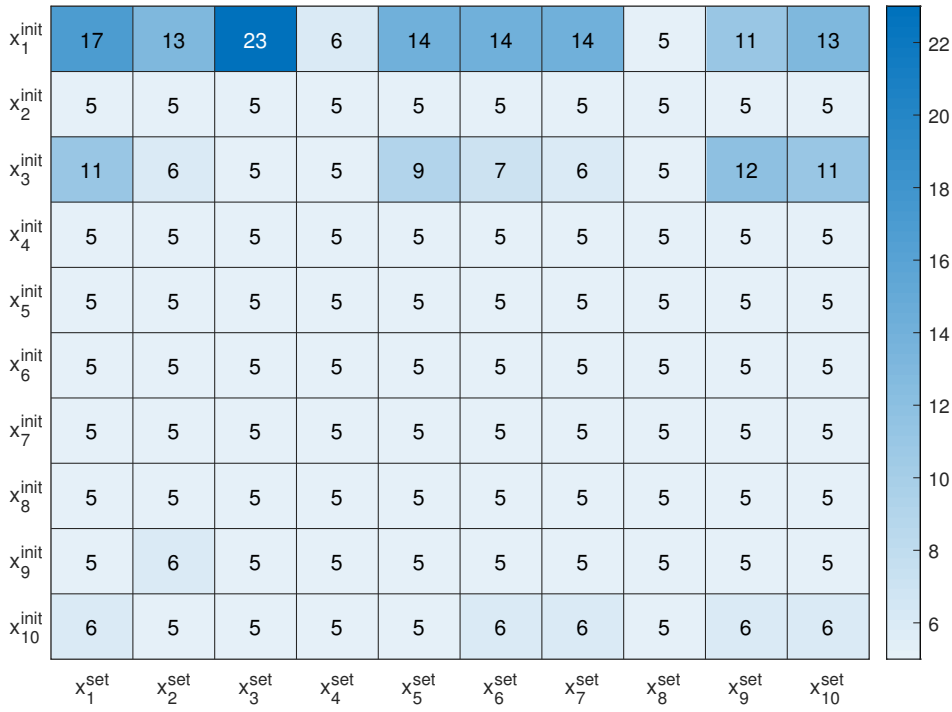

Figure 5.9: The number of iterations of CDDP at offset-free tracking of MLMPC

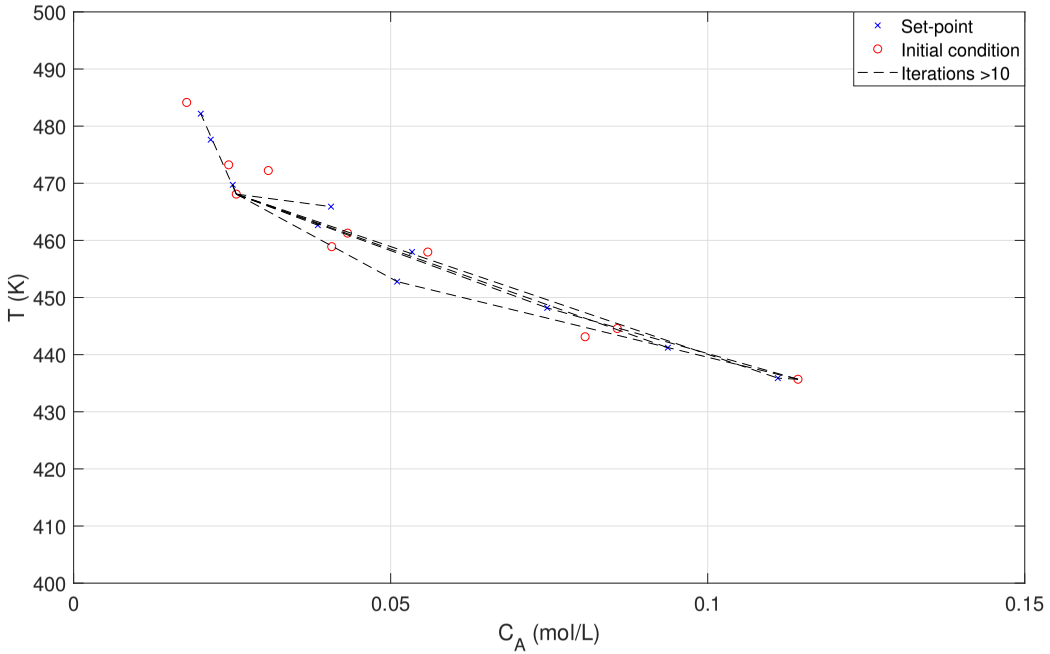

Figure 5.10: Initial points, set-points, and the pairs whose final iterations are larger than 10

Chapter 6

Design of data-driven linear time-varying model predictive control [4]

6.1 Introduction

In the previous chapter, we have introduced a framework for set-point tracking of nonlinear systems that considers input constraints without the knowledge of the model using CDDP and MLMPC. However, MLMPC cannot guarantee stability and feasibility before the state is located at the neighborhood of the set-point and MLMPC converges to linear MPC. Thus, we try to exploit LTVMPC until the state reaches the neighborhood of the set-point, which guarantees the stability and the feasibility around the nominal trajectory given by CDDP. In this chapter, we propose data-driven offset-free LTVMPC based on CDDP to track a set-point from an initial condition that considers box input constraints without the knowledge of the nonlinear or linear model. In the proposed scheme, CDDP improves the trajectory stably and iteratively, followed by obtaining the first derivatives of the process at the nominal trajectory. Then the classification of the trajectory into the transient and steady-state region is proposed,

[4] This chapter is an adapted version of B. Park, J. W. Kim, and J. M. Lee, “Data-driven model predictive control design for offset-free tracking of nonlinear systems", In preparation

where the regions are controlled by LTVMPC and offset-free MPC, respectively. The selection of the model for offset-free MPC is also proposed based on gap metric. LTVMPC to track the trajectory in the transient region is designed, which includes the proof of the recursive feasibility. Last, Offset-free MPC to track the trajectory in the steady-state region is designed, which includes the stability at the set-point is proven using gap stability margin.

6.2 Design of data-driven linear time-varying model predictive control

Although the CDDP improves the trajectory, the number of the iteration to steer the state to a set-point is unknown. Also, it is vulnerable to disturbance as a proportional (P) controller is used at each time step. Linear offset-free MPC does not work at a state far from the state where the model is identified. LTVMPC is exploited to follow a provided trajectory, which is not the optimal trajectory achieving offset-free set-point tracking. In this work, we proposed a novel offset-free LTVMPC algorithm that exploits the LTV models from a suboptimal trajectory without a nonlinear model. It consists of the algorithms for determining the neighborhood of the set-point, the model for offset-free MPC, and calculating of LTVMPC and offset-free MPC.

6.2.1 Gap metric-based model selection

To choose the representative model from LTV models of CDDP, the neighborhood of a set-point is determined first. We assume that a suboptimal trajectory to track a set-point satisfies the following: (1)

At the early stage of the operation, the transient behavior is required to approach the set-point. (2) Then, the output and input are close to the set-point and the corresponding steady-state input, which means the gap metric between dynamics at a point in a part of the trajectory and the set-point is small enough to satisfy Theorem 2.3. First, we choose a starting point of an interval from the trajectory that satisfies

$$\begin{aligned} &\min \ n_s \\ &\text{s.t.} \max_k |r_k - y_k| < e_{th}, \ \ k = n_s, \cdots, N \\ &\quad \text{Var}(\mathbf{U}_{n_s}) < \Sigma_{th} \\ &\quad \text{Var}(\mathbf{X}_{n_s}) < \Lambda_{th} \\ &\quad n_s \geq 0 \end{aligned} \tag{6.1}$$

where componentwise inequality is applied. If there does not exist n_s satisfying Eq. (6.1) or n_s is too large to reach the set-point before the operation finishes, we give up to get the model from the trajectory and an improved suboptimal trajectory is provided by the CDDP. Then, LTV models from the n_s^{th} to the $(N-1)^{th}$ step are regarded as $N - n_s$ linear time-invariant (LTI) models. We choose the model whose maximum gap metric with a model in other $N - n_s - 1$ models is minimum.

$$\begin{aligned} med = &\text{arg} \min_{i \neq j, n_s \leq i \leq N-1} \max_{n_s \leq j \leq N-1} \ \delta_g(P_i, P_j) \\ &\text{s.t.} \max_{n_s \leq j \leq N-1} \ \delta_g(P_i, P_j) < \gamma_{th} \end{aligned} \tag{6.2}$$

where P_i, is the LTV model (2.37) at the i^{th} time step. The first constraint helps to find an interval where the outputs are close to the set-

point. The second and third constraints are included to find an interval at steady-state because state and input do not change at steady-state. The last constraint can be removed if the output is the state because the first constraint makes the states in the interval have a small variance. If there does not exist the solution of (6.2), n_s increases and solve the problem again. If n_s is too large, we discard the constraint and obtain the model from (6.2). The Algorithm 16 summarizes determining the steady-state interval and the representative model.

Algorithm 16: Gap metric-based model selection algorithm

Input: $\{P_i\}$, $\mathbf{X}_0$, $\mathbf{U}_0$

Parameter: e_{th}, Σ_{th}, γ_{th}, n_{min}, n_{max}

Result: Starting point of steady-state n_s

Time index for the representative model med

Feasibility of the algorithm $flag_s$

if

$|r_k - y_k| < e_{th}(k \geq n_{max}) Var(\mathbf{U}_{n_{max}}) < \Sigma_{th},\ Var(\mathbf{X}_{n_{max}}) < \Lambda_{th}$

then

$flag_s \leftarrow 1, flag_m \leftarrow flag_s, n_s \leftarrow n_{max}$

while $flag_m$ **do**

$n_s \leftarrow n_s - 1$

if $|r_k - y_k| < e_{th}(k \geq n_s), Var(\mathbf{U}_{n_s}) < \Sigma_{th}, Var(\mathbf{X}_{n_s}) < \Lambda_{th}$ **then**

$flag_m \leftarrow 1$

else

$flag_m \leftarrow 0$

end

if $n_s < n_{min}$ **then**

break

end

end

$n_s \leftarrow n_s + 1$

Compute $\Delta_{i,j} \leftarrow \delta_g(P_i, P_j),\ \ n_s \leq i, j \leq N-1$

$\Delta^m \leftarrow \min_i \max_j \Delta_{i,j},\ \ n_s \leq i, j \leq N-1$

while $\Delta^m > \gamma_{th}$ **do**

$n_s \leftarrow n_s + 1$

$\Delta^m \leftarrow \min_i \max_j \Delta_{i,j},\ \ n_s \leq i, j \leq N-1$

$med \leftarrow arg\min_i(\max_j \Delta_{i,j}),\ \ n_s \leq i, j \leq N-1$

if $n_s > n_{max}$ **then**

break

end

end

else

$flag_s \leftarrow 0, n_s \leftarrow none, med \leftarrow none$

end

6.2.2 Offset-free linear time-varying model predictive control

Based on CDDP, offset-free MPC, and LTVMPC, we propose to design a offset-free LTVMPC algorithm that achieves offset-free control without the nonlinear model. First, we assume that the CDDP Algorithms 11, 12, and 13 generates a suboptimal trajectory for MPC in finite iterations.

Assumption 6.1. *There exists the number of iterations n_{cddp} such that CDDP Algorithms 11, 12, and 13 generate the n_{cddp}^{th} trajectory $\mathbf{Z}_0$ and $\mathbf{A}_0$ and LTV models $\{P_i\}$ for which Algorithm 16 is feasible.*

Once a suboptimal trajectory from CDDP satisfies Assumption 6.1, $\mathbf{X}_0$ and $\mathbf{U}_0$ from $\mathbf{Z}_0$ is a feasible trajectory of the system (2.1). Trajectory following of LTVMPC (2.38) can be guaranteed with some assumptions and the computation of the terminal constraint $\mathcal{X}_f$.

Assumption 6.2. *There is no model mismatch between the prediction model and the plant, i.e.,*

$$A_k = \left.\frac{\partial f}{\partial x}\right|_{x_k,u_k}, B_k = \left.\frac{\partial f}{\partial u}\right|_{x_k,u_k} \tag{6.3}$$

In order to compute $\mathcal{X}_f$, consider the system (2.1) controlled by time-varying feedback control law to track a reference trajectory $u^v(k) = K_k x^v(k)$, i.e., the autonomous system:

$$x_{k+1} = f_a(x_k) := f(x_k, K_k(x_k - \bar{x}_k) + \bar{u}_k) \tag{6.4}$$

where $\{\bar{x}_k\}$ and $\{\bar{u}_k\}$ are the reference trajectory, and x^v and u^v are the deviation variables defined in (2.37). Then one-step controllable

set, N-step controllable set, positive invariant set, and maximal positive invariant set are defined.

Definition 6.1. *For the system* (2.1) *and the reference trajectory* $\{\bar{x}_k\}$, $\{\bar{u}_k\}$, *the one-step controllable set to the set* $\mathcal{S}$ *at the* k^{th} *time step is defined as*

$$Pre^k(\mathcal{S}) = \{x^v(k) \in \mathbb{R}^n | \exists u^v(k) \in \mathcal{U}_k^v, f(x_k, u_k) - f(\bar{x}_k, \bar{u}_k) \in \mathcal{S}\} \tag{6.5}$$

where $\mathcal{U}_k^v$ is the set for the input constraints in (2.1).

Definition 6.2. *For a given target set* $\mathcal{S}$, *the N-step controllable set* $\mathcal{K}_N^k(\mathcal{S})$ *of system* (2.1) *and the reference trajectory* $\{\bar{x}_k\}$, $\{\bar{u}_k\}$ *at the* k^{th} *time step is defined recursively*

$$\mathcal{K}_j^k(\mathcal{S}) = Pre^k(\mathcal{K}_{j-1}^{k+1}(\mathcal{S})), \;\; \mathcal{K}_0^{k+j}(\mathcal{S}) = \mathcal{S} \tag{6.6}$$

Definition 6.3. $Pre_{LQR}^k(\mathcal{S})$ *is defined as the one-step controllable set to the set* $\mathcal{S}$ *at the* k^{th} *time step, where time-varying LQR gain* (2.40) *is applied.*

$$Pre_{LQR}^k(\mathcal{S}) = \{x^v(k) \in \mathbb{R}^n | K_k x^v(k) \in \mathcal{U}_k^v, f(x_k, u_k) - f(\bar{x}_k, \bar{u}_k) \in \mathcal{S}\} \tag{6.7}$$

Definition 6.4. $\mathcal{K}_{LQR,N}^k(\mathcal{S})$ *of system* (2.1) *and the reference trajectory* $\{\bar{x}_k\}$, $\{\bar{u}_k\}$ *at the* k^{th} *time step is defined recursively*

$$\mathcal{K}_{LQR,j}^k(\mathcal{S}) = Pre_{LQR}^k(\mathcal{K}_{LQR,j-1}^{k+1}(\mathcal{S})), \;\; \mathcal{K}_{LQR,0}^{k+j}(\mathcal{S}) = \mathcal{S} \tag{6.8}$$

Then the recursive feasiblity of LTVMPC (2.38) is ensured by

computing the terminal constraint $\mathcal{X}_{k+p}^f$.

Theorem 6.1. *LTVMPC* (2.38) *for the system* (2.1) *is feasible for all* $0 \leq k \leq N$ *and the terminal constraint* $\mathcal{X}_N^f$ *if* p *is chosen such that* P_{k+p} *is calculated by* (2.40), $\mathcal{X}_{k+p}^f = \mathcal{K}_{LQR,N-k-p}^{k+p}(\mathcal{X}_N^f)$ *for* $0 \in \mathcal{X}_N^f = \mathcal{K}_{LQR,0}^N$, *and if* $x^v(k) \in \mathcal{K}_p^k(\mathcal{X}_{k+p}^f)$.

Proof *If* $x^v(k) \in \mathcal{K}_p^k(\mathcal{X}_{k+p}^f)$, *then the system is feasible at* $t = k$. *By definition of* $\mathcal{K}_{LQR,N-k-p}^{k+p}$, *there exists a sequence of inactive input by which the state is steered to* $\mathcal{K}_{LQR,0}^N$ *at the final time step. Thus, the system* (2.1) *is feasible for all* $k \leq t \leq N$. ■

We define the optimal predicted cost of the trajectory at the k^{th} time step to evaluate the predicted trajectory by LTVMPC (2.38).

$$J_k^*(x_k) := \sum_{i=0}^{k-1} \|y^v(i)\|_Q^2 + \|u^v(i)\|_R^2 + \sum_{j=k}^{N-1} \left\|y_{j|k}^v\right\|_Q^2 + \left\|u_{j|k}^v\right\|_R^2 + \left\|y_{N|k}^v\right\|_Q^2 \tag{6.9}$$

where $u_{j|k}$ is the optimal input at time j obtained by solving LTVMPC (2.38) at time k, and $x_{j|k}$ is the predicted state at time j by applying $\{u_{j|k}\}$

Theorem 6.2. *Consider the system* (2.1), *the LTV models* (2.37), *the LTVMPC* (2.38), *and the terminal constraint* $0 \in \mathcal{X}_N^f$ *to track* $\mathbf{X}_0$ *and* $\mathbf{U}_0$. *Let Assumptions 6.1 and 6.2 hold. Suppose that there exists the prediction horizon* p_0 *at the* 0^{th} *time step for LTVMPC* (2.38) *such that there exists the terminal constraint at the* 0^{th} *time step,* $\mathcal{X}_{p_0}^f = \mathcal{K}_{LQR,N-p_0}^{p_0}$, *and* $x^v(0) \in \mathcal{K}_{p_0}^0(\mathcal{X}_{p_0}^f)$. *Then LTVMPC at time* k *is always feasible by reducing the prediction horizon, i.e,* $0 \leq p_k \leq p_{k-1}$. *In addition, the optimal predicted cost of the trajectory at the* k^{th} *time step,* J_k^*, *does not increase as* k *increases.*

Proof *Let p be the prediction horizon at time k. Consider the state of the process* (2.1) *at k^{th} time step that satisfies $x^v(k) \in \mathcal{K}_p^k(\mathcal{X}_{k+p}^f)$. Let $\mathbf{U}_{k,p}^* = \{u_{k|k}^*, \cdots, u_{k+p-1|k}^*\}$, be the optimal input sequence of LTVMPC* (2.38) *and $\mathbf{X}_{k,p}^* = \{x_{k|k}^*, \cdots, x_{k+p|k}^*\}$ be the corresponding optimal state. Let $J_k^*(x_k)$ be the optimal predicted cost of* (2.38) *when applying $\mathbf{U}_{k,p}^*$ to the system state x_k.*

Suppose that LTVMPC at the $(k+1)^{th}$ time step is feasible for the prediction horizon $p > 0$. The upper bound of $J_{k+1}^*(x_{k+1})$ can be obtained by applying $\mathbf{U}_{k+1,p} = \{u_{k+1|k}^*, \cdots, u_{k+p-1|k}^*, K_{k+p}(x_{k+p|k}^* - \bar{x}_{k+p}) + \bar{u}_{k+p}\}$. The resulting sequence of the state is

$$\mathbf{X}_{k+1,p} = \{x_{k+1|k}^*, \cdots, x_{k+p|k}^*, (A_{k+p} + B_{k+p}K_{k+p})(x_{k+p|k}^* - \bar{x}_{k+p}) + \bar{x}_{k+p+1}\} \tag{6.10}$$

Let $J_{k+1}(x_{k+1})$ be the predicted cost of (2.38) *when applying $\mathbf{U}_{k+1,p}$ to the system state x_{k+1}. Then*

$$\begin{aligned}
&J_{k+1}(x_{k+1}) = J_k^*(x_k) - (\left\|y_{k|k}^v\right\|_Q^2 + \left\|u_{k|k}^v\right\|_R^2 + \left\|x_{k+p|k}^v\right\|_{P_{k+p}}^2) \\
&+ \left\|y_{k+p|k+1}^v\right\|_Q^2 + \left\|u_{k+p|k+1}^v\right\|_R^2 + \left\|x_{k+p+1|k+1}^v\right\|_{P_{k+p+1}}^2 \\
&= J_k^*(x_k) - (\left\|y_{k|k}^v\right\|_Q^2 + \left\|u_{k|k}^v\right\|_R^2 + \left\|x_{k+p|k}^v\right\|_{P_{k+p}}^2) \\
&+ (x_{k+p|k+1}^v)^T(C_{k+p+1}^T Q C_{k+p+1} + K_{k+p}^T R K_{k+p} \\
&+ (A_{k+p} + B_{k+p}K_{k+p})^T C_{k+p+1}^T P_{k+p+1} C_{k+p+1}(A_{k+p} + B_{k+p}K_{k+p}))x_{k+p|k+1}^v \\
&= J_k^*(x_k) - (\left\|y_{k|k}^v\right\|_Q^2 + \left\|u_{k|k}^v\right\|_R^2)
\end{aligned} \tag{6.11}$$

The last equality in (6.11) *comes from* (2.40)*. Thus,*

$$J_{k+1}^*(x_{k+1}) - J_k^*(x_k) \leq J_{k+1}(x_{k+1}) - J_k^*(x_k) = -(\left\|y_{k|k}^v\right\|_Q^2 + \left\|u_{k|k}^v\right\|_R^2) \tag{6.12}$$

Because R is positive definite and Q is positive semidefinite, the cost decrease if the x_k is not located at the reference $\bar{x}_k$.

Suppose that LTVMPC at the $(k+1)^{th}$ time step is infeasible for the prediction horizon $p > 0$. As LTVMPC at time k is feasible and Assumption 6.2 is satisfied, $x^v(k+1) \in \mathcal{K}_{p-1}^{k+1}(\mathcal{X}_{k+p}^f)$. If we choose the prediction horizon as $p-1$, there exists the terminal cost and constraint, i.e., $P_{k+1+p-1} = P_{k+p}$, $\mathcal{X}_{k+1+p-1}^f = \mathcal{X}_{k+p}^f$. In addition, $x^v(k+1) \in \mathcal{K}_{p-1}^{k+1}(\mathcal{X}_{(k+1)+(p-1)}^f)$. Hence, LTVMPC at time $(k+1)$ is feasible because it satisfies the conditions in Theorem 6.1. The cost does not change because the terminal state and weight is equal to those of LTVMPC at time k.

If $p = 0$ at time k, it means that $x^v(k) \in \mathcal{K}_{LQR,N-k}^k$ and LQR solution is applied for input, i.e., $u^v(k) = K_k x^v(k)$. Because Assumption 6.2 is satisfied, $x^v(k+1) \in \mathcal{K}_{LQR,N-k-1}^k$. Then $p = 0$ makes LTVMPC at time $(k+1)$ feasible. The cost does not change due to the relation (2.40)*.*

Hence, LTVMPC at time k is always feasible by reducing the prediction horizon, and the cost of the trajectory J_k^* does not increase as k increases. ■

Hence, the initial deviation x_0^v and the terminal constraint $\mathcal{X}_N^f$ determines the feasibility of tracking the reference trajectory. If it is feasible and the final state of the resulting trajectory is close to the set-point, Offset-free MPC can be exploited to track the set-point, where the final state of LTVMPC is the initial state of offset-free MPC. The following theorem shows that a linear offset-free MPC can track a set-point if the initial point is near the set-point, and the gap metric between the model and the dynamics at the set-point is small.

Theorem 6.3. *Suppose MPC* (2.9)*,* (2.10)*, and* (2.6) *is applied to the system* (2.1) *given a linear model* P_m *and a set-point* r*. Assume that* Q_T *is the solution of DARE for* P_m *and the weights* Q *and* R *in* (2.31) *and* K_{lqr} *is the corresponding LQR gain. Let* (x_r, u_r) *and* P_r *be the equilibrium point corresponding to* r *and the linearized system at* (x_r, u_r)*. If* $b_{P_m,K_{lqr}} > \delta_g(P_m, P_r)$*, then the equilibrium state* x_r *corresponding to the set-point* r *in closed-loop system* (2.1)*,* (2.9)*, and* (2.10)*, and* (2.6) *is asymptotically stable.*

Proof *If the state* $x(k)$ *is in the neighborhood of* x_r *in which the solution of the MPC problem* (2.9) *equivalent to that of the unconstrained case, the solution is equivalent to the solution of the following LQR [1]:*

$$\begin{aligned} J^*_\infty(\delta x_0) = \min \sum_{j=0}^{\infty} \|\delta x_j\|^2_Q + \|\delta u_j\|^2_R = \|\delta x_0\|^2_{Q_T} \\ s.t.\ \delta x_{j+1} = A\delta x_j + B\delta u_j \\ \delta x_0 = \hat{x}(k) - \bar{x}(k),\ \delta u_0 = u(k) - \bar{u}(k). \end{aligned} \tag{6.13}$$

Then the closed-loop system is

$$\begin{aligned} x(k+1) = f(x(k), -K_{lqr}\delta x + \bar{u}(k)), \\ y(k) = h(x(k)). \end{aligned} \tag{6.14}$$

The linearized and discretized system of (6.14) *at* x_θ *is*

$$\begin{aligned} \delta x(k+1) &= A_r \delta x(k) + B_r \delta u(k) \\ &= (A_r - B_r K_{lqr}) \delta x(k), \\ \delta y(k) &= C_r \delta x(k), \end{aligned} \tag{6.15}$$

where k *is the sampling instant;* $\delta x(k) = x(k) - x_r$*;* $\delta u(k) = u(k) - u_r$*;* $\delta y(k) = y(k) - r$*;* (A_r, B_r, C_r) *is the matrices corresponding to* P_r*; Because LQR gain* K_{lqr} *satisfies Theorem 2.3, the autonomous system* (6.15) *is asymptotic stable and the eigenvalues of* $(A_r - B_r K_{lqr})$ *are inside the unit circle, which is hurwitz. Hence,* (x_r, u_r) *in the closed-loop system is asymptotically stable.* ■

Hence, the overall scheme of model-free offset-free LTVMPC starting with a suboptimal trajectory is proposed as follows. First, Algorithm 13 identifies LTV models around the suboptimal trajectory. CDDP is conducted by the model and Algorithm 11 and 12 until the resulting trajectory is improved. Then, Algorithm 16 classifies the trajectory into the transient and steady-state region, and select a model for offset-free MPC among the models in the steady-state region. Then Offset-free LTVMPC is applied to the system. First, LTVMPC tries to follow the nominal trajectory in the transient region, and offset-free MPC is applied in the steady-state region and tries to track the set-point, not the trajectory in the steady-state region. Algorithm 17 summarizes the overall scheme.

Remark 6.1. *For LTVMPC in the transient region, Assumption 6.2 is difficult to satisfy, because identified models and the dynamics away from the nominal trajectory can be different from the dynamics at the nominal trajectory. Thus, the proposed scheme implicitly assumes*

that the gap metric between the identified LTV models and the dynamics at the nominal trajectory is small, and the state controlled by LTVMPC is close to the nominal trajectory. Even if the state at the switching time is far from the nominal state due to model-plant mismatch and LMPC does not achieve offset-free control, we can optimize the trajectory, obtain the models, and apply the proposed LTVMPC and LMPC by running CDDP.

Remark 6.2. *For offset-free MPC in the steady-state region, The gap metric between the model of the offset-free MPC and the linearized system at the set-point should be small enough to satisfy Theorem 3.1. Because the trajectory from CDDP converges to the set-point as the number of iterations increases, we can obtain an improved model in terms of the gap metric with the linearized system at the set-point after additional iterations of the proposed CDDP. However, we do not know when the state reaches the set-point. Thus, Algorithm 16 determines the steady-state region according to the output error and variance of input. In addition, it chooses the model for offset-free MPC according to the gap metric among other models in the steady-state region, because we do not know the dynamics and the steady-state input at the set-point.*

Algorithm 17: Data-driven offset-free LTVMPC

Input: $\mathbf{Z}_0$, $\mathbf{A}_0$, $\{r_k\}$
Parameter: nf, J_{th}, $\mathcal{X}_{n_s}^f$
Result: $\mathbf{X}_0^{MPC}$,$\mathbf{U}_0^{MPC}$
$flag \leftarrow 1$
while $flag$ **do**
  $flag_s \leftarrow 0$, get LTV Models $\{g_{z,k}\}$ and $\{g_{a,k}\}$ from Algorithm 13
  while $\sim flag_s$ **do**
    Get the optimal gains $\{K_{g,k}\}$ and $\{K_{c,k}\}$ from Algorithm 11
    Update $\mathbf{Z}_0$, $\mathbf{A}_0$ from Algorithm 12
    Get LTV Models $\{g_{z,k}\}$ and $\{g_{a,k}\}$ from Algorithm 13
    Get $flag_s$, n_s, med from Algorithm 16
  end
  Get LTV Models $\{(A_k, B_k, C_k)\}$ from $\{g_{z,k}\}$
  Get the reference trajectory $\bar{\mathbf{X}}_0$ and $\bar{\mathbf{U}}_0$ from $\mathbf{Z}_0$, $\mathbf{A}_0$
  Set LTVMPC (2.38) for final step n_s and prediction horizon p
  Set offset-free MPC (2.9) using $\{(A_{med}, B_{med}, C_{med})\}$
  Set $x^v(0) \in \mathcal{K}_p^0(\mathcal{X}_p^f)$
  for $k = 0, 1, \cdots, N-1$ **do**
    if $k < n_s$ **then**
      if *LTVMPC* (2.38) *is infeasible* **then**
        $p \leftarrow p - 1$
      end
      Apply u_k^* by solving LTVMPC (2.38)
    else
      Apply u_k^* by solving offset-free MPC (2.9)
    end
  end
  Get $\mathbf{X}_0^{MPC} = \{x_0, \cdots, x_N\}$ and $\mathbf{U}_0^{MPC} = \{u_0, \cdots, u_{N-1}\}$
  Get $\mathbf{Z}_0^{MPC}$ and $\mathbf{A}_0^{MPC}$ from $\mathbf{Z}_0^{MPC}$ and $\mathbf{A}_0^{MPC}$
  Calculate the cost of CDDP, $J_{nf}(\mathbf{Z}_{nf}^{MPC}, \mathbf{A}_{nf}^{MPC})$
  $flag \leftarrow J_{nf}(\mathbf{Z}_{nf}^{MPC}, \mathbf{A}_{nf}^{MPC}) < J_{th}$
end

6.3 Results and discussions

Consider a multi-input multi-output (MIMO) continuous stirred tank reactor (CSTR) [53].

$$\begin{aligned}
\dot{C}_A(t) =& \frac{q}{V}[C_{A0} - C_A(t)] - k_0 C_A(t) \mathrm{e}^{-E/RT(t)}, \\
\dot{T}(t) =& \frac{q}{V}[T_0 - T(t)] - \frac{\Delta H k_0}{\rho C_p} C_A(t) \mathrm{e}^{-E/RT(t)} \\
&+ \frac{\rho_c C_{pc}}{\rho C_p V} q_c(t)[1 - \mathrm{e}^{-\frac{h_A}{\rho_c C_{pc} q_c(t)}}][T_{c0} - T(t)], \\
y(t) =& [C_A(t) \;\; T(t)]^T.
\end{aligned} \tag{6.16}$$

It consists of an irreversible, exothermic reaction. The concentration C_A and the temperature T are controlled by manipulating the flow rate of A, q and the coolant flow rate, q_c. The parameters and initial values of the variables in the system are shown in Table 6.1.

The equilibrium point of the process is determined uniquely if the values of the inputs are given. Considering the magnitude constraint of the inputs, the steady-state input-output relationship is shown in Figure 6.1. The goal of the controller is steering the state from an equilibrium point to another equilibrium point in Figure 6.1, assuming that the system dynamics and the input values at the set-point are unknown.

Table 6.1: MIMO CSTR Parameters and Initial Values

Product concentration	C_A	0.0245 mol/L
Coolant flow rate	q_c	70 L min^{-1}
Feed concentration	C_{A0}	1 mol/L
Inlet coolant temperature	T_{C0}	350 K
Heat transfer term	h_A	7×10^5 cal/min K
Activation energy term	E/R	1×10^4 K
Liquid densities	ρ, ρ_c	1×10^3 g/L
Reactor temperature	T	473.23 K
Process flow rate	q	120 L min^{-1}
Feed temperature	T_0	350 K
CSTR volume	V	100 L
Heat of reaction	ΔH	-2×10^5 cal/mol
Specific heats	C_p, C_{pc}	1 cal g^{-1} K^{-1}
Reaction rate constant	k_0	7.2×10^{10} min^{-1}
Constraints on the flow rate	q_{min}, q_{max}	95, 150 L min^{-1}
Constraints on the coolant flow rate	$q_{c,min}, q_{c,max}$	60, 110 L min^{-1}
Constraints on the flow rate	$\Delta q_{c,max}, \Delta q_{max}$	5 L min^{-1}
Constraints on the coolant flow rate	$\Delta q_{c,min}, \Delta q_{min}$	-5 L min^{-1}

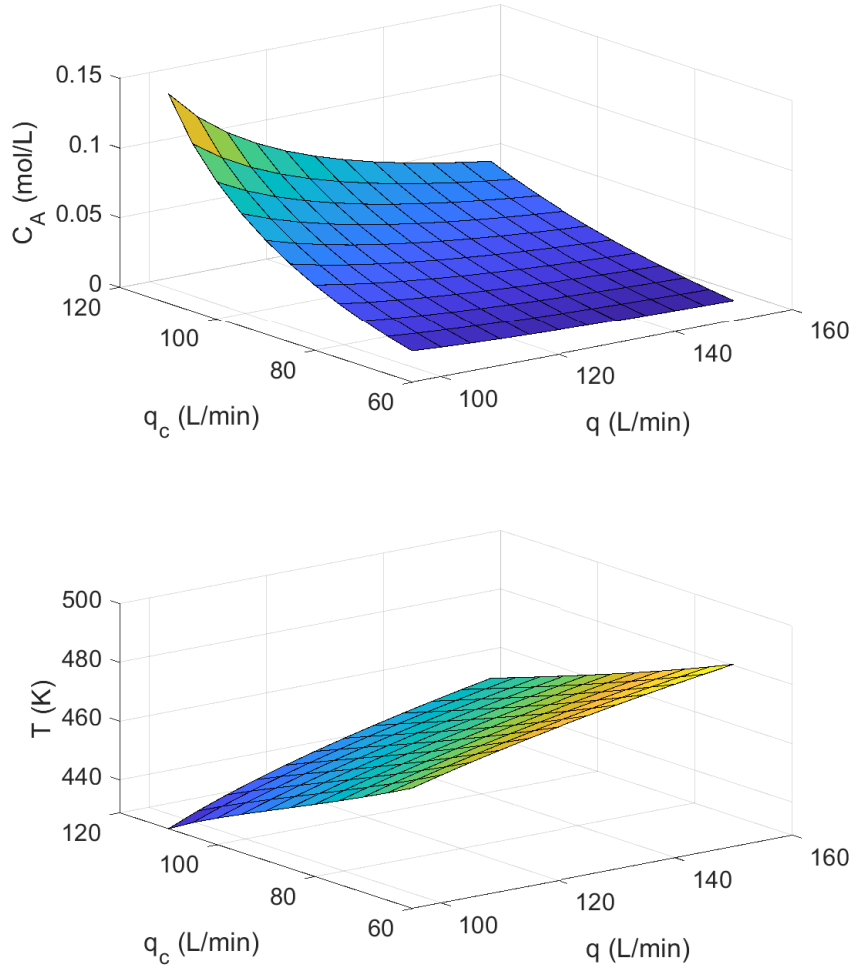

Figure 6.1: Steady-state input-output map for MIMO CSTR

First, we assume that the optimal input sequences can steer the initial state to the set-point in finite time steps and decide the terminal time step $N = 100$. Then CDDP is conducted to obtain suboptimal trajectories and models for MPC. The initial values for α_g and α_c in Algorithm 11 and 12 are 0.02 and 0.02, respectively. We choose $\beta_1, \beta_2 \sim \sqrt{n}$ and $\alpha_1, \alpha_2 \sim 1/\sqrt{n}$ in Algorithm 12, where n is the number of iterations of CDDP. In order to obtain the models for CDDP, the standard deviation $\Sigma_{a_k}^{0.5}$ to sample input in Algorithm 13 is chosen as 1% of the maximum difference of inputs, i.e., $\frac{(\Delta q_{max} - \Delta q_{min})}{100}$ and $\frac{(\Delta q_{c,max} - \Delta q_{c,min})}{100}$. The parameters of CDDP is shown in Table 6.2, and the result is shown in Figure 6.2 and 6.3. The cost, i.e., $J_0(\mathbf{Z_0}, \mathbf{A_0})$, decreases and the states steers to the set-point as the number of the iterations increases by adjusting α_g and α_c.

Table 6.2: Parameters of CDDP for MIMO CSTR

Q	$5I$
R	$0.1I$
P_N	$5I$
α_g	0.02
α_c	0.02
Σ_{a_k}	$0.01I$

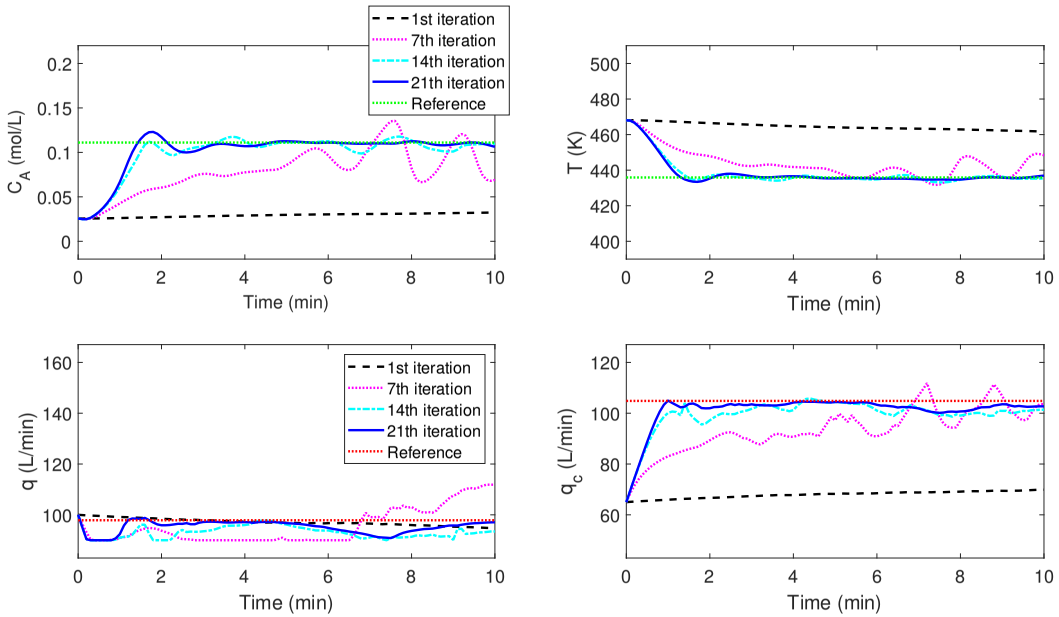

Figure 6.2: Trajectories from CDDP

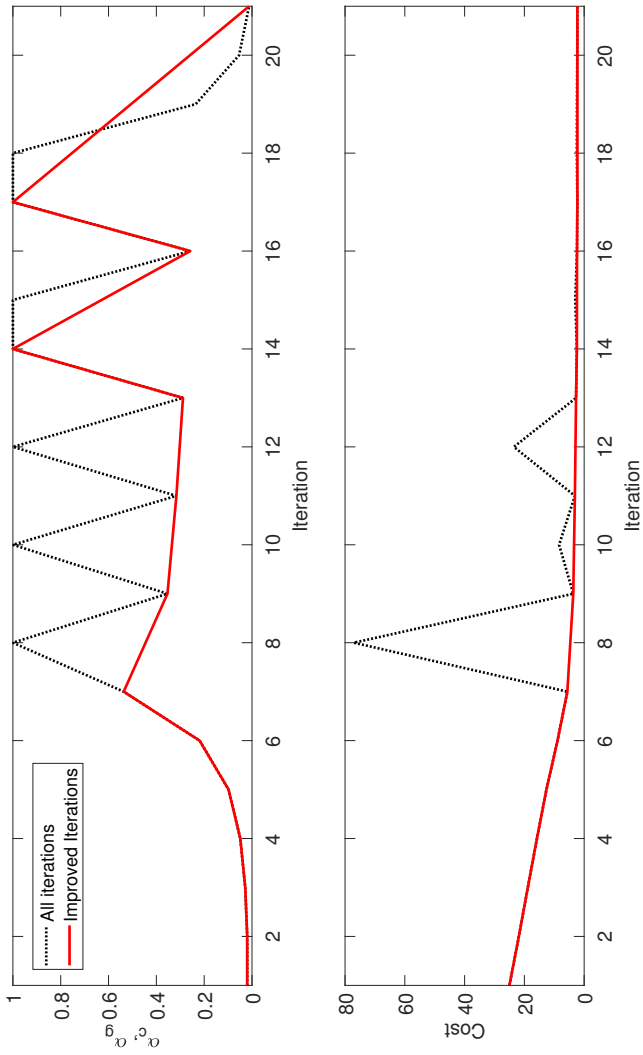

Figure 6.3: Cost of CDDP

To achieve offset-free tracking, the proposed offset-free LTVMPC is applied every time CDDP generates an improved trajectory in terms of the cost, the steady-state region exists, and the model for the steady-state region obtained accord to the condition for gap metric. The parameters of Algorithms 16 and 17 are shown in Table 6.3 and 6.4, respectively. To check offset-free tracking, the state and measurement noise are not added. Kalman filter is chosen for the observer of MPC as it is convenient to control the weight between the model and the measurement. The result is shown in Figure 6.4. The proposed MPC achieves offset-free tracking using the models from the 21st trajectory of the CDDP. The steady-state region starts at 7 (min) in all iterations where offset-free LTVMPC is applied. The MPC track the suboptimal trajectory in the transient region. However, the trajectories in the steady-state region have oscillation before it achieves offset-free tracking. The maximum gap metric between the model of offset-free MPC and the models in the steady-state region and the distance between the set-point and the linearized point are shown in Table 6.5. Because we do not know the dynamics at the set-point, we assume that the maximum gap metric is larger than the gap metric between the model and the dynamics at the set-point. The gap metrics in all iterations are close or equal to one, which does not enjoy stability result from Theorem 3.1. It is checked that the distance is also important to track the set-point. If the distance between the linearized point of a model and the set-point is large, the model may not accurately predict the behavior around the set-point because the model is valid around the linearized point. In this case, it does not achieve offset-free tracking the value if the distance of the first input Δq is larger than 3. Thus, CDDP is required to provide the trajectory whose

states and inputs are close to the equilibrium point at the set-point. We also verify that the proposed MPC rejects disturbance by injecting step input disturbances whose magnitudes are -10 for the first input at 7 min and 5 for the second input at 15 min, respectively. Figure 6.5 shows the proposed MPC rejects the disturbances effectively. The disturbance is injected after the state reaches the steady-state region because LTVMPC in the transient region is designed to track the nominal input and state trajectories, which conflicts with tracking the input that compensates the disturbance.

Table 6.3: Parameters of gap metric-based model selection for MIMO CSTR

e_{th}	$[0.02\ 10]^T$
Σ_{th}	$[7\ 7]^T$
γ_{th}	0.9
n_{min}	30
n_{max}	70

Table 6.4: Parameters of offset-free LTVMPC for MIMO CSTR

nf	10
J_{th}	1610
$\mathcal{X}_{n_s}^f$	$\{0\}$
Q	I
R	$0.1I$

Table 6.5: Gap metric between model and steady-state region and distance between model and set-point

Δ^m	$\lvert\Delta C_A\rvert$(mol/L)	$\lvert\Delta T\rvert$(K)	$\lvert\Delta q\rvert$ (L/min)	$\lvert\Delta q_c\rvert$ (L/min)
0.97	0.0081	2.61	7.19	3.96
0.9	0.0111	3.11	7.16	7.24
1	0.0141	3.61	4.39	2.56
1	0.0141	1.11	2.96	5.14

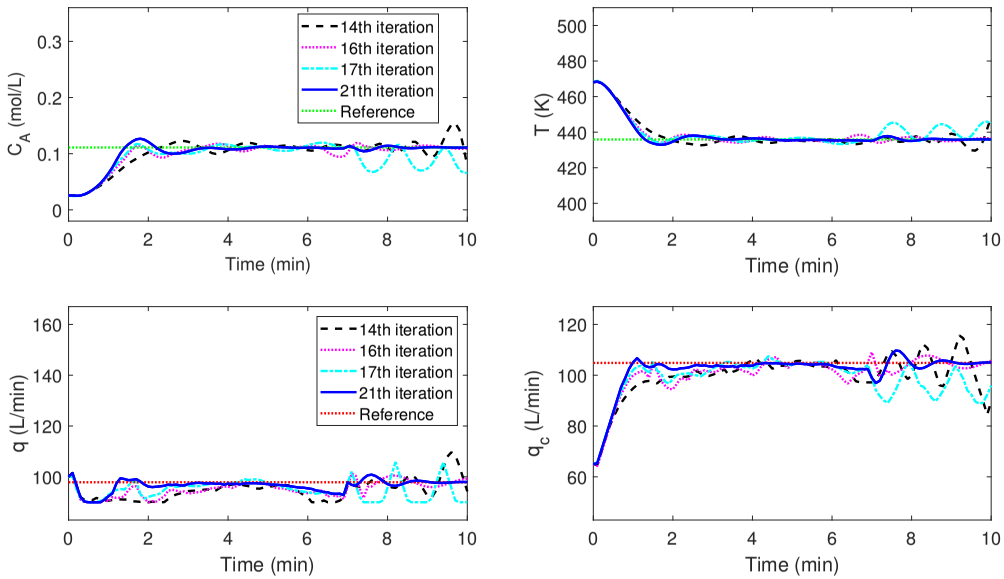

Figure 6.4: Trajectories from offset-free LTVMPC

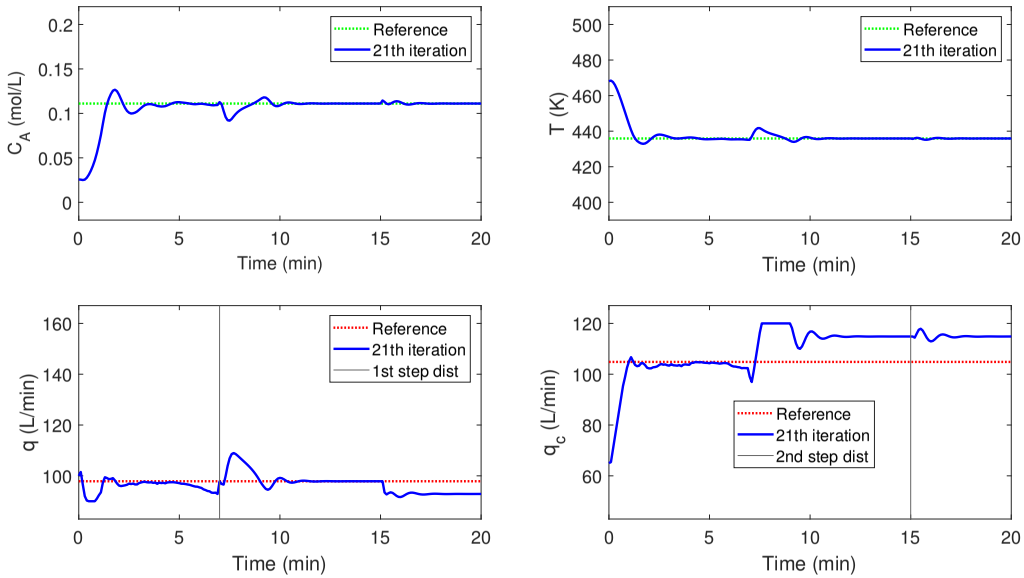

Figure 6.5: Disturbance rejection of offset-free LTVMPC

We applied the proposed scheme for various initial conditions and set-points to check the tracking property extensively. We apply the MPC with the improved trajectories of the CDDP after 4^{th} iteration of CDDP, because initial trajectories are away from the set-point. Tables 6.6, 6.7, 6.8 and 6.9 show the initial conditions, set-points, the number of iterations of CDDP and offset-free LTVMPC to achieve offset-free tracking, and the maximum gap metric between the model of offset-free MPC and the models in the steady-state region, respectively. The means of the iterations to achieve offset-free tracking is 6.65 and 1.62 for CDDP and MPC, respectively. It means most of the cases have a suboptimal trajectory with a steady-state region before the 6th iteration of CDDP and achieve offset-free tracking in two iterations of MPC. Note that the proposed scheme achieves offset-tracking fast if the maximum gap metric between the model for offset-free MPC and the models in the steady-state region is small. For the cases whose maximum gap is less than 0.7, The mean of the iterations to achieve offset-free tracking is 5.14 and 1.03 for CDDP and MPC, respectively. Otherwise, The means are 11.7 and 3.61. Thus, reducing the maximum gap metric can be helpful to achieve fast offset-free tracking. It can be accomplished in two ways. First, the state and input in the steady-state region have to be close to not only each other but also the set-point. It is achieved by improving the trajectory by CDDP. Second, the model-plant mismatch between the models and the dynamics at the trajectory in the steady-state region should be minimized, which can be attained by adjusting the parameter for input excitation Σ_a in Algorithm 13. The models from some episodes of Algorithm 16 can be compared in terms of gap metric. Σ_a is reduced if the gap metric is too large. Practically, Σ_a can be reduced until

the difference of the output can be distinguished by the measurement noise.

Table 6.6: Initial conditions

	$C_A(mol/L)$	$T(K)$
x_1^{init}	0.026	468.10
x_2^{init}	0.018	484.15
x_3^{init}	0.114	435.69
x_4^{init}	0.086	444.57
x_5^{init}	0.043	461.27
x_6^{init}	0.056	457.96
x_7^{init}	0.041	458.90
x_8^{init}	0.081	443.13
x_9^{init}	0.024	473.22
x_{10}^{init}	0.030	472.22

Table 6.7: Set-points

	$C_A(mol/L)$	$T(K)$
x_1^{set}	0.111	435.89
x_2^{set}	0.041	465.91
x_3^{set}	0.094	441.22
x_4^{set}	0.022	477.63
x_5^{set}	0.020	482.18
x_6^{set}	0.025	469.73
x_7^{set}	0.038	462.66
x_8^{set}	0.053	457.98
x_9^{set}	0.051	452.79
x_{10}^{set}	0.075	448.19

Table 6.8: The number of iterations of CDDP and MPC at offset-free tracking

	x_1^{set}	x_2^{set}	x_3^{set}	x_4^{set}	x_5^{set}	x_6^{set}	x_7^{set}	x_8^{set}	x_9^{set}	x_{10}^{set}
x_1^{init}	(21,4)	(20,5)	(10,4)	(7,3)	(18,9)	(8,4)	(11,4)	(6,2)	(29,11)	(15,4)
x_2^{init}	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)
x_3^{init}	(15,1)	(18,3)	(5,1)	(5,1)	(5,1)	(5,1)	(9,5)	(6,2)	(20,5)	(17,8)
x_4^{init}	(5,1)	(5,1)	(7,1)	(6,1)	(5,1)	(5,1)	(5,1)	(6,1)	(5,1)	(5,1)
x_5^{init}	(5,1)	(5,1)	(7,1)	(6,1)	(5,1)	(5,1)	(5,1)	(6,1)	(5,1)	(5,1)
x_6^{init}	(5,1)	(5,1)	(6,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)
x_7^{init}	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)
x_8^{init}	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)
x_9^{init}	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)	(5,1)
x_{10}^{init}	(6,2)	(9,1)	(5,1)	(5,1)	(6,2)	(5,1)	(6,2)	(5,1)	(7,2)	(5,1)

Table 6.9: The maximum gap metric between the model for offset-free MPC and models in the steady-state region

	x_1^{set}	x_2^{set}	x_3^{set}	x_4^{set}	x_5^{set}	x_6^{set}	x_7^{set}	x_8^{set}	x_9^{set}	x_{10}^{set}
x_1^{init}	1	1	1	1	0.94	0.99	0.99	1	0.98	0.98
x_2^{init}	0.3	0.19	0.38	0.24	0.28	0.28	0.33	0.31	0.27	0.31
x_3^{init}	1	1	1	0.96	1	0.57	1	0.77	1	1
x_4^{init}	0.34	0.21	0.27	0.24	0.23	0.3	0.34	0.26	0.2	0.19
x_5^{init}	0.29	0.22	0.29	0.19	0.18	0.36	0.31	0.17	0.19	0.23
x_6^{init}	0.18	0.41	0.29	0.37	0.28	0.18	0.19	0.27	0.2	0.25
x_7^{init}	0.31	0.19	0.45	0.28	0.28	0.25	0.34	0.34	0.35	0.32
x_8^{init}	0.44	0.27	0.36	0.21	0.29	0.3	0.4	0.38	0.23	0.39
x_9^{init}	0.6	0.24	0.41	0.43	0.33	0.6	0.36	0.47	0.41	0.39
x_{10}^{init}	0.52	0.76	0.63	0.52	0.68	0.91	0.99	0.46	0.75	0.41

Chapter 7

Conclusions and future works

7.1 Conclusions

Set-point tracking of a nonlinear continuous chemical process is crucial for uniform production. Wide operating ranges and short sampling time limit employing both linear MPC and NMPC. In order to address these issues, many MPC algorithms based on multiple linear models have been developed. In this thesis, we propose MLMPC and LTVMPC algorithms based on gap metric that can be applied with and without the knowledge of the linear or nonlinear models of processes. The first part is about offset-free multilinear model predictive control based on gap metric. Three systematic algorithms are developed based on the gap metric and the stability margin: (1) the gridding algorithm, (2) the clustering algorithm, (3) the combination of the local MPCs. Compared with the conventional MMPC algorithms, The proposed gridding and clustering algorithms systematically construct a model bank regardless of the dimension of the scheduling vector. In addition, the proposed weighting method combines a prediction-based and a gap metric-based method, which improves the prediction performance and shows the stability at several set-points. A multilinear model predictive control based on gap metric and switching

strategy is proposed in the second chapter. The algorithms to design MLMPC are developed: (1)selecting boundaries, (2) graph construction, and (3) switching strategy. Two novel algorithms are proposed to construct the graph. The MLMPCs designed by the algorithms exploit a series of linear MPC to steer the state to a set-point through a series of subregions, in each of which a linear MPC is employed to guarantee stability in the subregion. In the third part, a framework is proposed for set-point tracking of nonlinear systems that considers input constraints when the nonlinear model and the steady-state input at the set-point are unknown. It consists of a constrained DDP (CDDP), modeling for CDDP and MLMPC, and MLMPC. Closed-loop simulations demonstrate that the CDDP generates improved trajectory, reaching a set-point, as the number of iterations increases. MLMPC performs well in both set-point tracking and disturbance rejection control if CDDP provides a trajectory around the set-point, the resulting model is constructed around the set-point, and the gap metric between the model and the dynamics at the set-point is small. The last part proposes offset-free LTVMPC based on model-free CDDP. It tracks a suboptimal trajectory from CDDP using LTVMPC until the state reaches the neighborhood of the set-point. Then, linear offset-free MPC is employed to track the set-point and reject disturbance.

7.2 Future works

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

- In order to achieve offset-free track from an initial point to a set-point, the approaches in the thesis should design a controller. A

global controller, which achieves offset-free tracking from a set of initial points to a set of set-points, can be designed by learning the controllers, each of which is designed for a pair of an initial point and a set-point. Guided policy search (GPS) [73], which learns the controllers designed by iterative LQR [26] can be a ingredient. However, GPS does not consider that the reward changes according to the set-point change. Contextual policy search [74] aims to provide the function that maps the objective into the optimal controller. Combining GPS and contextual policy search can be an answer.

- Trajectory optimization and modeling by DDP that considers state constraints: In many continuous processes, state constraints are imposed for safety or profitability. The CDDP used in the thesis only considers input constraints. There exist research about CDDP with nonlinear constraints [31], but it cannot contain pure state constraints. To consider general state and input constraints, DDP combined with a primal-dual interior-point method, called interior-point DDP (IPDDP), has been proposed [75]. The variation of the value function is expressed with respect to not only state and input, but also dual variable. Combining it with MPC can be more general and practical to apply for set-point tracking of nonlinear processes.

Bibliography

[1] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[2] H. Kwakernaak and R. Sivan, *Linear optimal control systems*, vol. 1. Wiley-interscience New York, 1972.

[3] O. Galán, J. A. Romagnoli, A. Palazoğlu, and Y. Arkun, "Gap metric concept and implications for multilinear model-based controller design," *Industrial & engineering chemistry research*, vol. 42, no. 10, pp. 2189–2197, 2003.

[4] M. Kuure-Kinsey and B. W. Bequette, "Multiple model predictive control strategy for disturbance rejection," *Industrial & Engineering Chemistry Research*, vol. 49, no. 17, pp. 7983–7989, 2010.

[5] J. Du and T. A. Johansen, "A gap metric based weighting method for multimodel predictive control of mimo nonlinear systems," *Journal of Process Control*, vol. 24, no. 9, pp. 1346–1357, 2014.

[6] B. Aufderheide and B. W. Bequette, "Extension of dynamic matrix control to multiple models," *Computers & chemical engineering*, vol. 27, no. 8-9, pp. 1079–1096, 2003.

[7] J. Du and T. A. Johansen, "Integrated multilinear model predictive control of nonlinear systems based on gap metric," *Industrial & Engineering Chemistry Research*, vol. 54, no. 22, pp. 6002–6011, 2015.

[8] M. Kuure-Kinsey and B. W. Bequette, "Multiple model predictive control: a state estimation based approach," in *2007 American Control Conference*, pp. 3739–3744, IEEE, 2007.

[9] M. Soliman, O. Malik, and D. Westwick, "Multiple model multiple-input multiple-output predictive control for variable speed variable

pitch wind energy conversion systems," *IET renewable power generation*, vol. 5, no. 2, pp. 124–136, 2011.

[10] M. Soliman, O. Malik, and D. T. Westwick, "Multiple model predictive control for wind turbines with doubly fed induction generators," *IEEE Transactions on Sustainable Energy*, vol. 2, no. 3, pp. 215–225, 2011.

[11] V. Lystianingrum, B. Hredzak, and V. G. Agelidis, "Multiple model estimator based detection of abnormal cell overheating in a li-ion battery string with minimum number of temperature sensors," *Journal of Power Sources*, vol. 273, pp. 1171–1181, 2015.

[12] K. Xiao, S. Mao, and J. K. Tugnait, "Congestion control for infrastructure-based crns: A multiple model predictive control approach," in *2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2016.

[13] K. Xiao, S. Mao, and J. K. Tugnait, "Maq: A multiple model predictive congestion control scheme for cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 4, pp. 2614–2626, 2017.

[14] K. Zhou and J. C. Doyle, *Essentials of robust control*, vol. 104. Prentice hall Upper Saddle River, NJ, 1998.

[15] E. Arslan, M. C. Çamurdan, A. Palazoglu, and Y. Arkun, "Multimodel scheduling control of nonlinear systems using gap metric," *Industrial & engineering chemistry research*, vol. 43, no. 26, pp. 8275–8283, 2004.

[16] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat, "Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation," *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, vol. 18, no. 8, pp. 862–875, 2008.

[17] P. Falcone, M. Tufo, F. Borrelli, J. Asgari, and H. E. Tseng, "A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems," in *2007 46th IEEE Conference on Decision and Control*, pp. 2980–2985, IEEE, 2007.

[18] A. Bemporad and C. Rocchi, "Decentralized linear time-varying model predictive control of a formation of unmanned aerial vehicles," in *2011 50th IEEE conference on decision and control and European control conference*, pp. 7488–7493, IEEE, 2011.

[19] M. M. G. Plessen and A. Bemporad, "Reference trajectory planning under constraints and path tracking using linear time-varying model predictive control for agricultural machines," *Biosystems engineering*, vol. 153, pp. 28–41, 2017.

[20] R. Sharma, D. Nesić, and C. Manzie, "Idle speed control using linear time varying model predictive control and discrete time approximations," in *2010 IEEE International Conference on Control Applications*, pp. 1140–1145, IEEE, 2010.

[21] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[22] A. V. Rao, "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, vol. 135, no. 1, pp. 497–528, 2009.

[23] C. C. Françolin, D. A. Benson, W. W. Hager, and A. V. Rao, "Costate approximation in optimal control using integral gaussian quadrature orthogonal collocation methods," *Optimal Control Applications and Methods*, vol. 36, no. 4, pp. 381–397, 2015.

[24] L. N. Trefethen, *Approximation theory and approximation practice*, vol. 164. Siam, 2019.

[25] D. H. Jacobson and D. Q. Mayne, *Differential dynamic programming*. North-Holland, 1970.

[26] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems.," in *ICINCO (1)*, pp. 222–229, 2004.

[27] L.-z. Liao and C. A. Shoemaker, "Advantages of differential dynamic programming over newton's method for discrete-time optimal control problems," tech. rep., Cornell University, 1992.

[28] D. H. Jacobson, "New second-order and first-order algorithms for determining optimal control: A differential dynamic programming approach," *Journal of Optimization Theory and Applications*, vol. 2, no. 6, pp. 411–440, 1968.

[29] D. M. Murray and S. J. Yakowitz, "Constrained differential dynamic programming and its application to multireservoir control," *Water Resources Research*, vol. 15, no. 5, pp. 1017–1027, 1979.

[30] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1168–1175, IEEE, 2014.

[31] Z. Xie, C. K. Liu, and K. Hauser, "Differential dynamic programming with nonlinear constraints," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 695–702, IEEE, 2017.

[32] G. Pannocchia and J. B. Rawlings, "Disturbance models for offset-free model-predictive control," *AIChE journal*, vol. 49, no. 2, pp. 426–437, 2003.

[33] B. A. Francis, "A course in h∞ control theory," *Lecture notes in control and information sciences*, vol. 88, 1987.

[34] T. T. Georgiou and M. C. Smith, "Optimal robustness in the gap metric," in *Proceedings of the 28th IEEE Conference on Decision and Control,*, pp. 2331–2336, IEEE, 1989.

[35] L. Qui and E. Davison, "Feedback stability under simultaneous gap metric uncertainties in plant and controller," *Systems & Control Letters*, vol. 18, no. 1, pp. 9–22, 1992.

[36] G. Zames, "Unstable systems and feedback: The gap metric," in *Proc. of the Allerton Conference, 1980*, pp. 380–385, 1980.

[37] M. Vidyasagar, "The graph metric for unstable plants and robustness estimates for feedback stability," *IEEE Transactions on Automatic Control*, vol. 29, no. 5, pp. 403–418, 1984.

[38] L. Qiu and E. J. Davison, "Pointwise gap metrics on transfer matrices," in *29th IEEE Conference on Decision and Control*, pp. 2431–2436, IEEE, 1990.

[39] R. Murray-Smith and T. Johansen, *Multiple model approaches to nonlinear modelling and control*. CRC press, 1997.

[40] B. A. Foss, T. A. Johansen, and A. V. Sørensen, "Nonlinear predictive control using local models–applied to a batch fermentation process," *Control Engineering Practice*, vol. 3, no. 3, pp. 389–396, 1995.

[41] K. K. Kumar and S. C. Patwardhan, "Nonlinear predictive control of systems exhibiting input multiplicities using the multimodel approach," *Industrial & engineering chemistry research*, vol. 41, no. 13, pp. 3186–3198, 2002.

[42] P.-J. Van Overloop, S. Weijs, and S. Dijkstra, "Multiple model predictive control on a drainage canal system," *Control Engineering Practice*, vol. 16, no. 5, pp. 531–540, 2008.

[43] C. Porfirio, E. A. Neto, and D. Odloak, "Multi-model predictive control of an industrial c3/c4 splitter," *Control engineering practice*, vol. 11, no. 7, pp. 765–779, 2003.

[44] L. Özkan and M. V. Kothare, "Stability analysis of a multi-model predictive control algorithm with application to control of chemical reactors," *Journal of Process Control*, vol. 16, no. 2, pp. 81–90, 2006.

[45] A. Vachon, A. Desbiens, E. Gagnon, and C. Bérard, "Launch ascent guidance by discrete multi-model predictive control," *Acta Astronautica*, vol. 95, pp. 101–110, 2014.

[46] J. Du, C. Song, Y. Yao, and P. Li, "Multilinear model decomposition of mimo nonlinear systems and its implication for multilinear model-based control," *Journal of Process Control*, vol. 23, no. 3, pp. 271–281, 2013.

[47] A. El-Sakkary, "The gap metric: Robustness of stabilization of feedback systems," *IEEE Transactions on Automatic Control*, vol. 30, no. 3, pp. 240–247, 1985.

[48] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*, vol. 2. Nob Hill Publishing Madison, WI, 2017.

[49] U. Maeder, F. Borrelli, and M. Morari, "Linear offset-free model predictive control," *Automatica*, vol. 45, no. 10, pp. 2214–2222, 2009.

[50] H.-S. Park and C.-H. Jun, "A simple and fast algorithm for k-medoids clustering," *Expert systems with applications*, vol. 36, no. 2, pp. 3336–3341, 2009.

[51] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.

[52] W. Tan, H. J. Marquez, T. Chen, and J. Liu, "Multimodel analysis and controller design for nonlinear processes," *Computers & chemical engineering*, vol. 28, no. 12, pp. 2667–2675, 2004.

[53] M. García-Alvarado and I. Ruiz-López, "A design method for robust and quadratic optimal mimo linear controllers," *Chemical Engineering Science*, vol. 65, no. 11, pp. 3431–3438, 2010.

[54] L. A. Wolsey, *Integer programming*, vol. 52. John Wiley & Sons, 1998.

[55] R. Bellman, "On a routing problem," *Quarterly of applied mathematics*, vol. 16, no. 1, pp. 87–90, 1958.

[56] A. Shimbel, "Structure in communication nets," in *Proceedings of the symposium on information networks*, pp. 119–203, Polytechnic Institute of Brooklyn, 1954.

[57] E. F. Moore, "The shortest path through a maze," in *Proc. Int. Symp. Switching Theory, 1959*, pp. 285–292, 1959.

[58] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[59] D. B. Johnson, "Efficient algorithms for shortest paths in sparse networks," *Journal of the ACM (JACM)*, vol. 24, no. 1, pp. 1–13, 1977.

[60] M. L. Fredman and R. E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *Journal of the ACM (JACM)*, vol. 34, no. 3, pp. 596–615, 1987.

[61] M. Thorup, "Integer priority queues with decrease key in constant time and the single source shortest paths problem," *Journal of Computer and System Sciences*, vol. 69, no. 3, pp. 330–353, 2004.

[62] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*, vol. 3. Prentice hall Upper Saddle River, NJ, 2002.

[63] J. Du, C. Song, and P. Li, "Application of gap metric to model bank determination in multilinear model approach," *Journal of Process Control*, vol. 19, no. 2, pp. 231–240, 2009.

[64] J. K. Gugaliya, R. D. Gudi, J. Mo, and G. T. Tan, "Decomposition of nonlinear dynamics using multiple model approach and gap metric analysis," Dec. 6 2011. US Patent 8,073,659.

[65] B. J. Park, D. H. Jeong, and J. M. Lee, "Offset-free multilinear model predictive control based on gap metric," *Journal of Process Control, Under Review*.

[66] T.-T. Tay, I. Mareels, and J. B. Moore, *High performance control*. Springer Science & Business Media, 2012.

[67] D. Mayne, "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems," *International Journal of Control*, vol. 3, no. 1, pp. 85–95, 1966.

[68] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives," *Pacific Journal of mathematics*, vol. 16, no. 1, pp. 1–3, 1966.

[69] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[70] E. Theodorou, Y. Tassa, and E. Todorov, "Stochastic differential dynamic programming," in *Proceedings of the 2010 American Control Conference*, pp. 1125–1132, IEEE, 2010.

[71] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[72] G. Williams, A. Aldrich, and E. A. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 344–357, 2017.

[73] S. Levine and V. Koltun, "Guided policy search," in *International Conference on Machine Learning*, pp. 1–9, 2013.

[74] A. G. Kupcsik, M. P. Deisenroth, J. Peters, G. Neumann, *et al.*, "Data-efficient generalization of robot skills with contextual policy search," in *Proceedings of the 27th AAAI Conference on Artificial Intelligence, AAAI 2013*, pp. 1401–1407, 2013.

[75] A. Pavlov, I. Shames, and C. Manzie, "Interior point differential dynamic programming," *arXiv preprint arXiv:2004.12710*, 2020.

초 록

모델예측제어 (model predictive control) 는 산업에서 널리 쓰이는 고급 공정 제어 기법으로, 다변수 시스템의 동역학과 제약 조건을 고려하여 실시간으로 현재 상태에 대해 최적해를 도출해낸다. 선형 모델을 이용하는 선형 모델예측제어 (linear model predictive control) 가 가장 간단하고 많은 이론들이 정립되어 있으나, 실제 비선형 공정에서는 선형 모델이 근사할 수 있는 좁은 운전 조건에서만 사용할 수 있다는 한계가 있다. 비선형 모델예측제어 (nonlinear model predictive control) 는 비선형 모델을 이용하여 넓은 운전 조건에서도 최적해를 제공할 수 있지만, 비선형 최적화 문제를 실시간으로 풀어야 하기 때문에 샘플링 타임이 작을 경우, 적용하기 어렵다는 한계가 있다. 다중 선형 모델예측제어 (multilinear model predictive control) 또는 선형 시변 모델예측제어 (linear time-varying model predictive control) 는 여러 개의 선형 모델을 이용하여 넓은 운전 범위에서 공정의 거동을 표현하고 최적에 가까운 해를 제공할 수 있기 때문에 앞선 두 가지 제어 기법의 한계를 보완할 수 있다. 모델예측제어 기법을 실제 비선형 공정에 적용하기에 또 다른 어려운 점은 실제 공정의 비선형 모델을 얻기가 힘들다는 것이다. 미분동적계획법 (differential dynamic programming) 은 이러한 상황에서 현재 공정 운전 데이터에 기반해 동적 최적화 (dynamic optimization) 를 수행하여 초기 조건에서 설정점까지의 최적에 가까운 경로를 찾아 모델예측제어 기법을 적용할 수 있도록 도움을 줄 수 있다. 구체적으로, 미분동적계획법은 현재 공정 운전 데이터를 이용해 운전 데이터 근처의 거동을 묘사하는 선형 모델들을 얻고, 이를 이용하여 반복적으로 다음 운전에서의 최적해를 제공하여 운전 궤적을 개선한다.

본 학위 논문에서는 넓은 운전 범위를 가진 공정에서 운전 조건을 변경하기에 적합한 다중 선형 모델예측제어와 선형 시변 모델예측제어 전략을 제시한다. 첫번째로, gap metric을 이용하여 설정점에서 모델예측제어를 적용한 시스템의 안정성을 보장하고 잔류 편차를 제거하는 다중 선형 모델예측제어 기법을 제시한다. 두번째로, 다중 선형 모델예측제어기의 진동 가능성을 막기 위해, 초기 조건에서 설정점까지의 구간을 gap metric에 기반하여 나누고, 각각의 구간에서의 하위 설정점들을 정하여 초기 조건에서 설정점까지의 경로를 하위 설정점들의 그래프로 표현하고, 각 하위 설정점까지 각각 배정된 선형 모델예측제어기를 이용하여 설정점까지 도달하게 하는 제어 전략을 제시한다. 다음으로는 공정의 모델이 없을 때, 공정의 입력 제약 조건을 고려하는 미분동적계획법을 이용하여 최적에 가까운 개루프 (open-loop) 제어 입력과 해당 운전 데이터 근방을 근사하는 선형 모델들을 얻어, 다중 선형 모델예측제어를 적용하여 설정점까지 도달하고 잔류 편차를 제거하는 제어 전략을 제시한다. 마지막으로, 미분동적계획법이 제공하는 준최적 (suboptimal) 운전 데이터를 활용하기 위해, 선형 시변 모델예측제어 기법과 잔류편차-제거 모델예측제어 기법을 이어서 사용하는 전략이 제시되었다. 구체적으로, 제공된 준최적 운전데이터를 과도 응답 (transient response) 과 정상 상태 응답 (steady-state response) 이 나타나는 구간으로 나누고, 선형 시변 모델예측제어를 통해 과도 응답에서의 준최적 궤적을 추적하고 상태 변수가 정상 상태에 가까워지면 잔류편차-제거 모델예측제어를 적용해 설정점에 도달하도록 한다. 제안된 기법들을 공정 예제에 적용하여 공정의 모델 유무와 관계없이 선형 모델들을 이용한 모델예측제어 기법이 넓은 운전 조건을 가진 비선형 공정의 공정 조건을 이동하기에 적합한

방법론임을 검증하였다.

주요어 : 최적제어, 동적최적화, 미분동적계획법, 모델예측제어
학번 : 2015-21061