# Carnegie Mellon University

## CARNEGIE INSTITUTE OF TECHNOLOGY

## THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF Doctor of Philosophy

**TITLE** Advanced-multi-step and Economically Oriented Nonlinear Model Predictive Control

**PRESENTED BY** Xue Yang

**ACCEPTED BY THE DEPARTMENT OF**

Chemical Engineering

| Lorenz Biegler | 5/4/15 |
|---|---|
| ADVISOR AND DEPARTMENT HEAD | DATE |

**APPROVED BY THE COLLEGE COUNCIL**

| Vijayakumar Bhagavatula | 5/4/15 |
|---|---|
| DEAN | DATE |

# Advanced-multi-step and Economically Oriented Nonlinear Model Predictive Control

Submitted in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

in

Chemical Engineering

Xue Yang

B.S., Control Science and Engineering, Zhejiang University

Carnegie Mellon University

Pittsburgh, PA

May, 2015

# Acknowledgement

I am grateful for Department of Chemical Engineering at Carnegie Mellon University for giving me the opportunity to finish my Ph.D. study and write this thesis. I would like to thank my advisor, Prof. Lorenz T. Biegler, for taking me as his student, and tutoring me with great patience. I feel very lucky to work with Larry. He is knowledgeable, patient, and always able to point me to the right directions. His enthusiasm and devotion to work have inspired me tremendously. I would like to thank Prof. Ignacio Grossmann, Prof. Erik Ydstie, Prof. Nikolaos Sahinidis, Prof. Chrysanthos Gounaris, and all the CHEME faculty and staff for their help and support during my Ph.D. study.

I would like to thank my committee members, Prof. Biegler, Prof. Grossmann, Prof. Ydstie and Prof. Hug for their advices and suggestions. I would like to thank the Biegler group for the inspiring discussions we had. Specifically I would like to thank Rodrigo López-Negrete, Rui Huang, Johannes Jäschke, Eranda Harinath and Devin Griffith for their guidance and help on my projects. I would like to thank all the students in the Process Systems Engineering group and I am very lucky to work with so many smart and talented people.

Many thanks go to National Science Foundation for sponsoring the work in this dissertation. Also I would like to thank Mr. and Mrs. Bertucci for their generosity to establish the Bertucci Fellowship to support Ph.D. students.

Last but not least, I would like to thank my parents for their unconditional and endless love and support. I would like to thank Juan Du, Qi Zhang and Jun Shi for being such wonderful friends and supporting me during the most difficult time. I would like to thank myself for never giving up. Life is hard, but there's got to be a way to "control and optimize" it.

# Abstract

This dissertation addresses two issues that arise in the field of Nonlinear Model Predictive Control (NMPC): computational delay and stability of economically oriented NMPC.

NMPC has gained wide attention through the application of dynamic optimization. It has the ability to handle variable bounds and multi-input-multi-output systems. However, computational delay caused by large size of nonlinear programming (NLP) problems may lead to deterioration of controller performance and system stability. In this thesis we propose an advanced-multi-step formulation of NMPC (amsNMPC) based on NLP sensitivity. The basic idea of amsNMPC is to solve a background NLP problem in advance to get predictions of future manipulated variables. These are then updated online using NLP sensitivity when the actual states are obtained. This method could be applied to optimization problems whose solutions require multiple sampling times. We then analyze the nominal and robust stabilities of the two approaches. Two examples are studied to evaluate the performance of amsNMPC.

The ultimate goal of any operation strategy for a process plant is to make profit. Traditionally this goal could be achieved by a two-layer Real-time Optimization (RTO) system, where the upper layer solves a steady state problem aiming at optimizing economic performance to get the optimal setpoints for the controlled variables in the layer below. The lower layer then keeps the controlled variables at their given setpoints using MPC/NMPC. However, there are some problems with this two-layer structure. One of the solutions is to combine these two layers and include the economic criterion directly into the cost function of the lower layer controller when an optimization-based controller such as MPC is used. This approach is often referred to as Economic MPC. The issue with Economic NMPC is that the controller may not be stable. In this dissertation we analyze its Lyapunov stability property and propose to stabilize it by adding quadratic regularization terms to the objective function, and we also provide a method to calculate the most appropriate weights on regularization terms to ensure the stability of Economic NMPC while achieving the best possible economic performance. Several challenging case studies are used to demonstrate these concepts.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

In this chapter, we discuss the background of the research problems in this dissertation. The hierarchy of plant operations will be introduced with an emphasise on the two-layer structure of Real-time Optimization (RTO). Then the research problem is defined and the thesis outline is listed.

## 1.1 Hierarchy Structure

The typical operation of a chemical process involves several layers structured as a pyramid. As shown in Fig. 1.1, it is composed of the following layers: planning, scheduling,

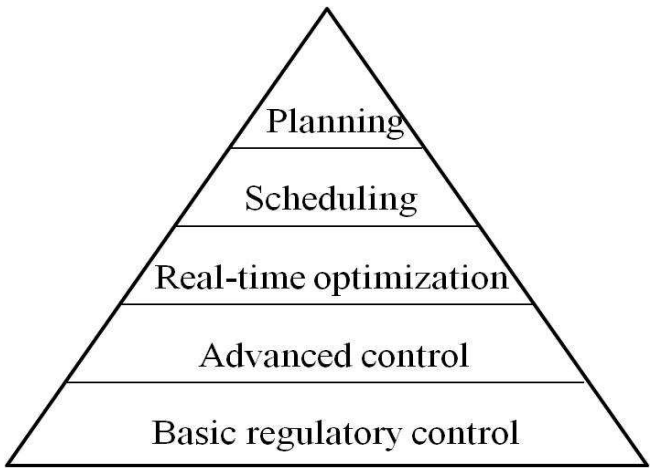



Figure 1.1: Optimization hierarchy

RTO, advanced control, and basic regulatory control. As we move from top to bottom, the decisions are generated at increased frequency. The planning layer makes decisions about production goals. For example, what products to produce and how much to produce, what raw materials to buy and how much to buy. The time scale of planning is in terms of months or weeks. The scheduling layer then makes decisions about how to arrange the manufacturing sequence and when to start a process given the decision made by the planning layer. This is usually determined by availability of equipments. The time scale of scheduling is in terms of weeks or days. The two layers also provide parameters of economic objective, e.g., prices of products and raw material, cost; and economic constraints, e.g., amount of raw material. Given such information, the RTO layer then generates setpoints of the advanced control layer aiming at optimizing economic performance in the presence of changes and long term disturbances. The time scale of RTO is in terms of hours. The advanced control layer then makes control decisions to drive the system to its setpoint in the presence of short-term disturbances. The time scale of advanced control is in terms of minutes. Then the control decision is sent to the regulatory control layer, whose time scale is in terms of seconds.

### 1.1.1 Real-time Optimization (RTO)and Advanced Control

The RTO layer solves a steady state problem aiming at maximizing profit or minimizing cost to generate setpoint of the advanced control layer. The two-layer RTO and advanced control structure is shown in Fig.1.2.

The RTO problem is usually a nonlinear programming (NLP) problem with an economic objective function. The process model included in the NLP is a steady state model. This NLP is solved online on a scale of hours in order to generate the real-time set point considering the change in economic parameters and models, long-term disturbances, etc.

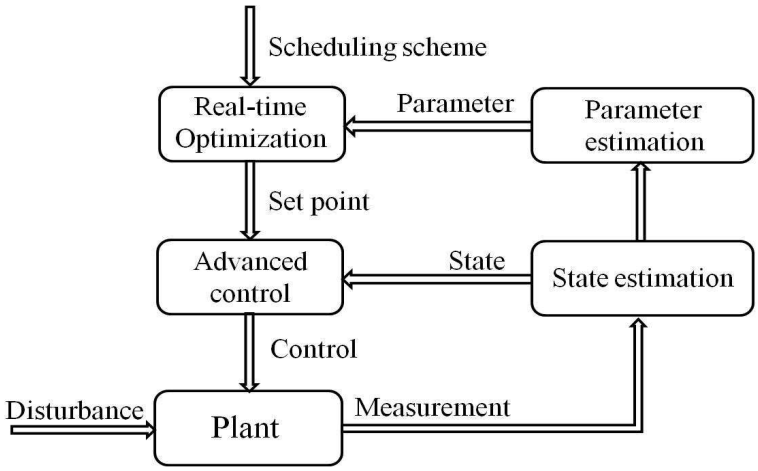



Figure 1.2: Two-layer RTO and advanced control structure

The advanced control layer generates control actions to be sent to the lower layer of regulatory control. The most commonly used advanced controller in industry is the Model Predictive Controller (MPC). It utilizes a linear model of the controlled system to generate control decisions. Normally, in the areas such as refining and petrochemicals, the process is slightly nonlinear, and a linearized model is used by MPC. However, in areas such as chemicals, polymers, gas plants and pulp & paper, processes are strongly nonlinear. In this case MPC has not been applied yet. However, if we have the dynamic model of the nonlinear processes, Nonlinear MPC (NMPC) could be used instead.

MPC uses a dynamic model of the process to predict future behavior of the process over a time horizon. It then solves an NLP to generate the control actions within this horizon that leads to the shortest transition time or minimal control effort. Besides the dynamic model, current state is also required as the initial condition of the NLP. However in reality, current state is not measured directly. The output of the plant is measurement, then state is estimated by a state estimator using the output measurement and the dynamic model. The estimated state is then sent to the advanced controller as the actual state. Some commonly

used state estimators are Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Moving Horizon Estimation (MHE). There are a lot of interesting research topics in the field of state estimation, but in this dissertation we focus on NMPC and assume that all the actual states are already obtained by state estimation.

Due to the formulation of the NLP, MPC has a lot of advantages compared to classical controllers, such as PID controllers. MPC could control multi-input-multi-output systems with no decoupling needed; and it could handle bounds on states, outputs and inputs. Therefore MPC has become the current dominant practice in the process industry.

There are several problems with the current two-layer RTO and advanced control structure. First, the RTO layer has no information on real-time disturbances in the plant. As a result, the steady state it generates may not be economically optimal in the presence of real-time disturbance. Second, the RTO layer and control layer have different time scales. Therefore, delay on the RTO layer is inevitable. Finally, model inconsistency between the two layers and unresolved transient behavior may lead to unreachable setpoints. Therefore the combination of these two parts has ignited people's interests in the recent decades, which is the idea of Economic NMPC (eNMPC). Due to its formulation, it could optimize economic profit while driving the system to its steady state in the presence of dynamic disturbances. We will discuss more about eNMPC in Chapter 7.

## 1.2 Research Problem Statement

Normally, the solution time of NLP problems are not negligible. If the scale of the system is very large, solution time of the NLP is usually comparable to, or even longer than the sampling time. As a result, there is time delay between the time that the input is implemented and the time that the plant state is obtained. This delay will lead to incorrect control

actions which will deteriorate the controller performance and even system stability.

There are a lot of fast NMPC methods to handle computational delay, which will be discussed in Chapter 4. And we also propose a fast NMPC method which takes advantage of the existing fast NMPC methods and allows the NLP to be solved beyond one sampling time.

Economic NMPC aims at maximizing economic profit directly. Since the setpoint is not known in advance, there are no setpoints or terminal regions formulated in the NLP. Stability analysis of setpoint tracking NMPC has been well established because its objective function, which is composed of quadratic tracking terms, satisfies some certain criteria that will be introduced in Chapter 4. However, the stability analysis of setpoint tracing NMPC cannot be directly applied to Economic NMPC because since the objective function is composed of economic terms, those criteria may not be satisfied. In this dissertation we propose a scheme to analyze the stability of an Economic NMPC controller. Moreover, if the controller is unstable, we propose a method to stabilize it while obtaining optimal economic performance.

## 1.3 Thesis Outline

This dissertation is organized as follows:

Chapter 2 provides literature review of MPC and NMPC. The original work and variations of MPC and NMPC are discussed. The commonly used notations, basic formulation of process model and NMPC are also introduced.

Chapter 3 provides numerical methods we use to solve differential algebraic equations (DAEs) and NLPs. In particular, orthogonal collocation on finite elements is used to discretize the time-continuous DAE models and the interior point method is used to solve

NLPs. These methods are discussed in detail in this chapter. Then we discuss NLP sensitivity, which is a very important concept that is utilized in advanced-step NMPC (asNMPC), Chapter 4 and advanced-multi-step NMPC (amsNMPC), Chapter 5. IPOPT, an NLP solver that uses interior point method, and sIPOPT, an IPOPT version that utilizes NLP sensitivity, are also introduced.

Chapter 4 discusses fast NMPC strategies and Lyapunov stability analysis. To avoid delay that occurs due to non-negligible computational time, several fast NMPC methods are proposed by different research groups all over the world and three of them are discussed in this chapter. Then reformulation of NLPs using soft constraints is introduced. We also present the formulation and implementation of asNMPC, a fast NMPC method proposed by Zavala, a previous member in the Biegler group, which lays the foundation of the work in this dissertation. Then we introduce the concept of Lyapunov stability and analyze nominal and robust stabilities of ideal NMPC and asNMPC. When we discuss robust stability of ideal NMPC we study three toy examples to show that robustness could be obtained by using a longer NLP horizon or reformulating the NLP with soft constraints and $\ell_1$ penalty terms.

Chapter 5 presents the two approaches of advanced-multi-step NMPC, the serial approach and the parallel approach. amsNMPC avoids computational delay by solving NLPs in advance with predictions of future states as initial conditions. In order to transit smoothly from asNMPC to amsNMPC, we start with blocked amsNMPC, which is asNMPC but with longer sampling time. Stability of the blocked amsNMPC is discussed first, which will be used as a reference when stability of the two approaches are analyzed. Implementations and stability analysis of the two different approaches are discussed.

Chapter 6 compares the performance of ideal NMPC, asNNPC, the parallel approach and the serial approach. Two nonlinear cases, a continuous stirred tank reactor (CSTR) and

a large-scale propane-propylene distillation column (C3 splitter), are studied. Both the serial and the parallel approaches are applied to the CSTR example to show the pros and cons of amsNMPC. The C3 splitter, which has 79750 variables and 79700 constraints and whose NLP solution time exceeds one sampling time, is used to show the capability of the parallel approach on large-scale systems. Moreover, we also apply asNMPC/amsNMPC to the three toy examples studied in Chapter 4 and show that robustness is preserved when asNMPC or amsNMPC is applied.

Chapter 7 presents Economic NMPC. Economic NMPC has advantages over traditional two-layer RTO advanced control structure because it drives the controller directly with economic stage costs. However, a critical issue of Economic NMPC is that stability cannot be guaranteed if there are only economic costs in the objective. We propose a strategy to stabilize an Economic NMPC controller. Moreover, we also provide a method to obtain the balance between optimizing economic profit and maintaining the stability of the controlled system. Two case studies, a small CSTR and two distillation columns in sequence, which has 136154 variables and 77754 constraints, are used to demonstrate our stabilizing strategy.

Chapter 8 concludes the dissertation by stating the concluding remarks of each chapter. Then we discuss the contribution of our work and point out possible directions for future work.

# Chapter 2

# Literature Review

Model predictive control (MPC) is a widely used feedback control strategy. Compared with classical controllers, it has the advantage of handling variable bounds and dealing directly with multi-input-multi-output systems. Often used to track setpoints, it solves an optimization problem derived from dynamics of a real system. If the system is nonlinear, then a nonlinear programming (NLP) problem must be solved, leading to nonlinear model predictive control (NMPC) considered in our work.

## 2.1 Background

Before dynamic matrix control (DMC) was proposed in the 1970s ([11, 10]), the dominant advanced control method in industry was proportional-integral-derivative (PID) control. The PID controller attempts to minimize the error between the measurement of the process state and a setpoint. They are applicable to many control problems. However, they do not have information about the exact process model and usually do not provide "optimal" control and have difficulties in some situations, for example, when the process is strongly nonlinear, or there are large disturbances, etc. Also for a multi-input-multi-output system, decoupling is needed since PID controller works better for a single-input-single-output system. So the dominant advanced control method has become DMC which was proposed in 1979 by Cutler and Ramaker [11].

DMC is a model predictive controller. It generates control decisions by solving an optimization problem to minimize the deviation between the states and their setpoint within the shortest time or with the minimum control effort. It uses a linear finite impulse or step response model to represent the process and the optimization problem is solved by the computer. The calculation of DMC decisions does not require decoupling and inequality constraints are also considered when the optimization problem is solved. DMC has had a significant impact on process industry. Up till now, DMC is the dominant advanced process controller that is widely applied in oil industry and chemical industry. DMC is the first generation of industrial MPC technology. Qin and Badgwell [49] reviewed many MPC technologies, and features of products that used these technologies developed by various companies.

Model predictive control (MPC), on the other hand, is the generalization of DMC. Morari and Lee summarized the past MPC development, currently existing variations of MPC, and future directions in [42]. The MPC formulation in [42] is listed as the following:

$$\begin{aligned} J_N(x_0) := \min_{u(\cdot)} \quad & x^T(N)P_N x(N) + \sum_{i=0}^{N-1} x^T(i)Qx(i) + \sum_{i=0}^{m-1} u^T(i)Ru(i) \\ \text{s.t. } x(k+1) &= Ax(k)+Bu(k), \;\; k=0,\ldots,N-1 \\ x(0) &= x_0 \\ Ex(k)+Fu(k) &\leq \psi(k). \end{aligned} \tag{2.1}$$

where $N$ is the length of the prediction horizon and $m$ is the length of the control horizon. When $N=\infty$, this problem is referred to as an infinite horizon problem. For an infinite horizon problem, stability is more likely to be guaranteed; however, it is not practical to solve Problem (2.1) with $N=\infty$ numerically. Therefore the infinite horizon problem needs to be reformulated to a finite horizon problem. The common way is to set $m=N$ and use a terminal cost $x^T(N)P_N x(N)$ to replace the summation $\sum_{i=N}^{\infty} x^T(i)Qx(i)+u^T(i)Ru(i)$ where $P_N$ could be obtained by solving the Riccati equation.

More recently, MPC has been extended to Nonlinear Model Predictive Control (NMPC) in order to realize high-performance control of highly nonlinear processes. The objective function of NMPC problems has the same formulation as in (2.1). However, for NMPC, the weight of terminal cost cannot be obtained by solving the Riccati equation. Different approaches have been proposed to reformulate the infinite horizon problem to the finite horizon problem for NMPC. For example, Keerthi and Gilbert [35] proposed to add a terminal constraint or terminal region to force the state to the setpoint or terminal region. Michalska and Mayne [41] proposed to set the horizon length as a variable. Chen and Allgöwer [8, 9] proposed to replace the terminal cost with an upper bound of $\sum_{i=N}^{\infty} x^T(i)Qx(i)+u^T(i)Ru(i)$. More recently, Pannocchia et al. [47] showed that when there are no state constraints, terminal constraint could be replaced by an appropriate terminal penalty without loss of robustness. Another commonly used approach in industry to generate MPC decisions for nonlinear processes is to linearize the process model and solve an MPC problem. However when the process is strongly nonlinear this approach does not always lead to satisfactory results. For the rest of this chapter we introduce the general formulation of NMPC problems.

## 2.2 Nonlinear Programming (NLP) Formulation

The basic concepts and development of NMPC can be found in Rawlings and Mayne [53]. The current plant state and the plant model are used to predict future plant states. Based on these states, an NLP problem is solved to get the corresponding manipulated variables, among which the first is injected into the plant. Here we assume that the dynamics of the plant can be described by

$$x(k+1) = f(x(k), u(k)) \tag{2.2}$$

where $x(k) \in \Re^{n_x}$ is the plant state at time $t_k$ and $u(k) \in \Re^{n_u}$ is the manipulated variable at $t_k$. The mapping $f : \Re^{n_x+n_u} \mapsto \Re^{n_x}$ is continuous and $f(0,0) = 0$. In a realistic scenario the evolution of the system could be described by the following discrete-time nonlinear dynamic model:

$$\begin{aligned} x(k+1) &= \tilde{f}(x(k),u(k),w(k)) \\ &= f(x(k),u(k)) + g(x(k),u(k),w(k)) \end{aligned} \tag{2.3}$$

where $w(k) \in \mathscr{W} \subset \Re^{n_w}$ is disturbance, $g : \Re^{n_x+n_u+n_w} \mapsto \Re^{n_x}$ is continuous and used to describe modeling errors, estimation errors and disturbances. We use $z_l$ and $v_l$ to denote predicted values of $x(k+l)$ and $u(k+l)$ respectively. The NLP problem for NMPC at time $t_k$ is formulated as

$$\begin{aligned} J_N(x(k)) := \min_{z_l, v_l} \quad & \Psi(z_N) + \sum_{l=0}^{N-1} \psi(z_l, v_l) \\ \text{s.t. } z_{l+1} &= f(z_l, v_l) \quad l = 0, \ldots N-1 \\ z_0 &= x(k) \\ z_l &\in \mathbb{X}, v_l \in \mathbb{U}, z_N \in \mathbb{X}_f. \end{aligned} \tag{2.4}$$

We assume that the states $z_l$ and controls $v_l$ are restricted to the domains $\mathbb{X}$ and $\mathbb{U}$, respectively. $\mathbb{X}_f$ is the terminal set and $\mathbb{X}_f \subset \mathbb{X}$. We assume that the horizon length $N$ is sufficiently large such that $z_N \in \mathbb{X}_f$ is always true for the solution of (2.4). The set $\mathbb{U}$ is compact and contains the origin; the sets $\mathbb{X}$ and $\mathbb{X}_f$ are closed and contain the origin in their interiors.

The stage cost is given by $\psi(\cdot,\cdot) : \Re^{n_x+n_u} \to \Re$, while the terminal cost is denoted by $\Psi(\cdot) : \Re^{n_x+n_u} \to \Re$. For tracking problems, we can assume that the states and control variables can be defined with respect to setpoint and reference values, and that the nominal model has the property, $f(0,0) = 0$. The optimal solution $\{z_0^*, \ldots, z_N^*, v_0^*, \ldots, v_{N-1}^*\}$ leads to

the optimal objective function value $J_N(x(k))$, and $v_0^*$ is injected into the plant as $u(k)$. Once $x(k+1)$ is known, the horizon moves one step forward and the next NMPC problem (2.4) is solved for $u(k+1)$. This recursive strategy gives rise to the feedback law,

$$u(k) = \kappa(x(k)) \tag{2.5}$$

with $\kappa(\cdot) : \Re^{n_x} \mapsto \Re^{n_u}$. With the feedback law (2.5) system (2.3) becomes

$$\begin{aligned} x(k+1) &= \tilde{f}(x(k), \kappa(x(k)), w(k)) \\ &= f(x(k), \kappa(x(k))) + g(x(k), \kappa(x(k)), w(k)) \\ &= f(x(k), \kappa(x(k))) + g(x(k), w(k)) \end{aligned} \tag{2.6}$$

Hence, we replace $g(x(k), u(k), w(k))$ with $g(x(k), w(k))$ since $u(k) = \kappa(x(k))$. As stated above, the input to the NMPC controller is the current plant state, while its output is the manipulated variable to be injected into the plant which is achieved by solving an NLP problem. Ideally it is expected that the input is injected to the plant right after the measurement is obtained. We refer to this strategy as *ideal NMPC* (iNMPC), where the on-line calculation time is neglected. However, in reality the NLP solution always requires non-negligible on-line computational time, which leads to computational delay between obtaining the state and injecting the control. This delay could lead to deterioration of controller performance and system stability. To prevent computational delay, researchers have come up with a number of fast NMPC strategies. We will discuss several fast NMPC strategies in Chapter 4.

## 2.3 Concluding Remarks

In this chapter we start from the history of MPC and then move on to NMPC and discuss the reformulation of infinite horizon problems to finite horizon problems. We introduce

the commonly used notations and the basic formulation of NMPC. At each time $t_k$, NMPC generates control decisions by solving optimization Problem (2.4) repeatedly with the actual state $x(k)$ as initial condition. In the next chapter we discuss how Problem (2.4) is solved; we then discuss reformulation of Problem (2.4) in Chapter 4.

# Chapter 3

# Numerical Solutions of DAE and NLP

In this chapter, we discuss the numerical methods we use to solve process models composed of differential and algebraic equations (DAEs). We also discuss numerical methods to solve nonlinear programming (NLP) problems.

## 3.1 DAE-constrained Optimization

Based on how the process model is built, there are two kinds of models: data-driven models and first principle models. Data-driven models are built using on computational intelligence and machine learning methods based on outputs and inputs. First principle models are built based on the dynamic properties of the process. In this dissertation we study first principle models.

First principle models are usually composed of continuous-time DAEs. However, there are no easy ways to solve NLPs that contain DAEs directly. Therefore we need to discretize the continuous equations first. There are different approaches to handle the DAEs for dynamic optimization, such as single shooting, multiple shooting, and collocation method [6]. Typically collocation is performed by using orthogonal collocation on finite elements. It represents states and controls in each element with piecewise polynomials. The discretization utilizes sparsity and structure, avoids convergence difficulties that appear with the shooting methods, and sensitivity calculations for the DAEs are replaced by direct gra-

dient and Hessian evaluations within the NLP formulation. However, discretization of the DAE model increases size of the NLP problem, so efficient large-scale NLP solvers are required. In this dissertation orthogonal collocation is used to discretize the continuous-time process models.

We consider the following DAE:

$$\begin{aligned} \frac{dz}{dt} &= \hat{f}(z(t),y(t),u(t)),\ z(0)=z_0 \\ 0 &= \hat{g}(z(t),y(t),u(t)) \end{aligned} \tag{3.1}$$

where $z,y$ are the differential and algebraic variables of the system respectively. If (3.1) is index 1, we can represent $y(t)$ as $y^{imp}(t)$, an implicit function of $z(t)$ and $u(t)$, and we can rewrite (3.1) as

$$\begin{aligned} z(t_i) &= z(t_{i-1}) + \int_{t_{i-1}}^{t_i} \hat{f}(z(t),y^{imp}(t),u(t))dt \\ &= z(t_{i-1}) + h_i \int_0^1 \hat{f}(z(t_{i-1}+\tau h_i),y^{imp}(t_{i-1}+\tau h_i),u(t_{i-1}+\tau h_i))d\tau, \end{aligned} \tag{3.2}$$

where $h_i = t_i - t_{i-1}$ and $\tau \in [0,1]$. We would like to discretize System (3.1) to the following formulation:

$$z(k+1) = f(z(k),u(k)) \tag{3.3}$$

where we set $t_{i-1} = k$ and $t_i = k+1$. We use orthogonal collocation on finite elements to discretize system (3.1) to (3.3). We partition the time domain $[0,t_f]$ into $N$ stages where the domain inside each element $i$ is given by $t \in [t_{i-1},t_i]$ with $i = 1,\ldots,N, t_0 = 0$ and $t_{N+1} = t_f$. Using this representation we could reformulate System (3.1) as

$$\begin{aligned} \frac{dz_i}{dt} &= \hat{f}(z_i(t),y_i(t),u_i(t)),\ z(0)=z_0 \\ 0 &= \hat{g}(z_i(t),y_i(t),u_i(t)) \\ i &= 1,\ldots,N \end{aligned} \tag{3.4}$$

Then over each single finite element, we use a polynomial of order $K$ to approximate the continuous function $\hat{f}$ in the above integral (3.2), and therefore there are $K$ interpolation points in each element $i$. If the length of element $i$ is $h_i$, we can represent any time $t \in [t_{i-1}, t_i]$ as $t = t_{i-1} + h_i\tau, \tau \in [0,1]$ and the state in element $i$ is presented as

$$z_i^K(t) = \sum_{j=0}^{K} l_j(\tau) z_{ij} \tag{3.5}$$

where $l_j(\tau) = \prod_{k=0,\neq j}^{K} \frac{\tau-\tau_k}{\tau_j-\tau_k}, \tau_0 = 0, \tau_j \leq \tau_{j+1}, j = 0,\ldots,K$. Similarly we could represent the algebraic variable $y$ as

$$y_i^K(t) = \sum_{j=1}^{K} \bar{l}_j(\tau) y_{ij} \tag{3.6}$$

where $\bar{l}_j(\tau) = \prod_{k=1,\neq j}^{K} \frac{\tau-\tau_k}{\tau_j-\tau_k}$. $u_i(t)$ can be represented as in (3.6), or as a lower order polynomial. In this dissertation we use piece-wise linear control given as $u_i(t) = u_{i-1}$ which is constant within each finite element.

Since the state $z_i^K(t)$ is represented as a polynomial of order $K+1$, its time derivative could be represented as a polynomial of order $K$ with $K$ interpolation points. From (3.4)-(3.6) we have

$$\begin{aligned} z_i^K(t_i) &= z_i^K(t_{i-1}) + h_i \int_0^1 \hat{f}(z(\tau), y(\tau), u(\tau)) d\tau \\ &= z_i^K(t_{i-1}) + h_i \int_0^1 \sum_{j=1}^{K} \dot{z_{ij}} l_j(\tau) d\tau \\ &= z_i^K(t_{i-1}) + h_i \sum_{j=1}^{K} \dot{z_{ij}} \int_0^1 l_j(\tau) d\tau \\ &= z_i^K(t_{i-1}) + h_i \sum_{j=1}^{K} \Omega_j(1) \dot{z_{ij}} \end{aligned} \tag{3.7}$$

where $\Omega_j(\tau) = \int_0^\tau l_j(\tau')d\tau', \ \tau \in [0,1], \dot{z_{ij}} = \hat{f}(z_{ij}, y_{ij}, u_{i-1}, t_{ij})$. $z_{ij}$ could be represented in

a similar formulation. They are represented as

$$z_i^K(t) = z_i^K(t_{i-1}) + h_i \sum_{j=1}^{K} \Omega_j(\tau)\dot{z}_{ij} \tag{3.8a}$$

$$z_{ij} = z_i^K(t_{i-1}) + h_i \sum_{k=1}^{K} \Omega_k(\tau_j)\dot{z}_{ik} \tag{3.8b}$$

where $z_i^K(t_{i-1}) = z_{i,0}$.

Eventually we replace (3.1) with the following two equations:

$$\dot{z}_{ij} = \hat{f}(z_{ij}, y_{ij}, u_{i-1}, t_{ij}), j = 1, \ldots, K \tag{3.9a}$$

$$0 = \hat{g}(z_{ij}, y_{ij}, u_{i-1}, t_{ij}) \tag{3.9b}$$

$$z_{i,0} = z_{i-1,0} + h_i \sum_{j=1}^{K} \Omega_j(1)\dot{z}_{ij} \tag{3.9c}$$

Assuming that the DAE system is index 1, we can eliminate $z_{ij}, y_{ij}$ and $\dot{z}_{ij}$ implicitly using (3.8b), (3.9a),(3.9b) so that (3.9c) leads to the dynamic model (3.3). In this dissertation we use Radau collocation and 3 collocation points within each finite element.

## 3.2 IPOPT Algorithm

IPOPT(**I**nterior **P**oint **OPT**imizer) is a large-scale NLP solver that is designed to find local solution of NLPs. It uses the interior point method to solve NLPs. IPOPT is used for all the case studies in this dissertation.

### 3.2.1 Interior Point Method

To explore continuity and sensitivity properties of Problem (2.4), we represent this problem as:

$$min_{\mathbf{x}} F(\mathbf{x}, p),\ s.t.\ c(\mathbf{x}, p) = 0,\ x \geq 0 \tag{3.10}$$

where $\mathbf{x} \in \Re^n$ is the variable vector containing the states and controls, and $p$ is a *fixed* data vector used to represent uncertain external parameters such as disturbances. The equality constraints are $c(\mathbf{x}, p) : \Re^n \to \Re^m$. In interior-point solvers, the inequality constraints of problem (3.10) are handled *implicitly* by adding barrier terms to the objective function,

$$\begin{aligned} min_{\mathbf{x}} \quad & F(\mathbf{x}, p) - \mu \sum_{j=1}^{n_{\mathbf{x}}} \ln(\mathbf{x}^{(j)}), \\ s.t. \quad & \mathbf{c}(\mathbf{x}, p) = 0 \end{aligned} \tag{3.11}$$

where $\mathbf{x}^{(j)}$ denotes the $j^{th}$ component of vector $\mathbf{x}$. Solving (3.11) for the sequence of $\mu^l \to 0$, with $l = 0, 1, 2, \ldots, \infty$ leads to solution of the original NLP (3.10).

We define the primal-dual Lagrange function $L(\mathbf{x}, \lambda, \nu, p) = F(\mathbf{x}, p) + c(\mathbf{x}, p)^T \lambda - \mathbf{x}^T \nu$ and the solution vector $s = [\mathbf{x}^T, \lambda^T, \nu^T]^T$, where $\lambda, \nu$ are multipliers of equality constraints and bound constraints respectively, $\nu = \lim_{\mu \to 0} \mu X^{-1} e$. For a given barrier parameter value $\mu$, IPOPT [55] solves the primal-dual optimality conditions of barrier problems (3.11) directly,

$$\nabla_{\mathbf{x}} L(\mathbf{x}, \lambda, \nu, p) = 0, \tag{3.12a}$$

$$\mathbf{c}(\mathbf{x}, p) = 0, \tag{3.12b}$$

$$\mathbf{X}\mathbf{V}e = \mu e, \tag{3.12c}$$

where $\mathbf{X} = \text{diag}(\mathbf{x}), \mathbf{V} = \text{diag}(\nu)$, $e \in \Re^n$ is a vector of ones, and $\lambda \in \Re^m$ and $\nu \in \Re^n$ are Lagrange multipliers. To solve this system of nonlinear equations, IPOPT uses an exact Newton method and starts the iteration sequence at point $s_o^T := [\mathbf{x}_o^T \, \lambda_o^T \, \nu_o^T]$. At the $i$th Newton iteration, the search direction $\Delta s_i = s_{i+1} - s_i$ is computed by linearization of the KKT conditions (3.12),

$$\begin{bmatrix} \nabla_{\mathbf{xx}} L(s_i(p)) & \nabla_{\mathbf{x}}(s_i(p)) & -I \\ \nabla_{\mathbf{x}}(s_i(p))^T & 0 & 0 \\ \mathbf{V}_i & 0 & \mathbf{X}_i \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x}_i \\ \Delta \lambda_i \\ \Delta \nu_i \end{bmatrix} = - \begin{bmatrix} \nabla_{\mathbf{x}} L(s_i, p) \\ \mathbf{c}(\mathbf{x}_i, p) \\ \mathbf{X}_i \mathbf{V}_i e - \mu e \end{bmatrix} \tag{3.13}$$

where $\mathbf{X} = diag(\mathbf{x}), \mathbf{V} = diag(v)$. After solving a sequence of barrier problems for $\mu \to 0$, the solver returns the solution triple $s^{*,T}(\eta) = [\mathbf{x}^{*,T}\ \lambda^{*,T}\ v^{*,T}]$ for Problem (3.10).

Having introduced Lagrange function and KKT conditions, we introduce the concepts of SC, LICQ, and SSOSC. We first reformulate Problem (3.10) as:

$$\begin{aligned} min_{\mathbf{x}} \quad & F(\mathbf{x}, p) \\ s.t. \quad & \mathbf{c}(\mathbf{x}, p) = 0 \\ & \mathbf{g}(\mathbf{x}, p) \leq 0 \end{aligned} \tag{3.14}$$

**Definition 1** (Strict Complementarity[44])**.** *Given a vector p, a local solution* $\mathbf{x}^*$ *of* (3.14) *and vectors* $(\lambda, v)$*, we say that the strict complementarity condition (SC) holds for* $\lambda, v$ *only if* $v_j - g_j(\mathbf{x}^*, p) > 0$ *for each* $j = 1, \ldots, n_g$*.*

A constraint qualification is required for a local minimizer of (3.14) to be a KKT point [44].

**Definition 2** (Linear Independence Constraint Qualification)**.** *Given a vector p and a point* $\mathbf{x}^*$*, the linear independence constraint qualification (LICQ) holds at* $\mathbf{x}^*$ *if the gradient vectors*

$$\begin{aligned} &\nabla c_i(\mathbf{x}^*, p); \quad i = 1, \ldots, n_c, \\ &\nabla g_j(\mathbf{x}^*, p); \quad j \in J \text{ where } J = \{j | g_j(\mathbf{x}^*, p) = 0\} \end{aligned} \tag{3.15}$$

*are linearly independent.*

The LICQ implies that the multipliers $\lambda, v$ are unique. Sufficient conditions for $\mathbf{x}^*$ to be a minimum are given by the following second order conditions:

**Definition 3** (Second Order Sufficient Conditions)**.** *[6] Suppose that* $\mathbf{x}^*$ *and the multipliers* $\lambda^*$*,* $v^*$ *satisfy the KKT conditions (3.12) and*

$$q^T \nabla_{\mathbf{xx}} L(\mathbf{x}^*, \lambda, v, p) q > 0 \quad \text{for all } q \neq 0 \tag{3.16}$$

*such that*

$$\begin{aligned} \nabla_{\mathbf{x}} c_i(\mathbf{x}^*, p)^T q = 0, \quad & i = 1, .., n_c \\ \nabla_{\mathbf{x}} g_j(\mathbf{x}^*, p)^T q = 0, \quad & for\ \nu_j > 0, \\ \nabla_{\mathbf{x}} g_j(\mathbf{x}^*, p)^T q \leq 0, \quad & for\ \nu_j = 0 \end{aligned} \tag{3.17}$$

*then $x^*$ is a strict local solution of (3.14).*

**Definition 4** (Strong Second Order Sufficient Conditions [31])**.** *The strong second order sufficient conditions (SSOSC) hold at $\mathbf{x}^*$ with multipliers $\lambda$, and $\nu$ if*

$$q^T \nabla_{\mathbf{xx}} L(\mathbf{x}^*, \lambda, \nu, p) q > 0 \quad for\ all\ q \neq 0 \tag{3.18}$$

*such that*

$$\begin{aligned} \nabla_{\mathbf{x}} c_i(\mathbf{x}^*, p)^T q = 0, \quad & i = 1, .., n_c \\ \nabla_{\mathbf{x}} g_j(\mathbf{x}^*, p)^T q = 0, \quad & \nu_j > 0. \end{aligned} \tag{3.19}$$

**Theorem 1** (Implicit function theorem applied to optimality conditions)**.** *Let $\mathbf{x}^*(p)$ be a KKT point that satisfies* (3.12)*, and assume that SC, LICQ and SSOSC hold at $\mathbf{x}^*$. Further let the functions $F, c, g$ be at least $k+1$ times differentiable in $\mathbf{x}$ and $k$ times differentiable in $p$. Then*

- *$\mathbf{x}^*$ is an isolated minimizer, and the associated multipliers $\lambda$ and $\eta$ are unique.*
- *for $p$ in a neighborhood of $p_0$ the set of active constraints remains unchanged,*
- *for $p$ in a neighborhood of $p_0$ there exists a $k$ times differentiable function $s(p) = \left[\mathbf{x}^*(p)^T,\ \lambda(p)^T,\ \nu(p)^T\right]$, that corresponds to a locally unique minimum for* (3.10)*.*

*Proof.* See Fiacco [18]. □

### 3.2.2 NLP Sensitivity

In the solution of an NLP, solving (3.13) is the most computationally demanding step during each iteration. However, the KKT matrix is always very sparse and structured. Efficient sparse linear solvers are applied to factorize the KKT matrix, e.g., HSL library with the METIS option [16]. Moreover, the structure of the KKT matrix does not change between iterations, so the linear solver needs to analyze the sparsity pattern only once.

Having the optimal solution $s^*(p_0)$ (superscript '*' indicates optimal solution), the KKT conditions of (3.11) are listed as the following equations:

$$\Phi(s^*(p_0)) = \begin{bmatrix} \nabla_{\mathbf{x}} L(s^*(p_0)) \\ \mathbf{c}(\mathbf{x}^*(p_0)) \\ X^* v - \mu e \end{bmatrix} = 0 \tag{3.20}$$

where $X^* = diag(\mathbf{x}^*)$, $e = [1,1,...,1]^T$. Expanding the KKT conditions at the optimal solution with parameter $p \neq p_0$ leads to the following:

$$0 = \nabla_s \Phi(s^*(p)) = \nabla_s \Phi(s^*(p_0)) + \frac{d}{dp}(\nabla_s \Phi(s^*(p_0)))\Delta p + O(|\Delta p|^2) \tag{3.21}$$

where $\nabla_s \Phi(s^*(p_0)) = 0$ and $\Delta p = p - p_0$. Consequently,

$$\frac{d}{dp}(\nabla_s \Phi(s^*(p_0)))\Delta p = (M\frac{ds^{*,T}}{dp} + N_p)\Delta p = 0 \tag{3.22}$$

where $M = \begin{bmatrix} \nabla_{\mathbf{xx}} L(s^*(p_0)) & \nabla_{\mathbf{x}} c(\mathbf{x}^*(p_0)) & -I \\ \nabla_{\mathbf{x}} c(\mathbf{x}^*(p_0))^T & 0 & 0 \\ V(p_0) & 0 & X^*(p_0) \end{bmatrix}$ is the KKT matrix, $V = diag(v)$, and $N_p = \nabla_p \Phi = \begin{bmatrix} \nabla_{\mathbf{x}p} L(s^*(p_0)) \\ \nabla_p c(\mathbf{x}^*(p_0))^T \\ 0 \end{bmatrix}$. From the right hand side of (3.22) with $\Delta s = s^*(p) -$

$s^*(p_0) = \frac{ds^{*,T}}{dp}\Delta p + O(|\Delta p|^2)$ and $N = \nabla_s \Phi(s^*(p)) - \nabla_s \Phi(s^*(p_0)) = N_p \Delta p + O(|\Delta\, p|^2)$, we have:

$$M\Delta s(p) \approx -N. \tag{3.23}$$

Thus, for every perturbation $\Delta p$, the solution of the perturbed problem can be approximated by (3.23). If we use $\tilde{s}(p)$ to denote the estimate, then from (3.23) we have

$$\tilde{s}(p) = s^*(p_0) + \frac{ds^*}{dp}(p - p_0) \tag{3.24}$$

And applying Taylor expansion to $s^*(p)$ around $p_0$ we have

$$s^*(p) = s^*(p_0) + \frac{ds^*}{dp}(p - p_0) + O(|p - p_0|^2) \tag{3.25}$$

Subtracting (3.24) from (3.25), we have

$$|s^*(p) - \tilde{s}(p)| = O(|p - p_0|^2) \tag{3.26}$$

From continuity and differentiability of the optimal solution vector $s$, there exits a positive Lipschitz constant $L_q$ such that

$$|s^*(p) - \tilde{s}(p)| \leq L_q |p - p_0|^2. \tag{3.27}$$

**Definition 5** (Mangasarian-Fromovitz Constraint Qualification)**.** *For Problem (3.10), the Mangasarian-Fromovitz constraint qualification (MFCQ) holds at the optimal point* $\mathbf{x}^*(p)$ *if and only if*

*a) the vectors* $\nabla_{\mathbf{x}} c_i(\mathbf{x}^*, p)$ *are linearly independent for all* $i = 1, \ldots, n_c$

*b) there exists a vector w such that*

$$\begin{aligned} \nabla_{\mathbf{x}} c_i(\mathbf{x}^*, p)^T w = 0 \quad & \textit{for } i = 1, \ldots, n_c \\ \nabla_{\mathbf{x}} g_j(\mathbf{x}^*, p)^T w < 0 \quad & \textit{for } g_j(\mathbf{x}^*, p) = 0. \end{aligned} \tag{3.28}$$

The MFCQ implies that the set of KKT multipliers is a closed convex polytope [23]. If we reformulate $\mathbf{g}(\mathbf{x}, p) = 0$ as soft constraints by using slack variables: $\mathbf{g}(\mathbf{x}, p) \leq \varepsilon$ and adding $\ell_1$ penalty to the objective function, then the reformulated NLP satisfies MFCQ.

Another constraint qualification we need is the constant rank constraint qualification.

**Definition 6** (CRCQ [30])**.** *For Problem (3.10), the constant rank constraint qualification holds at* $(\mathbf{x}^*, p_0)$*, if for any subset* $S \subset J, J = \{j | g_j(\mathbf{x}^*, p) = 0\}$ *of active constraints the family*

$$\left\{\nabla_{\mathbf{x}} g_j(\mathbf{x}, p)\ j \in S, \quad \nabla_{\mathbf{x}} c_i(\mathbf{x}, p)\ i = 1, .., n_c\right\} \tag{3.29}$$

*remains of constant rank near the point* $(\mathbf{x}^*, p_0)$*.*

Note that the CRCQ is neither stronger nor weaker than MFCQ in the sense that one implies the other [30]. For the formulated (3.14), the inequalities are linear, e.g. constraints on $\mathbf{x}$ are simple bounds, and $g_j(\mathbf{x}, p) \leq \varepsilon$ is linear in $\mathbf{x}$, then CRCQ holds. If MFCQ and CRCQ hold, then the objective function is Lipschitz continuous in $p$. Moreover, if we use barrier terms to eliminate the inequality constraints on $\mathbf{x}$, then LICQ, MFCQ and CRCQ hold, and in this case the objective function is differentiable in $p$.

**Definition 7** (GSSOSC)**.** *The general strong second order sufficient condition (GSSOSC) is said to hold at* $\mathbf{x}^*$ *if the SSOSC holds for all multipliers* $\lambda, \nu$ *in the set of KKT multipliers.*

It has been shown by Kojima [36] that the conditions KKT-point, MFCQ, and GSSOSC are the weakest ones under which the perturbed solution of the (3.14) is locally unique. Under these general conditions we cannot expect the solution $\mathbf{x}^*(p)$ to be differentiable any longer (because of active set changes). However, it can be shown that the solution $\mathbf{x}^*(p)$ is directionally differentiable, and for obtaining sensitivity updates in an NMPC context, directional differentiability is sufficient.

**sIPOPT**

As mentioned previously in this chapter, the KKT matrix is already factorized and available when the NLP problem is solved. When one or more parameters in the NLP formulation change, this matrix could be directly used to generate fast approximations to optimal solutions. In order to take advantage of NLP sensitivity more efficiently, Pirnay et.al. developed an NLP sensitivity extension for IPOPT named *sIPOPT* in [48]. The parameter $p$ is defined as a variable in the AMPL interface, although its old and new values are determined separately. The optimal solution is calculated with the old values of the parameters, and an update is done when the new values of parameters are determined, thus leading to an approximation of the optimal solution at the new value. sIPOPT is applied later in our case studies.

## 3.3 Concluding Remarks

In this chapter we introduce numerical methods to discretize DAE systems and solve NLP problems. We use orthogonal collocation on finite elements to discretize DAEs and use interior point method to solve NLP problems. These two methods are discussed in detail in this chapter. IPOPT, an NLP solver that solves NLP using the interior point method, is introduced here. We also introduce the concept and calculation of NLP sensitivity. NLP sensitivity will be used in Chapters 4 and 5 for control update. sIPOPT, an IPOPT version that provides fast suboptimal solution based on NLP sensitivity, is also discussed. In this dissertation all the NLPs are solved with IPOPT or sIPOPT. We also introduce some commonly used notations and concepts for Lyapunov stability analysis.

# Chapter 4

# Fast NMPC Strategies

As we mentioned in Chapter 2, solving the NLP (2.4) takes a non-negligible amount of time, especially when the scale of the system is large. As a result computational delay occurs, and may deteriorate controller performance and system stability. To avoid computational delay has been an interesting topic studied by people from different research groups all over the world. In this chapter, we first briefly introduce some fast MPC/NMPC methods, and then focus on advanced-step NMPC (asNMPC), proposed by Zavala in [60]. asNMPC lays the foundation of the fast NMPC method proposed in this dissertation. Then we talk about Lyapunov stability analysis of NMPC and prove stability of both ideal NMPC and asNMPC.

## 4.1 Background

The past decade has seen the development of modifications to (N)MPC that address computational delay. First, to solve the (NP-hard) quadratic programming (QP) problem for linear MPC, a partial enumeration of active constraints was proposed for the QP calculation [46]. Here the most frequent active sets are stored in a list, and searched whenever the actual state is received. If the corresponding active set is present, the optimal solution is obtained directly. Otherwise a suboptimal MPC problem is determined quickly, while the full QP problem is solved in background with a solution that updates the list. For nonlinear

MPC, Findeisen and Allgöwer [19] extended the NMPC formulation as well as its stability properties to account for computational delay. In addition, a number of fast NMPC strategies have been developed including [38, 1, 14, 45, 43, 21, 5] . As early as in 1989, Li and Biegler [38] proposed the Newton-type strategy for constrained nonlinear processes, where the nonlinear model is linearized around a nominal trajectory, and a QP is solved every sampling time to get deviation from set point. Alamir [1] developed a strategy that updates the sampling time in nonlinear model-predictive control online and adjusts the number of iterations in the optimization problem. In addition, recent NMPC strategies have been developed that separate the NMPC strategy into a) an off-line NLP step, using a predicted state, and b) a fast on-line calculation for the actual state. Based on this concept, Diehl et al. proposed a suboptimal NMPC strategy [15], where an NLP problem is solved with undetermined initial condition and parameter, and manipulated variables are updated with SQP upon achievement of actual initial condition and parameter value. Then a real-time iteration NMPC was proposed by Diehl et al. [14] where only one Newton or QP iteration of the NLP is executed on-line at every sampling time, instead of solving the NLP completely. In contrast, a neighboring extremal update (NEU) approach was proposed in [57], where an optimal control problem is solved over all sampling intervals. Then, during each sampling time a fast update, determined by a QP, is performed for the manipulated variable. Feasibility and optimality criteria are then checked to estimate the performance of each update. If they are not satisfied, additional QP iterations are triggered. Wolf et al. [56] extended NEU by adding an a-priori estimate of deviation between the NEU and optimal solution and using this estimate to decide whether additional QP iterations will improve the controller performance; the optimal number of QP iterations is also determined on-line.

In addition to QP iterations, *NLP sensitivity* is also used to update optimal solutions of NMPC. Ganesh and Biegler developed an optimal sensitivity analysis for NLPs in process flowsheets [22]. Kadam and Marquardt applied this sensitivity on dynamic real time

optimization to update reference trajectories [33, 34]. Related to these approaches is the *advanced-step NMPC* (asNMPC) strategy [61, 60], where an NLP problem is solved off-line one step in advance with prediction of the next state as initial condition, and *NLP sensitivity* is used to update manipulated variable on-line when the real-time measurement of the state is achieved. Since the update is a single backsolve for a linear system, it requires negligible computation time. Hence, the correct manipulated variable is available almost immediately after the state is estimated. The asNMPC strategy also enjoys nominal and robust stability properties [60]. We will discuss this in more detail in Section 4.4.4.

## 4.2 Ideal NMPC

### 4.2.1 NMPC Problem Reformulation

One advantage of NMPC is its ability to deal with constrained problems directly. However, due to disturbances, some states might violate their bounds, thus leading to infeasibility of the problem. Even if the problem is feasible, dependent active sets could also make the system unstable under perturbations. Sometimes the instability could be avoided by increasing predictive horizon length, sometimes not. In order to tackle this situation, Oliveira and Biegler considered "soft constraints" and exact penalty terms in [12]. When violation of some constraints could be tolerated, we can impose soft constraints, such as for constraints on outputs, while some constraints cannot be violated, which are called hard constraints, such as for constraints on inputs and safety constraints. In the soft constraint strategy, constraint violations are penalized in the objective using exact ($\ell_1$) penalty functions.

We define $\mathbb{X}$ and $\mathbb{X}_f$ in Problem (2.4) by the inequalities $h(z_l) \leq 0$. These regions are

closed and contain the origin in their interior. To develop a robust problem formulation we replace $\mathbb{X}$ and $\mathbb{X}_f$ by $\ell_1$ penalty terms and assume, without loss of generality, that $\mathbb{U}$ can be represented by simple upper and lower bounds on $v_l$. We add slack variables to the constraints $h(z_l) \leq 0$, and add $\ell_1$ penalty functions to the objective. We also assume that $N$ is sufficiently long such that $z_N \in \mathbb{X}_f$ can always be satisfied. In our case studies, we have used a sufficiently long optimization horizon such that this condition always holds. This horizon length is determined through repeated off-line simulations. As a result, the terminal constraints can be reformulated as terminal cost with penalties in our problem formulation and state constraints appear as penalty terms in the stage costs. Analogously, robust stability of this formulation is also shown in [47] with sufficiently large terminal cost.

To maintain differential objective and constraint functions, we reformulate Problem (2.4) to take the following form:

$$\begin{aligned} \min_{z_l, v_l, \varepsilon_l} \quad & \Psi(z_N) + \rho \varepsilon_N^T \mathbf{1} + \sum_{l=0}^{N-1} (\psi(z_l, v_l) + \rho \varepsilon_l^T \mathbf{1}) \\ \text{s.t.} \quad & z_{l+1} = f(z_l, v_l), \quad l = 0, ..., N-1 \\ & z_0 = x(k) \\ & h(z_l) \leq \varepsilon_l, \varepsilon_l \geq 0, \quad l = 0, ..., N \\ & v_l \in \mathbb{U}, \quad l = 0, ..., N-1. \end{aligned} \tag{4.1}$$

where $\varepsilon_l$ is a slack variable and $\mathbf{1} = [1, 1, ..., 1]^T$. If a solution to Problem (4.1) exists, then it is easy to see that the gradients of the equality constraints contain a nonsingular basis matrix, and are therefore linearly independent. Moreover, at the solution $\mathbf{x}^*$ of (4.1) where $\mathbf{x}^T = [{z_0}^T \ldots, {z_N}^T, {v_0}^T, \ldots, {v_{N-1}}^T, {\varepsilon_1}^T \ldots, {\varepsilon_N}^T]$ and with active equality and inequality constraints represented by $c(\mathbf{x}) = 0$, $c_I(\mathbf{x}) \leq 0$, it is straightforward to find a vector

$d^T = [d_{z_0}^T, \ldots, d_{z_N}^T, d_{v_0}^T, \ldots, d_{v_{N-1}}^T, d_{\varepsilon_1}^T \ldots, d_{\varepsilon_N}^T]$ that satisfies

$$\nabla c(\mathbf{x}^*)^T d = 0, \nabla c_I(\mathbf{x}^*)^T d < 0.$$

Consequently, the Mangasarian-Fromovitz constraint qualification (MFCQ, see Definition 5) always holds for Problem (4.1), as shown in [31]. If generalized second order sufficient conditions (GSOSC) are also satisfied, then selecting $\rho$ larger than a finite threshold, $\rho > \bar{\rho}$, will drive $\varepsilon_l$ to zero, where $\bar{\rho}$ is the dual norm of the multipliers at the solution of Problem (2.4). If $\varepsilon_l = 0$, then the solution of (4.1) is identical to the solution of Problem (2.4). Therefore, stability properties of the soft constrained problem (4.1) are identical to the hard constrained problem (2.4). Since a solution with $\varepsilon_l > 0$ for arbitrarily large values of $\rho$ implies that Problem (2.4) is locally infeasible, we assume that a finite $\bar{\rho}$ can be found as long as Problem (2.4) is well-posed. This corresponds to the common assumption that there exists a feasible input sequence, which steers the system to the terminal set. Among other considerations, this requires that the horizon $N$ be long enough to satisfy the terminal conditions.

**Definition 8.** *A set $\Gamma \subset \mathbb{X}$ is a robust positively invariant (RPI) set for system (2.3) if $f(x,u,w) \in \Gamma, \forall x \in \Gamma$ and $\forall w \in \mathscr{W}$.*

With the reformulation, $\Re^{n_x}$ is always a robust positively invariant (RPI) set. Moreover, if we apply barrier terms to the inequalities in (4.1), we obtain an arbitrarily close approxi-

mation to the solution of (4.1) by setting a sufficiently small positive value of $\mu$, as follows:

$$\begin{aligned}
\min_{z_l, v_l, \varepsilon_l} \quad & (\Psi(z_N) + \rho \varepsilon_N^T \mathbf{1} - \mu \sum_j ln(\varepsilon_N^{(j)} - h^{(j)}(z_N))) \\
& + \sum_{l=0}^{N-1} (\psi(z_l, v_l) + \rho \varepsilon_l^T \mathbf{1} - \mu \sum_j ln(\varepsilon_l^{(j)} - h^{(j)}(z_l))) \qquad (4.2) \\
\text{s.t.} \quad & z_{l+1} = f(z_l, v_l), \quad z_0 = x(k) \\
& \varepsilon_l \geq 0; v_l \in \mathbb{U}, \quad l = 0, ..., N-1, \\
& \varepsilon_N \geq 0.
\end{aligned}$$

Through the redefinitions: $v_l := [v_l^T, \varepsilon_l^T]^T$, $\psi(z_l, v_l) := \psi(z_l, v_l) + \rho \varepsilon_l^T \mathbf{1} - \mu \sum_j ln(\varepsilon_l^{(j)} - h^{(j)}(z_l))$ and $\Psi(z_N) := \Psi(z_N) + \rho \varepsilon_N^T \mathbf{1} - \mu \sum_j ln(\varepsilon_N^{(j)} - h^{(j)}(z_N))$, Problem (4.2) can be simplified to

$$\begin{aligned}
J_N(x(k)) = \min_{v_l, z_l} \quad & \Psi(z_N) + \sum_{l=0}^{N-1} \psi(z_l, v_l) \qquad (4.3) \\
\text{s. t.} \quad & z_{l+1} = f(z_l, v_l),\ z_0 = x(k),\ v_l \in \mathbb{U} \quad l = 0, \ldots N-1.
\end{aligned}$$

and this replaces Problem (2.4). In the subsequent development we refer to Problem (4.3) as $\mathscr{P}_N(x(k))$. Since the equality constraint gradients are linearly independent and the active bounds for $v_l$ and $\varepsilon_N$ are independent, it is clear that that the solution of $\mathscr{P}_N(x(k))$ satisfies LICQ. Along with the presence of bounds as the only inequalities, this property implies the weaker Constant Rank and Mangasarian-Fromovitz Constraint Qualifications (CRCQ, Definition 6, and MFCQ, Definition 5), and leads to unique, bounded multipliers. Moreover, if the solutions of (4.3) also satisfy General Strong Second Order Sufficient Conditions (GSSOSC, Definition 7) then the solution vector, $u(k) = \kappa(x(k))$ and the optimal objective function are continuous in $x(k)$.

In the development and analysis of the robust controllers in Section 4.4.2 and beyond, we will assume slightly stronger assumptions: LICQ, SOSC and *strict complementarity* at the solutions of $\mathscr{P}_N(x(k))$. These conditions guarantee the *differentiability* of the solution

vector $x^*(p)$ as well. Based on our experience, these stronger assumptions hold as long as the active set is unique at the solution with nonzero multipliers for the control bounds. Therefore, this leads to continuity and differentiability of the solution of $\mathscr{P}_N(x(k))$ with respect to disturbances.

## 4.3 Advanced-step NMPC

In this chapter we introduce the advanced-step NMPC (asNMPC) strategy. asNMPC is proposed by Zavala in [60]. The idea of asNMPC is to use the prediction of the future state to solve the NLP problem within the current sampling time. Once the actual state is obtained (or estimated), *NLP sensitivity* is used to update the manipulated variable online. Since the update is only a single backsolve, it requires negligible computation time. Hence, the manipulated variable is available right after the actual state is obtained and computational delay is avoided.

### 4.3.1 Formulation and Implementation

Given $\bar{x}(k)$ to denote the prediction of state at $t_k$, at time $t_{k-1}$, the NLP solved by asNMPC has the following formulation:

$$\begin{aligned} J_N(\bar{x}(k)) := \quad & \min_{v_l, z_l} \quad \Psi(z_N) + \sum_{l=0}^{N-1} \psi(z_l, v_l) \\ \text{s. t.} \quad & z_{l+1} = f(z_l, v_l),\ z_0 = \bar{x}(k)\ ; v_l \in \mathbb{U} \quad l = 0, \ldots N-1. \end{aligned} \tag{4.4}$$

Note that the only difference from $\mathscr{P}_N(x(k))$(4.3) is that the initial condition $x(k)$ is replaced by $\bar{x}(k)$.

In the NLP formulation of asNMPC, the state prediction $\bar{x} = p_0$ in (3.10) and the actual state $x$ is the perturbed parameter $p$. Applying NLP sensitivity, asNMPC is implemented

as follows:

- *Online*: at $t_k$, having $x(k)$, update $v_0$ using $(s^*(p_0)+\Delta s(p))$ from (3.23), where $p_0 = \bar{x}(k)$, $p = x(k)$. Implement the updated $v_0$ as $u(k)$ to the plant. Predict $x(k+1)$ using $x(k)$ and $u(k)$: $\bar{x}(k+1) = f(x(k),u(k))$.
- *Background*: move the horizon one step forward, take $\bar{x}(k+1)$ as initial condition $(z_0 = p_0)$ and solve NLP problem $\mathscr{P}_N(\bar{x}(k+1))$ (4.4).
- Set $k = k+1$ and repeat.

The above steps and positions of $x$, $u$, $\bar{x}$ and $v$ are shown in Fig. 4.1, where optimal solutions are shown in dashed line, with optimal control shown in light blue and optimal state shown in red; actual state trajectory is shown in dark blue solid line and actual control is shown in red solid line. The state prediction is shown in point-dashed line.

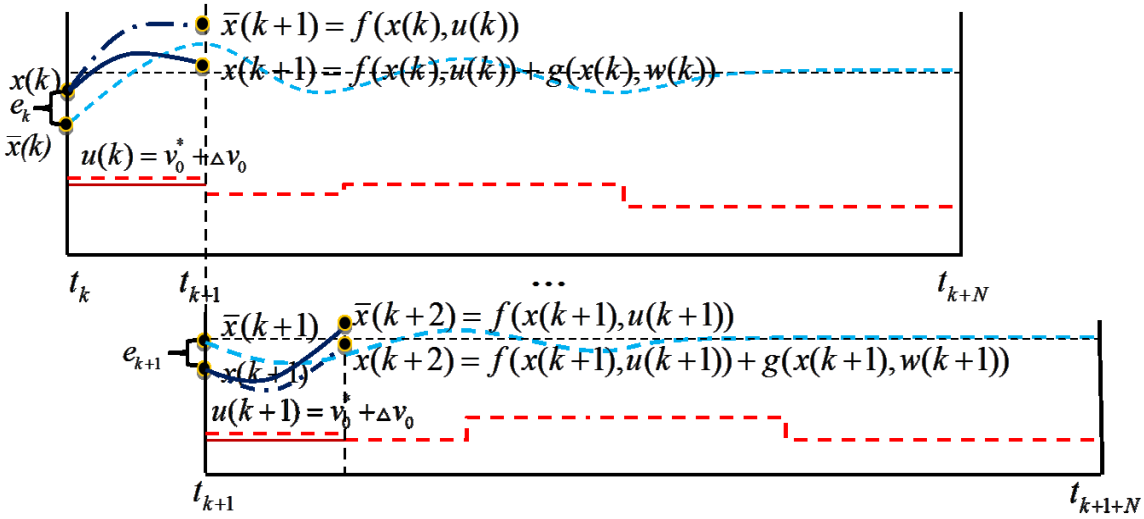



Figure 4.1: Implementation of asNMPC

To compute approximate neighboring solutions around an already available nominal solution $s^*(p_0)$, we invoke the following classical results.

**Theorem 2.** *(NLP sensitivity [18])If $f(\cdot,\cdot)$, $\psi(\cdot,\cdot)$ and $\Psi(\cdot)$ of the mixed constrained problem are twice continuously differentiable in a neighborhood of the nominal solution $s^*(p_0)$ and this solution satisfies the linear independence constraint qualifications (LICQ) and sufficient second-order conditions (SSOC) then,*

1. *there exists a positive Lipschitz constant $\alpha$ such that $|s^*(p) - s^*(p_0)| \leq \alpha|p - p_0|$;*
2. *there exists a positive Lipschitz constant $L_z$ such that the optimal cost values $J_N(p)$ and $J_N(p_0)$ satisfy $|J_N(p) - J_N(p_0)| \leq L_z|p - p_0|$.*
3. *if $\tilde{s}(p)$ is an approximate solution of $s^*(p)$ based on sensitivity, from continuity and differentiability of the optimal solution vector, there exists a positive Lipschitz constant $L_s$ such that, $|\tilde{s}(p) - s^*(p)| \leq L_s|p - p_0|^2$.*

Thus, for every perturbation $\Delta p$, the solution of the neighboring problem can be approximated by (3.23). Note that the KKT matrix $M$ is directly available and already factorized after the NLP problem is solved. Since in (3.23) only a backsolve is done, the update takes much less time than solving the NLP problem to get new solutions.

### 4.3.2 Active Set Changes

When the perturbation $p - p_0$ is large enough to induce a change in the active constraint set at the solution of (3.10), tracking the KKT conditions becomes nonsmooth, the linearization (3.23) is no longer valid and the approximate solution $\tilde{s}(p)$ may not even be feasible to $\mathbb{X}$ and $\mathbb{U}$. Since Theorem 2 no longer holds at the point where the transition in active sets occurs, $s^*(p)$ is no longer differentiable (and may not even be continuous) with respect to $p$. However, under Generalized Second Order Sufficient Conditions (GSOSC) and a relaxed set of constraint qualifications (CRCQ and MFCQ) (satisfied by $\mathscr{P}_N(x)$), one can

still obtain continuity of $s^*(p)$ and the value function with respect to $p$ as well as the *directional derivative* of $\mathbf{x}^*(p)$ for $p - p_0$.

As developed in [50], this directional derivative is determined by solving a specialized QP that can be viewed as an extension of (3.23), with linearized inequality constraints included and strongly active inequalities (with positive multipliers) expressed as equality constraints. Because multipliers at the active set transition are nonunique, a linear program (LP) in dual space must first be solved to maximize the linearized prediction of the Lagrange function. In [31] we apply this formulation to develop a path-following algorithm to track $\mathbf{x}^*(p)$ with respect to $p$. Detailed presentation of this path-following approach and general underlying concepts that govern NLP sensitivity are also included in [31]. While this approach leads to a rigorous treatment of NLP sensitivity, it is more expensive than the simple update (3.23).

Instead of path-following approach, we can also apply a cheaper strategy called "clipping in the first interval," where we perturb the solution only up to the active set change so that Theorem 1 still holds. Here, feasibility of the soft constrained formulation (4.3) only requires $v_l \in \mathbb{U}$, and clipping ensures that the perturbed control variable value, $u(k)$ remains within its bounds:

$$v_0^L \leq v_0 + \tau \Delta v_0 \leq v_0^U, \tau \in [0,1] \tag{4.5}$$

Once $\tau$ and the updated variables are determined, the manipulated variable $u(\tau) = v_0 + \tau \Delta v_0$ is implemented to the plant. Because the clipping strategy requires no additional computational cost beyond (3.23), we incorporate this approach within the asNMPC and amsNMPC and use this approach for the case studies. Moreover, in [31] a comparison of the simple clipping approach with the path-following approach shows very good performance in the presence of active set changes.

## 4.4 Lyapunov Stability Analysis

One of the key issues of NMPC is stability. The stability of system without uncertainty is called *nominal stability*, while the stability of system with uncertainty is called *robust stability*. The uncertainty could be uncertain parameters, plant-model mismatch, additive disturbance or measurement noise, etc. In this section we study the stability properties of setpoint tracking NMPC.

We start with introducing some basic notations and definitions that are used for stability analysis.

**Definition 9.** *[40] A continuous function* $\alpha(\cdot) : \Re_{\geq 0} \mapsto \Re_{\geq 0}$ *is a* $\mathscr{K}$ *function if* $\alpha(0) = 0, \alpha(s) > 0, \forall s > 0$ *and it is strictly increasing.*
*A continuous function* $\alpha(\cdot) : \Re_{\geq 0} \mapsto \Re_{\geq 0}$ *is a* $\mathscr{K}_\infty$ *function if it is a* $\mathscr{K}$ *function and* $\alpha(s) \to +\infty$ *as* $s \to +\infty$*. A continuous function* $\alpha(\cdot,\cdot)$ *: is a* $\mathscr{KL}$ *function if* $\alpha(s,k) : \Re^2_{\geq 0} \mapsto \Re_{\geq 0}$ *is a* $\mathscr{K}$ *function in s for any* $k > 0$ *and for each* $s > 0$*,* $\alpha(s,\cdot)$ *is decreasing and* $\alpha(s,k) \to 0$ *as* $k \to \infty$*.*

**Definition 10.** *[6] A function* $f(x) : \Re^n \mapsto \Re$ *is continuous in* $\Re^n$ *if for every pair of points,* $x, x$ *and all* $\varepsilon > 0$ *there is a value* $\delta > 0$ *such that*

$$\| x - x' \| < \delta \implies \| f(x) - f(x') \| < \varepsilon \tag{4.6}$$

*and therefore* $\lim_{x \to \bar{x}} f(x) \to f(\bar{x})$.
*A continuous function* $f(x) : \Re^n \mapsto \Re$ *is Lipschitz continuous in* $\Re^n$ *if for any two points* $x, y \in \Re^n$ *there exists a finite* $L > 0$ *such that*

$$\| f(x) - f(y) \| < L \| x - y \| \tag{4.7}$$

**Definition 11.** *A set* $\mathscr{A} \subset \mathbb{X}$ *is a positively invariant set for system (2.2) if* $x \in \mathscr{A}$ *implies* $f(x,u) \in \mathscr{A}$*.*

We use *Lyapunov function* to prove the stability of the system. It is a function that takes positive values everywhere except at the equilibrium of the system, and is non-increasing along every trajectory. Lyapunov function could be viewed as an energy function of a system. The system is stable if we could construct a Lyapunov function for it.

### 4.4.1 Nominal Stability

When there is no uncertainty, the system evolves as (2.2). We use $\phi(k;x(0))$ to denote the solution of (2.2) at time $t_k$ with initial condition $x(0)$. For nominal stability analysis we use the concept of asymptotic stability [53].

**Definition 12.** *The (closed positive invariant) set $\mathscr{A}$ is locally stable for (2.2) if, for all $\varepsilon > 0$, there exists a $\delta > 0$ such that $|x|_{\mathscr{A}} < \delta$ implies $|\phi(k;x(0))|_{\mathscr{A}} < \varepsilon, k \in \mathbb{I}_{\geq 0}$.*

**Definition 13.** *The (closed positive invariant) set $\mathscr{A}$ is locally attractive if there exists $\eta > 0$ such that $|x|_{\mathscr{A}} < \eta$ implies $|\phi(k;x(0))|_{\mathscr{A}} \to 0$ as $k \to \infty$.*

**Definition 14.** *The closed positive invariant set $\mathscr{A}$ is locally asymptotically stable if it is locally stable and locally attractive.*

The definition of nominal stability is stated as follows:

**Definition 15.** *System (2.2) is said to be locally nominally stable in $\mathbb{X}$ if there exists a $\mathscr{KL}$ function $\beta(\cdot,\cdot)$ such that*

$$|x(k)| \leq \beta(|x(0)|,k),\ \forall k \geq 0,\ \forall x(0) \in \mathbb{X} \tag{4.8}$$

A function $V(\cdot)$ is called a Lyapunov function for system (2.2) if there exist a set $\mathbb{X}$, $\mathscr{K}_\infty$ functions $\alpha_1$, $\alpha_2$, $\alpha_3$ such that $\forall x \in \mathbb{X}$, we have:

$$\begin{aligned} \alpha_1(|x|) &\leq V(x) \leq \alpha_2(|x|) \\ \Delta V(x) &= V(f(x,u)) - V(x) \leq -\alpha_3(|x|) \end{aligned} \tag{4.9}$$

### 4.4.2 Robust Stability

In the presence of uncertainty, the system evolves as (2.6). Moreover, if at $t_k$, besides disturbance $w(k)$, there is also uncertainty due to old error, we introduce the following formulation of the system:

$$x(k+1) = f(x(k),u(k),d(k)) + w(k) = \tilde{f}(x(k),u(k),d(k),w(k)), \tag{4.10}$$

where $d(k) = \phi(w(k-j), j = 1,\ldots,N), d(k) \in \mathscr{D} \subset \Re^{n_d}$ is bounded. We use Input-to-State Stability (ISS) theory to prove the stability of (2.6) and Input-to-State Practical Stability (ISpS) theory to prove the stability of (4.10).

#### Input-to-State Stability

**Definition 16.** *System (2.6) is said to be locally Input-to-State Stable (ISS) in $\mathbb{X}$ if there exists a $\mathscr{KL}$ function $\beta$, and a $\mathscr{K}$ function $\gamma_1$ such that for all w in the bounded set $\mathscr{W}$,*

$$|x(k)| \leq \beta(|x(0)|,k) + \gamma_1(|w|),\ \forall k \geq 0,\ \forall x(0) \in \mathbb{X} \tag{4.11}$$

A function $V(\cdot)$ is called an ISS-Lyapunov function for system (2.6) if there exist a set $\mathbb{X}$, $\mathscr{K}_\infty$ functions $\alpha_4, \alpha_5, \alpha_6$ and $\mathscr{K}$ function $\sigma_1$ such that, $\forall x \in \mathbb{X}$ and $\forall w \in \mathscr{W}$, we have:

$$\begin{aligned} \alpha_4(|x|) &\leq V(x) \leq \alpha_5(|x|) \\ V(\tilde{f}(x,u,w)) - V(x) &\leq -\alpha_6(|x|) + \sigma_1(|w|) \end{aligned} \tag{4.12}$$

Moreover, if $\mathbb{X}$ is a robustly invariant set for system (2.6) and $V(\cdot)$ is an ISS-Lyapunov function for this system, then the resulting system is ISS in $\mathbb{X}$ [9, 40].

Notice that asymptotic stability is stronger than ISS. If $|w| \to 0$ then ISS implies asymptotic stability. Also, $|x(k)|$ is asymptotically bounded by $\gamma_1(|w|)$.

To prove the robust stability of iNMPC, we prove that the objective function of $\mathscr{P}_N(x(k))$ is an ISS-Lyapunov function. We make the following assumptions:

**Assumption 1.** *(Robust Stability Assumptions)*

*(i)* $g(x,w)$ *is Lipschitz with respect to all its arguments with Lipschitz constant* $L_g$ *:* $|g(x,w)| \leq |g(x,0)| + L_g|w|$.

*(ii)* $|g(x,w)|$ *satisfies*

$$|g(x,0)| < \frac{\rho}{\zeta}\alpha_p(|x|), \text{ and } |g(x,0)| \leq g_{max} \tag{4.13}$$

*where* $\rho \in (0,1)$ *is an arbitrary real number and* $\zeta > 0$ *is a prespecified constant.*

Theorem 4 will be proved in Chapter 3 after NLP sensitivity is introduced.

### Input-to-State Practical Stability

The concept of Input-to-State Practical Stability (ISpS) is introduced in [39].

**Definition 17.** *A system (4.10) is said to be locally Input-to-State Practical Stable (ISpS) in* $\mathbb{X}$ *if* $\mathbb{X}$ *is a robust positively invariant set for system (4.10) and if there exist a* $\mathscr{KL}$ *function* $\beta$*, and a* $\mathscr{K}$ *function* $\gamma_2$ *and a constant* $c \geq 0$ *such that for all* $w$ *in the bounded set* $\mathscr{W}$ *and* $d$ *in the bounded set* $\mathscr{D}$*,*

$$|x(k)| \leq \beta(|x(0)|,k) + \gamma_2(|w|) + c, \ \forall k \geq 0, \ \forall x(0) \in \mathbb{X} \tag{4.14}$$

**Definition 18.** *A function* $V(\cdot)$ *is called an ISpS-Lyapunov function for system (4.10) with respect to* $w$*, if there exist sets* $\mathbb{X}, \mathscr{W}, \mathscr{D}$ *and* $\mathscr{K}_\infty$ *functions* $\alpha_7, \alpha_8, \alpha_9$*, a* $\mathscr{K}$ *function* $\sigma_2$ *and a couple of constants* $c_1, c_2 \in \Re_{\geq 0}$ *such that* $\forall x \in \mathbb{X}$*,* $\forall d \in \mathscr{D}$ *and* $\forall w \in \mathscr{W}$*, we have:*

$$\begin{aligned} \alpha_7(|x|) &\leq V(x) \leq \alpha_8(|x|) + c_1 \\ V(\tilde{f}(x,u,w,d)) - V(x) &\leq -\alpha_9(|x|) + \sigma_2(|w|) + c_2 \end{aligned} \tag{4.15}$$

ISpS will be used to prove the robust stability of advanced-multi-step NMPC proposed in Chapter 5.

### 4.4.3 Ideal Setpoint Tracking NMPC

#### Nominal Stability

To prove the nominal stability of iNMPC we need to prove that the Lyapunov function is strictly decreasing except at equilibrium. We make the following assumptions:

**Assumption 2.** *(Nominal Stability Assumptions for ideal NMPC)*

*(i)* *The terminal cost* $\Psi(\cdot)$ *in (4.3) satisfies* $\Psi(x) > 0$.

*(ii)* *There exits a local control law* $u = \kappa_f(x)$ *for all* $x \in \mathbb{X}_f$*, some unspecified terminal region, where* $\Psi(f(x,\kappa_f(x))) - \Psi(x) \leq -\psi(x,\kappa_f(x))$.

*(iii)* $\psi(x,u)$ *satisfies* $\alpha_p(|x|) \leq \psi(x,u) \leq \alpha_q(|x|)$ *where* $\alpha_p(\cdot)$ *and* $\alpha_q(\cdot)$ *are* $\mathscr{K}$ *functions.*

**Theorem 3.** *(Nominal Stability of ideal NMPC) Consider the moving horizon Problem (4.3) and associated control law* $u = \kappa_f(x)$ *that satisfies Assumption 2. Then,* $J_N(x(k))$ *is a Lyapunov function for system (2.2) and the closed-loop system is asymptotically stable.*

*Proof.* We compare the optimal cost function of the two neighboring ideal NMPC problems

(4.3).

$$\begin{aligned} & J_N(x(k+1)) - J_N(x(k)) \\ \leq\ & \Psi(f(z_N, \kappa_f(z_N))) + \sum_{l=1}^{N-1} \psi(z_l, v_l) + \psi(z_N, \kappa_f(z_N)) - [\Psi(z_N) + \sum_{l=0}^{N-1} \psi(z_l, v_l)] \\ =\ & \Psi(f(z_N, \kappa_f(z_N))) - \Psi(z_N) + \psi(z_N, \kappa_f(z_N)) - \psi(z_0, v_0) \\ \leq\ & -\psi(z_0, v_0) \\ =\ & -\psi(x(k), u(k)) \end{aligned} \tag{4.16}$$

where the first inequality comes from the fact that the solution of $\mathscr{P}_N(x(k))$ is feasible for $\mathscr{P}_N(x(k+1))$. This leads to:

$$J_N(x(0)) \geq J_N(x(0)) - J_N(x(\infty)) = \sum_{k=0}^{\infty} (J_N(x(k)) - J_N(x(k+1))) \geq \sum_{k=0}^{\infty} \psi(x(k), u(k)) \tag{4.17}$$

and $\lim_{k\to\infty} \psi(x(k), u(k)) = 0$. Also, by Assumption 2(iii), $\lim_{k\to\infty} x(k) = 0$. Hence, $J_N(x(k))$ satisfies the conditions of Lyapunov function and nominal stability property of ideal NMPC in the Lyapunov sense is proved[60]. □

**Robust Stability**

Robust stability of the iNMPC controller can be established from the following theorem.

**Theorem 4.** *Robust ISS Stability of iNMPC (Theorem 2 in [40], see also [32, 60]) Under Assumptions 1 and 2 with $\zeta = L_J$, the cost function $J_N(x)$ obtained from the solution of $\mathscr{P}_N(x)$ (4.3) is an ISS-Lyapunov function and the resulting closed-loop system is ISS stable.*

Proof of Theorem 4 could be found is ([60]). We compare the costs of the neighboring problems $\mathscr{P}_N(x(k))$ and $\mathscr{P}_N(x(k+1))$. Without uncertainty the system evolves as $\bar{x}(k+1) = f(x(k), u(k))$ while with uncertainty the system evolves as $x(k+1) = f(x(k), u(k)) +$

$g(x(k),u(k))$. We define $\varepsilon(x(k+1)) := J_N(x(k+1)) - J_N(\bar{x}(k+1))$. Under Theorem 2, there exits a local positive Lipschitz constant $L_J$ such that $\forall x \in \mathbb{X}$,

$$|\varepsilon(x(k+1))| \leq L_J |g(x(k),w(k))| \tag{4.18}$$

$$\begin{aligned} & J_N(x(k+1)) - J_N(x(k)) \\ = \; & J_N(\bar{x}(k+1)) - J_N(x(k)) + J_N(x(k+1)) - J_N(\bar{x}(k+1)) \end{aligned} \tag{4.19}$$

Since the solution of $\mathscr{P}_N(x(k))$ is feasible for $\mathscr{P}_N(\bar{x}(k+1))$, from the proof of Nominal Stability of iNMPC in Chapter 2, we have $J_N(\bar{x}(k+1)) - J_N(x(k)) \leq -\psi(x(k),u(k))$. And from (4.18) we have $J_N(x(k+1)) - J_N(\bar{x}(k+1)) \leq L_J |g(x(k),w(k))|$. Therefore we have

$$\begin{aligned} & J_N(x(k+1)) - J_N(x(k)) \\ \leq \; & -\psi(x(k),u(k)) + L_J |g(x(k),w(k))| \\ \leq \; & -\alpha_p(|x(k)|) + L_J \frac{\rho}{\zeta} \alpha_p(|x(k)|) + L_J L_g |w(k)| \\ \leq \; & (\rho - 1)\alpha_p(|x(k)|) + \sigma |w(k)| \end{aligned} \tag{4.20}$$

where the second inequality results from Assumption 1 and the last inequality follows from $\zeta = L_J$ and $\sigma|w(k)| = L_J L_g |w(k)|$. Robust stability of ideal setpoint tracking NMPC (Theorem 4) is proved.

**Stabilizing a Non-robust MPC**

As developed in Section 4.2.1, we reformulate the NLP (2.4) by replacing state constraints with soft constraints and adding $\ell_1$ penalties to the stage costs. It is pointed out in [47] that when there are no state constraints, the terminal constraint could be replaced by an appropriate terminal penalty without loss of robustness. Also, we need to specify a sufficiently large prediction horizon (determined through off-line simulation) such that $x \in \mathbb{X}_f$

is satisfied and is not active. This satisfies the assumptions for Theorems 2 and 4, so the results still hold here. Moreover, if we assume that SOSC, LICQ and strict complementarity hold at the optimal solution of $\mathscr{P}_N(x(k))$ (problems (4.3), (3.10)), the value function and feedback law are continuous and differentiable. As a result, robustness will not be lost.

In [24], Grimm et al. presented three NMPC examples that lack robustness because of state constraints, terminal region or a short horizon. In this section we study those three examples and show that robustness can be obtained by a long enough horizon or reformulation of the NLPs.

**Example 1: Artstein's circles with state constraints** This example is the discretization of the following continuous system:

$$\begin{aligned} \dot{x}_1 &= (x_1^2 - x_2^2)u \\ \dot{x}_2 &= 2x_1^2 x_2^2 u \end{aligned} \tag{4.21}$$

One peculiarity of system (4.21) is that if the system is initially on the circle $x_1^2 + (x_2 - r)^2 = r^2$, where $r \in \Re$, then the trajectory stays on the same circle regardless of control. Zero-order hold is used to discretize the system with a sampling time of 1. The discretized system $\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), u(k))$ evolves as

$$\begin{aligned} x_1(k+1) &= \frac{-(x_1^2(k) + x_2^2(k))u(k) + x_1(k)}{1 + (x_1^2(k) + x_2^2(k))u^2(k) - 2x_1(k)u(k)} \\ x_2(k+1) &= \frac{x_2(k)}{1 + (x_1^2(k) + x_2^2(k))u^2(k) - 2x_1(k)u(k)} \end{aligned} \tag{4.22}$$

Then they introduce the MPC controller with input and state constraints $u \in [-1,1] =: \mathbb{U}$ and $\mathbf{x} \in \{x \in \Re^2 : x_1 \leq c,\ c \in (0,1)\} =: \mathbb{X}$. The terminal cost is the length of the shorter arc between the state and the origin:

$$g(\mathbf{x}) = |\mathbf{x}| cos^{-1} \frac{(x_2 - |\mathbf{x}|)(-|\mathbf{x}|)}{|\mathbf{x}| \sqrt{x_1^2 + (x_2 - |\mathbf{x}|)^2}}. \tag{4.23}$$

The stage cost is the length of the shorter arc between $\mathbf{x}$ and the closest reachable point from $\mathbf{x}$ to the origin in one step:

$$l(\mathbf{x},u) = |\mathbf{x}| cos^{-1} \frac{x_1 f_1(\mathbf{x},-1) + (x_2 - |\mathbf{x}|)(f_2(\mathbf{x},-1) - |\mathbf{x}|)}{\sqrt{x_1^2 + (x_2 - |\mathbf{x}|)^2}\sqrt{f_1(\mathbf{x},-1)^2 + (f_2(\mathbf{x},-1) - |\mathbf{x}|)^2}}. \tag{4.24}$$

The terminal region is $\mathbb{X}_f = \varepsilon \mathscr{B}_2$ where $\mathscr{B}_2$ is the unit circle. Moreover, the control law $\kappa_f(\mathbf{x})$ satisfies $\kappa_f(\mathbf{x}) = -x_1/|x_1|$ if $x_1 \neq 0$ and $\kappa_f(\mathbf{x}) = -1$ if $x_1 = 0$.

This NMPC algorithm is nominally stable but not robustly stable. The analysis could be found in [24]. In general, if the trajectory starts from a point in the first quadrant and moves clockwise, when it hits $x_1 = c$, it reverses its direction and moves counter-clockwise. Then at the next step it moves clockwise and hits $x_1 = c$ again. As a result, it will move back and forth between the current state and $x_1 = c$ and will not converge to the origin. We reformulate the NMPC by adding soft constraints to the state constraints and terminal region. As a result, for the terminal region we have

$$|\mathbf{x}_N| \leq \varepsilon + s_N \tag{4.25}$$

where $N$ is the horizon length, $s_N$ is a slack variable and $s_N \geq 0$. And the state constraint becomes

$$x_{i,1} \leq c + s_i,\ i = 0,\ldots,N-1 \tag{4.26}$$

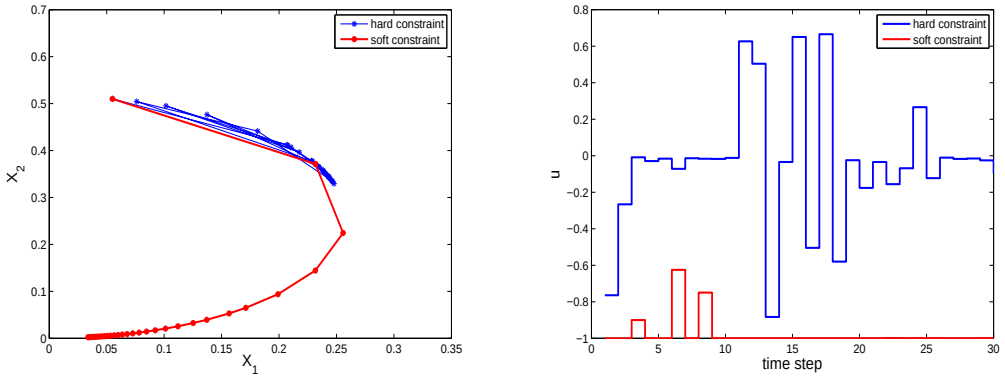
where $s_l$ is the slack variable and $s_l \geq 0$. And we add $\ell_1$ penalty to the objective function:

$$\sum_{i=0}^{N-1} l(\mathbf{x}_i,u_i) + g(x_N) + \rho_l \sum_{i=0}^{N-1} s_i + \rho_g s_N \tag{4.27}$$

As a result, when the state reaches the vertical line $x_1 = c$ on its way to converge to the origin, instead of going counter clockwise, it goes beyond $x_l = c$ and keeps moving clockwise, and then crosses $x_l = c$ and becomes feasible again and eventually converges to the origin.

We introduce additive disturbance which is Gaussian noise with zero mean and standard deviation of 0.05. Horizon length is $N = 10$ and simulation time is 30. We choose $c = 0.25$, $\varepsilon = 0.1$ and weight on slack variables are $\rho_l = \rho_g = 10^5$. The system starts from $(0.055, 0.51)$ such that it hits $x_1 = 0.25$ on its trajectory to the origin.

We first solve the original NLP problem with no soft constraints. The state trajectory and control trajectory are plotted in Fig. 4.2. It could be observed from Fig. 4.2(a) that before soft constraints are added, the states evolve back and forth, but are stopped by $x_1 = c$ and never converge to the origin; the control in Fig. 4.2(b) changes its direction every now and then. After soft constraints are applied using NLP (4.1), the state goes beyond $x_1 = c$ in one step and then converges to the origin.




(a) Trajectory of $\mathbf{x}$ (b) Trajectory of u

Figure 4.2: Trajectory of $\mathbf{x}$ and $u$. The original MPC is shown in blue; the soft-constrained MPC is shown in red.

**Example 2: origin as terminal constraint** For the second example, the discrete system $\mathbf{x}(k+1) = f(\mathbf{x}(k), u(k))$ evolves as

$$\begin{aligned} x_1(k+1) &= x_1(k)(1-u(k)) \\ x_2(k+1) &= |\mathbf{x}(k)|u(k) \end{aligned} \tag{4.28}$$

where $u \in [0,1] =: \mathbb{U}$ and $\mathbf{x} := (x_1, x_2) \in \Re^2 =: \mathbb{X}$. Then they introduce the MPC controller with input constraint $u \in \mathbb{U}$. The terminal cost is $g(\mathbf{x}) = 0$. The state cost $l(\mathbf{x}, u)$ satisfies $l(\mathbf{x}, u) \leq \alpha(|\mathbf{x}|)$ where $\alpha(\cdot)$ is a $\mathscr{K}_\infty$ function. We choose $l(\mathbf{x}, u) = x_1^2 + x_2^2$. The terminal constraint is $\mathbb{X}_f = \{0\}$. Horizon length is $N = 2$.

With this MPC algorithm the origin is nominally stable but not robustly stable with measurement error or additive disturbance. This analysis can be found in [24]. We reformulate the MPC by adding soft constraints to the terminal constraint to yield (4.1). As a result, for the terminal constraint we have

$$x_{N,1}^2 + x_{N,2}^2 = s \tag{4.29}$$

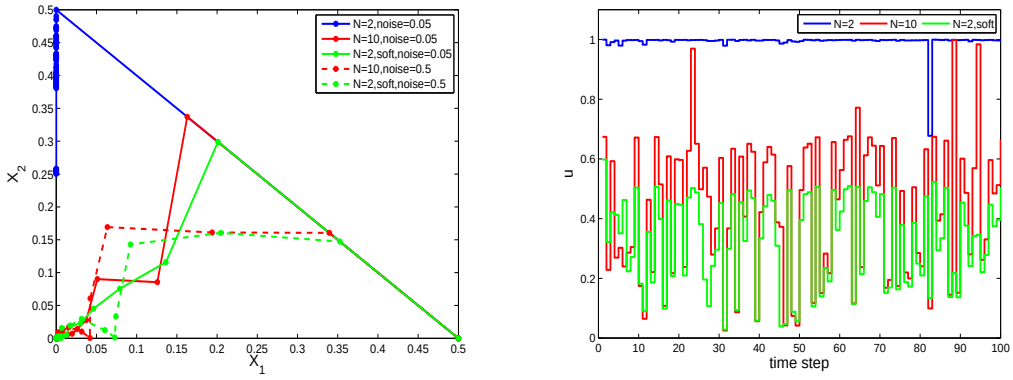
where $N$ is the horizon length, $s$ is the slack variable and $s \in \Re_{\geq 0}$. And we add $\ell_1$ penalty to the objective function:

$$\sum_{i=0}^{N-1} l(\mathbf{x}_i, u_i) + g(\mathbf{x}_N) + \rho s \tag{4.30}$$

We introduce additive disturbance which is Gaussian noise with zero mean and standard deviation of 0.05. Horizon length is $N = 2$ and simulation time is 100. We choose $\rho = 10^3$. The system starts from $(0.5, 0)$.

We first solve the original NLP problem with $N = 2$ and without soft constraints. We plot the trajectory of $\mathbf{x}$ within the simulation period. As we could see from Fig. 4.3(a), although we use a long simulation time (100), $\mathbf{x}$ still does not converge to $(0,0)$. If we look at the corresponding control profile in Fig. 4.3(b) we could see that $u$ stays close to 1, and as a result according to (4.28), $x_2(k+1) = |\mathbf{x}(k)|$ will not decade to 0. We then increase $N$ to 10 and observe that $\mathbf{x}$ converges to the origin within only a few steps. Keeping $N = 2$ but adding soft constraint (4.29), we observe that $\mathbf{x}$ converges within the first few steps from a slightly different trajectory. Also from Fig. 4.3(b) we could see that $u(k) \in [0,1]$ with a longer horizon $N = 10$ or with a soft state constraint, both $x_1$ and $x_2$ could decrease to 0. The reformulation schemes also work well when we increase the noise standard deviation

to 0.5.




(a) Trajectory of **x**, noise=0.05 or 0.5

(b) Trajectory of u, noise=0.05

Figure 4.3: Trajectory of **x** and $u$. $N = 2$ without soft constraint is shown in blue; $N = 10$ without soft constraint is shown in red; $N = 2$ with soft constraint is shown in green.

**Example 3: unit disk as terminal constraint** For the third example, the discrete system $\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k))$ evolves as

$$x_1(k+1) = (1+|x_1(k)|)\sin(u_1(k))\sin(u_2(k)) + \gamma(\theta_{\mathbf{x}(k)})\mathbf{x}_{\frac{\pi}{8}}(k)\cos(u_2(k)) \quad (4.31a)$$

$$x_2(k+1) = (1+|x_1(k)|)\cos(u_1(k))\sin(u_2(k)) + \gamma(\theta_{\mathbf{x}(k)})\mathbf{x}_{\frac{\pi}{8}}(k)\cos(u_2(k)) \quad (4.31b)$$

where $\mathbf{u} = [u_1; u_2] \in [0, \frac{\pi}{4}] \times [0, \frac{\pi}{2}] =: \mathbb{U}$ and $\mathbf{x} := [x_1; x_2] \in \Re^2 =: \mathbb{X}$. $\theta_x$ is the angle between the state and the positive vertical axis and $\mathbf{x}_{\frac{\pi}{8}}$ is the state $\mathbf{x}$ rotated $\frac{\pi}{8}$ counter clockwise. $\gamma$ is a piecewise linear function of $\theta$:

$$\gamma(\theta) = \begin{cases} 2 - \frac{2\times\theta}{\pi} & if\ \frac{\pi}{2} \le \theta < \pi \\ -2 + \frac{2\times\theta}{\pi} & if\ \pi \le \theta < \frac{3\pi}{2} \\ 1 & otherwise \end{cases} \quad (4.32)$$

Then they introduce the MPC controller with input constraint $\mathbf{u} \in \mathbb{U}$. The terminal cost is $g(\mathbf{x}) = \sum_{k=0}^{\infty} |\mathbf{x}(k)|$ with $\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{0})$. The state cost $l(\mathbf{x}, \mathbf{u})$ satisfies $l(\mathbf{x}, \mathbf{u}) \le$

$\alpha(|\mathbf{x}|)$ where $\alpha(\cdot)$ is a $\mathscr{K}_\infty$ function. We choose $l(\mathbf{x},\mathbf{u}) = x_1^2 + x_2^2 + u_1^2 + u_2^2$. The terminal constraint is the unit circle $\mathbb{X}_f = \mathscr{B}_2$. Horizon length is $N = 2$.

With this MPC algorithm the origin is nominally stable but not robustly stable. The analysis can be found in [24]. We reformulate the MPC by adding soft constraints to the terminal constraint. As a result, for the terminal constraint we have

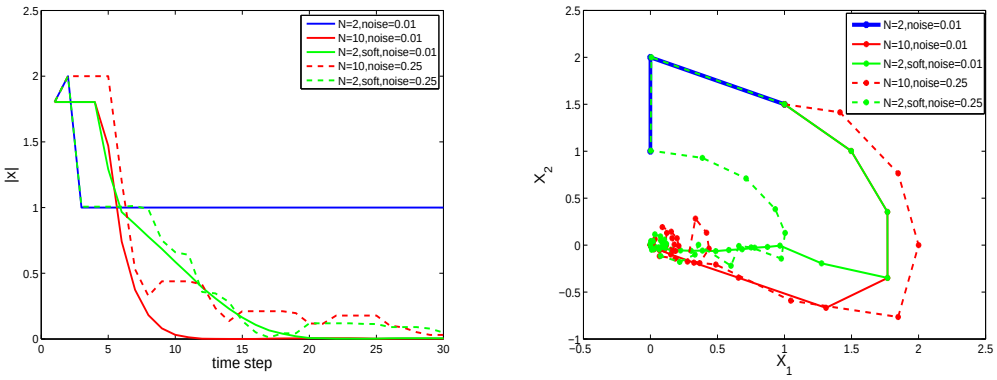
$$|\mathbf{x}_N| \leq 1 + s_N \tag{4.33}$$

where $N$ is the horizon length, $s_N$ is a slack variable, $s_N \geq 0$. And we add $\ell_1$ penalty to the objective function:

$$\sum_{i=0}^{N-1} l(\mathbf{x}_i, \mathbf{u}_i) + g(\mathbf{x}) + \rho s_N \tag{4.34}$$

We introduce additive disturbance which is Gaussian noise with zero mean and standard deviation of 0.01. Horizon length is $N = 2$ and simulation time is 30. We choose $\rho = 1$. The system starts from $(1, 1.5)$.

We first solve the original NLP problem with $N = 2$ and without soft constraints. We plot the trajectories of $|\mathbf{x}|$ and $\mathbf{x}$ within the simulation period. As we could see from Fig. 4.4(a), $|\mathbf{x}|$ stalls at $|\mathbf{x}| = 1$ and we can see from Fig. 4.4(b) that $\mathbf{x}$ oscillates within a very small region around $(0, 1)$. We then increase $N$ to 10 and observe that $|\mathbf{x}|$ eventually converges to 0 and $\mathbf{x}$ converges to the origin. Keeping $N = 2$ but adding soft constraint (4.33), we observe that $|\mathbf{x}|$ decreases to 0 from a different trajectory. The convergence rate is slower than with $N = 10$ because $N = 10$ allows more degrees of freedom for the controller to move around. We then increase the noise standard deviation to 0.25, and observe convergence to the origin, but with a slower rate and larger values of $|\mathbf{x}|$.




(a) Trajectory of $|\mathbf{x}|$,noise=0.01 or 0.25

(b) Trajectory of $\mathbf{x}$,noise=0.01 or 0.25

Figure 4.4: Trajectory of $|\mathbf{x}|$ and $\mathbf{x}$. $N = 2$ without soft constraint is shown in blue; $N = 3$ without soft constraint is shown in red; $N = 2$ with soft constraint is shown in green; asNMPC with $N = 2$ and with soft constraint is shown in magenta.

### 4.4.4 asNMPC

#### Nominal Stability

In the nominal case, there is no uncertainty, so $p = p_0$ and $\Delta v = 0$ in the NLP (3.10), the controls do not need update. Therefore asNMPC leads to the same solution as iNMPC. As a result, asNMPC has the same nominal stability as iNMPC.

#### Robust stability

To analyze the robustness of the asNMPC controller, we need to consider the effect of NLP sensitivity errors. Also we recognize that with $\bar{x}(k+1) = f(x(k),u(k))$, there exists a future mismatch $x(k+1) - \bar{x}(k+1) = g(x(k),w(k))$ at the next time step, and the plant will evolve with uncertain dynamics generating $x(k+1)$, giving rise to two different problems (4.3) $\mathscr{P}_N(\bar{x}(k+1))$ and $\mathscr{P}_N(x(k+1))$, with optimal costs $J_N(\bar{x}(k+1))$ and $J_N(x(k+1))$,

respectively. Moreover, we need to distinguish between iNMPC using $u^{id}(k)$ and asNMPC, which generates $u^{as}(k) = \kappa^{as}(x(k))$. To interpret this difference we consider an extended problem $\mathscr{P}_{N+1}(x(k), \hat{u}(k))$:

$$\hat{J}(x(k), \hat{u}(k)) := \min_{z_l, v_l} \quad \Psi(z_N) + \psi(x(k), \hat{u}(k)) + \sum_{l=0}^{N-1} \psi(z_l, v_l) \tag{4.35a}$$

$$\text{s.t. } z_{l+1} = f(z_l, v_l) \quad l = 1, \ldots N-1 \tag{4.35b}$$

$$z_0 = f(x(k), \hat{u}(k)) \tag{4.35c}$$

$$v_l \in \mathbb{U} \tag{4.35d}$$

This problem has an equivalent solution to problem $\mathscr{P}_N(z_0)$ and we consider $\hat{J}(x, \hat{u})$ as our candidate ISS Lyapunov function. We define $J^{as}(x(k)) := \hat{J}(x(k), u^{as}(k))$, $J^{id}(x(k)) := \hat{J}(x(k), u^{id}(k))$, and also $J^{id}(\overline{x}(k)) := \hat{J}(\overline{x}(k), \overline{u}^{id}(k))$, where $u^{id}(k)$ and $\overline{u}^{id}(k)$ are determined as variables, $\hat{u}(k) \in \mathbb{U}$, in $\mathscr{P}_{N+1}$. For the next time step, we define the following residuals as:

$$\varepsilon_s(x(k+1)) := J^{id}(x(k+1)) - J^{id}(\overline{x}(k+1)) \tag{4.36a}$$

$$\varepsilon_{as}(x(k+1)) := J^{as}(x(k+1)) - J^{id}(x(k+1)) \tag{4.36b}$$

where $\varepsilon_s$ accounts for the model mismatch as in $x(k+1)$ while $\varepsilon_{as}$ for approximation errors introduced by NLP sensitivity. If clipping is applied in case of active set change, (3.24) becomes

$$\tilde{s}(p) = s^*(p_0) + \frac{ds^*}{dp}\tau(p - p_0) \tag{4.37}$$

And as a result $(3.25) - (4.37)$ becomes

$$s^*(p) - \tilde{s}(p) = (1-\tau)\frac{ds^*}{dp}(p - p_0) + O(|p - p_0|^2) \tag{4.38}$$

Therefore we have

$$|\tilde{s}(p) - s^*(p)| \leq (1-\tau)\frac{ds^*}{dp}|p - p_0| + L_q|p - p_0|^2 \tag{4.39}$$

From (4.39) and Theorem 2 we have positive Lipschitz constants $L_J, L_u, L_{sv}$ and $L_{sq}$ such that $\forall x \in \mathbb{X}$,

$$\begin{aligned} \varepsilon_s(x(k+1)) &\leq L_J(|x(k+1) - \overline{x}(k+1)| \leq L_J|g(x(k), w(k))| && \text{(4.40a)} \\ \varepsilon_{as}(x(k+1)) &\leq L_u(|u^{as}(k+1) - u^{id}(k+1)|) \\ &\leq L_J((1-\tau)L_{sv} + L_{sq}|g(x(k), w(k))|) \cdot |g(x(k), w(k))| && \text{(4.40b)} \end{aligned}$$

By comparing the successive costs $J^{as}(x(k))$ and $J^{as}(x(k+1))$, we arrive at a similar ISS property as in Theorem 4.

**Theorem 5** (*Robust Stability of asNMPC*)**.** *Under Assumptions 2 and 1 with* $\zeta = L_J(1 + L_{sv} + L_{sq} g_{max})$ *in (4.13), the cost function* $J^{as}(x)$ *obtained from the solution of Problem (4.35) with* $u = u^{as}$ *is an ISS-Lyapunov function and the resulting closed-loop system is ISS stable.*

**Proof :** We compare the costs $J^{as}(x(k))$, $J^{as}(x(k+1))$ and use the mismatch terms in (4.40a)-(4.40b) to obtain,

$$\begin{aligned} & J^{as}(x(k+1)) - J^{as}(x(k)) \\ = \; & J^{id}(\overline{x}(k+1)) - J^{as}(x(k)) + J^{id}(x(k+1)) - J^{id}(\overline{x}(k+1)) \\ & + J^{as}(x(k+1)) - J^{id}(x(k+1)) \\ \leq \; & -\psi(x(k), u^{as}(k)) + \varepsilon_s(x(k+1)) + \varepsilon_{as}(x(k+1)) \\ \leq \; & -\alpha_p(|x(k)|) + \varepsilon_s(x(k+1)) + \varepsilon_{as}(x(k+1)). \end{aligned} \tag{4.41}$$

The last two inequalities follow by noting that the solution of Problem (4.35) at $k$ provides a feasible solution to Problem (4.35) at $k+1$, and from Assumption 2. Substituting the

bounds (4.40a)-(4.40b) for the error terms leads to:

$$\begin{aligned}
& \varepsilon_s(x(k+1)) + \varepsilon_{as}(x(k+1)) \\
\leq\ & L_J(1+(1-\tau)L_{sv}+L_{sq}(|g(x(k),0)|+L_g|w(k)|))(|g(x(k),0)|+L_g|w(k)|) \\
\leq\ & L_J(1+L_{sv}+L_{sq}g_{max})(\rho/\zeta)\alpha_p(|x|) \\
& +L_J(L_g(1+L_{sv}+(1+L_{sq})g_{max})|w(k)|+(L_g)^2|w(k)|^2) \\
\leq\ & \rho\alpha_p(|x(k)|)+\sigma|w(k)|
\end{aligned}$$

where the first two inequalities follow from Assumption 5 and from $\tau \geq 0$, and the last inequality follows from $\zeta = L_J(1+L_{sv}+L_{sq}g_{max})$ and $\sigma(|w|) = L_J(L_g(1+L_{sv}+(1+L_{sq})g_{max})|w(k)|+(L_g)^2|w(k)|^2)$. The theorem is proved by substituting this result into (4.41) to yield:

$$J^{as}(x(k+1)) - J^{as}(x(k)) \leq (\rho-1)\alpha_p(|x(k)|)+\sigma|w(k)|. \ \square$$

Note that $\zeta$ determines the bound on $g(x,0)$ in the ISS condition (4.13) and we can identify the following performance levels.

- If clipping ($\tau < 1$) is applied at frequent intervals as $k \to \infty$, we have the largest $\zeta$ and the smallest values of $|g(x,0)|$ allowed for robustness due to (4.13) .
- If $\tau = 1$ for all $k \geq k_1 > 0$ then we have the robust performance of unconstrained asNMPC derived in [60], whose additional loss is due to a term proportional to $|g(x(k),w(k))|^2$.
- If $g(x(k),w(k)) = 0$, then $x(k+1) = \bar{x}(k+1)$ and asNMPC recovers the nominal stability and performance of iNMPC, and clipping is not needed.

## 4.5 Concluding Remarks

In this chapter we start with some fast NMPC algorithms and then discuss the formulation and implementation of asNMPC. By applying NLP sensitivity, computational cost could be reduced by 2 or 3 orders of magnitude [60]. To handle active set change, "clipping in first interval" is proposed. We then prove nominal stability and robust stability for ideal NMPC and asNMPC. This chapter lays a basis for the amsNMPC method which we are going to discuss in the next chapter. The implementation of amsNMPC as well as stability analysis are based on the corresponding theories of asNMPC. We also study three MPC examples that lack robustness and show that robustness could be obtained with a long enough horizon or reformulation of NLP. The case studies of asNMPC will be shown in Chapter 6 in comparison with the performance of amsNMPC.

# Chapter 5

# Advanced-multi-step Nonlinear Model Predictive Control

In Chapter 4 we discussed the asNMPC strategy. It applies to the situation where the NLP solution time is less than one sampling time. In this chapter we propose the advanced-multi-step NMPC strategy, which could be used to avoid computational delay when the NLP solution time exceeds one sample time.

## 5.1 Motivation

Due to NLP sensitivity update which takes a negligible amount of time, asNMPC is able to reduce the computational time by 2 to 3 orders of magnitude and thus avoid computational delay. Moreover, in Chapter 4, it is proved to be nominally stable and robustly stable. However, asNMPC requires the NLP problem to be solved within one sampling time. This condition can be satisfied for many applications, but there are exceptions if the problem scale is very large or if faster updates are needed. If NLP solution takes longer, neither iNMPC or asNMPC will work properly. On the other hand, there are several fast MPC or NMPC methods that deal with this case. They have been discussed in Chapter 2. For the case where the NLP computation requires more than one sampling time, we first study the the alternative of simply slowing down the sampling rate. For instance, consider

the continuous stirred tank reactor (CSTR) example (6.1) in in [25]. The reaction is a third order reaction $A \rightarrow B$. The model is composed of two ordinary differential equations that describe mass balance and energy balance. The states are concentration of A and temperature in the CSTR. The goal is to drive the system to its steady state from a state far from the steady state. More details of the model is presented in Chapter 6. In our simulation we assume that there is no noise in the system, and we set the sampling time as 1 *min*, 2 *mins* and 3 *mins* and compare the state profiles generated using those sampling times. The simulation horizon is 60 *mins*. The results are shown in Fig. 5.1. We use $T_{samp}$ to denote length

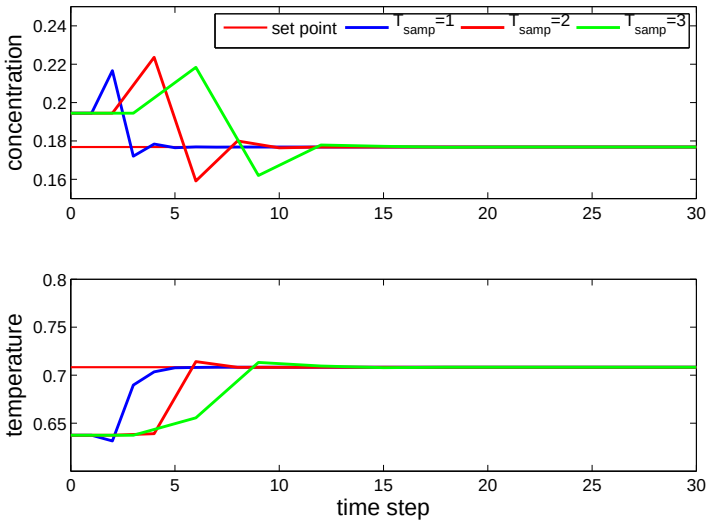



Figure 5.1: Effects of different sampling time lengths on a CSTR.

of sampling time. From Fig. 5.1 we could observe that when the sampling time is 1*min*, as the blue line shows, the state converges to the steady state within the first 5 steps. When $T_{samp} = 2$, the state converges to the steady state within the first 10 steps. With $T_{samp} = 3$, the state converges to the steady state within the first 15 steps. The convergence rate is slower with increased sampling time. Therefore, slowing down sampling is a less suitable option, as it will deteriorate the performance of the NMPC controller.

Based on asNMPC, we propose the amsNMPC method. We first define $N_s = \lceil \frac{NLP\ solution\ time}{sampling\ time} \rceil$. The idea of amsNMPC is simple: if it takes $N_s$ sampling times to solve the NLP, then we predict the state $N_s$ steps ahead and this prediction is used as initial condition to solve an NLP $N_s$ sampling times ahead of time.

We develop two variants of this strategy: the parallel approach and the serial approach. The parallel approach applies multiple processors to solve a new NLP problem at every sampling time. When the controller receives the actual state, the solution of a previous NLP problem is updated to obtain the corresponding manipulated variable, and a free processor is applied to a new NLP problem. Each time an NLP problem is solved, the processor is then freed to solve other NLP problems. Each NLP solution and corresponding NLP sensitivity is used for update *only once*. On the other hand the serial approach uses only one processor. It updates the manipulated variable every sampling time but solves the NLP problem at a lower frequency. Each NLP solution and corresponding NLP sensitivity matrix is updated *every $N_s$ sampling times*.

Before presenting the details of the parallel and serial approaches, we start with a direct extension of asNMPC, called blocked amsNMPC.

## 5.2 Blocked amsNMPC

For the following discussions we use $N_s$ to indicate the number of sampling times it takes to solve the NLP. The positions of $x$, $u$, $z$, $v$ and $\bar{x}$ are shown in Fig. 5.2. At $t_k$, we have the solution of the *previous* NLP problem, which was started at $t_{k-N_s}$ with prediction $\bar{x}(k)$ based on information at $x(k-N_s)$ and $u(k-N_s+i), i = 0,\ldots,N_s-1$. At $t_{k-N_s}$, the NLP

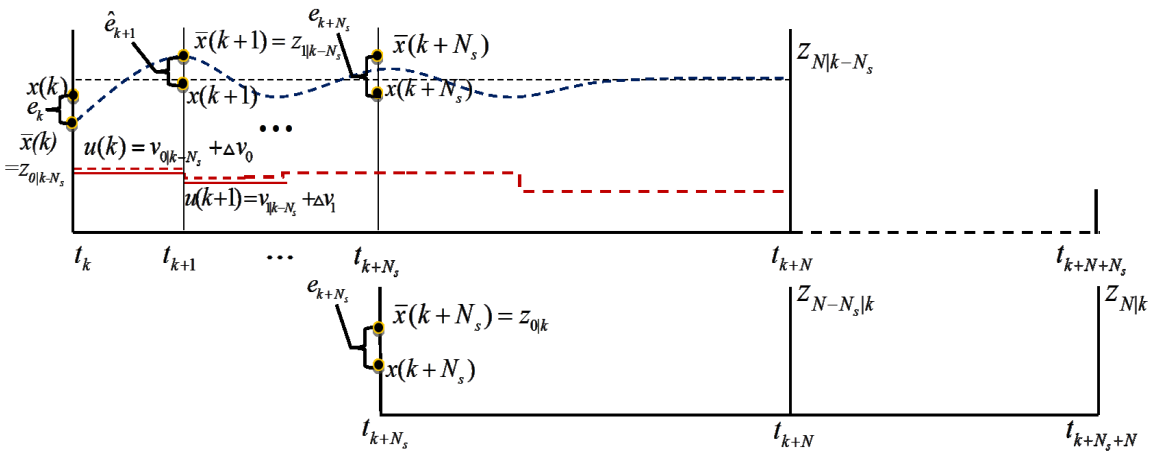



Figure 5.2: Positions of $x$, $u$, $\bar{x}$ and $v$ for blocked amsNMPC

formulation is shown by (5.1):

$$\begin{aligned} \min_{v_l, z_l} \quad & J_N := \Psi(z_N) + \sum_{l=0}^{N-1} \psi(z_l, v_l) \\ \text{s. t.} \quad & z_{l+1} = f(z_l, v_l),\ z_0 = \bar{x}(k) = z_{0|k-N_s}\ ; v_l \in \mathbb{U} \quad l = 0, \ldots N-1. \end{aligned} \tag{5.1}$$

We denote the optimal state and control variables of this problem as ($z_{l|k-N_s}$, $z_{N|k-N_s}$, $v_{l|k-N_s}$), $l = 0, 1, \ldots, N-1$, which are predicted states and controls in a horizon given state information at time $t_{k-N_s}$. Also we define the error between actual state $x(k)$ and state prediction $\bar{x}(k)$ as $e_k = x(k) - \bar{x}(k)$.

The blocked amsNMPC is a direct extension of asNMPC. The NLP (5.1) is solved every $N_s$ sampling times. At time $t_k$, when $x(k)$ is obtained, instead of updating the first control $u(k)$, as asNMPC does, the first $N_s$ controls are updated based on $e_k = x(k) - \bar{x}(k)$ and the same NLP sensitivity. Then $\bar{x}(k+N_s)$ is predicted and a new NLP is started. The blocked amsNMPC is implemented as follows:

- *On-line*: at $t_k$, having $x(k)$, update $v_{n|k-N_s}$, $n = 0, \ldots, N_s - 1$ from NLP sensitivity and $e_k = x(k) - \bar{x}(k) = p - p_0 = \Delta p$ to get $\Delta v_n = M_{z_0}^{v_n} e_k$. At $t_{k+n}$, inject the updated

$u(k+n) = v_{n|k-N_s} + \tau\Delta v_n \in \mathbb{U}$ as $u(k+n)$ to the plant. Here $\tau \in [0,1]$ is the step size when clipping is applied.

- *Background*: at $t_k$, having $x(k)$ and updated controls $u(k+n)$, predict $\bar{x}(k+N_s)$ from (5.4) as the initial value and solve the NLP problem (5.1).
- Set $k = k+N_s$ and repeat the cycle.

In blocked amsNMPC, $u(k+n), n = 0,\ldots,N_s-1$ are updated by the sensitivity matrix evaluated around $z_{n|k-N_s}$. Here $z_{n|k-N_s}$ are assigned as predictions and $u(k+n) = v_{n|k-N_s} + \tau M_{z_0}^{v_n} e_k$, $n = 0,\ldots,N_s-1$ are the updated controls, matrix $M_{z_0}^{v_n}$ is part of the inverse of KKT matrix $M$, reflecting the perturbation in $v_{n|k-N_s}$ due to the perturbation in $\Delta p = x(k) - z_{0|k-N_s}$. From (3.23) we have $\Delta s = -M^{-1}N\Delta p$, where $-M^{-1}N$ could be decomposed as $[M_z^z \;\; M_z^v \;\; M_z^\lambda \;\; M_z^v]$.

We must make it clear that when $u(k+i), i = 0,\ldots,N_s-1$ are updated, the sensitivity matrix is evaluated around $z_{i|k-N_s}$, so $z_{i|k-N_s}$ are assigned as predictions $\bar{x}(k+i)$:

$$\bar{x}(k+i) = z_{i|k-N_s} \tag{5.2}$$

If we define $F$ as the general form of evolution of states while the NLP is solved, $F$ should contain the initial state, all the controls during this period, and disturbance:$x(k+N) = F(x(k),u(k),,u(w(k)k+1),w(k+1),\ldots,u(k+N-1),w(k+N-1))$. The optimal predicted trajectory from (5.1) evolves as

$$z_{i|k-N_s} = F(\bar{x}(k), v_{0|k-N_s}, 0, v_{1|k-N_s}, 0\ldots, v_{i-1|k-N_s}, 0), i = 1,\ldots,N_s \tag{5.3}$$

The way $\bar{x}(k+N_s)$ is predicted is that when $x(k)$ is obtained, $u(k+i), i = 0,\ldots,N_s-1$ are updated, then:

$$\begin{aligned} \bar{x}(k+N_s) \;\; = \;\; & F(x(k), v_{0|k-N_s} + \tau M_{z_0}^{v_0} e_k, 0, v_{1|k-N_s} + \tau M_{z_0}^{v_1} e_k, 0, \ldots, \\ & v_{N_s-1|k-N_s} + \tau M_{z_0}^{v_{N_s-1}} e_k, 0) \end{aligned} \tag{5.4}$$

For $x(k+N_s)$, $u(k+N_s)$ is updated based on $x(k+N_s)-\bar{x}(k+N_s)=x(k+N_s)-z_{0|k}$ using a new NLP solution obtained with $\bar{x}(k+N_s)$ as initial condition.

For the new NLP problem (5.1) $z_{0|k}=\bar{x}(k+N_s)$. The predictions $\bar{x}(k)$, $\bar{x}(k+1)$ and $\bar{x}(k+N_s)$ are shown in Fig.5.2.

### 5.2.1 Nominal Stability Analysis

For nominal stability, we have $x(k)=\bar{x}(k)$ and we assume that all the states are measured. Problem (5.1) can be formulated as

$$\begin{aligned} \min_{v_l,z_l} \quad & J_N(\bar{x}(k+N_s)) := \Psi(z_N)+\sum_{l=0}^{N-1}\psi(z_l,v_l) \\ \text{s.t.} \quad & z_{l+1}=f(z_l,v_l),\ z_0=x(k+N_s),\ l=0,\ldots,N-1 \\ & v_l\in\mathbb{U} \end{aligned} \tag{5.5}$$

For the nominal case, Problem (5.5) is solved instead, and both $\Delta p=0$ and $\Delta s=0$ at all $t_k$.

To start, we refer to assumptions due to Magni and Scattolini [40]:

**Definition 19.** *A continuous function $\alpha(\cdot):\Re\to\Re$ is a $\mathscr{K}$ function if $\alpha(0)=0,\alpha(s)>0,\forall s>0$ and it is strictly increasing.*

**Assumption 3.** *(Nominal Stability Assumptions of amsNMPC)*

*(i) The terminal cost $\Psi(\cdot)$ satisfies $\Psi(x)>0$.*

*(ii) There exits a local control law $u=\kappa_f^{ams}(x)\in\mathbb{U}$ for all $x\in\mathbb{X}_f$, some unspecified terminal region, where $\Psi(f(x,\kappa_f^{ams}(x)))-\Psi(x)\le-\psi(x,\kappa_f^{ams}(x))$.*

*(iii) $\psi(x,u)$ satisfies $\alpha_p(|x|)\le\psi(x,u)\le\alpha_q(|x|)$ where $\alpha_p(\cdot)$ and $\alpha_q(\cdot)$ are $\mathscr{K}$ functions.*

According to the implementation of the blocked amsNMPC, the NLP problem is solved every $N_s$ sampling times, and solution of each NLP problem is used repeatedly until the next solution is obtained. We note that $x(k+N_s) = z_{0|k}$, and that $(z_{N_s+i|k-N_s}, v_{N_s+i|k-N_s})$, $i = 0,\ldots,N-1$ from $J_N(x(k))$ are feasible values for $(z_i, v_i)$, $i = 0,\ldots,N-1$ in $J_N(x(k+N_s))$. This allows us to compare the objective function values of these two NLP problems:

$$\begin{aligned} & J_N(x(k)) - J_N(x(k+N_s)) \\ \geq\ & \Psi(z_N) + \sum_{l=0}^{N-1} \psi_l(z_l, v_l) - \Psi(z_{N+N_s}) - \sum_{l=N_s}^{N_s+N-1} \psi_l(z_l, v_l) \\ =\ & \Psi(z_N) - \Psi(z_{N+N_s}) + \sum_{l=0}^{N_s-1} \psi_l(z_l, v_l) - \sum_{l=N}^{N_s+N-1} \psi_l(z_l, v_l) \end{aligned} \tag{5.6}$$

Using Assumption (3)(ii), we can show

$$\begin{aligned} \Psi(z_N) - \Psi(z_{N+N_s}) &= \sum_{i=1}^{N_s} (\Psi(z_{N+i-1}) - \Psi(z_{N+i})) \\ &\geq \sum_{i=1}^{N_s} \psi_{N+i-1}(z_{N+i-1}, v_{N+i-1}) \end{aligned} \tag{5.7}$$

Combining (5.6) and (5.7) leads to

$$J_N(x(k)) - J_N(x(k+N_s)) \geq \sum_{l=0}^{N_s-1} \psi_l(z_l, v_l) = \sum_{l=0}^{N_s-1} \psi_l(x(k+l), u(k+l)) \tag{5.8}$$

Summing over $k = 0,\ldots,\infty$ leads to

$$\begin{aligned} \sum_{k=0}^{N_s-1} J_N(x(k)) &\geq \sum_{k=0}^{\infty} J_N(x(k)) - J_N(x(k+N_s)) \\ &\geq \sum_{k=0}^{N_s-1} (k+1)\psi(x(k), u(k)) + N_s \sum_{k=N_s}^{\infty} \psi(x(k), u(k)) \end{aligned} \tag{5.9}$$

which implies $\lim_{k\to\infty} \psi(x(k), u(k)) = 0$, which leads to the following result.

**Theorem 6.** *(Nominal Stability of blocked amsNMPC) Consider the moving horizon problem (5.5) and associated control law $u = \kappa^{ams}(x)$ that satisfies Assumption 3. Then, $J_N(x(k))$ is a Lyapunov function and the closed-loop system is asymptotically stable.*

### 5.2.2 Robust Stability Analysis

For amsNMPC approach we consider the following system:

$$x(k+1) = f(x(k),u(k)) + w(k) \tag{5.10}$$

where $w(k)$ is additive disturbance. In order for the system to be robustly stable, we have the following assumptions:

**Assumption 4.** *(Robust Stability Assumption of Blocked amsNMPC)*

1. *$w(k)$ is bounded, $|w(k)| \leq w_{max}$.*

From time $t_k$ to $t_{k+N_s}$, the actual state trajectory evolves as below:

$$\begin{aligned}
x(k+1) &= f(x(k),u(k)) + w(k) = f(x(k), v_{0|k-N_s} + \tau M_{z_0}^{v_0} e_k) + w(k) \\
x(k+2) &= f(x(k+1),u(k+1)) + w(k+1) \\
&= f(f(x(k), v_{0|k-N_s} + \tau M_{z_0}^{v_0} e_k) + w(k), v_{1|k-N_s} + \tau M_{z_0}^{v_1} e_k) + w(k+1) \\
x(k+n) &= F(x(k), v_{0|k-N_s} + \tau M_{z_0}^{v_0} e_k, w(k), v_{1|k-N_s} + \tau M_{z_0}^{v_1} e_k, w(k+1), ..., \\
&\quad v_{n-1|k-N_s} + \tau M_{z_0}^{v_{n-1}} e_k, w(k+n-1)) \\
&\ldots \\
x(k+N_s) &= F(x(k), v_{0|k-N_s} \tau M_{z_0}^{v_0} e_k, w(k), v_{1|k-N_s} + \tau M_{z_0}^{v_1} e_k, w(k+1), ..., \\
&\quad v_{N_s-1|k-N_s} + \tau M_{z_0}^{v_{N_s-1}} e_k, w(k+N_s-1))
\end{aligned} \tag{5.11}$$

Next we start from $x(k)$ and show how error accumulates from $t_k$ to $t_{k+N_s}$. At $t_k$ we have

$$x(k+1) = f(x(k),u(k)) + w(k) \tag{5.12}$$

and

$$e_k = x(k) - \bar{x}(k) \tag{5.13}$$

Applying Taylor expansion around $(z_{0|k-N_s}, v_{0|k-N_s})$ from (5.3), we get

$$\begin{aligned} x(k+1) &= z_{1|k-N_s} + \hat{f}_{z_0}(z_{0|k-N_s}, v_{0|k-N_s})^T (x(k) - z_{0|k-N_s}) \\ &\quad + \hat{f}_{v_0}(z_{0|k-N_s}, v_{0|k-N_s})^T (u(k) - v_{0|k-N_s}) + w(k) \\ &= z_{1|k-N_s} + \hat{f}_{z_0}(z_{0|k-N_s}, v_{0|k-N_s})^T e_k \\ &\quad + \hat{f}_{v_0}(z_{0|k-N_s}, v_{0|k-N_s})^T \tau M_{z_0}^{v_0} e_k + w(k) \end{aligned} \tag{5.14}$$

where

$$\hat{f}_\xi = \int_0^1 \frac{df(z,v)}{d\xi}[p(k) + t(q(k) - p(k))]^T (q(k) - p(k)) dt \tag{5.15}$$

$$q(k) = [x(k) \;\; u(k)]^T \tag{5.16}$$

$$p(k) = [z_{0|k-N_s} \;\; v_{0|k-N_s}]^T \tag{5.17}$$

Thus we have

$$\begin{aligned} \hat{e}_{k+1} &= x(k+1) - z_{1|k-N_s} \\ &= \hat{f}_{z_0}(z_{0|k-N_s}, v_{0|k-N_s})^T e_k + \hat{f}_{v_0}(z_{0|k-N_s}, v_{0|k-N_s})^T \tau M_{z_0}^{v_0} e_k + w(k) \end{aligned} \tag{5.18}$$

Similarly

$$x(k+2) = f(z_{1|k-N_s} + \hat{e}_{k+1}, v_{1|k-N_s} + \tau M_{z_0}^{v_1} e_k) + w(k+1) \tag{5.19}$$

$$\begin{aligned} \hat{e}_{k+2} &= x(k+2) - z_{2|k-N_s} \\ &= \hat{f}_{z_1} \hat{e}_{k+1} + \hat{f}_{v_1} \tau M_{z_0}^{v_1} e_k + w(k+1) \\ &= \hat{f}_{z_1}(\hat{f}_{z_0} e_k + \hat{f}_{v_0} \tau M_{z_0}^{v_0} e_k + w(k)) + \hat{f}_{v_1} \tau M_{z_0}^{v_1} e_k + w(k+1) \\ &= [\hat{f}_{z_1} \hat{f}_{z_0} + \hat{f}_{z_1} \hat{f}_{v_0} \tau M_{z_0}^{v_0} + \hat{f}_{v_1} \tau M_{z_0}^{v_1}] e_k + \hat{f}_{z_1} w(k) + w(k+1) \end{aligned} \tag{5.20}$$

$$\begin{aligned} \hat{e}_{k+3} &= \hat{f}_{z_2} \hat{e}_{k+2} + \hat{f}_{v_2} \tau M_{z_0}^{v_2} e_k + w(k+2) \\ &= \hat{f}_{z_2}[(\hat{f}_{z_1} \hat{f}_{z_0} + \hat{f}_{z_1} \hat{f}_{v_0} \tau M_{z_0}^{v_0} + \hat{f}_{v_1} \tau M_{z_0}^{v_1}) e_k + \hat{f}_{z_1} w(k) + w(k+1)] + \hat{f}_{v_2} \tau M_{z_0}^{v_2} e_k + w(k+2) \\ &= (\hat{f}_{z_2} \hat{f}_{z_1} \hat{f}_{z_0} + \hat{f}_{z_2} \hat{f}_{z_1} \hat{f}_{v_0} \tau M_{z_0}^{v_0} + \hat{f}_{z_2} \hat{f}_{v_1} \tau M_{z_0}^{v_1} + \hat{f}_{v_2} \tau M_{z_0}^{v_2}) e_k \\ &\quad + \hat{f}_{z_2} \hat{f}_{z_1} w(k) + \hat{f}_{z_2} w(k+1) + w(k+2) \end{aligned} \tag{5.21}$$

...

$$\hat{e}_{k+n} = (\prod_{i=0}^{n-1} \hat{f}_{z_i} + \sum_{i=0}^{n-1} (\prod_{j=i+1}^{n-1} \hat{f}_{z_j}) \hat{f}_{v_i} \tau M_{z_0}^{v_i}) e_k + \sum_{i=0}^{n-1} (\prod_{j=i+1}^{n-1} \hat{f}_{z_j}) w(k+i) \quad (5.22)$$

where $n = 1, 2, ..., N_s - 1$. If we compare (5.4) with (5.11), we could see that the only difference is that (5.4) does not include $w(k+i)$. So $e_k$ and $e_{k+N_s}$ should evolve as

$$x(k) - \bar{x}(k) = e_k = \sum_{i=0}^{N_s-1} (\prod_{j=i+1}^{N_s-1} \hat{f}_{z_{j-N_s}}) w(k+i-N_s), \quad (5.23)$$

$$x(k+N_s) - \bar{x}(k+N_s) = e_{k+N_s} = \sum_{i=0}^{N_s-1} (\prod_{j=i+1}^{N_s-1} \hat{f}_{z_j}) w(k+i) \quad (5.24)$$

respectively. Note that the notations of $e_k$, $e_{k+N_s}$ and the errors in the states between $x(k)$ and $x(k+N_s)$ (e.g., $\hat{e}_{k+n}$, $n = 1, 2, ..., N_s - 1$) are different. This is because $\bar{x}(k)$ and $\bar{x}(k+N_s)$ are predicted differently from $z_{n|k-N_s}, n = 1, ..., N_s - 1$; $\bar{x}(k+n)$ are the optimal predicted states taken from the optimal solution directly; while $\bar{x}(k)$ and $\bar{x}(k+N_s)$ are the integral from the current state with *updated* controls from the most recently solved NLP $N_s$ steps ago, as stated in (5.4). Correspondingly, $z_{0|k-N_s} = \bar{x}(k)$ and $z_{0|k} = \bar{x}(k+N_s)$ are the initial conditions of two subsequent NLPs. At time $t_k$, the optimal solution of the NLP with $\bar{x}(k)$ as initial condition is obtained, the sensitivity update is done around the *optimal solution* to get $u(k+n), n = 1, ..., N_s - 1$. Therefore the predictions for $x(k+n), n = 1, ..., N_s - 1$ are the optimal states $z_{n|k-N_s}$; whereas $u(k+N_s)$ is not obtained in this iteration. Instead, $x(k+N_s)$ is predicted to start a new NLP problem at $t_k$, and $u(k+N_s)$ is obtained by updating the control in the first interval of the new NLP solution.

By Lipschitz continuity of (5.22), we have

$$|\hat{e}_{k+n}| \leq L_e |e_k| + \sum_{i=0}^{n-1} (\prod_{j=i+1}^{n-1} \hat{f}_{z_j}) w(k+i) \quad (5.25)$$

where $L_e$ satisfies

$$|(\prod_{i=0}^{n-1} \hat{f}_{z_i} + \sum_{i=0}^{n-1} (\prod_{j=i+1}^{n-1} \hat{f}_{z_j}) \hat{f}_{v_i} \tau M_{z_0}^{v_i}) \leq L_e \quad (5.26)$$

Similarly we could find an upper bound $L_{gi}$ for $|\prod_{j=i+1}^{n-1} \hat{f}_{z_j}|$ such that

$$|\hat{e}_{k+n}| \leq L_e|e_k| + \sum_{i=0}^{n-1} L_{gi}|w(k+i)| \tag{5.27}$$

where $n = 1,...,N_s-1$ and

$$|e_{k+N_s}| \leq \sum_{i=0}^{N_s-1} L_{gi}|w(k+i)| \tag{5.28}$$

For the analysis of robust stability of the blocked ideal NMPC, it is necessary to include the effect of NLP sensitivity errors. In the nominal case, $w(k+n) = 0, n = 0,...,N_s-1$, the forward simulation $\bar{x}(k+N_s) = F(x(k),u(k),0,u(k+1),0,...,u(k+N_s-1),0)$ would lead to the control action $\bar{u}^{id}(k+N_s+n) = \kappa^{id}(\bar{x}(k+N_s+n)) = \kappa^{ams}(\bar{x}(k+N_s+n)), n = 0,...,N_s-1$, where $u(k+N_s+n) = \kappa^{ams}(\bar{x}(k+N_s+n))$ is the control action of amsNMPC with $\bar{x}(k+N_s)$ as initial condition. $\kappa^{id}(\bar{x}(k+N_s+i))$ would then be used to start Problem (5.1). As shown in Figure 5.2 we can write:

$$\begin{aligned} J^{id}(\bar{x}(k+N_s)) &:= J_N(\bar{x}(k+N_s), \kappa^{id}(\bar{x}(k+N_s)), \kappa^{id}(\bar{x}(k+N_s+1)),..., \kappa^{id}(\bar{x}(k+2N_s-1))) \\ &= \Psi(z_{N|k}) + \sum_{l=0}^{N-1} \psi(z_{l|k}, v_{l|k}) \end{aligned} \tag{5.29}$$

while $x(k)$ and $\kappa^{ams}(x(k+n))$ would then be used to start Problem (5.1) for the prediction of $\bar{x}(k+N_s)$ with the cost

$$J^{ams}(x(k)) = \Psi(z_{N-N_s|k}) + \psi(x(k),u^{ams}(k)) + \sum_{l=1}^{N_s-1} \psi(\bar{x}(k+l),u^{ams}(k+l)) + \sum_{l=0}^{N-N_s-1} \psi(z_{l|k},v_{l|k}) \tag{5.30}$$

In the first step of the blocked amsNMPC algorithm, $\bar{x}(k+n), n = 1,\ldots,N_s$ is generated according to the following predicted states, controls and errors:

$$\begin{aligned} \bar{x}(k+n) &= F(x(k), v_{0|k-N_s} + \tau M_{z_0}^{v_0} e_k, 0, v_{1|k-N_s} + \tau M_{z_0}^{v_1} e_k, 0, ..., v_{n-1|k-N_s} + \tau M_{z_0}^{v_{n-1}} e_k, 0) \\ u^{ams}(k+n) &= v_{0|k-N_s} + \tau M_{z_0}^{v_n} e_k \\ e_{k+n} &= x(k+n) - \bar{x}(k+n) = \sum_{i=0}^{n-1} (\prod_{j=i+1}^{N_s-1} \hat{f}_{z_j}) w(k+i), \quad n = 1,\ldots N_s. \end{aligned} \tag{5.31}$$

At $t_{k+N_s}$, the plant further evolves with disturbances generating state $x(k+N_s)$. With ideal NMPC, $x(k+N_s)$ would lead to the control action $\kappa^{id}(x(k+N_s+n)), n=0,...,N_s-1$ which is used to start Problem (5.1) with cost

$$J^{id}(x(k+N_s)) := J_N(x(k+N_s), \kappa^{id}(x(k+N_s)), \kappa^{id}(x(k+N_s+1)), ..., \kappa^{id}(x(k+2N_s-1))).$$

With amsNMPC, $x(k+N_s)$ would lead to the control action $\kappa^{ams}(x(k+N_s+n))$ and thus the new Problem (5.1) with cost

$$J^{ams}(x(k+N_s)) := J_N(x(k+N_s), \kappa^{ams}(x(k+N_s)), \kappa^{ams}(x(k+N_s+1)), ..., \kappa^{ams}(x(k+2N_s-1))).$$

The error of blocked amsNMPC is composed of two parts: prediction error, which comes from the difference between actual states and their predictions, and sensitivity error, which is due to discarding higher order terms $O(|p-p_0|^2)$. We define these two kinds of errors at $t_k$ as:

$$\varepsilon_s(k+N_s) := J^{id}(x(k+N_s)) - J^{id}(\bar{x}(k+N_s))$$

and

$$\varepsilon_{ams}(k+N_s) := J^{ams}(x(k+N_s)) - J^{id}(x(k+N_s)).$$

**Lemma 1.** *There exist positive Lipschitz constants $L_z$ and $L_\kappa$ such that $\forall x \in \mathbb{X}$,*

$$\begin{aligned} |\varepsilon_s(x(k+N_s))| &\leq L_z(|x(k+N_s) - \bar{x}(k+N_s)| + |\kappa^{id}(x(k+N_s)) - \kappa^{id}(\bar{x}(k+N_s)|) \\ &\leq L_z(1+L_\kappa)|x(k+N_s) - \bar{x}(k+N_s)| = L_z(1+L_\kappa)|e_{k+N_s}| \end{aligned} \tag{5.32}$$

*Proof.* According to the implementation of blocked ideal NMPC, $\kappa^{id}(x(k+N_s))$ is achieved from $s^*(x(k+N_s))$, while $\kappa^{id}(\bar{x}(k+N_s))$ is achieved from $s^*(\bar{x}(k+N_s))$. According to Theorem 2(1), there exists $L_\kappa$ such that $|\kappa^{id}(x(k+N_s)) - \kappa^{id}(\bar{x}(k+N_s))| \leq L_\kappa|(x(k+N_s) - \bar{x}(k+N_s))|$. Using Theorem 2 (2) leads to the result for Lemma 1. □

Using Theorem 2(3), the following lemma follows:

**Lemma 2.** *There exist positive Lipschitz constants $L_z$ and $L_\kappa^{ams}$ such that $\forall x \in \mathbb{X}$,*

$$\begin{aligned} |\varepsilon_{ams}(x(k+N_s))| &\leq L_z(|x(k+N_s)-x(k+N_s)|+|\kappa^{ams}(x(k+N_s))-\kappa^{id}(x(k+N_s))|) \\ &\leq L_z L_\kappa^{ams}|x(k+N_s)-\bar{x}(k+N_s)|^2 = L_z L_\kappa^{ams}|e_{k+N_s}|^2 \end{aligned} \tag{5.33}$$

*Proof.* If we use $\hat{s}(p)$ to denote the estimate of perturbed optimal solution with parameter $p$, due to sensitivity, we have

$$\hat{s}(p) = s^*(p_0) + \frac{\partial s^*}{\partial p}(p-p_0) \tag{5.34}$$

while the accurate optimal solution satisfies

$$s^*(p) = s^*(p_0) + \frac{\partial s^*}{\partial p}(p-p_0) + O(|p-p_0|^2) \tag{5.35}$$

Subtracting (5.35) from (5.34), we have

$$|\hat{s}(p)-s^*(p)| = O(|p-p_0|^2) \tag{5.36}$$

In the NLP formulation of the blocked amsNMPC, $p = (x(k+N_s), \kappa^{ams}(x(k+N_s)))$, $p_0 = (x(k+N_s), \kappa^{id}(x(k+N_s)))$, $\kappa^{ams}(x(k+N_s))$ is a subvector of $\hat{s}(x(k+N_s))$ and $\kappa^{id}(x(k+N_s))$ is a subvector of $s^*(x(k+N_s))$. So there exists a constant $L_\kappa^{ams}$ such that

$$|\kappa^{ams}(x(k+N_s))-\kappa^{id}(x(k+N_s))| \leq L_\kappa^{ams}|x(k+N_s)-\bar{x}(k+N_s)|^2 = L_\kappa^{ams}|e_{k+N_s}|^2 \tag{5.37}$$

□

Due to the implementation of the blocked amsNMPC, between $t=k$ and $t=k+N_s$, only two NLPs need be considered.

$$\begin{aligned} J^{ams}(x(k+N_s))-J^{ams}(x(k)) &= J^{id}(\bar{x}(k+N_s))-J^{ams}(x(k))+J^{id}(x(k+N_s))-J^{id}(\bar{x}(k+N_s)) \\ &\quad +J^{ams}(x(k+N_s))-J^{id}(x(k+N_s)) \\ &= J^{id}(\bar{x}(k+N_s))-J^{ams}(x(k))+\varepsilon_s(x(k+N_s))+\varepsilon_{ams}(x(k+N_s)) \end{aligned} \tag{5.38}$$

From Figure 5.2 we see the solution of $J^{ams}(x(k))$ is feasible for $J^{id}(\bar{x}(k+N_s))$ and we compare the objective functions (5.30) and (5.29) as follows:

$$\begin{aligned} J^{id}(\bar{x}(k+N_s)) - J^{ams}(x(k)) &\leq (\Psi(z_{N|k}) - \Psi(z_{N-N_s|k}) + \sum_{l=0}^{N_s-1} \psi(z_{N-N_s+l|k}, v_{N-N_s+l|k})) \\ &\quad - \psi(x(k), u^{ams}(k)) - \sum_{l=1}^{N_s-1} \psi(\bar{x}(k+l), u^{ams}(k+l)) \\ &\leq \sum_{l=0}^{N_s-1} [\Psi(z_{N-N_s+l+1|k}) - \Psi(z_{N-N_s+l|k}) + \psi(z_{N-N_s+l|k}, v_{N-N_s+l|k})] \\ &\quad - \psi(x(k), u^{ams}(k)) - \sum_{l=1}^{N_s-1} \psi(\bar{x}(k+l), u^{ams}(k+l)) \end{aligned}$$

From Assumption 1, $\Psi(z_{l+1}) - \Psi(z_l) + \psi(z_l, v_l) \leq 0$ for $z_l \in \mathbb{X}_f$ and we have:

$$\begin{aligned} J^{id}(\bar{x}(k+N_s)) - J^{ams}(x(k)) &= -\psi(x(k), u^{ams}(k)) - \sum_{l=1}^{N_s-1} \psi(\bar{x}(k+l), u^{ams}(k+l)) \qquad (5.39) \\ &= -\psi(x(k), u^{ams}(k)) - \sum_{l=1}^{N_s-1} \psi(x(k+l) - e_{k+l}, u^{ams}(k+l)) \\ &\leq -\psi(x(k), u^{ams(k)}) - \sum_{l=1}^{N_s-1} (\psi(x(k+l), u^{ams}(k+l)) - L_z e_{k+l}) \\ &\leq \sum_{l=0}^{N_s-1} -\psi(x(k+l), u^{ams}(k+l)) + \sum_{i=0}^{l-1} L_z L_f |w(k+i)| \qquad (5.40) \end{aligned}$$

where the last inequality follows from (5.28) and (5.31). Combining (5.40) with Lemma 1 and Lemma 2, (5.38) becomes

$$\begin{aligned} & J^{ams}(x(k+N_s)) - J^{ams}(x(k)) \\ = & \sum_{l=0}^{N_s-1} -\psi(x(k+l), u^{ams}(k+l)) + L_z(1 + L_\kappa + L_\kappa^{ams}|e_{k+N_s}|)|e_{k+N_s}| + \sum_{l=1}^{N_s-1} L_z e_{k+l} \end{aligned} \qquad (5.41)$$

Combining with (5.28), we have

$$\begin{aligned}
& J^{ams}(x(k+N_s)) - J^{ams}(x(k)) \\
\leq & \sum_{l=0}^{N_s-1} -\psi(x(k+l), u^{ams}(k+l)) + L_f(\sum_{n=0}^{N_s-1} |w(k+n)|)[L_z(1+L_\kappa) + L_z L_\kappa^{ams}(\sum_{n=0}^{N_s-1} L_f|w(k+n)|)] \\
& + \sum_{l=0}^{N_s-1}\sum_{n=0}^{l-1} L_f L_z|w(k+n)|) \\
= & \sum_{l=0}^{N_s-1} \{-\psi(x(k+l), u^{ams}(k+l)) + L_z L_f(L_\kappa + N_s - l))|w(k+l)|\} \\
& + L_z L_\kappa^{ams}(L_f \sum_{i=0}^{N_s-1} |w(k+i)|)^2. \qquad (5.42)
\end{aligned}$$

This constitutes a blocked form of the ISS property, where the stage costs and noise are written in terms of blocks, instead of individual sampling times. Using (5.42) and Definition 16 leads to the following result.

**Theorem 7.** *(Robust stability of the blocked amsNMPC approach) For the moving horizon problem (5.1) and associated control law $u = \kappa^{ams}(x)$ that satisfies Assumption 3 and Assumption 4 is an ISS-Lyapunov function and the closed-loop system is robustly stable.*

## 5.3 Serial approach

The serial approach is an extension of asNMPC. Instead of predicting the *next* state, solving an NLP problem *one* step in advance and updating only the *first* input within each horizon, it predicts the state $N_s$ sampling times ahead, solves an NLP problem $N_s$ sampling times in advance, and updates the *first* $N_s$ inputs within each horizon. It should be noticed that in the formulation of the NLP problem, the initial constraint $z_{0|k-N_s} = \bar{x}(k)$ is included, and $\bar{x}(k)$ is considered a parameter. In the KKT matrix, the row related to this initial constraint is a unit vector, whose elements corresponding to $z_{0|k-N_s}$ have the value of $I$.

The positions of $x$, $u$, $\bar{x}$, $z$ and $v$ of the serial approach are shown in Fig. 5.3. Solid lines are for the actual control profiles, while dashed lines are for predicted state and control profiles. Dot-dashed lines indicate solutions of different NLP problems (5.1).

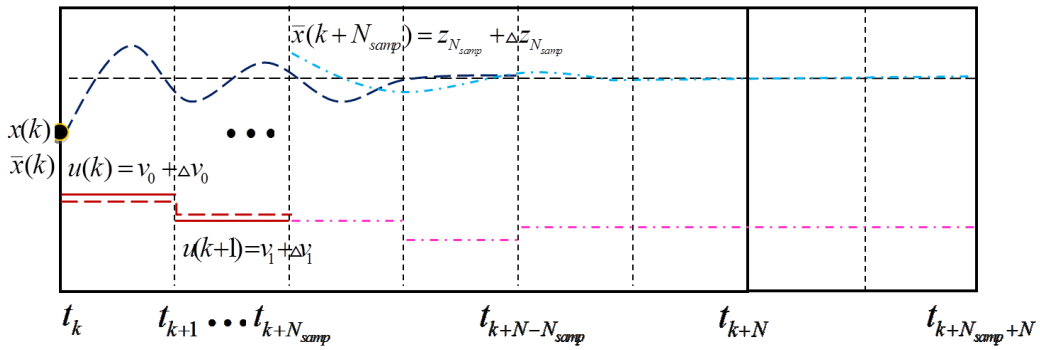



Figure 5.3: Positions of $x$, $u$, $\bar{x}$, $z$ and $v$ for the serial approach

When the measurement is obtained, $z_0$ changes from $\bar{x}(k)$ to $x(k)$, the KKT matrix $M$ is directly applied to update the first input. This is not the case when it comes to the other states within the same horizon, because they are not considered as parameters in the NLP formulation; thus there are no corresponding unit elements in the KKT matrix. As a result, when the other inputs are updated, besides current NLP sensitivity, additional constraints need to be added. Those constraints are reflected as additional columns and rows of the KKT matrix. In other words, when the inputs other than the first are updated, the KKT matrix needs to be updated.

### 5.3.1 Updating NLP Sensitivity

When the $i^{th}$ input is updated ($i \neq 1$), $p_0 + \Delta p = x(k)$ is determined implicitly by an additional constraint $v_{i|k-N_s} + \Delta v_i = x(k+i)$. This constraint is treated through adding unit elements to the rows and columns of the KKT matrix, and through adding additional con-

ditions to the right hand side. The system (3.23) is reformulated as:

$$\left[\begin{array}{c|c} M & E_{1i} \\ \hline E_{2i} & 0 \end{array}\right] \left[\begin{array}{c} \Delta s \\ \hline \Delta p \end{array}\right] = - \left[\begin{array}{c} 0 \\ \hline r_i \end{array}\right] \tag{5.43}$$

where $r_i = x(k+i) - z_i$, $E_{2i}$ is matrix whose element corresponding to $z_i$ has the value of $i$ while other elements are 0, and $E_{1i} = E_{2i}^T$. System (5.43) is solved using the Schur complement method developed in [48]:

$$M\Delta s = -E_{1i}(E_{2i}M^{-1}E_{1i})^{-1}r_i \tag{5.44}$$

Thus we can calculate the approximation of the perturbed solution using

$$\tilde{s}(x(k+i)) = s^*(z_i) + \Delta s(x(k+i)) \tag{5.45}$$

and the updated manipulated variable is contained in the perturbed solution vector $\tilde{s}(x(k+i))$.

### 5.3.2 Implementation

Suppose the optimal solution of the last NLP problem is known at $t_k$. Knowing $x(k)$, $v_{0|k-N_s}$ is updated using (3.23) and injected into the plant as $u(k)$, and $\bar{x}(k+N_s)$ is predicted using (5.4). Then between $t_k$ and $t_{k+N_s}$, Problem (5.1) is solved in background using $\bar{x}(k+N_s)$ as the initial value. In the meantime, the current manipulated variables $v_{i|k-N_s}, i = 1,2,...,N_s-1$ are updated online using the sensitivity $(s^*(z_{i|k-N_s}) + \Delta s(x(k+i)))$ based on solution of the previous NLP problem and (5.43). The serial approach works as follows:

- *Background*: At $t_k$, having $x(k)$ and $u(k)$, update $v_{0|k-N_s}$ and $z_{n|k-N_s}, n = 1,...,N_s$ using $e_k = x(k) - \bar{x}(k)$. Evaluate $\bar{x}(k+N_s)$ by (5.4) and solve Problem (5.1) between $t_k$ and $t_{k+N_s}$ with $\bar{x}(k+N_s)$ as the initial value.

- *On-line*: for $n = 1 : N_s - 1$, At $t_{k+n}$, having $x(k+n)$, update the augmented sensitivity system in (5.43) and update $v_{n|k-N_s}$ using $\hat{e}_{k+n} = x(k+n) - z_{n|k-N_s}$. Inject the updated $u(k+n) = v_{n|k-N_s} + M_{z_n}^{v_n}(x(k+n) - z_{n|k-N_s})$ to the plant.
- Set $k = k + N_s$ and repeat the cycle.

### 5.3.3 Nominal Stability Analysis

For the nominal case, Problem (5.5) is solved instead, and both $\Delta p = 0$ and $\Delta s = 0$ at all $t_k$. According to the implementation of the serial approach, the NLP problem is solved every $N_s$ sampling times, and solution of each NLP problem is used repeatedly until the next solution is obtained. In addition to the objective function $J_N(x(k))$, we define a modified objective function given by:

$$\hat{J}_{N-j}(x(k)) = J_N(x(k)) - \sum_{l=0}^{j-1} \psi(z_l, v_l) \tag{5.46}$$

$\hat{J}_{N-j}(x(k))$ is obtained from solution of (5.5) with the optimal cost $J_N(x(k))$ and subtracting the first $j$ stage costs from it. When the $i^{th}$ manipulated variable $v_i$ is injected to the plant, $\hat{J}_{N+1-i}(x(k))$ is the objective function corresponding to the states and manipulated variables within a shrinking horizon starting from $k+i-1$.

It is obvious that

$$\begin{aligned} J_N(x(k)) - \hat{J}_{N-1}(x(k)) &= \psi(z_0, v_0) \\ \hat{J}_{N-j}(x(k)) - \hat{J}_{N-j-1}(x(k)) &= \psi(z_j, v_j), j = 1, \ldots, N_s - 2 \end{aligned} \tag{5.47}$$

Next we compare $\hat{J}_{N-N_s+1}(x(k))$ and $J_N(x(k+N_s))$.

$$\begin{aligned} & \hat{J}_{N-N_s+1}(x(k)) - J_N(x(k+N_s)) \\ = \; & J_N(x(k)) - \sum_{l=0}^{N_s-2} \psi(z_l, v_l) - J_N(x(k+N_s)) \end{aligned} \tag{5.48}$$

As in the parallel approach, $(z_{N_s+i}(x(k)), v_{N_s+i}(x(k)))$, $i = 0,\ldots,N-N_s-1$ from $J_N(x(k))$ are feasible values for $(z_i(x(k+N_s)), v_i(x(k+N_s)))$, $i = 0,\ldots,N-N_s-1$ in $J_N(x(k+N_s))$. Hence, from (5.8) we have

$$J_N(x(k)) - J_N(x(k+N_s)) \geq \sum_{l=0}^{N_s-1} \psi(z_l, v_l)$$

Substituting into (5.48) we get

$$\begin{aligned} & \hat{J}_{N-N_s+1}(x(k)) - J_N(x(k+N_s)) \\ \geq & \sum_{l=0}^{N_s-1} \psi(z_l, v_l) - \sum_{l=0}^{N_s-2} \psi(z_l, v_l) \\ = & \psi(z_{N_s-1}, v_{N_s-1}) \end{aligned} \tag{5.49}$$

Therefore both $J_N(x(k))$ and $\hat{J}_{N-j}(x(k))$ satisfy the conditions of a Lyapunov function, Assumption 3(ii) leads to $\lim_{k\to\infty} x(k) = 0$ and the following nominal stability property follows for the serial approach.

**Theorem 8.** *(Nominal Stability of the Serial Approach) Consider the moving horizon problem (5.86) and associated control law $u = \kappa^{ams}(x)$ that satisfies Assumption 3. Then, $J_N(x(k))$ and $\hat{J}_{N-j}(x(k))$ are Lyapunov functions and the closed-loop system is asymptotically stable.*

### 5.3.4 Robust Stability Analysis

We start with the case where clipping is not needed. Compared with blocked amsNMPC, the optimal NLP trajectory (5.3) does not change, neither do the predictions of $z_{n|k-N_s}, n = 1,\ldots,N_s$. The only difference from (5.11) lies in the evolution of actual trajectory:

$$\begin{aligned} x(k+n) = & F(x(k), v_{0|k-N_s} + M_{z_0}^{v_0} e_k, w(k), v_{1|k-N_s} + M_{z_1}^{v_1} \hat{e}_{k+1}, w(k+1), \ldots, \\ & v_{n-1|k-N_s} + M_{z_{n-1}}^{v_{n-1}} \hat{e}_{k+n-1}, w(k+n-1)) \end{aligned} \tag{5.50}$$

Starting from $\hat{e}_{k+1}$, we have

$$\begin{aligned}
\hat{e}_{k+1} &= \hat{f}_{z_0}(z(k),v(k))e_k + \hat{f}_{v_0}(z(k),v(k))M_{z_0}^{v_0}e_k + w(k) = (\hat{f}_{z_0} + \hat{f}_{v_0}M_{z_0}^{v_0})e_k + w(k) \\
\hat{e}_{k+2} &= \hat{f}_{z_1}\hat{e}_{k+1} + \hat{f}_{v_1}M_{z_1}^{v_1}\hat{e}_{k+1} + w(k+1) \\
&= (\hat{f}_{z_1} + \hat{f}_{v_1}M_{z_1}^{v_1})[(\hat{f}_{z_0} + \hat{f}_{v_0}M_{z_0}^{v_0})e_k + w(k)] + w(k+1) \\
&= (\hat{f}_{z_1} + \hat{f}_{v_1}M_{z_1}^{v_1})(\hat{f}_{z_0} + \hat{f}_{v_0}M_{z_0}^{v_0})e_k + (\hat{f}_{z_1} + \hat{f}_{v_1}M_{z_1}^{v_1})w(k) + w(k+1) \\
\hat{e}_{k+3} &= \hat{f}_{z_2}\hat{e}_{k+2} + \hat{f}_{v_2}M_{z_2}^{v_2}\hat{e}_{k+2} + w(k+2) \\
&= (\hat{f}_{z_2} + \hat{f}_{v_2}M_{z_2}^{v_2})\{(\hat{f}_{z_1} + \hat{f}_{v_1}M_{z_1}^{v_1})[(\hat{f}_{z_0} + \hat{f}_{v_0}M_{z_0}^{v_0})e_k + w(k)] + w(k+1)\} + w(k+2) \\
&= \prod_{i=0}^{2}(\hat{f}_{z_i} + \hat{f}_{v_i}M_{z_i}^{v_i})e_k + \prod_{i=1}^{2}(\hat{f}_{z_i} + \hat{f}_{v_i}M_{z_i}^{v_i})w(k) + (\hat{f}_{z_2} + \hat{f}_{v_2}M_{z_2}^{v_2})w(k+1) + w(k+2) \\
&\dots \\
\hat{e}_{k+n} &= \prod_{i=0}^{n-1}(\hat{f}_{z_i} + \hat{f}_{v_i}M_{z_i}^{v_i})e_k + \sum_{i=0}^{n-1}\prod_{j=i+1}^{n-1}(\hat{f}_{z_j} + \hat{f}_{v_j}M_{z_j}^{v_j})w(k+i) \qquad (5.51)
\end{aligned}$$

where $n = 1,\dots,N_s-1$.

Comparing (5.4) and (5.50), we get

$$e_{k+N_s} = \sum_{i=0}^{N_s-1}\prod_{j=i+1}^{N_s-1}(\hat{f}_{z_j} + \hat{f}_{v_j}M_{z_j}^{v_j})w(k+i) - \sum_{i=1}^{N_s-1}\hat{f}_{v_i}M_{z_0}^{v_i}\prod_{j=i+1}^{N_s-1}(\hat{f}_{z_j} + \hat{f}_{v_j}M_{z_j}^{v_j})e_k \qquad (5.52)$$

Similarly as with the blocked amsNMPC approach, we could find $L_e^S$ and $L_{\hat{e}}^S$ such that

$$|\prod_{n=0}^{n-1}(\hat{f}_{z_i} + \hat{f}_{v_i}M_{z_i}^{v_i})e_k| \le L_e^S|e_k| \qquad (5.53)$$

$$|\sum_{n=1}^{N_s-1}\hat{f}_{v_i}M_{z_0}^{v_i}\prod_{j=i+1}^{N_s-1}(\hat{f}_{z_j} + \hat{f}_{v_j}M_{z_j}^{v_j})e_k| \le L_{\hat{e}}^S|e_k| \qquad (5.54)$$

And we could find an upper bound $L_w^S$ for $\sum_{n=0}^{n-1}\prod_{j=i+1}^{n-1}(\hat{f}_{z_j} + \hat{f}_{v_j}M_{z_j}^{v_j})$ such that

$$|\hat{e}_{k+n}| \le L_e^S|e_k| + \sum_{n=0}^{n-1}L_w^S|w(k+n)| \qquad (5.55)$$

when $n = 1,\dots,N_s-1$ and

$$|e_{k+N_s}| \le L_{\hat{e}}^S|e_k| + \sum_{n=0}^{N_s-1}L_w^S|w(k+n)| \qquad (5.56)$$

Applying recursion on (5.56) we obtain for $m = 0, 1, \ldots$ and $k = mN_s$:

$$|e_{(m+1)N_s}| \leq (L_{\hat{e}}^S)^{m+1}|e_0| + \sum_{l=0}^{m}[(L_{\hat{e}}^S)^l \sum_{i=0}^{N_s-1} L_w^S |w((m-l)N_s+i)|] \tag{5.57}$$

since the NLP problems are only solved at $k = mN_s$. Because $k \to \infty$ a memory effect persists in the accumulation of errors $e_k$ and it is not clear that this accumulation remains bounded. This implies that robust stability may not hold for serial amsNMPC.

### Simplifying Assumptions for the Serial Approach

From the sensitivity equations, we will introduce the following approximation:

$$M_{z_n}^{v_n}\hat{e}_{k+n} \quad = \quad M_{z_0}^{v_n}e_k + \sum_{i=0}^{n-1} M_{z_i}^{v_n} w(k+i) \tag{5.58}$$

**Justification for Approximation (5.58):** To see this, we first consider the nominal case with $w(k+i) = 0, i = 1, \ldots, N_s - 1$. From the sensitivity system we define,

$$u(k) - v_{0|k+N_s} = \Delta v_0 = M_{z_0}^{v_0} e_k = M_{z_0}^{v_0}(x(k) - \overline{x}(k))$$

as with the blocked approach. From the evolution of the *linear sensitivity equations* we can write:

$$x(k+1) - z_{1|k-N_s} = \Delta\overline{z}_1 = f_{z_0}e_k + f_{v_0}M_{z_0}^{v_0}e_k = (f_{z_0} + f_{v_0}M_{z_0}^{v_0})e_k \tag{5.59}$$

where the Jacobians $f_{z_i}, f_{v_i}$ are elements of the KKT matrix evaluated at $x(k-N_s)$. By induction, assume that

$$\Delta\overline{z}_{i+1} = [(f_{z_i} + f_{v_i}M_{z_i}^{v_i}), \ldots, (f_{z_1} + f_{v_1}M_{z_1}^{v_1})(f_{z_0} + f_{v_0}M_{z_0}^{v_0})]e_k \tag{5.60}$$

and

$$\begin{aligned} \Delta v_{i+1} \quad &= \quad M_{z_{i+1}}^{v_{i+1}}\Delta\overline{z}_{i+1} = M_{z_{i+1}}^{v_{i+1}}[(f_{z_i} + f_{v_i}M_{z_i}^{v_i}), \ldots, (f_{z_1} + f_{v_1}M_{z_1}^{v_1})(f_{z_0} + f_{v_0}M_{z_0}^{v_0})]e_k \\ &= \quad M_{z_0}^{v_{i+1}}e_k. \end{aligned} \tag{5.61}$$

Then

$$\begin{aligned}\Delta v_{i+2} &= M_{z_{i+2}}^{v_{i+2}} \Delta \bar{z}_{i+2} = M_{z_{i+2}}^{v_{i+2}} [f_{z_{i+1}} \Delta \bar{z}_{i+1} + f_{v_{i+1}} \Delta \bar{v}_{i+1}] \\ &= M_{z_{i+2}}^{v_{i+2}} [(f_{z_{i+1}} + f_{v_{i+1}} M_{z_{i+1}}^{v_{i+1}}), \ldots, (f_{z_1} + f_{v_1} M_{z_1}^{v_1})(f_{z_0} + f_{v_0} M_{z_0}^{v_0})] e_k \\ &= M_{z_0}^{v_{i+2}} e_k. \end{aligned} \tag{5.62}$$

Hence, with $w(k+i) = 0, i = 1, \ldots, N_s - 1$, we have $\Delta v_i = M_{z_i}^{v_i} \Delta \bar{z}_i = M_{z_0}^{v_i} e_k$.

We now consider the case where $w(k+i) \neq 0$ and $w(k+j) \neq 0, j = 1, \ldots, i-1$ and state the following assumption.

**Assumption 5.** *(Robust Stability Assumptions of the Serial Approach) The errors due to differences in the Jacobians $f_{z_i}, f_{v_i}$ and $\hat{f}_{z_i}, \hat{f}_{v_i}$ can always be bounded by the uncertainty term $|w'(k+i-1) - w(k+i-1)|$ as follows:*

$$\begin{aligned} & |(\hat{f}_{z_{i+1}} + \hat{f}_{v_{i+1}} M_{z_{i+1}}^{v_{i+1}}), \ldots, (\hat{f}_{z_0} + \hat{f}_{v_0} M_{z_0}^{v_0}) e_k - (f_{z_{i+1}} + f_{v_{i+1}} M_{z_{i+1}}^{v_{i+1}}) \ldots, (f_{z_0} + f_{v_0} M_{z_0}^{v_0}) e_k| \\ \leq\ & |w'(k+i-1) - w(k+i-1)|, \end{aligned} \tag{5.63}$$

*where $w'(k+i-1), w(k+i-1) \in \mathscr{W}$, $i = 1, \ldots N_s - 1$*

This assumption allows the evolution of the errors to be rewritten as:

$$\begin{aligned} e(k+i) &= x(k+i) - z_{i|k-N_s} \\ &= (\hat{f}_{z_{i+1}} + \hat{f}_{v_{i+1}} M_{z_{i+1}}^{v_{i+1}}), \ldots, (\hat{f}_{z_1} + \hat{f}_{v_1} M_{z_1}^{v_1})(\hat{f}_{z_0} + \hat{f}_{v_0} M_{z_0}^{v_0}) e_k + w(k+i-1) \\ &= (f_{z_{i+1}} + f_{v_{i+1}} M_{z_{i+1}}^{v_{i+1}}), \ldots, (f_{z_1} + f_{v_1} M_{z_1}^{v_1})(f_{z_0} + f_{v_0} M_{z_0}^{v_0}) e_k + w'(k+i-1). \end{aligned} \tag{5.64}$$

From the Schur complement extension of the sensitivity equations ([58]), we have $x(k+i) - \bar{x}(k+i)$ (i.e., $w(k+i)$) specified in the right hand sides and $p - p_0$ is back-calculated.

For $w(k+i) \neq 0, i = 1, \ldots, N_s - 1$ we have:

$$\begin{aligned}
\hat{e}_{k+1} &= (f_{z_0} + f_{v_0} M_{z_0}^{v_0}) e_k + w(k) \\
\hat{e}_{k+2} &= (f_{z_1} + f_{v_1} M_{z_1}^{v_1})(f_{z_0} + f_{v_0} M_{z_0}^{v_0}) e_k + (f_{z_1} + f_{v_1} M_{z_1}^{v_1}) w(k) + w(k+1) \\
\hat{e}_{k+n} &= \prod_{i=0}^{n-1} (f_{z_i} + f_{v_i} M_{z_i}^{v_i}) e_k + \sum_{i=0}^{n-1} \prod_{j=i}^{n-1} (f_{z_j} + f_{v_j} M_{z_j}^{v_j}) w(k+i).
\end{aligned} \tag{5.65}$$

Similarly, we can write:

$$\begin{aligned}
M_{z_n}^{v_n} \hat{e}_{k+n} &= M_{z_n}^{v_n} [\prod_{i=0}^{n-1} (f_{z_i} + f_{v_i} M_{z_i}^{v_i}) e_k + \sum_{i=0}^{n-1} \prod_{j=i}^{n-1} (f_{z_j} + f_{v_j} M_{z_j}^{v_j}) w(k+i)] \\
M_{z_n}^{v_n} \hat{e}_{k+n} &= M_{z_0}^{v_n} e_k + \sum_{i=0}^{n-1} M_{z_i}^{v_n} w(k+i).
\end{aligned} \tag{5.66}$$

As a result of this approximation, the evolution (5.50) can be rewritten as (5.67). Note that this has strong similarities to (5.11) from the blocked amsNMPC approach.

$$\begin{aligned}
x(k+n) &= F(x(k), v_{0|k-N_s} + M_{z_0}^{v_0} e_k, w(k), v_{1|k-N_s} + M_{z_0}^{v_1}(e_k + w(k)), w(k+1), \ldots, \\
&\quad v_{n-1|k-N_s} + M_{z_0}^{v_{n-1}} e_k + \sum_{i=0}^{n-2} M_{z_i}^{v_n} w(k+i), w(k+n-1)).
\end{aligned} \tag{5.67}$$

In a similar manner as with blocked amsNMPC, we again compare (5.4) with (5.67), and derive $e_k$ and $e_{k+N_s}$ expressions analogous to (5.23), (5.24):

$$x(k) - \overline{x}(k) = e_k = \sum_{i=0}^{N_s-1} [\prod_{j=i+1}^{N_s-1} \hat{f}_{z_{j-N_s}} + \sum_{j=i+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_{p-N_s}}) \hat{f}_{v_{j-N_s}} M_{z_i}^{v_j}] w(k+i-N_s) \tag{5.68}$$

$$x(k+N_s) - \overline{x}(k+N_s) = e_{k+N_s} = \sum_{i=0}^{N_s-1} [\prod_{j=i+1}^{N_s-1} \hat{f}_{z_j} + \sum_{j=i+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j} M_{z_i}^{v_j}] w(k+i) \tag{5.69}$$

Defining an upper bound $L_f^S$ for $\prod_{j=i+1}^{N_s-1} \hat{f}_{z_j} + \sum_{j=i+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j} M_{z_i}^{v_j}$ we then have:

$$|e_k| \leq \sum_{i=0}^{N_s-1} L_f^S |w(k-N_s+i)| \text{ and } |e_{k+N_s}| \leq \sum_{i=0}^{N_s-1} L_f^S |w(k+i)| \tag{5.70}$$

Plugging (5.70) into (5.65) we get

$$\begin{aligned}|\hat{e}_{k+n}| &\leq \prod_{i=0}^{n-1}(\hat{f}_{z_i}+\hat{f}_{v_i}M_{z_i}^{v_i})\sum_{i=0}^{N_s-1}L_f^S|w(k-N_s+i)|+\sum_{i=0}^{n-1}\prod_{j=i}^{n-1}(\hat{f}_{z_j}+\hat{f}_{v_j}M_{z_j}^{v_j})|w(k+i)| \\ &\leq L_e^S\sum_{i=0}^{N_s-1}L_f^S|w(k-N_s+i)|+\sum_{i=0}^{n-1}L_w^S|w(k+i)| \end{aligned} \tag{5.71}$$

where $L_e^S$ and $L_w^S$ satisfy

$$|\prod_{i=0}^{n-1}(\hat{f}_{z_i}+\hat{f}_{v_i}M_{z_i}^{v_i})| \leq L_e^S \tag{5.72}$$

$$\prod_{j=i}^{n-1}(\hat{f}_{z_j}+\hat{f}_{v_j}M_{z_j}^{v_j}) \leq L_w^S \tag{5.73}$$

For the serial approach, (5.38), Lemma 1 and Lemma 2 are still true. However, we also observe the following changes:

$$\begin{aligned} & J^{id}(\overline{x}(k+N_s))-J^{ams}(x(k)) \\ \leq\ & -\psi(x(k),u^{ams}(k))-\sum_{l=1}^{N_s-1}\psi(z_{l|k-N_s},u^{ams}(k+l)) \\ =\ & -\psi(x(k),u^{ams}(k))-\sum_{l=1}^{N_s-1}\psi(x(k+l)-\hat{e}_{k+l},u^{ams}(k+l)) \\ \leq\ & -\psi(x(k),u^{ams}(k))+\sum_{l=1}^{N_s-1}(-\psi(x(k+l),u^{ams}(k+l))+L_z\hat{e}_{k+l}) \\ \leq\ & \sum_{l=0}^{N_s-1}-\psi(x(k+l),u^{ams}(k+l))+\sum_{l=1}^{N_s-1}(L_z(\sum_{i=0}^{N_s-1}L_e^SL_f^S|w(k+i-N_s)|+\sum_{i=0}^{l-1}L_w^S|w(k+i)|) \end{aligned} \tag{5.74}$$

Therefore

$$\begin{aligned} & J^{ams}(x(k+N_s))-J^{ams}(x(k)) \\ =\ & \sum_{l=0}^{N_s-1}-\psi(x(k+l),u^{ams}(k+l))+L_z(1+L_h+L_h^{ams}|e_{k+N_s}|)|e_{k+N_s}|+\sum_{l=1}^{N_s-1}(L_z\hat{e}_{k+l}) \end{aligned} \tag{5.75}$$

Combining with (5.70) and (5.71) we have

$$\begin{aligned}
& J^{ams}(x(k+N_s)) - J^{ams}(x(k)) \\
\leq & \sum_{l=0}^{N_s-1} -\psi(x(k+l), u^{ams}(k+l)) + L_f^S(\sum_{n=0}^{N_s-1} |w(k+n)|)[L_z(1+L_h) + L_z L_h^{ams}(\sum_{n=0}^{N_s-1} L_f^S |w(k+n)|)] \\
& + \sum_{l=1}^{N_s-1} L_z(\sum_{n=0}^{N_s-1} L_e^S L_f^S |w(k+n-N_s)| + \sum_{n=0}^{l-1} L_w^S |w(k+n)|) \\
= & \sum_{l=0}^{N_s-1} \{-\psi(x(k+l), u^{ams}(k+l)) + L_z[(L_f^S(1+L_h) + L_w^S(N_s-1-l))|w(k+l)|]\} \\
& + L_z L_h^{ams}(L_f^S \sum_{i=0}^{N_s-1} |w(k+i)|)^2 + c_2. \qquad (5.76)
\end{aligned}$$

where the old error is represented by $c_2 = \sum_{l=1}^{N_s-1} L_z(\sum_{i=0}^{N_s-1} L_e^S L_f^S |w(k-N_s+i)|)$.

Due to the implementation of the serial approach, $J^{ams}(x(k+i)), i = 1,2,...,N_s-1$ can be calculated as follows:

$$J^{ams}(x(k+i)) = J^{id}(\overline{x}(k)) - \sum_{l=0}^{i-1} \psi(z_{l|k-N_s}, v_{l|k-N_s}) + \hat{J}_{i,z}(x(k))[x(k+i) - z_{i|k-N_s}] \qquad (5.77)$$

where

$$J_i(\overline{x}(k)) = J^{id}(\overline{x}(k)) - \sum_{l=0}^{i-1} \psi(z_{l|k-N_s}, v_{l|k-N_s}) \qquad (5.78)$$

$$\hat{J}_{i,z}(x(k)) = \int_0^1 \frac{dJ_i(\overline{x}(k))}{dz_i}[z_{i|k-N_s} + t(x(k+i) - z_{i|k-N_s})]^T (x(k+i) - z_{i|k-N_s}) dt \qquad (5.79)$$

and we assume that for all $x$ $|\hat{J}_{i,z}(x)| \leq L_z$, a Lipschitz constant. Next we compare $J^{ams}(x(k+$

$i+1))$ and $J^{ams}(x(k+i))$, $i=0,\ldots,N_s-2$:

$$
\begin{aligned}
& J^{ams}(x(k+i+1)) - J^{ams}(x(k+i)) \\
= \;& J^{id}(\overline{x}(k)) - \sum_{l=0}^{i} \psi(z_{l|k-N_s}, v_{l|k-N_s}) + \hat{J}_{i+1,z}(x(k))\hat{e}_{k+i+1} \\
& -[J^{id}(\overline{x}(k)) - \sum_{l=0}^{i-1} \psi(z_{l|k-N_s}, v_{l|k-N_s}) + \hat{J}_{i,z}(x(k))\hat{e}_{k+i}] \\
= \;& -\psi(z_{i|k-N_s}, v_{i|k-N_s}) + \hat{J}_{i+1,z}(x(k))\hat{e}_{k+i+1} - \hat{J}_{i,z}(x(k)))\hat{e}_{k+i} \\
\leq \;& -\psi(x(k+i) - \hat{e}_{k+i}, u^{ams}(k+i) - M_{z_i}^{v_i}\hat{e}_{k+i}) + |\hat{J}_{i+1,z}(x(k))|(L_e^S|e_k| + \sum_{l=0}^{i} L_w^S|w(k+l)|) \\
& +|\hat{J}_{i,z}(x(k))|(L_e^S|e_k| + \sum_{l=0}^{i-1} L_w^S|w(k+l)|)) \\
\leq \;& -\psi(x(k+i), u^{ams}(k+i)) + (2L_z+L_v)\{L_e^S \sum_{l=0}^{N_s-1} L_f^S|w(k-N_s+l)| + \sum_{l=0}^{i-1} L_w^S|w(k+l)|\} \\
& +L_z\{L_e^S \sum_{l=0}^{N_s-1} L_f^S|w(k-N_s+l)|) + \sum_{l=0}^{i} L_w^S|w(k+l)|\} \\
= \;& -\psi(x(k+i), u^{ams}(k+i)) + L_z(x(k))L_w^S|w(k+i)| + c_3 \qquad (5.80)
\end{aligned}
$$

where $c_3 = (3L_z+L_v)\{L_e^S\sum_{l=0}^{N_s-1} L_f^S|w(k-N_s+l)| + \sum_{l=0}^{i-1} L_w^S|w(k+l)|\}$.

$$
\begin{aligned}
& J^{ams}(x(k+N_s)) - J^{ams}(x(k+N_s-1)) \\
= \;& J^{id}(\overline{x}(k+N_s)) + \hat{J}_{0,z}(x(k+N_s))e_{k+N_s} \\
& -[J^{id}(\overline{x}(k)) - \sum_{l=0}^{N_s-2} \psi(z_{l|k-N_s}, v_{l|k-N_s}) + \hat{J}_{N_s-1,z}(x(k))\hat{e}_{k+N_s-1}] \\
\leq \;& -\sum_{l=0}^{N_s-1} \psi(z_{l|k-N_s}, v_{l|k-N_s}) + \sum_{l=0}^{N_s-2} \psi(z_{l|k-N_s}, v_{l|k-N_s}) \\
& +|\hat{J}_{0,z}(x(k+N_s))||e_{k+N_s}| + |\hat{J}_{N_s-1,z}(x(k))||\hat{e}_{k+N_s-1}| \\
\leq \;& -\psi(z_{N_s-1|k-N_s}, v_{N_s-1|k-N_s}) + L_z \sum_{l=0}^{N_s-1} L_f^S |w(k+l)|) \\
& +L_z\{L_e^S \sum_{l=0}^{N_s-1} L_f^S |w(k-N_s+l)|) + \sum_{l=0}^{N_s-2} L_w^S |w(k+l)|)\} \\
\leq \;& -\psi(x(k+N_s-1) - \hat{e}_{k+N_s-1}, u^{ams}(k+N_s-1) - M_{z_{N_s-1}}^{v_{N_s-1}} \hat{e}_{k+N_s-1}) + L_z \sum_{l=0}^{N_s-1} L_f^S |w(k+l)|) \\
& L_z\{L_e^S \sum_{l=0}^{N_s-1} L_f^S |w(k-N_s+l)|) + \sum_{l=0}^{N_s-2} L_w^S |w(k+l)|)\} \\
\leq \;& -\psi(x(k+N_s-1), u^{ams}(k+N_s-1)) + L_z \sum_{l=0}^{N_s-1} L_f^S |w(k+l)|) \\
& +(2L_z+L_v)\{L_e^S \sum_{l=0}^{N_s-1} L_f^S |w(k-N_s+l)|) + \sum_{l=0}^{N_s-2} L_w^S |w(k+l)|\} \\
= \;& -\psi(x(k+N_s-1), u^{ams}(k+N_s-1)) + L_z L_f^S |w(k+N_s-1)| + c_4
\end{aligned} \tag{5.81}
$$

where $c_4 = L_z \sum_{l=0}^{N_s-2} L_f^S |w(k+l)| + (2L_z+L_v)\{L_e^S \sum_{l=0}^{N_s-1} L_f^S |w(k-N_s+l)| + \sum_{l=0}^{N_s-2} L_w^S |w(k+l)|\}$. From (5.81) and Definition 17, we obtain the following result.

**Theorem 9.** *(Robust stability of the serial approach) For the moving horizon problem (5.1) and associated control law $u = h^{ams}(x)$ that satisfies Assumption 3, Assumption 4 and 5 with $c_3$ and $c_4$ is ISpS Lyapunov function and the closed-loop system is robustly stable.*

If clipping is applied, we obtain $u(k) - v_{0|k+N_s} = \Delta v_0 = \tau_0 M_{z_0}^{v_0} e_k$. Then (5.59) becomes

$$\Delta z_1 = f_{z_0} e_k + f_{v_0} \tau_0 M_{z_0}^{v_0} e_k = (f_{z_0} + f_{v_0} \tau_0 M_{z_0}^{v_0}) e_k \tag{5.82}$$

and (5.60) becomes

$$\Delta \bar{z}_{i+1} = [(f_{z_i} + f_{v_i} \tau_i M_{z_i}^{v_i}), \ldots, (f_{z_1} + f_{v_1} \tau_1 M_{z_1}^{v_1})(f_{z_0} + f_{v_0} \tau_0 M_{z_0}^{v_0})] e_k \tag{5.83}$$

However $M_{z_{i+i}}^{v_{i+1}} [(f_{z_i} + f_{v_i} \tau_i M_{z_i}^{v_i}), \ldots, (f_{z_1} + f_{v_1} \tau_1 M_{z_1}^{v_1})(f_{z_0} + f_{v_0} \tau_0 M_{z_0}^{v_0})] \neq M_{z_0}^{v_{i+1}}$. Therefore (5.58), (5.61) and (5.62) are not true any more. As a result Assumption 5 and the ISpS property do not hold with clipping.

## 5.4 Parallel Approach

Among the two approaches, the parallel approach is more direct and easier to implement because it is a combination of multiple asNMPC iterations, each executed by one processor. The amsNMPC approach handles NLP problems whose solutions require multiple sampling times to compute. Suppose $N_s \geq 1$ sampling times are needed to solve the NLP problem. Here we would like to solve the predicted NLP problem $N_s$ sampling times in advance to get the predicted manipulated variable for the current state.

We define $\bar{x}(k)$ as the *prediction* of the state at time $t_k$. Fig. 5.4 shows the positions of $x$, $u$, $\bar{x}$ and $v$. Solid lines are for the actual control profiles, while dashed lines are for predicted state and control profiles. Also, different dashed line styles indicate solutions of different NLP problems (5.85).

At $t_k$, it is assumed that the solution of the NLP problem is calculated based on the information from time $t_{k-N_s}$, and the solution is $\{z_{0|k-N_s}, \ldots, z_{N|k-N_s}, v_{0|k-N_s}, \ldots, v_{N-1|k-N_s}\}$. Once $x(k)$ is obtained, $v_{j|k-N_s}$, $j = 0, \ldots, N_s - 1$ are updated from (3.23), with $\bar{x}(k) = p_0$ and

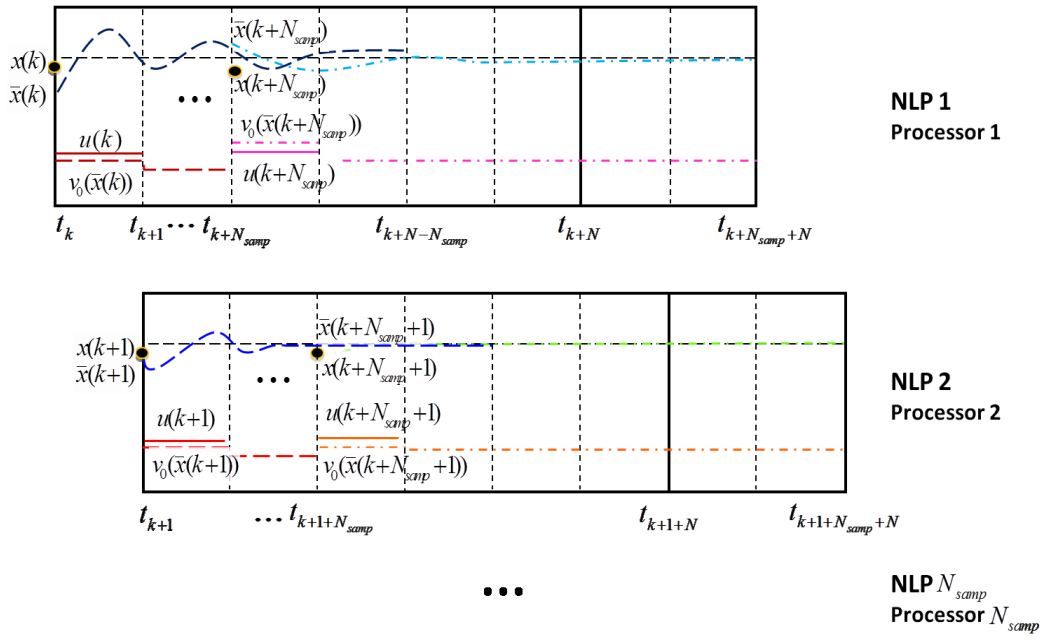



Figure 5.4: Positions of $x$, $u$, $\bar{x}$ and $v$ for the parallel approach

$x(k) = p$: We then predict $\bar{x}(k+N_s)$ with $x(k), v_{j|k-N_s}, j = 0,\ldots,N_s-1$, which becomes the initial value and parameter $p_0$ of the next NLP problem solved by the same processor:

$$\bar{x}(k+N_s) = F(x(k), v_{0|k-N_s} + M^0_{v_0} e_k, 0, v_{1|k-N_s} + M^0_{v_1} e_k, 0, \ldots, v_{N_s-1|k-N_s} + M^0_{v_{k-N_s}} e_k, 0) \tag{5.84}$$

We use the superscript of $M$ to show which processor generated the current matrix that is being used. Similarly, the superscript of $\hat{f}$ shows which processor generates the solution that is used to evaluate the derivative.

Note that the manipulated variables are updated using (3.23) only. At each sampling time the solution of an NLP problem and a new linearized KKT system are obtained, and the first manipulated variable of each solution is updated with respect to the current state measurement. Since it takes $N_s$ sampling times to solve one NLP, $N_s$ processors are applied at each sampling time in a staggered manner.

### 5.4.1 Implementation

The parallel approach is implemented on $N_s$ processors as follows: For $i = 0 : N_s - 1$,

- *Online*: at $t_{k+i}$, having $x(k+i)$ , given $\overline{x}(k+i) = p_0$ and $x(k+i) = p$, update $v_{j|k-N_s+i},\ j = 0,\ldots,N_s-1$ and $z_{N_s|k-N_s+i}$ using $(s^*(p_0) + \Delta s(p))$ from (3.23). Inject the updated $v_{0|k-N_s+i} + \Delta v_0 \in \mathbb{U}$ as $u(k+i)$ to the plant.
- *Background*: predict $x(k+N_s+i)$ by (5.84) as the initial value and solve the NLP problem (5.85) using the $i^{th}$ processor.

$$\begin{aligned} \min_{v_l, z_l} \quad & J_N(\overline{x}(k+i+N_s)) := \Psi(z_N) + \sum_{l=0}^{N-1} \psi_l(z_l, v_l) \\ \text{s.t.} \quad & z_{l+1} = f(z_l, v_l),\ z_0 = \overline{x}(k+i+N_s),\ l = 0,...,N-1 \\ & v_l \in \mathbb{U} \end{aligned} \tag{5.85}$$

Set $k = k + N_s$ and repeat the cycle.

### 5.4.2 Nominal Stability Analysis

For the stability proof we need to assume that $N \to \infty$ so that all solutions with $x(k+i+N_s)$ correspond to each other. For nominal stability we have $x(k) = \overline{x}(k)$ and all the states are measured. Problem (5.85) can be formulated as

$$\begin{aligned} \min_{v_l, z_l} \quad & J_N(x(k+i+N_s)) := \Psi(z_N) + \sum_{l=0}^{N-1} \psi_l(z_l, v_l) \\ \text{s.t.} \quad & z_{l+1} = f(z_l, v_l),\ z_0 = x(k+i+N_s),\ l = 0,...,N-1 \\ & v_l \in \mathbb{U} \end{aligned} \tag{5.86}$$

where $i = 0,...,N_s-1$. The subscript of $J$ indicates the length of the horizon.

From (5.8) we have

$$J_N(x(k)) - J_N(x(k+N_s)) \geq \sum_{l=0}^{N_s-1} \psi_l(z_{l|k-N_s}, v_{l|k-N_s}) = \sum_{l=0}^{N_s-1} \psi_l(x(k+l), u(k+l))$$

Since

$$\begin{aligned} & \sum_{j=0}^{\infty} \sum_{i=0}^{N_s-1} (J_N(x(k+jN_s+i))) - J_N(x(k+(j+1)N_s+i)) \\ = & \sum_{i=0}^{N_s-1} J_N(x(k+i)) - J_N(x(\infty)) \\ \leq & \sum_{i=0}^{N_s-1} J_N(x(k+i)), \end{aligned}$$

we can combine with (5.8) to yield

$$\sum_{i=0}^{N_s-1} J_N(x(k+i)) \geq \sum_{j=0}^{\infty} \sum_{l=0}^{N_s-1} \psi_{k+l}(z_{k+l+jN_s}, v_{k+l+jN_s}) = \sum_{m=0}^{\infty} \psi_m(x(k+m), u(k+m)) \quad (5.87)$$

which leads to

$$\lim_{k\to\infty} \psi_l(x(k), u(k)) = 0 \quad (5.88)$$

So $J_N(x(k))$ satisfies the conditions of Lyapunov function, and the process on the processor being analyzed is nominally stable. We note that this analysis is similar to the average performance proposed by Angeli et al. [3], where the average cost function of $N_s$ adjacent NLP problems satisfies the conditions of Lyapunov function.

$$\bar{J}_{N,j} = \frac{\sum_{i=0}^{N_s-1} J_N(x(k+i+jN_s))}{N_s} \quad (5.89)$$

where $j = 0, ..., \infty$. From (5.8) we get

$$\bar{J}_{N,j} - \bar{J}_{N,j+1} \geq \sum_{l=j*N_s}^{(j+1)*N_s-1} (\psi_l(z_l, v_l)/N_s \quad (5.90)$$

Since $\bar{J}_{N,j}$ satisfies the conditions of Lyapunov function, the following nominal stability property is proved.

**Theorem 10.** *(Nominal Stability of the Parallel Approach) For the moving horizon problem (5.86) with $N \to \infty$ and associated control law $u = \kappa^{ams}(x)$ that satisfies Assumption 3, $\bar{J}_{N,j}$ is a Lyapunov function and the closed-loop system is asymptotically stable.*

### 5.4.3 Robust Stability Analysis

For the parallel approach, all the states are predicted identically; therefore we do not add a bar above the error to show the difference. Since inputs are calculated by different processors, it is difficult to figure out how error accumulates from $e_k$ to $e_{k+N_s}$ if N is finite. Therefore for robust stability proof we assume infinite horizon length. Having $x(k)$, with the first processor we have (5.84):

$$\bar{x}(k+N_s) = F(x(k), v_{0|k-N_s} + M^{v_0}_{z_0,0} e_k, 0, v_{1|k-N_s} + M^{v_1}_{z_0,0} e_k, 0, ..., v_{N_s-1|k-N_s} + M^{v_{N_s-1}}_{z_0,0} e_k, 0) \tag{5.91}$$

We use the second subscript of $M$ to show which processor generated the current matrix that is being used. Similarly, the superscript of $\hat{f}$ shows which processor generates the solution that is used to evaluate the derivative.

For the actual trajectory we have

$$\begin{aligned} x(k+N_s) &= F(x(k), v_{0|k-N_s} + M^{v_0}_{z_0,0} e_k, w(k), v_{0|k-N_s+1} + M^{v_0}_{z_0,1} e_{k+1}, w(k+1), ..., \\ &\quad v_{0|k-1} + M^{v_0}_{z_0,N_s-1} e_{k+N_s-1}, w(k+N_s-1)) \end{aligned} \tag{5.92}$$

For the parallel approach all processors are executed equivalently, so for processor $n, n = 0,\ldots,N_s-1$ we have

$$\begin{aligned}\bar{x}(k+N_s+n) &= F(x(k+n), v_{0|k-N_s+n}+M^{v_0}_{z_0,n}e_{k+n}, 0, v_{1|k-N_s+n}+M^{v_1}_{z_0,n}e_{k+n}, 0, ..., \\ &\quad v_{N_s-1|k-N_s+n}+M^{v_{N_s-1}}_{z_0,n}e_{k+n}, 0) \qquad (5.93)\\ x(k+N_s+n) &= F(x(k+n), v_{0|k-N_s+n}+M^{v_0}_{z_0,n}e_{k+n}, w(k+n), v_{0|k-N_s+1+n}+M^{v_0}_{z_0,n+1}e_{k+n+1}, \\ &\quad w(k+n+1), ..., v_{0|k-1+n}+M^{v_0}_{z_0,n+N_s-1}e_{k+n+N_s-1}, w(k+n+N_s-1)) \qquad (5.94)\end{aligned}$$

From (5.93) we could see that $z_{0|k+n} = \bar{x}(k+N_s+n)$ contains the information of $x(k+n), e_{k+n}, w(k+n),\ldots,w(k+n+N_s-1)$. Therefore memory effect also exists with the parallel approach. Moreover, $v_{i|k-N_s+n}, i = 0,\ldots,N_s-1$ contains the information of $x(k-N_s+n)$, therefore $M^{\mathbf{v}}_{\mathbf{z},n}(x(k+N_s+n))$ contains the information of $x(k-N_s+n), w(k-N_s+n),\ldots,w(k-N_s+n-1))$. Similarly, $M^{\mathbf{v}}_{\mathbf{z},n}(x(k+n))$ contains the information of $x(k-2N_s+n), w(k-2N_s+n),\ldots,w(k-2N_s+n-1))$.

For the parallel approach, since the controls are calculated by different processors, it is difficult to study the error accumulation. Therefore we need to make assumptions to link solutions of processor $n, n = 1,\ldots,N_s-1$ to processor 0:

**Assumption 6.** *(Robust Stability Assumptions of the Parallel Approach)*

1. *$N \to \infty$ such that when there is no uncertainty in the system, we have $M^{v_0}_{z_0,n} = M^{v_n}_{z_n,0}$.*
2. *We assume the error $q_{k+n} = (M^{v_0}_{z_0,n} - M^{v_n}_{z_n,0})e_{k+n}$ can always be bounded by an uncertainty term $L_M \sum_{i=0}^{n-1} |w(k+i)| + |w(k+i-N_s)|$, i.e.,*

$$|q_{k+n}| = |(M^{v_0}_{z_0,n} - M^{v_n}_{z_n,0})e_{k+n}| \le L_M \sum_{i=0}^{n-1} |w(k+i)| + |w(k+i-N_s)| \qquad (5.95)$$

*where $L_M$ is a Lipschitz constant for $M^v_z$.*

Therefore we could reformulate (5.92) as

$$\begin{aligned} x(k+N_s) &= F(x(k), v_{0|k-N_s} + M_{z_{0,0}}^{v_0} e_k, w(k), v_{1|k-N_s} + M_{z_{1,0}}^{v_1} e_{k+1} + q_{k+1}, \\ &\quad w(k+1), ..., v_{N_s-1|k-N_s} + M_{z_{N_s-1,0}}^{v_{N_s-1}} e_{k+N_s-1} + q_{k+N_s-1}, w(k+N_s-1)) \end{aligned} \tag{5.96}$$

Notice that this is similar to the evolution of the serial approach (5.50). So we could use Assumption 5 and replace $M_{z_n,0}^{v_n} e_{k+n}$ with $M_{z_0,0}^{v_n} e_k + \sum_{i=0}^{n-1} M_{z_i,0}^{v_n} w(k+i)$ (5.58) and end up with

$$\begin{aligned} & x(k+N_s) \\ = \; & F(x(k), v_{0|k-N_s} + M_{z_{0,0}}^{v_0} e_k, w(k), v_{1|k-N_s} + M_{z_{0,0}}^{v_1}(e_k + w(k)) + q_{k+1}, w(k+1), ..., \\ & v_{N_s-1|k-N_s} + M_{z_0,0}^{v_{N_s-1}} e_k + \sum_{i=0}^{n-2} M_{z_i,0}^{v_n} w(k+i) + q_{k+N_s-1}, w(k+N_s-1)) \end{aligned} \tag{5.97}$$

Comparing (5.97) and (5.84), we have

$$\begin{aligned} & x(k+N_s) - \overline{x}(k+N_s) = e_{k+N_s} \\ = \; & \sum_{i=0}^{N_s-1} [\prod_{j=i+1}^{N_s-1} \hat{f}_{z_j} + \sum_{j=i+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j} M_{z_m,0}^{v_j}] w(k+i) \\ & + \sum_{i=0}^{N_s-1} [\sum_{j=i+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j} q_{k+j}] \\ \leq \; & \sum_{i=0}^{N_s-1} \{\prod_{j=i+1}^{N_s-1} \hat{f}_{z_j} + \sum_{j=i+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j} M_{z_i,0}^{v_j} + L_M \sum_{m=i}^{N_s-1} [\sum_{j=m+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j}]\} |w(k+i)| \\ & + \sum_{i=0}^{N_s-1} \{L_M \sum_{m=i}^{N_s-1} [\sum_{j=m+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j}]\} |w(k-N_s+i)| \end{aligned} \tag{5.98}$$

Defining an upper bound $L_{f1}^P$ for $\{\prod_{j=i+1}^{N_s-1} \hat{f}_{z_j} + \sum_{j=i+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j} M_{z_i,0}^{v_j} + L_M \sum_{m=i}^{N_s-1} [\sum_{j=m+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j}]\}$ and an upper bound $L_{f2}^P$ for $\{L_M \sum_{m=i}^{N_s-1} [\sum_{j=m+1}^{N_s-1} (\prod_{p=j+1}^{N_s-1} \hat{f}_{z_p}) \hat{f}_{v_j}]\}$ we

then have:

$$|e_{k+N_s}| \leq \sum_{i=0}^{N_s-1} [L_{f1}^P |w(k+i)| + L_{f2}^P |w(k+i-N_s)|] \tag{5.99a}$$

$$|e_k| \leq \sum_{i=0}^{N_s-1} [L_{f1}^P |w(k+i-N_s)| + L_{f2}^P |w(k+i-2N_s)|] \tag{5.99b}$$

Similarly as (5.65), for processor 0, we have

$$\begin{aligned}
& |\hat{e}_{k+n}| \\
= \; & |\prod_{i=0}^{n-1}(f_{z_i} + f_{v_i} M_{z_i,0}^{v_i}) e_k + \sum_{i=0}^{n-1} \prod_{j=i}^{n-1} (f_{z_j} + f_{v_j} M_{z_j,0}^{v_j}) w(k+i) + \sum_{i=1}^{n-1} \prod_{j=i}^{n-1} (f_{z_j} + f_{v_j} M_{z_j,0}^{v_j}) \hat{f}_{v_i} q_{k+i}| \\
\leq \; & \prod_{i=0}^{n-1}(f_{z_i} + f_{v_i} M_{z_i,0}^{v_i}) |e_k| + \sum_{i=0}^{n-1} \prod_{j=i}^{n-1} (f_{z_j} + f_{v_j} M_{z_j,0}^{v_j}) |w(k+i)| \\
& + L_M \sum_{i=1}^{n-1} \sum_{m=i}^{n-1} \prod_{j=m}^{n-1} (f_{z_j} + f_{v_j} M_{z_j,0}^{v_j}) \hat{f}_{v_i} (|w(k+i)| + |w(k-N_s+i)|)
\end{aligned} \tag{5.100}$$

From (5.99b) and defining suitable upper bounds leads to:

$$|\hat{e}_{k+n}| \leq L_1^P \sum_{i=0}^{N_s-1} |w(k-N_s+i)| + L_2^P \sum_{i=0}^{N_s-1} |w(k-2N_s+i)| + L_{w1}^P \sum_{i=0}^{n-1} |w(k+i)| \tag{5.101}$$

For the parallel approach, (5.38), Lemma 1 and Lemma 2 are still true. However, we also

observe the following changes:

$$\begin{aligned}
& J^{id}(\overline{x}(k+N_s)) - J^{ams}(x(k)) \\
\leq\ & -\psi(x(k),u^{ams}(k)) - \sum_{l=1}^{N_s-1} \psi(z_{l|k-N_s},u^{ams}(k+l)) \\
=\ & -\psi(x(k),u^{ams}(k)) - \sum_{l=1}^{N_s-1} \psi(x(k+l)-\hat{e}_{k+l},u^{ams}(k+l)) \\
\leq\ & -\psi(x(k),u^{ams}(k)) + \sum_{l=1}^{N_s-1} (-\psi(x(k+l),u^{ams}(k+l)) + L_z\hat{e}_{k+l}) \\
\leq\ & \sum_{l=0}^{N_s-1} -\psi(x(k+l),u^{ams}(k+l)) \\
& + \sum_{l=1}^{N_s-1} [L_z(L_1^P \sum_{i=0}^{N_s-1} |w(k-N_s+i)| + L_2^P \sum_{i=0}^{N_s-1} |w(k-2N_s+i)| + L_{w1}^P \sum_{i=0}^{l-1} |w(k+i)|]
\end{aligned} \tag{5.102}$$

Therefore

$$\begin{aligned}
& J^{ams}(x(k+N_s)) - J^{ams}(x(k)) \\
=\ & \sum_{l=0}^{N_s-1} -\psi(x(k+l),u^{ams}(k+l)) + L_z(1+L_\kappa + L_\kappa^{ams}|e_{k+N_s}|)|e_{k+N_s}| + \sum_{l=1}^{N_s-1} (L_z\hat{e}_{k+l})
\end{aligned} \tag{5.103}$$

Combining with (5.70) and (5.71) we have

$$
\begin{aligned}
& N_s(J^{ams}(x(k+N_s)) - J^{ams}(x(k))) \\
\leq & \sum_{l=0}^{N_s-1} -\psi(x(k+l), u^{ams}(k+l)) \\
& + \sum_{i=0}^{N_s-1} [L_{f1}^P |w(k+i)| \\
& + L_{f2}^P |w(k+i-N_s)|][L_z(1+L_\kappa) + L_z L_\kappa^{ams}(\sum_{i=0}^{N_s-1} [L_{f1}^P |w(k+i)| + L_{f2}^P |w(k+i-N_s)|])] \\
& + \sum_{l=1}^{N_s-1} [L_z(L_1^P \sum_{i=0}^{N_s-1} |w(k-N_s+i)| + L_2^P \sum_{i=0}^{N_s-1} |w(k-2N_s+i)| + L_{w1}^P \sum_{i=0}^{l-1} |w(k+i)|] \\
= & \sum_{l=0}^{N_s-1} \{-\psi(x(k+l), u^{ams}(k+l)) + L_z[(L_{f1}^P(1+L_\kappa) + L_{w1}^P(N_s-1-l))|w(k+l)|]\} \\
& + L_z L_\kappa^{ams}(L_{f1}^P \sum_{i=0}^{N_s-1} |w(k+i)|)^2 + c_5. \qquad (5.104)
\end{aligned}
$$

where the old error is represented by $c_5 = \sum_{l=0}^{N_s-1}\{L_z[(L_{f1}^P(1+L_\kappa) + L_{w1}^P(N_s-1-l))|w(k+l-N_s)|]\} + L_z L_\kappa^{ams}(L_{f1}^P \sum_{i=0}^{N_s-1} |w(k+i-N_s)|)^2 + \sum_{l=1}^{N_s-1}[L_z(L_1^P \sum_{i=0}^{N_s-1} |w(k-N_s+i)| + L_2^P \sum_{i=0}^{N_s-1} |w(k-2N_s+i)|]$.

So $J^{ams}(x(k))$ satisfies the conditions of Lyapunov function, and the process on the processor 0 is nominally stable. We note that this analysis is similar to the average performance proposed by Angeli et al. [3], where the average cost function of $N_s$ adjacent NLP problems satisfies the conditions of Lyapunov function.

$$
\bar{J}_j^{ams} = \frac{\sum_{i=0}^{N_s-1} J^{ams}(x(k+i+jN_s))}{N_s} \qquad (5.105)
$$

where $j = 0, \ldots, \infty$. From (5.104) we get

$$\begin{aligned} & N_s(\bar{J}^{ams}_{j+1} - \bar{J}^{ams}_{j}) \\ \leq & \sum_{l=jN_s}^{(j+i)N_s-1} \{ \sum_{i=0}^{N_s-1} -\psi(x(k+l+i), u^{ams}(k+l+i)) \\ & + L_z[(L^P_{f1}(1+L_\kappa) + L^P_{w1}(N_s-1-l))|w(k+l+i)|] \\ & + L_z L^{ams}_\kappa (L^P_{f1} \sum_{i=0}^{N_s-1} |w(k+l+i)|)^2\} + 3c_5 \end{aligned} \tag{5.106}$$

which satisfies the conditions of ISpS Lyapunov function. Since $N_s\bar{J}_{N,j}$ satisfies the conditions of Lyapunov function, $\bar{J}_{N,j}$ could be used as the ISpS Lyapunov function and the following nominal stability property is proved.

**Theorem 11.** *(Robust stability of the parallel approach) For the moving horizon problem (5.1) and associated control law $u = \kappa^{ams}(x)$ that satisfies Assumption 3, Assumption 4 and Assumption 6 with $c_5$, $J^{ams}_j$ is an ISpS Lyapunov function and the closed-loop system is robustly stable.*

## 5.5 Concluding Remarks

In this chapter we propose the advanced-multi-step NMPC scheme to handle NLP problems whose solution takes multiple sampling times while avoiding computational delay. Two approaches, parallel and serial, are developed. For the parallel approach multiple processors are applied and an NLP problem is solved every sampling time. For the serial approach only one processor is applied and an NLP problem is solved every $N_s$ sampling times. Nominal stability and robust stability of amsNMPC are also analyzed in this chapter. Without uncertainty in the system, amsNMPC is nominally stable; with uncertainty there is memory effect due to error accumulation and inaccurate NLP sensitivity. However robust

stability could also be obtained with certain reasonable assumptions to bound this effect. We will evaluate the performance of both amsNMPC approaches on a CSTR model as well as a distillation column model. The results and analysis will be shown in Chapter 6.

# Chapter 6

# Case Studies

In this chapter we apply iNMPC, asNMPC and amsNMPC to two examples. The first example is a CSTR in which reaction $A \rightarrow B$ takes place. We set the order of reaction to 1 and 3. Setting order of reaction to 1 gives us a bilinear system while setting order of reaction to 3 gives us a more nonlinear system. We would like to use this example to show the influence that system nonlinearity has on the performance of amsNMPC. The second example is a large-scale propane-propylene distillation column. When we discretize the model and formulate it into an NLP, the solution time exceeds sampling time. We use this example to show the importance of amsNMPC for large-scale systems.

We compare the performance of iNMPC, asNMPC and amsNMPC through simulation and show how they support the results in the previous chapter.

Moreover, in this chapter, we also study the three MPC examples in [24] that lack robustness. We will show that both using a longer horizon or reformulating the NLP with soft constraints will lead to robustness. And robustness will not be lost when asNMPC or amsNMPC is applied.

## 6.1 Comparisons of iNMPC, asNMPC and amsNMPC

iNMPC is the ideal case where NLP solution time is negligible. In this case, there is no delay between obtaining the actual state and implementing the corresponding control.

Therefore the control is calculated with the exact state and no error exists. iNMPC should be the NMPC strategy that has the best performance.

asNMPC avoids computational delay by using prediction as initial condition to solve the NLP one step in advance and using NLP sensitivity to update the first control within the solution. If with uncertainty in the process, prediction error and sensitivity error will exist, and prediction error at $t_k$ comes from $g(x(k-1), w(k-1))$ at $t_{k-1}$. When there is no uncertainty, asNMPC has the same performance as iNMPC; however with uncertainty asNMPC will not perform as well as iNMPC with additional error as described in (4.36b).

amsNMPC predicts the state $N_s$ sampling times ahead of time. From (5.57) we can see that prediction error $\hat{e}_{k+n}$ at $t_{k+n}$ depends on prediction error $e_k$ at $t_k$ and additive disturbance $w$ from $t_k$ to $t_{k+n-1}$, and $e_k$ is the accumulation of the error at the first step $e_0$ and the additive disturbance $w$ from time $t_0$ to $t_{k-1}$. This memory effect will have significant influence on the performance of amsNMPC as well as its stability, and the larger $N_s$ is, the greater the influence will be. Besides prediction error, NLP sensitivity error also exists with amsNMPC.

The serial approach and the parallel approach have strong similarities. They both solve the NLP problems $N_s$ steps in advance using the predicted states at $k+N_s$ from the current optimal solution. The difference is that the serial approach uses only one processor, so the NLP problem is solved every $N_s$ sampling times, and the first $N_s$ manipulated variables in the horizon are updated by (3.23) and (5.43). The same KKT matrix is used repeatedly with update before the next KKT system is obtained. The parallel approach uses multiple processors, so the NLP problem is solved every sampling time, but only the first manipulated variable is updated using (3.23), and different KKT systems are used at every sampling time. With either approach, amsNMPC is expected to reduce the on-line computational cost by two to three orders of magnitude, because backsolves take much less time than

solving the NLP problem.

## 6.2 Demonstrations

### 6.2.1 Continuous stirred tank reactor

A dynamic CSTR example is used to demonstrate both amsNMPC approaches [58].

The reaction $A \rightarrow B$ is exothermic and model proposed by [25] is listed as follows:

$$\frac{dc}{dt} = \frac{F}{V}(c_f - c) - ke^{(-E/RT)}c \tag{6.1a}$$

$$\frac{dT}{dt} = \frac{F}{V}(T_f - T) + \frac{-\Delta H}{\rho C_p}ke^{-E/RT}c - \frac{AU}{V\rho C_p}(T - T_c) \tag{6.1b}$$

$$c_L \leq c \leq c_U,\ T_L \leq T \leq T_U \tag{6.1c}$$

$$F_L \leq F \leq F_U,\ U_L \leq U \leq U_U \tag{6.1d}$$

where $c$ and $T$ are the concentration of $A$ and temperature in the CSTR respectively, $F$ is the flow rate of feed, $V$ is the CSTR volume which is assumed to be a constant, $\beta$ is the order of the reaction. The dimensionless model takes the form

$$\frac{dz_1}{dt} = \frac{F}{V}(1 - z_1) - k'exp(-E'/z_2)z_1^{\beta} \tag{6.2a}$$

$$\frac{dz_2}{dt} = \frac{F}{V}(z_f - z_2) + k'exp(-E'/z_2)z_1^{\beta} - \alpha\nu(z_2 - z_c) \tag{6.2b}$$

where $z_1$ and $z_2$ are the dimensionless concentration of $A$ and temperature in the CSTR respectively, $z_f = 0.395$ and $z_c = 0.382$ are the dimensionless concentration and temperature of feed respectively, $k' = 300 \times 7.6^{\beta-1}$ is the dimensionless rate constant, $E' = 5$ is the dimensionless ratio of the activation energy and gas constant, $\alpha = 1.95 \times 10^{-4}$ is dimensionless heat transfer area and $\nu$ is the reactor jacket heat-transfer coefficient which is influenced by the coolant flow rate. For the order of reaction we have $\beta = 1,3$. For $\beta = 1$

we use $\frac{F}{V}$, and for $\beta = 3$ we use $\frac{V}{F}$ and $v$ as manipulated variables. After using orthogonal collocation to discretize the problem with $\Delta t = t_{k+1} - t_k = 1$, the objective function takes the form

$$J_N := \min \sum_{l=0}^{N} 10^6[(z_{1l} - z_{1des})^2 + (z_{2l} - z_{2des})^2] + \sum_{l=0}^{N-1} 2 \times 10^{-3}[(u_{1l} - u_{1des})^2 + (u_{2l} - u_{2des})^2] \tag{6.3}$$

where $N = 50$, $z_{1des} = 0.1768$, $z_{2des} = 0.7083$, $u_{1des} = 800$, $u_{2des} = 10$, and $\mathbf{z}$ and $\mathbf{u}$ are bounded by $0 \leq z_1 \leq 1$, $0 \leq z_2 \leq 1$, $0 \leq u_1 \leq 2500$, $1 \leq u_2 \leq 40$.

We now compare the performance of ideal NMPC and amsNMPC. For ideal NMPC, we assume that the NLP solution time is negligible; thus there is no computational delay. We use $N_s = 0$ to indicate ideal NMPC. For amsNMPC we include the cases $N_s = 1$, 2 and 3. Note that the asNMPC approach corresponds to $N_s = 1$, where the parallel approach and the serial approach are the same and only one processor is applied.

For the order of the reaction in (6.2a) and (6.2b) we consider cases where $\beta = 1$ and $\beta = 3$. We first consider the case where $\beta = 1$, as in [25] with a set point change and measurement noise. Next we make the problem artificially more nonlinear by setting $\beta = 3$ and by using $v$ and $\frac{V}{F}$, instead of $\frac{F}{V}$, as manipulated variables. We begin with no measurement noise or plant-model mismatch and show that amsNMPC can track set point changes immediately, if the change is known in advance. The results show that ideal NMPC and amsNMPC have virtually identical performance with any differences due to numerical precision and truncation errors. This also indicates the nominal stability of ideal NMPC and amsNMPC. Next we introduce different levels of measurement noise or plant-model mismatch and look for the trend of amsNMPC performance as noise, mismatch level and $N_s$ increase. Measurement noise has the form of Gaussian noise with standard deviation of a certain percentage of set point, and is added to the output. Plant-model mismatch is added by changing the parameter $k'$ in (6.2a) and (6.2b) in the simulation model. The comparison of

QP and clipping is also shown. All NLP problems are solved using IPOPT, and the update of NLP sensitivity and Schur complement are done manually using MATLAB.

**$\beta = 1$, with measurement noise** With $\beta = 1$, as in [25], $z_2$ and $v$ are fixed, and only $F/V$ is used as the manipulated variable. This leads to a nearly linear model and is the simplest case we consider. Fig. 6.1 shows the state profile solved by the serial approach with 10% set point change and 5% measurement noise. It could be seen that amsNMPC (with $N_s = 2,3$) behaves identically as ideal NMPC and asNMPC, with the difference at the beginning due to different start-up strategies. Note that in this case since the system is almost linear, sensitivity error due to discarding higher order terms $O(|p-p_0|^2)$ could be neglected, and Assumptions 5 and 6 hold from Chapter 5.

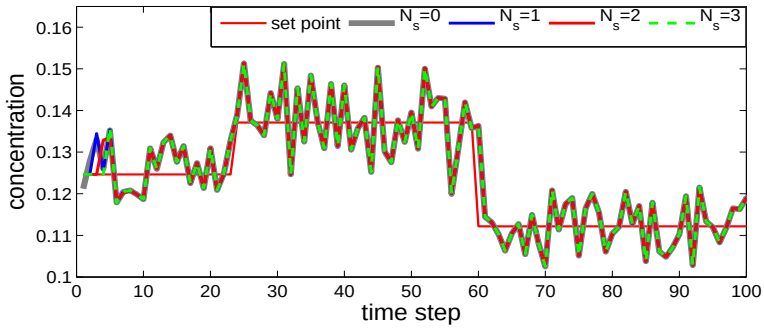



Figure 6.1: Performance with 10% set point change and 5% measurement noise

**$\beta = 3$, with set point change** Assuming there is no measurement noise or plant-model mismatch, we first change the set points of the two states by 10% at $time = 25$ and then reverse the set point change at $time = 60$. The state profiles are shown in Fig. 6.2. The parallel approach behaves exactly the same as the serial approach and ideal NMPC, asNMPC and amsNMPC (with $N_s \leq 3$) all have identical performance in tracking the set point

with small overshoots. In this case, since there is no uncertainty in the process, the memory effect does not exist.

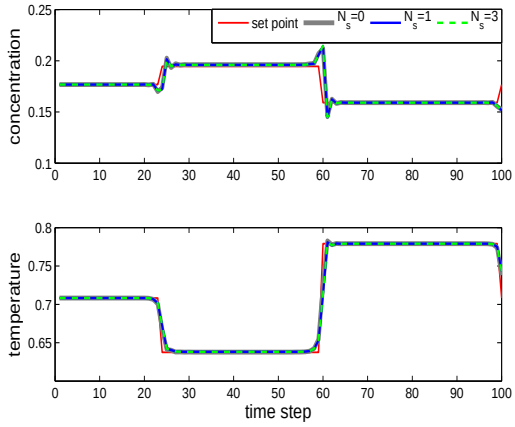



Figure 6.2: Performance with 10% set point change

$\beta = 3$**, with measurement noise** In addition to the set point change, Gaussian measurement noise is added to the plant with a standard deviation of 3% of the set point, which is relatively small. In this case the parallel approach still behaves identically as the serial approach, so only the performance of the latter is shown here. Since there is measurement noise, an on-line update of the state is introduced, and clipping is applied to deal with active set changes. From the state profiles shown in Fig. 6.3, the performance of amsNMPC is very similar to ideal NMPC and asNMPC. Also it could be noticed that different $N_s$ leads to no difference among the states. this is because the noise level is not large enough to cause large sensitivity error and strong memory effect.

When the noise level is increased to 5%, the results in Fig. 6.4 are observed. Through the comparison of Fig. 6.4(a) and Fig. 6.4(b), a performance difference can be observed between the parallel and serial approaches. First, with $N_s = 3$, the serial approach has larger offset from set point than the parallel approach between $time = 50$ and 60 due to

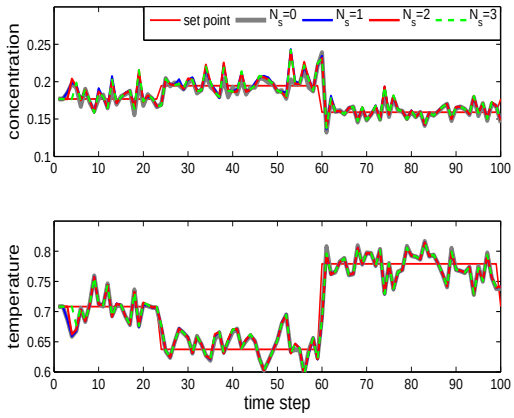
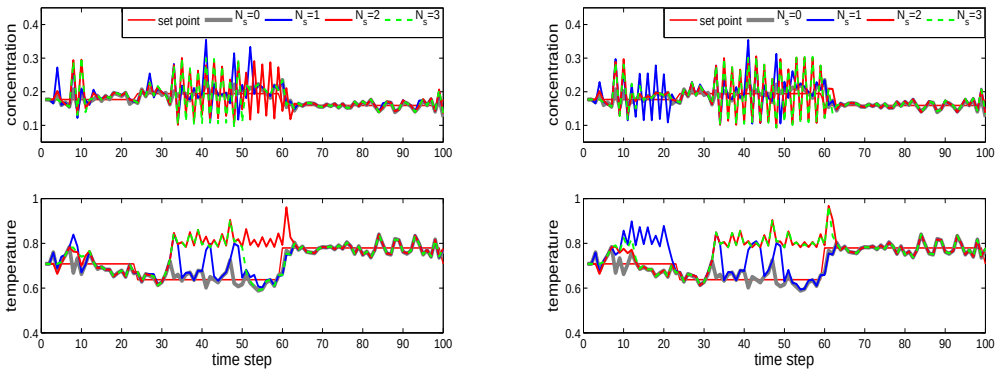



Figure 6.3: Performance with 10% set point change and 3% measurement noise

the use of previous, inaccurate KKT linearizations. Second, we observe the influence of memory effect when $N_s$ gets large. asNMPC ($N_s = 1$) generally performs worse than ideal NMPC, and performance of amsNMPC deteriorates as $N_s$ increases, as offsets from the set point grow.



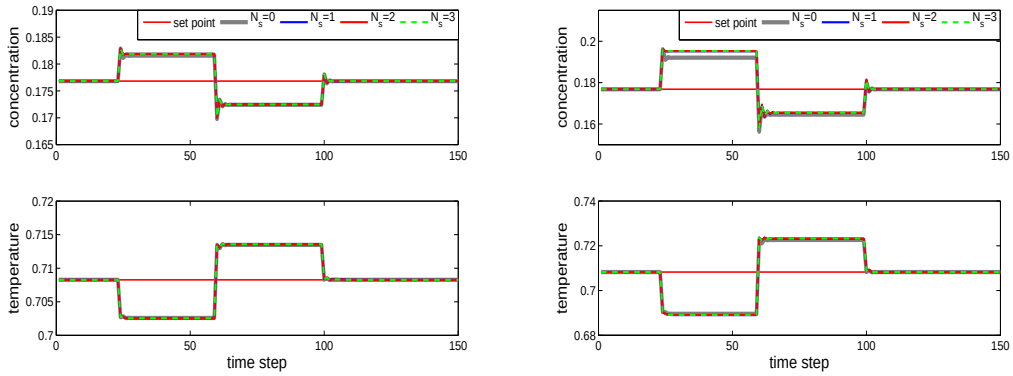

(a) Parallel approach (b) Serial approach

Figure 6.4: Performance with 10% set point change and 5% measurement noise

If we compare Fig. 6.4(b) with Fig. 6.1, we observe that although the same set point change and measurement noise are used, amsNMPC works much better with the less nonlinear

model, $\beta = 1$.

**$\beta = 3$, with plant-model mismatch** Instead of measurement noise, plant-model mismatch is added by changing the parameter $k'$ in (6.2a) and (6.2b) in the simulation model. Again the performance of the parallel approach and the serial approach is identical, so only the serial approach is shown. In Fig. 6.5(a) $k'$ is first increased by 10% then decreased by the same amount, while in Fig. 6.5(b) $k'$ is changed by 30%. Through comparison of these two subfigures, it could be seen that when the plant-model mismatch level is relatively small, amsNMPC performs identically as ideal NMPC and asNMPC. If the mismatch level is large, amsNMPC produces a larger offset than ideal NMPC and asNMPC. However, no matter what kind of NMPC controller is used, offset is unavoidable. The offset could be eliminated by adding an integrator to the controller or by adding a state observer.




(a) 10% plant-model mismatch

(b) 30% plant-model mismatch

Figure 6.5: Performance of the serial approach with plant-model mismatch

### 6.2.2 Distillation Column

Our second example is a large-scale distillation column. Distillation columns are used to separate feed streams and to refine final products. The model we study is a propane-propylene distillation column, also known as a C3 splitter. It is used to separate propane and propylene, which are the two products, and the feed has 2 compounds. The top stream of the C3 column consists of propylene while the bottom product contains propane. Modeled on an actual refinery unit [20], the binary distillation column consists of $N = 158$ trays with feed on the 43rd tray, a total condenser, and a thermosyphon reboiler. Generally the distillation model are nonlinear functions of temperature, pressure and composition, and consists of dynamic mass balance, equilibrium, summation and heat balance (MESH equations) for each tray along with vapor-liquid equilibrium, and tray hydraulics as listed in (6.4) - (6.7b). The overall mass balance on each tray is given by:

$$\begin{aligned} \frac{dM_1}{dt} &= L_2 - L_1 - V_1 && \text{(6.4a)} \\ \frac{dM_i}{dt} &= F_i + L_{i-1} - L_i + V_{i+1} - V_i && \text{(6.4b)} \\ \frac{dM_N}{dt} &= V_{N-1} - D - L_N - V_N && \text{(6.4c)} \\ i &= 1, \ldots, N \end{aligned}$$

where $N$ is the number of trays; $M_i, F_i$ are the liquid holdup and feed flowrate on the $i$th tray respectively; $L_i, V_i$ are the liquid and vapor leaving the $i$th tray respectively.

If we consider only liquid holdup, the model can be simplified by eliminating variation of pressure on each tray. This formulation allows the density, holdup, pressure defining equations to be eliminated. As a result the simplified component balances on each tray are:

$$\frac{M_1 dx_{1,j}}{dt} = L_2(x_{2,j} - x_{1,j}) - V_1(y_{1,j} - x_{i,j}) \tag{6.5a}$$

$$\frac{M_i dx_{i,j}}{dt} = F_i(x_{i,j} - x^F_{i,j}) + L_{i+1}(x_{i+1,j} - x_{i,j}) + V_{i-1}(y_{i-1,j} - x_{i,j}) - V_i(y_{i,j} - x_{i,j}) \tag{6.5b}$$

$$\frac{M_N dx_{N,j}}{dt} = V_{N-1}(y_{N-1,j} - x_{N,j}) - V_N(y_{N,j} - x_{N,j}) \tag{6.5c}$$

$$i = 1,\ldots,N$$

$$j \in COMP$$

where $COMP$ is the set of components; $x_{i,j}, y_{i,j}, x^F_{i,j}$ are the liquid composition and vapor composition and feed composition of the component $j$ on the $i$th tray respectively.

The energy balance on each tray is given by:

$$\frac{M_1 dh^L_1}{dt} = L_2(h^L_2 - h^L_1) - V_1(h^V_1 - h^L_1) + Q_r \tag{6.6a}$$

$$\frac{M_i dh^L_i}{dt} = F_i(h^L_i - \kappa^L_f) + L_{i+1}(h^L_{i+1} - h^L_i) + V_{i-1}(h^V_{i-1} - h^L_i) - V_i(H^V_i - h^L_i) \tag{6.6b}$$

$$\frac{M_N dh^L_N}{dt} = V_{N-1}(h^V_{N-1} - h^L_N) - V_N(h^V_N - h^L_N) - Q_c \tag{6.6c}$$

$$i = 1,\ldots,N$$

$$j \in COMP$$

where $h^L_i, h^V_i, \kappa_f$ are the liquid, vapor and feed enthalpy respectively; $Q_r$ and $Q_c$ are the reboiler and condenser loads. Finally, the equilibrium and summation equations on each tray are:

$$y_{i,j} = K_{i,j}(T_i, P_i, x_i)x_{i,j} \tag{6.7a}$$

$$0 = \sum_{j \in COMP} y_{i,j} - \sum_{j \in COMP} x_{i,j} \tag{6.7b}$$

$$i = 1,...,N$$

$$j \in COMP$$

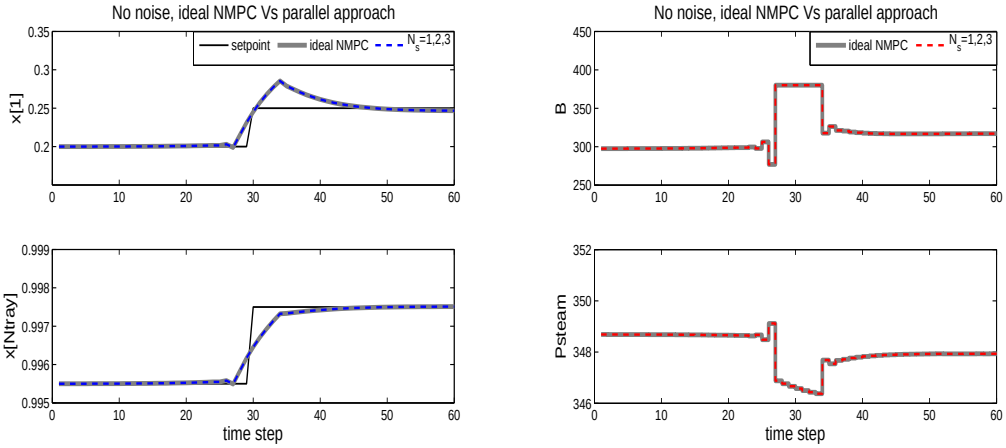
where $T_i, P_i$ are the temperature and pressure on each tray respectively. For the propane-propylene system, equilibrium constants are determined from DePriester nomographs. A detailed description of these equations, their reformulation from index 2 to index 1, along with model of the thermosyphon reboiler, can be found in [20]. It is assumed that the pressure in the column is controlled, and pressure drop is constant from tray to tray.

To obtain plantwide optimization requirements the compositions of the top and bottom products must be controlled. In this study the controlled states are the concentrations of propylene of the first and last trays, while the manipulated variables are the steam pressure in the reboiler and bottoms flow rate. The objective is to keep the states at their setpoints, so the objective function is composed of quadratic tracking terms.

For this case study we compare the performance of ideal NMPC, asNMPC and the parallel approach of amsNMPC. The cases we consider here are pure setpoint change and setpoint change with measurement noise. We tried different horizon lengths to assure that the solution $u = \kappa(z)$ exists and converges to the optimum. The horizon length we use is 25. After discretizing the ordinary differential equations with orthogonal collocation, the NLP problem has 79750 variables and 79700 constraints. Solution of Problem (4.3) requires 90 CPU seconds, but with a sampling time of only 60 s; practical implementation requires $N_s > 1$, and amsNMPC must be used. On the other hand, the on-line computation (solving (3.23)) requires less than 1 CPU second. To shown the difference between different NMPC formulations, we assume that we can choose $N_s = 0$ (iNMPC), $N_s = 1$ (asNMPC) and $N_s = 2, 3$.

Because the size of the corresponding KKT matrix exceeds the capabilities of MATLAB, and the Schur complement decomposition (5.43) is not yet implemented in sIPOPT, only the parallel approach with sIPOPT is shown here.

**Performance with setpoint change** For this case we change the set point at $time = 30$ and the change is known in advance. From Fig. 6.6 we could see that both iNMPC and




(a) States profile (b) Control profile

Figure 6.6: Performance of the parallel approach with setpoint change

the parallel approach are able to catch the set point change, and they behave identically. Moreover, since there is no uncertainty, the memory effect does not exist, and $N_s$ does not make any difference.

**Performance with setpoint change and measurement noise** For this case, we change the setpoint at $t = 30$ and introduce Gaussian noise with a standard deviation of 1% of the setpoint on all outputs but $x[Ntray]$. If this level of noise is added to $x[Ntray]$, its upper bound will be violated, so only 0.1% noise is added to $x[Ntray]$. The performance of ideal NMPC and the parallel approach with $N_s = 1, 2, 3$ are compared. The noise level is then increased to 3% and 5%. For each level, the noise is generated randomly only once and then the same noise sequence is used by iNMPC, asNMPC and the parallel approach, so that all results are consistent. Fig.6.7 shows that with 1% of noise, set points are tracked very well, and there is not much difference among different cases, in the state profile as well

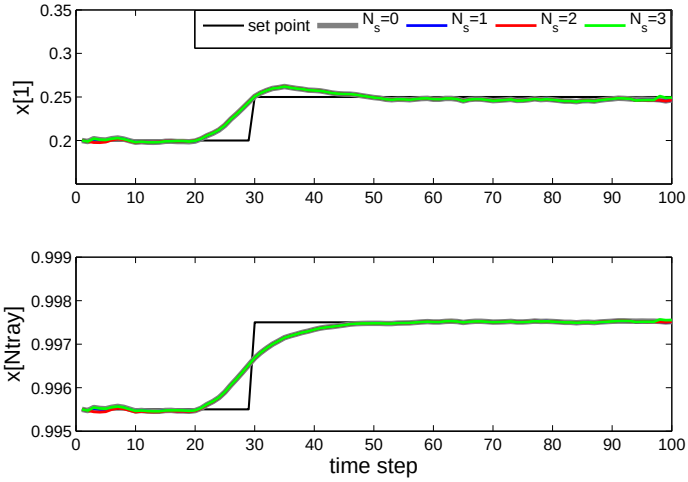
as the control profile. Increasing the noise level to 3% in Fig.6.8, shows that the difference in state and control profiles becomes larger as noise level increases. With 5% additive noise, Fig.6.9 shows that the difference is even more significant. For large noise levels, we would expect the performance of amsNMPC to deteriorate as $N_s$ increases, because the linearization used for the sensitivity update becomes less accurate. On the other hand, if the model responses to noise are not highly nonlinear, then performance loss with increasing $N_s$ does not occur. We observe this effect in this case study; the state profiles generated with different $N_s$ remain similar in all noise cases.

For the serial approach, we also worked on the nominal case with pure setpoint change and its performance is identical to the parallel approach. In case of measurement noise or plant-model mismatch, if $N_s \geq 1$, the NLP sensitivity needs to be updated. Because general implementation of the Schur complement sensitivity modification is part of our future work, the serial case with noise is not presented here.

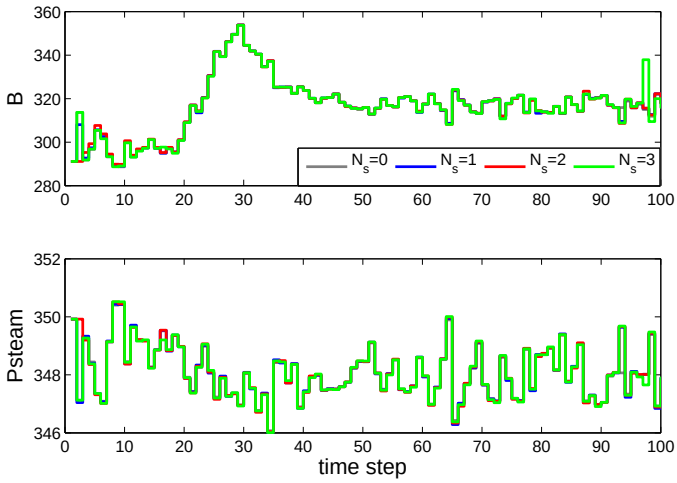
### 6.2.3 asNMPC/amsNMPC on Grimm Examples

As shown in [24], ISS may be lost if there are state constraints, or terminal constraints coupled with short optimization horizons. This necessarily leads to a discontinuous feedback law and cost function. However, with reformulation, robustness could be obtained. One way is to use a horizon long enough such that an optimal solution is obtained. This gives the controller more degrees of freedom to move around. The other way is to apply soft constraints on states or outputs and add penalties to the objective function, as stated in Chapter 4.

In [24], Grimm et al. demonstrated three NMPC examples that lack robustness because of state constraints, terminal region or a short horizon. In Chapter 4 we study those examples and show that robustness could be obtained by applying a longer horizon or reformulating

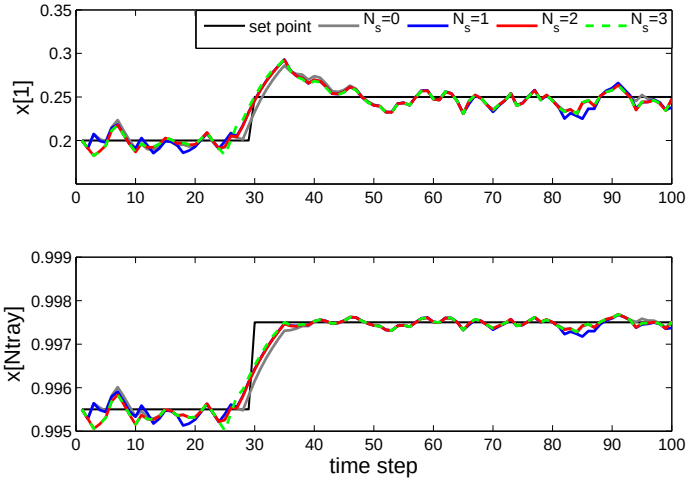



(a) State profiles

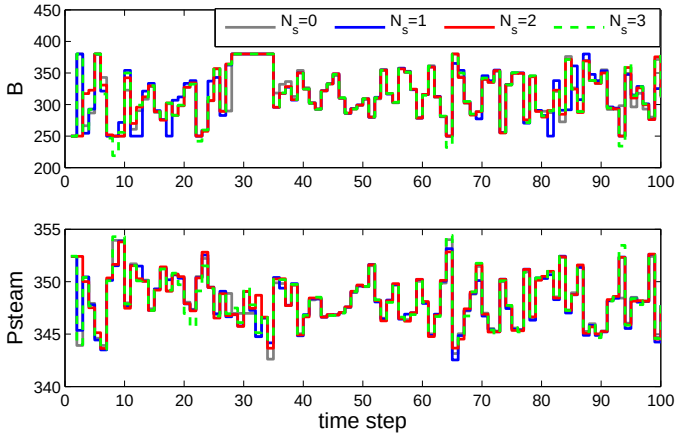



(b) Control profiles

Figure 6.7: 1% additive noise. The results of ideal NMPC are shown in gray; the results of the parallel amsNMPC are shown in blue, red and green with $N_s = 1, 2, 3$.

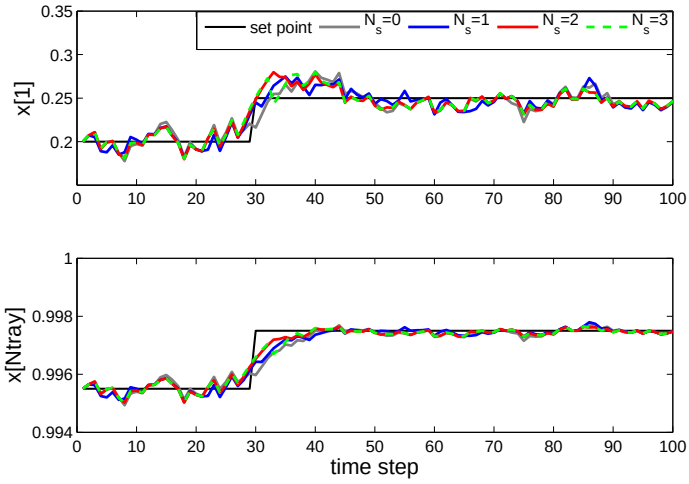



(a) State profiles

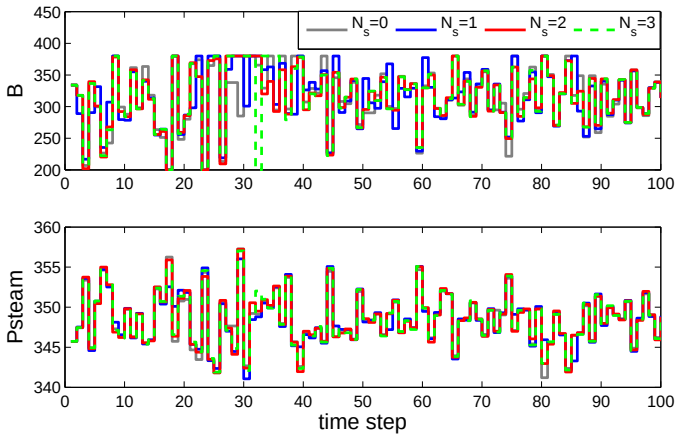



(b) Control profiles

Figure 6.8: 3% additive noise. The results of ideal NMPC are shown in gray; the results of the parallel amsNMPC are shown in blue, red and green with $N_s = 1, 2, 3$.




(a) State profiles




(b) Control profiles

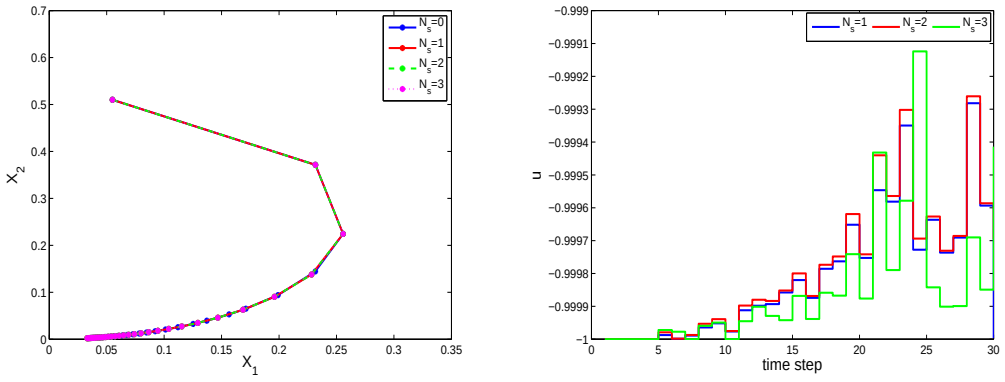
Figure 6.9: 5% additive noise. The results of ideal NMPC are shown in gray; the results of the parallel amsNMPC are shown in blue, red and green with $N_s = 1, 2, 3$.

the NLP with soft constraints. In this chapter we are going to show that robustness will not be lost if asNMPC or amsNMPC are applied.

**Example 1: Artstein's circles with state constraints**

In Chapter 4 we have shown that by reformulating the MPC problem with soft constraints robustness is obtained. Here we show that applying amsNMPC will not deteriorate robustness.

Again we introduce additive disturbance, which is Gaussian noise with zero mean and standard deviation of 0.05. Horizon length is $N = 10$ and simulation time is 30. We choose $N_s = 1, 2, 3$. The MPC is reformulated with soft constraints. The results are plotted in Fig. 6.10. From Fig. 6.10(a) we could see that amsNMPC with different $N_s$ lead to the similar results as iNMPC, so robustness is preserved; and it could be observed in Fig. 6.10(b) that the control profiles generated with $N_s = 1, N_s = 2, N_s = 3$ are very close.




(a) Trajectory of **x**,noise=0.05 (b) Trajectory of u, noise=0.05

Figure 6.10: Trajectory of **x** and $u$ with soft constraints. iNMPC is shown in blue; $N_s = 1$ (asNMPC) is shown in red; $N_s = 2$ is shown in green; $N_s = 3$ is shown in magenta.

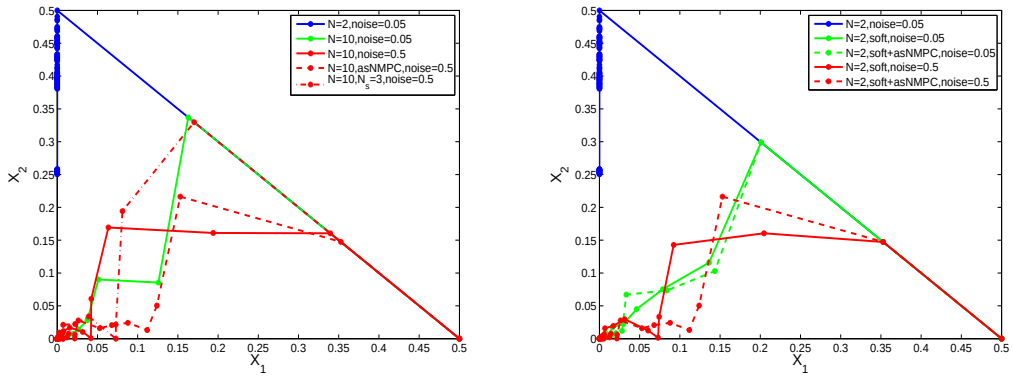
**Example 2: origin as terminal constraint**

In order to show that applying the advanced step strategy will not deteriorate robustness, we study the following cases:

1. Apply amsNMPC with $N_s = 1,3$ to the problem with only hard constraints and with $N = 10$;
2. Apply asNMPC to the soft constrained problem with $N = 2$.

$N = 10$**, amsNMPC with** $N_s = 1,3$**, no soft constraints** In this case the robustness is obtained by using a long enough horizon with $N = 10$. The noise is additive disturbance, which is zero mean Gaussian noise with standard deviation of 0.05. We then increase the standard deviation to 0.5 and observe that the states still converge to the origin, shown in Fig. 6.11(a). Then we apply asNMPC and amsNMPC with $N = 3$ and observe that robustness is not lost when the advanced-step strategy is applied. However we could see that the state converges the fastest without the advanced-step strategies.

$N = 2$**, asNMPC, with soft constraints** In this case the horizon length $N = 2$ and robustness is obtained by use of soft constraints. We increase the standard deviation of the noise from 0.05 to 0.5 and observe that the states still converge to the origin, shown in Fig. 6.11(b). Then we apply asNMPC and observe that robustness is not lost when the advanced-step strategy is applied. And again the state converges faster without the advanced-step strategy.




(a) $N = 10$, amsNMPC, no soft constraints (b) $N = 2$, asNMPC, with soft constraints

Figure 6.11: Trajectory of $\mathbf{x}$. $N = 2$ without soft constraint is shown in blue; noise standard deviation=0.05 is shown in green; noise standard deviation=0.25 is shown in red.

**Example 3: unit disk as terminal constraint**

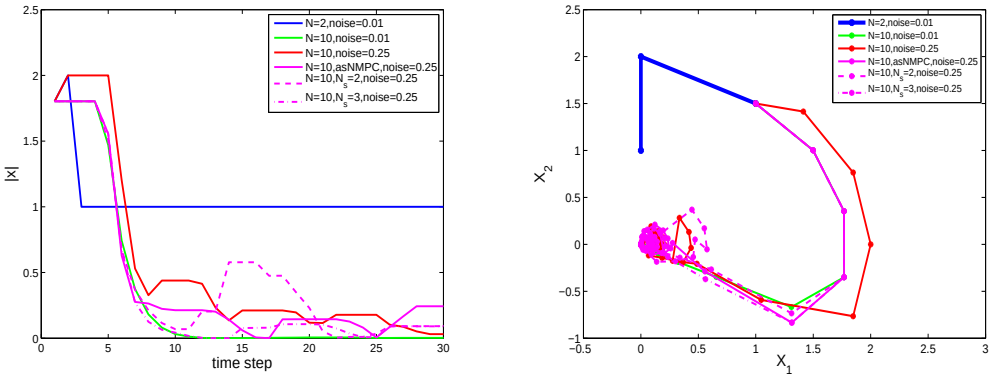
In order to show that applying the advanced step strategy will not deteriorate robustness, we study the following cases:

1. Apply amsNMPC with $N_s = 1, 2, 3$ to the problem with only hard constraints and with $N = 10$;

2. Apply asNMPC to the soft constrained problem with $N = 2$.

$N = 10$**, amsNMPC with** $N_s = 1, 2, 3$**, no soft constraints** In this case the robustness is obtained by using a long enough horizon $N = 10$. The noise is additive disturbance, which is zero mean Gaussian noise with standard deviation of 0.01. We then increase the standard deviation to 0.25 and observe that $|\mathbf{x}|$ decreases to 0 and the states converge to the origin, shown in Fig. 6.12. It is worth pointing out that with a larger level of noise $|\mathbf{x}|$ decreases to 0 at a lower rate and may grow before it eventually decreases to 0. Then we

apply asNMPC and amsNMPC with $N = 2, 3$ and observe that robustness is not lost when the advanced-step strategy is applied.



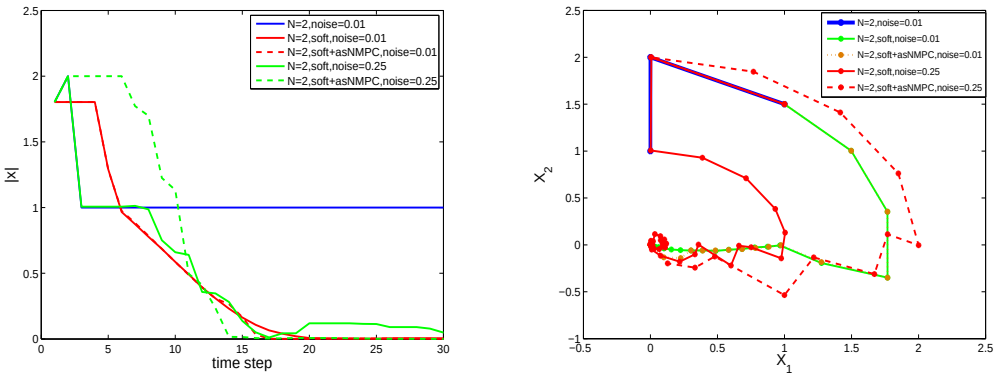

(a) Trajectory of $|\mathbf{x}|$ (b) Trajectory of $\mathbf{x}$

Figure 6.12: $N = 10$, amsNMPC, no soft constraints, noise=0.01 or 0.25.

**$N = 2$, asNMPC, with soft constraints** In this case the horizon length $N = 2$ and robustness is obtained by use of soft constraints. We increase the standard deviation of the noise from 0.01 to 0.25 and observe that the states still converges to the origin, shown in Fig. 6.13. Then we apply asNMPC and observe that robustness is not lost when the advanced-step strategy is applied. It is worth pointing out that with a larger level of noise, the convergence rate is smaller. Moreover, if we compare Fig. 6.13(a) with Fig. 6.12(a), we could see that when the standard deviation of noise is 0.25, case 2 converges more slowly than case 1, but there are fewer oscillations when $|\mathbf{x}|$ gets small.

## 6.3 Concluding Remarks

In this chapter we show the performance of iNMPC, asNMPC and amsNMPC with a CSTR model and a large-scale distillation column model. By applying different levels of measure-




(a) Trajectory of $|\mathbf{x}|$ (b) Trajectory of $\mathbf{x}$

Figure 6.13: $N = 2$, asNMPC, with soft constraints, noise=0.01 or 0.25.

ment noise and different values of $N_s$ to models with different nonlinearity, we observe that when there is no uncertainty in the system, the performance of iNMPC, asNMPC and amsNMPC are identical. Moreover, amsNMPC behaves similarly as iNMPC and asNMPC when the model nonlinearity is not strong or with small measurement noise or $N_s$ value. However, with large measurement noise and a strongly nonlinear system, the memory effect becomes more significant with increasing $N_s$. As a result the performance of amsNMPC deteriorates.

We also show that for MPC formulations that lack robustness, robustness could be obtained by using a longer horizon or reformulating the NLP with soft constraints and $\ell_1$ penalties. Moreover, robustness will not be lost when the advanced-(multi-)step strategy is applied.

# Chapter 7

# Economic Nonlinear Model Predictive Control

In this chapter we discuss the formulation of Economic NMPC. It is formulated by combining the two-layer RTO advanced control approach that we discussed in Chapter 1. First, real-time optimization (RTO) optimizes an economic objective with steady state models, leading to a setpoint handled by the lower-level control layer. The advanced control layer (using, e.g., NMPC) then tracks the setpoint to achieve a new steady state. However, this two-layer approach assumes that model disturbances and transients are neglected in the RTO layer [17]. Moreover, model inconsistency between layers and unresolved transient behavior may lead to unreachable setpoints [54]. Also, since the control layer has no information on dynamic economic performance, it may generate trajectories that simply track suboptimal setpoints to steady state [51, 31].

Recent studies on dynamic real-time optimization (D-RTO) have reported significant performance improvements with economically-oriented NMPC formulations [59, 51, 17, 2]. It uses economic stage costs to drive the controller directly. As a result, Economic NMPC usually leads to better economic performance. However, unlike setpoint tracking NMPC, there are no tracking terms in the objective function and no terminal constraints. Therefore, the system may not converge to a steady state, and instability occurs. Therefore, stability theory supporting economically-oriented NMPC requires development beyond the mature

results for setpoint tracking based on a discrete Lyapunov analysis. This problem formulation and stability analysis must be modified to ensure a stable and robust D-RTO implementation, especially if optimum steady state operation is expected. In this chapter, we propose a strategy to guarantee stability of Economic NMPC, and apply two case studies to evaluate the stabilizing strategy.

## 7.1 Formulation and Lyapunov Stability Analysis

The Economic NMPC problems have the same general formulation as (2.4). It usually shares the same model and constraints as NMPC, but the tracking terms in the objective function are replaced by economic stage cost, such as net profit. Additional variable bounds might also be added to make the problem formulation more practical. Such a controller formulation maximizes an economic objective while accounts for disturbances simultaneously using a well-tuned dynamic process model. As explored in [7], this formulation may lead to significant improvements in process performance. In particular, *artificial setpoints* used in (4.3) and determined from a steady state optimization are no longer needed. This single stage D-RTO thus has advantages over conventional RTO because the NMPC profile is driven by economic stage costs instead of mere tracking terms. As a result, this usually leads to better economic performance. In fact, significant differences between tracking to optimal steady set points and economic NMPC were observed in the case studies in [2], even for nominal cases.

However, Economic NMPC is not as simple as replacing the stage costs in (2.4). Unlike setpoint tracking NMPC, the stability analysis of the Economic NMPC is difficult, mostly because the objective functions would not satisfy the conditions of Lyapunov functions (Assumption 2 in Chapter 4), and cannot be used as the Lyapunov function directly.

Therefore, we need to define an appropriate Lyapunov function such that stability can be guaranteed. Stability properties of different formulations of economically oriented NMPC have been studied in [13] with the assumption of strong duality of steady-state problem. In [4] and [28], the assumption of strong duality is replaced by dissipativity and strong second order conditions, respectively, and it is proved that the best feasible equilibrium state is asymptotically stable with terminal region. Rawlings et al. proposed a setpoint tracking controller in [52], which handles unreachable setpoints better than traditional setpoint tracking controllers, and discuss its nominal stability. However the cost function is not strictly decreasing, so the Lyapunov stability theories for stability analysis no longer apply. Huang and Biegler proposed a Economic NMPC controller for a cyclic process by adding a periodic constraint and proved its nominal stability in [27]. Huang et al [29] studied both nominal and robust stability properties of an Economic NMPC with infinite horizon for cyclic processes.

We assume that Economic NMPC drives the process to a steady state. There are several reasons for making this assumption. In particular, production planning with economic models over long time scales is based on steady state models, and consistency with these models must be ensured. Also, plant stability and robustness are easier to analyze under steady state assumptions. Finally, steady state (or cyclic steady state) operation is easier to monitor, analyze and manage.

To ensure that Economic NMPC converges to steady state we revisit NMPC stability analysis through the following constructions from [13]. We first define the steady state problem and establish a suitable Lyapunov function by subtracting the optimal steady state from the original system and adding a *rotated* term. Here, the original system is asymptotically stable at the optimum if the rotated system is asymptotically stable at the origin. In addition, we observe that the rotated Lyapunov function is strictly decreasing.

To define implicit reference values for the states and controls, we consider the steady state optimization problem given by:

$$\min \psi(z,v),\ \text{s.t. } z = f(z,v), z \in \mathbb{X}, v \in \mathbb{U} \tag{7.1}$$

with the solution given by $(z^*, v^*)$. We then define the rotated states $z_l$ and controls $v_l$ by subtracting the optimum steady state from the predicted values of Problem (2.4):

$$\bar{z}_l = z_l - z^*, \quad \bar{v}_l = v_l - v^* \tag{7.2}$$

and the transformed state evolves according to

$$\bar{z}_{l+1} = \bar{f}(\bar{z}_l, \bar{v}_l) = f(\bar{z}_l + z^*, \bar{v}_l + v^*) - z^* \tag{7.3}$$

and $\bar{z}_l \in \bar{\mathbb{X}}$ and $\bar{u}_l \in \bar{\mathbb{U}}$, where $\bar{\mathbb{X}}$ and $\bar{\mathbb{U}}$ are the corresponding sets for the transformed system. Similarly we define the transformed stage cost and terminal cost as:

$$\bar{\psi}(\bar{z}_l, \bar{v}_l) = \psi(\bar{z}_l + z^*, \bar{v}_l + v^*) - \psi(z^*, v^*) \tag{7.4}$$

$$\bar{\Psi}(\bar{z}_N) = \Psi(\bar{z}_N + z^*) - \Psi(z^*) \tag{7.5}$$

At the optimal steady state of (7.1), $z_l = z^*,\ v_l = v^*$, therefore $\bar{z}_l = 0,\ \bar{v}_l = 0, \bar{z}_{l+1} = f(z^*, v^*) - z^* = z^* - z^* = 0$; $(0,0)$ is the optimal steady state for the transformed system. Moreover we have

$$\begin{aligned} \bar{\psi}(0,0) &= \psi(z^*, v^*) - \psi(z^*, v^*) = 0 \\ \bar{\Psi}(0) &= \Psi(z^*) - \Psi(z^*) = 0 \end{aligned}$$

The transformed iNMPC NLP is now given by:

$$\begin{aligned} &\min && \bar{\Psi}(\bar{z}_N) + \sum_{l=0}^{N} \bar{\psi}(\bar{z}_l, \bar{v}_l) \\ &\text{s.t.} && \bar{z}_{l+1} = \bar{f}(\bar{z}_l, \bar{v}_l), \quad l = 0, \ldots, N \\ &&& \bar{z}_0 = x(k) - x^*(k),\ \bar{z}_l \in \bar{\mathbb{X}},\ \bar{v}_l \in \bar{\mathbb{U}},\ \bar{z}_N \in \bar{\mathbb{X}}_f, \end{aligned} \tag{7.6}$$

And we reformulate Problem (7.6) to the formulation of Problem (4.3) by using soft constraints and adding penalty terms to the objective function. Therefore $\bar{\mathbb{X}} = \Re^n$.

Being the transformed stage costs and terminal cost, $\bar{\psi}$ and $\bar{\Psi}$ may not hold for Assumption 2. As shown in [13, 31], we introduce the rotated stage costs and terminal cost:

$$L(\bar{z}_l, \bar{v}_l) = \bar{\psi}(\bar{z}_l, \bar{v}_l) + \lambda^{*,T}(\bar{z}_l - \bar{f}(\bar{z}_l, \bar{v}_l)) \tag{7.7}$$

$$L_f(\bar{z}) = \bar{\Psi}(\bar{z}) + \lambda^{*,T}\bar{z}. \tag{7.8}$$

where $\lambda^*$ is the Lagrange multiplier of the equality constraint in (7.1). It is shown in [13, 31] that substitution of $L(\bar{z}_l, \bar{v}_l)$ and $L_f(\bar{z}_N)$ does not change the optimal solution of (7.6). Therefore, we could analyze the stability property of (7.6) while substituting $\bar{\psi}(\bar{z}_l, \bar{v}_l)$ and $\bar{\Psi}(\bar{z}_N)$ with $L(\bar{z}_l, \bar{v}_l)$ and $L_f(\bar{z}_N)$. If $L(\bar{z}_l, \bar{v}_l)$ and $L_f(\bar{z}_N)$ are strongly convex, Assumption 2 is satisfied. Therefore Problem (7.6) is asymptotically stable.

On the other hand, if $L(\bar{z}_l, \bar{v}_l)$ and $L_f(\bar{z}_N)$ are not strongly convex, regularization terms must be added to the original stage cost:

$$\varphi(z_l, v_l) := \psi(z_l, v_l) + 1/2\|z_l - z^*, v_l - v^*\|_Q^2 \tag{7.9}$$

$Q$ has to be large enough for $L(\bar{z}_l, \bar{v}_l)$ and $L_f(\bar{z}_N)$ to be strongly convex. However, if $Q$ is too large, economic performance might deteriorate. We use the Gershgorin theorem to find the 'minimal' $Q$ to make $L$ strongly convex.

As stated by the Gershgorin theorem, the eigenvalues $\sigma_i$ of a matrix $A = \nabla^2 L(\bar{z}_l, \bar{v}_l)$ satisfy the following inequalities:

$$a_{i,i} - \sum_{i \neq j} |a_{i,j}| \leq \sigma_i \leq a_{i,i} + \sum_{i \neq j} |a_{i,j}|. \tag{7.10}$$

where $a_{i,i}$ are diagonal elements of $A$ and $a_{i,j}$ are non-diagonal elements. If we add $Q$, a diagonal matrix with $q_i$ on its diagonal, the eigenvalues of $A + Q$, which are $\sigma_i + q_i$, have

to satisfy

$$0 < q_i + a_{i,i} - \sum_{i \neq j} |a_{i,j}| \leq \sigma_i + q_i, \tag{7.11}$$

In order for the new $L$ to be strongly convex, $A+Q$ has to be positive definite, therefore $\sigma_i + q_i$ has to be positive, i.e., $q_i > \sum_{i \neq j} |a_{i,j}| - a_{i,i}$. And (7.11) has to be satisfied for every $z$ and $v$. Next we evaluate the effect of regularization terms with two case studies.

## 7.2 Case Studies

### 7.2.1 CSTR

We illustrate some of the above concepts on a case study of a CSTR from [13] with a first order reaction $A \rightarrow B$. From a mass balance, we derive the following dynamic model

$$\begin{aligned} \frac{dc_A}{dt} &= \frac{\dot{m}}{V}(c_{Af} - c_A) - kc_A \\ \frac{dc_B}{dt} &= \frac{\dot{m}}{V}(-c_B) + kc_A. \end{aligned} \tag{7.12}$$

Here $c_A$ and $c_B$ denote the concentrations of components A and B, respectively. The reactor volume is $V = 10$ l, and the rate constant $k = 1.2$ l/(mol·min). Further, $\dot{m}$ denotes the manipulated input in l/min, and $c_{Af} = 1$mol/l denotes the feed concentration. Using state feedback NMPC, the economic stage cost in (7.1) is selected as

$$\psi(c_A, c_B, \dot{m}) = -\dot{m}\left(2c_B - \frac{1}{2}\right). \tag{7.13}$$

Different from [13], we set the variable bounds as:

$$10 \leq \dot{m} \leq 20 \tag{7.14}$$

$$0.45 \leq c_B \leq 1, \tag{7.15}$$

When large disturbances occur, variable bounds might be violated. Due to physical reasons, bounds on manipulated variables cannot be violated (hard constraints); the violation of state variable bounds may be tolerated when necessary. Therefore we treat the bounds of $c_B$ as soft constraints. As described in Chapter 2 and [12], we add a non-negative slack variable $\varepsilon$ to the lower and upper bounds of $c_B$,

$$0.45 - \varepsilon \leq c_B \leq 1 + \varepsilon, \tag{7.16}$$

and we add an exact $\ell_1$ penalty function that contains the slack variable to the stage cost:

$$\psi(c_A, c_B, \dot{m}) = -\dot{m}\left(2c_B - \frac{1}{2}\right) + \rho\varepsilon \tag{7.17}$$

where $\rho$ is a number large enough to drive $\varepsilon$ to zero. In our case we set $\rho = 1000$. Also, we note that the optimal steady states (from Problem (7.1)), are $c_A^* = 0.5, c_B^* = 0.5, \dot{m}^* = 12$.

**Regularization of the Stage Cost**

To ensure that the rotated stage cost is strongly convex, regularization terms $\frac{1}{2}[q_A(c_A - c_A^*)^2 + q_B(c_B - c_B^*)^2 + q_{\dot{m}}(\dot{m} - \dot{m}^*)^2]$ are added to the stage cost.

It can be shown that the Hessian matrix of the steady state optimization problem with regularized stage cost is

$$\nabla^2 V = A + Q = \begin{bmatrix} q_A & 0 & 1 \\ 0 & q_B & 0 \\ 1 & 0 & q_{\dot{m}} \end{bmatrix} \tag{7.18}$$

and we find the following Gershgorin bounds:

$$q_A > 1, q_B > 0, q_{\dot{m}} > 1 \tag{7.19}$$

Moreover, through an eigenvalue calculation, we can find smallest positive weights so that $q_A + q_B + Q_{\dot{m}}$ is minimized, while ensuring that $A + Q$ is positive definite. These weights correspond exactly to the bounds determined in (7.19).

**Simulation Results**

Our simulation results are organized such that we first demonstrate the effect of regularization on the CSTR with ideal NMPC with different levels of measurement noise and regularization weights. Then we proceed to show results for asNMPC where measurement noise causes the active set to change.

**Effect of Regularization** From the Gershgorin bounds, we set $q_A = 1+\delta$, $q_{\dot{m}} = 1+\delta$, $q_B = \delta$. For $\delta > 0$ we have strong convexity. The prediction horizon is chosen as $N = 30$ and we simulate for 50 sample times.

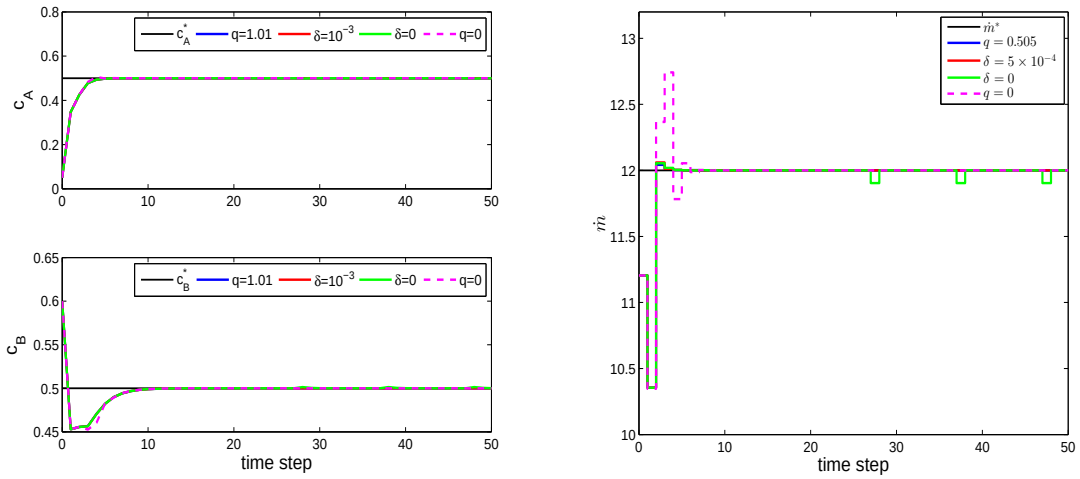
*Perfect case, no measurement noise*

We assume that all the states are known exactly, and start with the scenario where the model is known completely and there is no measurement noise. We consider different regularizations, starting with the limit, where we set the value of $\delta = 0$, and compare it with $\delta = 5 \times 10^{-4}$ and with the original weighting factors $q_{i=A,B,\dot{m}} = 1.1$ used in [13]. Finally, we consider the case without any regularization, i.e. $q_{i=A,B,\dot{m}} = 0$.

Figure 7.1 shows the state profiles and control profiles obtained by simulating the CSTR in closed loop with different weights on regularization terms.

From Figure 7.1 it can be seen that when we regularize, where $\delta > 0$, the Gershgorin weights are large enough so that their effects are the same as with the weights in [13]; i.e., the system is stable.

Note that when we have $\delta = 0$, Equation (7.19) is "almost satisfied"; thus the state profile is identical as with $q_i = 1.1$, but a few oscillations are observed in the control profile. These oscillations disappear as we increase time horizon.




(a) State profiles

(b) Control profiles

Figure 7.1: Ideal NMPC, no measurement noise. The results with $q_{i=A,B,\dot{m}} = 1.1$ are shown in blue; with $q_A = q_{\dot{m}} = 0.5 + 5 \times 10^{-4}, q_B = 5 \times 10^{-4}$ is shown in red; with $q_A = q_{\dot{m}} = 0.5, q_B = 0$ is shown in green; and with $q_{i=A,B,\dot{m}} = 0$ is shown in magenta.

Table 7.1: Cost of ideal NMPC with different noise levels

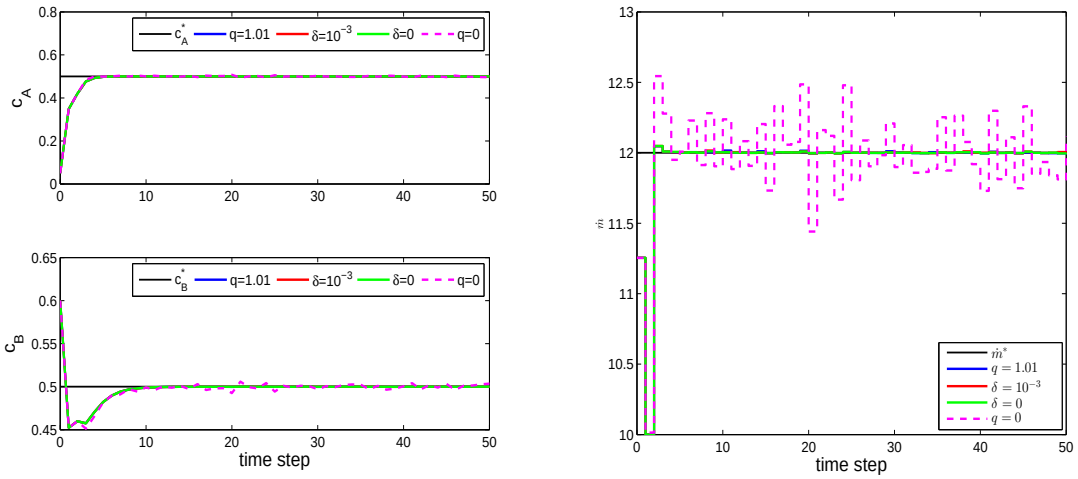
| Cost with | | No noise | 1% noise | 5% noise |
|---|---|---|---|---|
| No regularization | $q_{i=A,B,\dot{m}} = 0$ | -147.35 | -146.90 | -144.95 |
| Marginal regularization | $\delta = 0$ | -147.35 | -146.90 | -145.20 |
| Small regularization | $\delta = 0.0005$ | -147.35 | -146.90 | -145.20 |
| Large regularization | $q_{i=A,B,\dot{m}} = 1.1$ | -147.35 | -146.90 | -145.20 |

In the case without any regularization at all, $q_{i=A,B,\dot{m}} = 0$ we have an oscillatory control action at the beginning, and it takes time for the manipulated variable to converge to their steady state optimal values. Note that convergence to the steady state optimal values is not a general property of unregularized Economic NMPC. As Angeli et al. [4] have shown, there may also be cases, where a lower cost can be observed by not converging to the steady state optimal values.

The third column in Table 7.1 shows the accumulated stage cost $\sum_{k=1}^{50}[-\dot{m}_k(2c_{B,k} - \frac{1}{2})]$. It can be observed that the costs of all the cases are identical. Without noise, slack variables $\varepsilon$ of all cases are zero, and the state and control profiles with all weighting factors except $q_i = 0$ are the same, so the stage costs should be the same. Without any regularization $q_i = 0$, the state profiles are identical to the other cases, while the control profile above $\dot{m}^* = 12$ cancels with the profile below $\dot{m}^* = 12$ in order to yield the same accumulated cost.

*Cases with measurement noise at different levels*

For the two states we add measurement noise with standard deviations at 1% of their equilibrium points. Figure 7.2 shows the state profiles and control profiles. From Figure 7.2



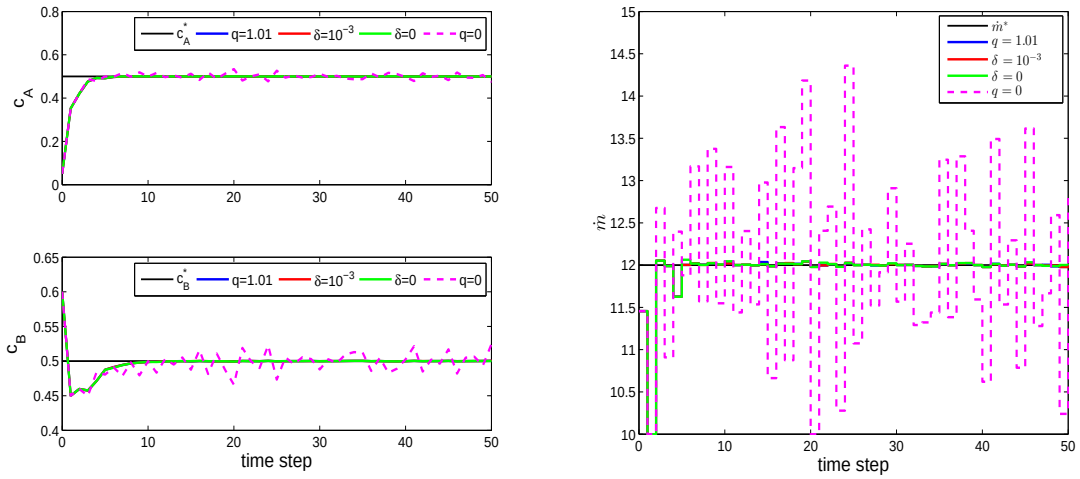

(a) State profiles

(b) Control profiles

Figure 7.2: Ideal NMPC, 1% of measurement noise. The results with $q_{i=A,B,\dot{m}} = 1.1$ are shown in blue; with $q_A = q_{\dot{m}} = 0.5 + 5 \times 10^{-4}, q_B = 5 \times 10^{-4}$ is shown in red; with $q_A = q_{\dot{m}} = 0.5, q_B = 0$ is shown in green; and with $q_{i=A,B,\dot{m}} = 0$ is shown in magenta.

we observe that because of the measurement noise, control profiles differ with different weights. But the difference is very small when (7.19) is satisfied. Without any regularization terms, $q_{i=A,B,\dot{m}} = 0$, there are oscillations in the control profile, which leads to small oscillations in the state profiles. Table 7.1 shows the accumulated stage costs. It can be seen that all of the accumulated stage costs are essentially the same for all weighting factors. However, even with a small level of measurement noise (1%), the lack of regularization ($q_{i=A,B,\dot{m}} = 0$) leads to significant oscillations in its control profile, which is unacceptable.

We then increase noise level to 5% of the equilibrium points. Figure 7.3 shows the state and control profiles. As the noise level increases, oscillations are larger in the control profile, which lead to larger oscillations in the state profiles compared with Figure 7.2. Table 7.1 shows the accumulated stage costs for this case, too. One would expect that these




(a) State profiles

(b) Control profiles

Figure 7.3: Ideal NMPC, 5% of measurement noise. The results with $q_{i=A,B,\dot{m}} = 1.1$ are shown in blue; with $q_A = q_{\dot{m}} = 0.5 + 5 \times 10^{-4}, q_B = 5 \times 10^{-4}$ is shown in red; with $q_A = q_{\dot{m}} = 0.5, q_B = 0$ is shown in green; and with $q_{i=A,B,\dot{m}} = 0$ is shown in magenta.

costs would decrease with decreasing weights on regularization terms. However, it seems

that regularization makes a positive contribution in the presence of measurement noise. Without regularization, i.e. $q_{i=A,B,\dot{m}} = 0$, the controller is not stabilizing, and we observe that its accumulated stage cost is the highest.

Moreover, from Table 7.1 we observe that accumulated stage costs tend to increase with increased noise levels. This is because the controller is optimizing based on incorrect information (without knowledge of the noise), so performance deteriorates with increased noise.

**Advanced-step NMPC with Economic Stage Costs** In this section we study the performance of asNMPC, where the noise level of 5 % is chosen so that the active sets differ for the predicted problem and the actual problem, for which a sensitivity based approximated solution is found. We set $q_{i=A,B,\dot{m}} = 1.1$ so that the controller is stable if optimal manipulated variables are injected. To better demonstrate the effect, we zoom into the first 12 sample times of closed loop simulations, and apply the advanced-step NMPC strategy to the CSTR example. We show results for three cases:

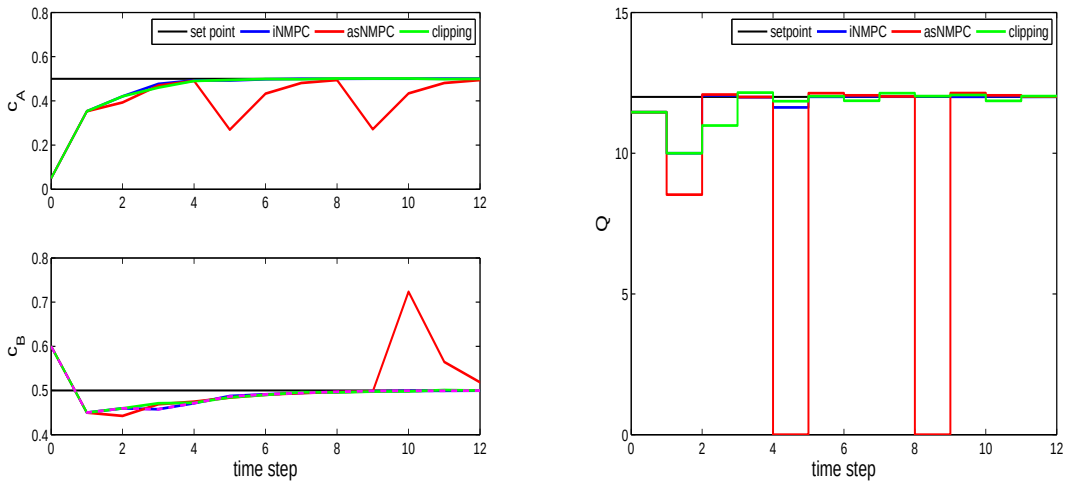
Case 1 Ideal NMPC, as a benchmark.

Case 2 asNMPC, using the sensitivity calculation using sIPOPT [48], based on the implicit function theorem (3.23). (Since this controller may violate bounds on manipulated variables, it should not be implemented in practice.)

Case 3 asNMPC, as in Case 2, but with the manipulated variables $v_0$ outside the bounds "clipped" to remain within bounds[58].

Figure 7.4 shows the state and input trajectories of the different cases. In particular, the lower bound of $\dot{m}$ becomes active at $time = 2$, and it is violated for Case 2 when (3.23) is applied directly. This follows because the lower bound is *inactive* at $time = 1$ and the

sensitivity prediction from (3.23) leads to a large (and inaccurate) step that does not include the lower bound, and consequently violates it. This violation is corrected by clipping (Case 3). The heuristic clipping approach gives good results in this case-study, and has been shown to perform well also in other contexts with input constraints [58].




(a) State profiles

(b) Control profiles

Figure 7.4: Comparison of the four cases, 5% of measurement noise. Case 1 is plotted in blue; Case 2 is plotted in red; Case 3 is plotted in green.

Table 7.2 shows accumulated stage costs of Cases 1 and 3. The cost of Case 2 is not given as it is infeasible. Interestingly, the ideal NMPC has the highest cost. Here the noise is not predictable, and its effect on the cost may be positive or negative. For this example it turns out that the effect of noise makes Ideal NMPC peform slightly worse. Note however, that the absolute difference in costs is very small. Finally, when these simulations are performed without measurement noise (no active set changes), we observe no differences between Cases 1 to 3.

Table 7.2: Cost of Economic asNMPC with 5% measurement noise and $q_{i=A,B,\dot{m}} = 1.1$

| Controller | Cost |
|---|---|
| Ideal NMPC | -33.00 |
| Clipping asNMPC | -33.15 |

### 7.2.2 Distillation Column Case Study for Economic NMPC

We illustrate stability concepts for Economic NMPC on a distillation column case study described in [37]. The model is composed of two distillation columns in sequence with a ternary component system. The three components are A, B and C, where A is the lightest component, B is the middle one and C is the heaviest component. B is the most valuable product. For our example the three components are benzene (A), toluene (B) and p-xylene (C). Each column has 41 trays including the total condenser and the partial reboiler; the feed enters column 1 at stage 21, and the bottom product from column 1 enters column 2 at stage 21. Component A leaves the first column from the top; the bottom flow of the first column becomes the feed of the second column; component B leaves the second column from the top and component C leaves the second column from the bottom. The molar holdups in the condenser drums and reboilers are assumed to be controlled by the distillate and bottom flow rates, respectively. Therefore manipulated variables are $u = [LT1, VB1, LT2, VB2]$ which are reflux and boilup for each column. The structure of the two columns in sequence is shown in Fig. 7.5. The columns are modeled with ideal thermodynamics, assuming constant relative volatility and vapor flow rate for all stages. The problem formulation is

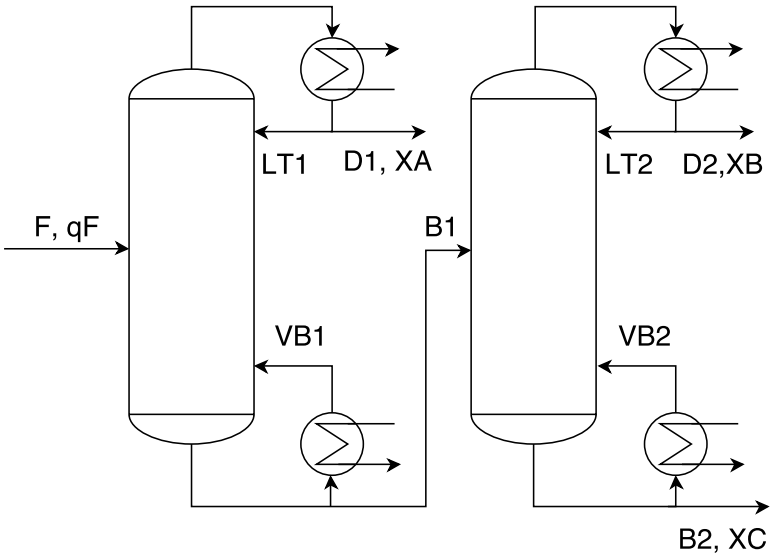



Figure 7.5: Structure of two distillation columns in sequence

given by:

$$\begin{aligned} \min_{u} \quad & J(u) = p_F F + p_V(VB1+VB2) - p_A D1 - p_B D2 - p_C B2 \qquad (7.20) \\ \text{s.t.} \quad & \textit{Mass balance}, \textit{Equilibrium} \\ & x_A \geq x_{A,min},\ x_B \geq x_{B,min},\ x_C \geq x_{C,min} \\ & 0 \leq LT1, LT2 \leq LT_{max},\ 0 \leq VB1, VB2 \leq VB_{max} \end{aligned}$$

where $D$ and $B$ are the distillate and bottom flow rate of each column; $p_F$, $p_V$, $p_A$, $p_B$, $p_C$ are prices for the feed, vapor, component A, component B and component C; their values and additional details of the model could be found in [37].

The Hessian matrix of the Lagrange function of the above steady state problem is not positive definite with $\sigma_{min} = -1.414$. Therefore if the current objective function is applied by eNMPC, the optimum steady state cannot be reached, and the system is not Lyapunov stable. Instead, the objective function must be modified to include *regularized* economic stage costs. This ensures that the first and third items of Assumption 3 in Chapter 5 are

satisfied. Moreover, through offline trials, we determined that $N = 20$ is a sufficiently long horizon so that the second item of Assumption 2 in Chapter 4 is satisfied as well.

In order to regularize the stage costs, we add quadratic terms to the objective function to make the Hessian matrix positive definite. The regularization terms have the form of (7.9) where $Q$ is a diagonal matrix and could be calculated by (7.11) for all values over a grid of $(z,v)$ over the entire domain. The objective function then becomes:

$$\begin{aligned} \min_u \ J(u) \quad &= \quad p_F F + p_V(VB1 + VB2) - p_A D1 - p_B D2 - p_C B2 \\ &\quad + \frac{1}{2} \parallel z - z^*, v - v^* \parallel_Q^2 \end{aligned} \tag{7.21}$$

Since the size of the steady state problem is large (979 variables, 975 constraints), and the Hessian matrix of the Lagrange function of the regularized steady state problem is not constant but changes with different variable values, Gershgorin weights could not be calculated directly. Therefore MATLAB was used along with AMPL to search for Gershgorin weights. To ensure that the rotated stage cost is strongly convex, the Hessian has to be positive definite with all possible values of variables. We divide the feasible value of each variable into 40 grid points and calculate value of each element of the Hessian matrix at each grid point. Finally, we have the following inequality:

$$q_i \geq \sum_{i \neq j} \max_{z,y,v} |a_{i,j}| - \min_{z,y,v} \ a_{i,i} \tag{7.22}$$

where $q_i = Q_{ii}$. To minimally regularize the economic stage costs we apply exact Gershgorin weights calculated by

$$q_i = \sum_{i \neq j} \max_{z,y,v} |a_{i,j}| - \min_{z,y,v} \ a_{i,i} \tag{7.23}$$

$Q$ is calculated offline. We use $Q^*$ to indicate the $Q$ matrix calculated using (7.23). Fig. 7.6 shows the logarithm of the diagonal elements $q_i^*$ in ascending order. In Fig. 7.6, the line does not start from $index = 0$. This is because some $q_i$ are 0 and as a result their logarithms

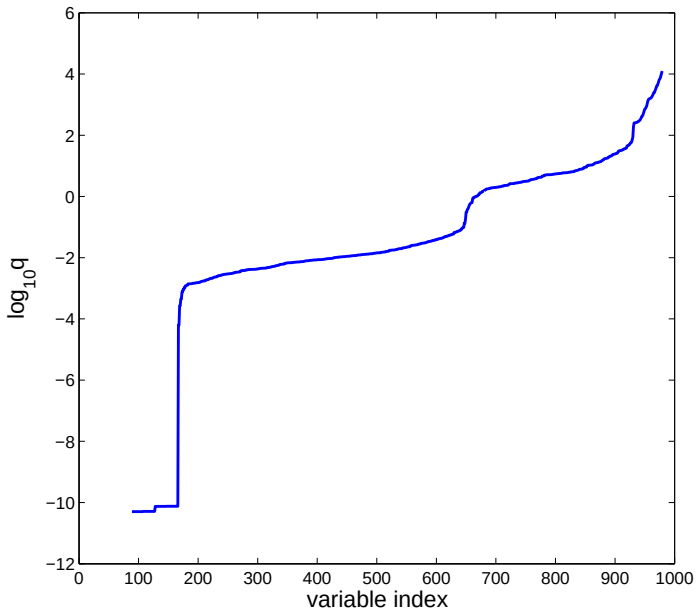



Figure 7.6: Gershgorin weights in ascending order

do not exist. We could see that about 300 of these $q_i^*$ are greater than 1, among which there are only about 50 that are greater than 100, and the rest of the $q_i$ are smaller than 1.

Next we show the effect of Gershgorin weights by comparing the state profiles generated without and with regularization, and without and with disturbances.

This case study considers two cases: no disturbance, and 5% disturbance; in the second case disturbances are introduced as additive noise. The disturbances are Gaussian noises with zero mean, and their standard deviations are 5% of the optimum steady state value of each state. For each case the objective function has three formulations: economic cost only, economic cost plus regularization terms calculated by Gershgorin Theorem, and tracking terms only. We start from a state that is not the optimum steady state and continue until $time = 100$. Here, each of the 100 NMPC problems has 136154 variables and 77754 constraints.

Fig.7.7 is generated without disturbance. From Fig.7.7(a) we could see that when the objective function is composed of only economic cost, as shown in blue, $x_A$, which is the composition of component A, drifts away from its optimum steady state, and $x_C$, the composition of component C, diverges. If the objective only includes tracking terms, as shown in green, $x_A$, $x_B$ and $x_C$ converge from the starting point to the optimum steady state very fast, and as Fig.7.7(b) shows, the corresponding control profiles converge to their set points very fast. If the objective includes both economic cost and regularization terms whose weights are calculated using the Gershgorin Theorem, $x_A$, $x_B$ and $x_C$ converge to the optimum steady state very fast as well.

Next, instead of using the full value of Gershgorin weights $Q^*$, we use $\alpha \times Q^*$ instead, where $\alpha = 0.01, 0.1, 0.5$, in order to show how the performance of Economic NMPC improves with increasing regularization weights. The results are shown in Fig. 7.8.

From Fig. 7.8(b) we could see that the control profiles get much closer to the optimal

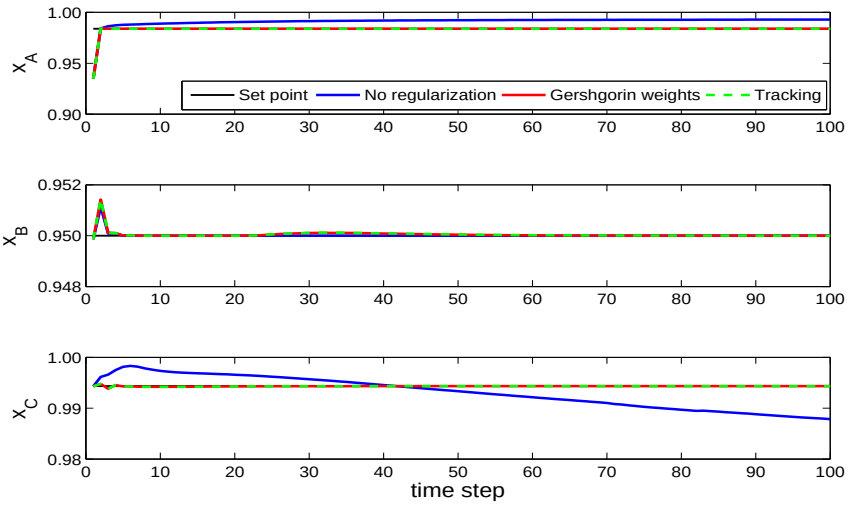
Table 7.3: Cost of NMPC with different noise levels and regularization weights

| Cost with | No regularization | $0.01 \times Q^*$ | $0.1 \times Q^*$ | $0.5 \times Q^*$ | $Q^*$ | Tracking |
|---|---|---|---|---|---|---|
| No noise | 2.476 | -20.4 | -22.09 | -22.26 | -22.27 | -22.27 |
| 5% noise | -19.97 | -21.77 | -22.97 | -20.1 | -22.03 | -22.03 |

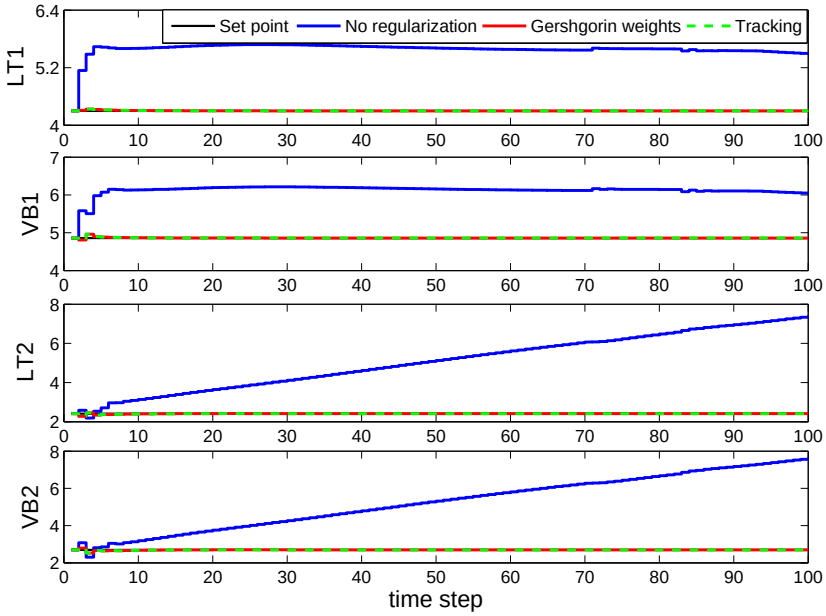
after regularization terms are added, even if the weights are small ($\alpha = 0.01$). Then as $\alpha$ increases the state profiles and control profiles get closer and closer to the economic optimal steady state. When $\alpha = 0.5$, the results are already very close to the results with $\alpha = 1$. In these cases $\sigma_{min}$ is not necessarily positive: with $\alpha = 0.01$, $\sigma_{min} = -1.3992$; with $\alpha = 0.1$, $\sigma_{min} = -1.2651$; with $\alpha = 0.5$, $\sigma_{min} = -0.6861$; however the state and control still converge to the steady state. This follows as positive definiteness of $L(\bar{z}_l, \bar{v}_l)$ and $L_f(\bar{z}_N)$ are just sufficient conditions for stability of Economic NMPC. We could also notice that the states and controls converge faster and closer with a larger value of $\sigma_{min}$.

In Table 7.3 we compare the time-accumulated economic cost, which is the objective function in (7.20), from $time = 1$ to $time = 100$. Here we observe that without regularization terms, the state does not approach the optimum steady state, and the cost is the highest. With regularization terms the cost is reduced significantly, and we can see how the cost decreases with increasing regularization weights.

Fig.7.9 and Fig. 7.10 are generated with 5% additive noise. It could be observed in Fig.7.4(a) that without regularization terms, the system is not Lyapunov stable; while regularization terms are able to keep the state within a small region around the optimum steady state. From Fig. 7.10(a) it could be seen that larger regularization weights keep the states closer to the setpoints, although this is not as obvious as the nominal case. Again, in terms of accumulated economic cost, the cost is reduced with regularization terms. However the

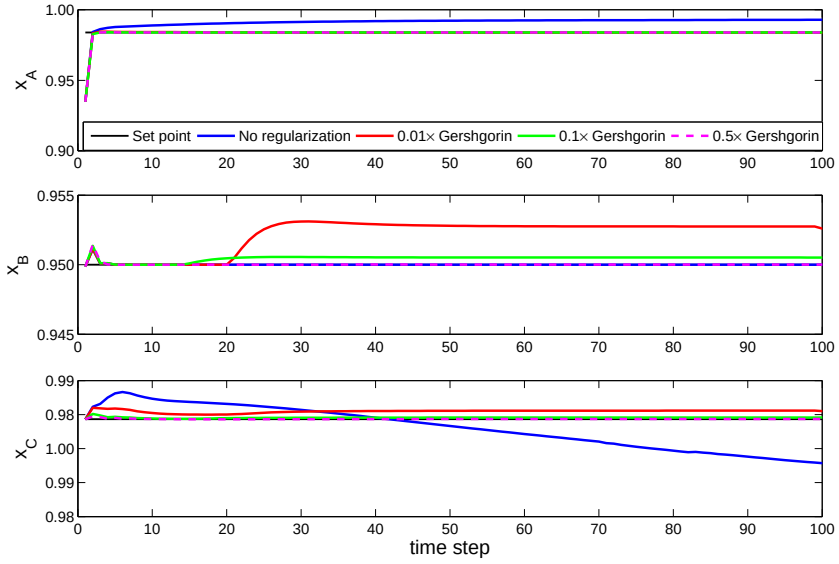



(a) State profiles

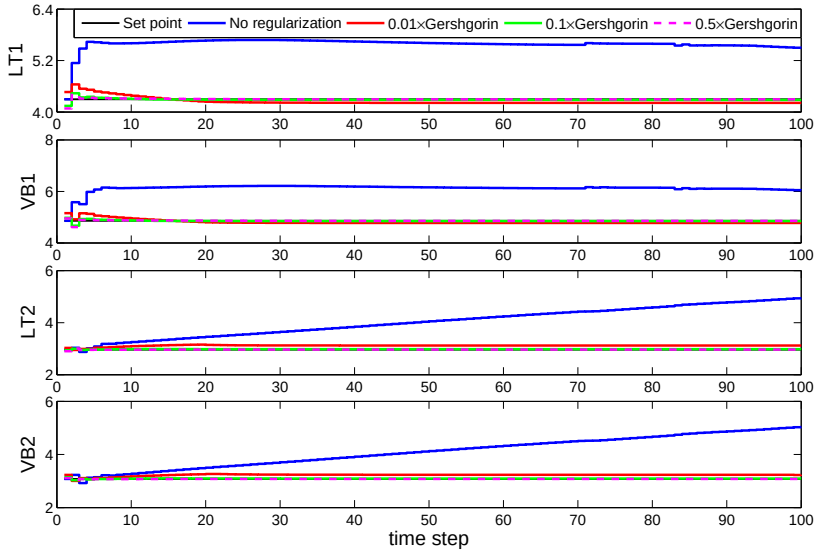



(b) Control profiles

Figure 7.7: No disturbance. The results without regularization terms are shown in blue; the results with regularization terms calculated with Gershgorin Theorem are shown in red; the results with only tracking terms are shown in green.




(a) State profiles




(b) Control profiles

Figure 7.8: No disturbance. Results with different partial values of Gershgorin weights.

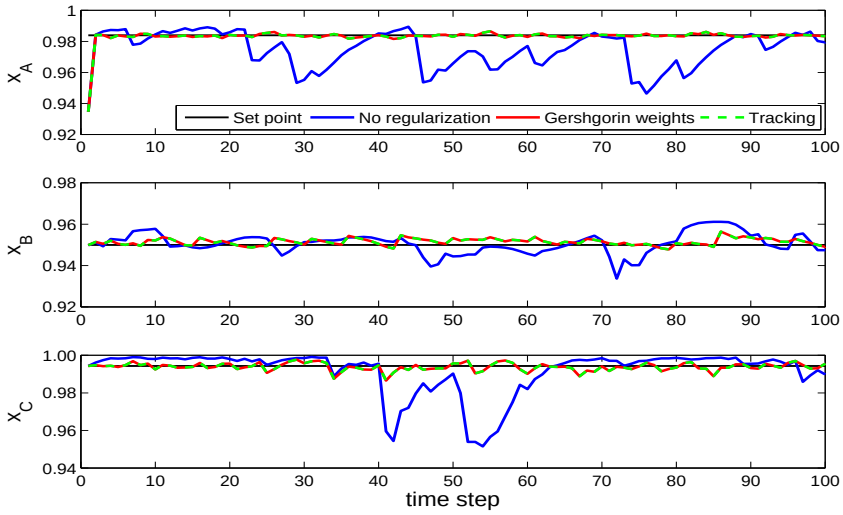
cost does not strictly decrease with increasing regularization weights because uncertainty exists in the system.

Hence, without regularization, the controller is not Lyapunov stable. However, by adding regularization terms to make the Hessian matrix of the Lagrange function of the steady state problem positive definite, the controller could be stabilized at the optimum steady state. Another observation worth mentioning is the significant improvement in CPU time. Without regularization, it takes around 6 CPU mins to solve each NLP, while with regularization it only takes about 55 CPU seconds to solve each NLP. So CPU time drops significantly with regularization terms. This is largely due to fewer inertia corrections of the KKT matrix in IPOPT.

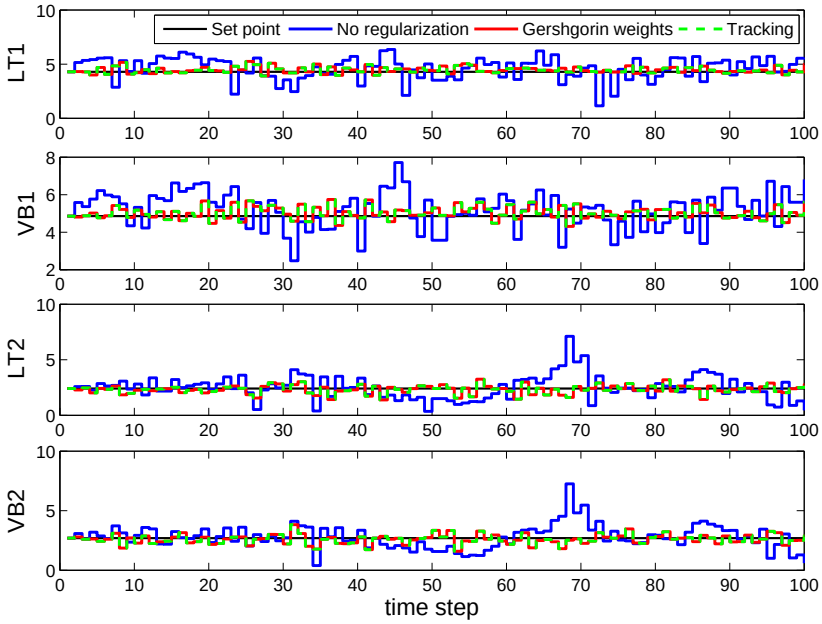
## 7.3 Concluding Remarks

In this chapter we analyze the nominal stability of Economic NMPC. If the rotated stage cost $L(\bar{z}_l, \bar{v}_l)$ and rotated terminal cost $L_f(\bar{z}_N)$ are strongly convex, the Economic NMPC controller is stable. Otherwise regularization terms must be added to the original stage costs and terminal cost. Moreover, we have presented a constructive way of calculating a 'minimal' stabilizing regularization and showed the effects of regularization terms and different regularization weights.

On the other hand, unknown inputs and model parameters have a strong influence on the steady state values $(z^*, v^*)$ in the Economic NMPC regularization. In future, we plan to handle these through on-line state and parameter estimation, where $(z^*, v^*)$ are also updated for each NMPC problem. For this, new state and parameter estimates will be used for solution of the steady state optimization problem, in order to update the regularized stage costs. An alternative strategy is to come up with better regularization terms that do not

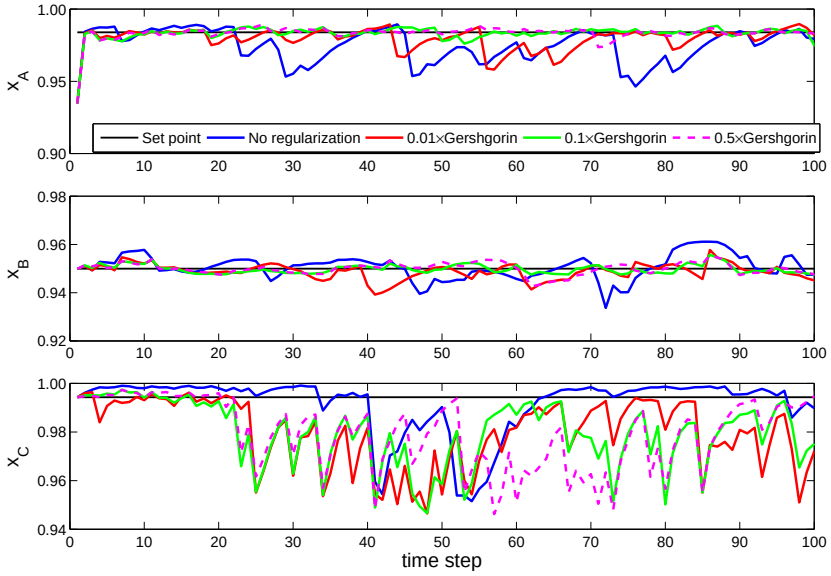



(a) State profiles

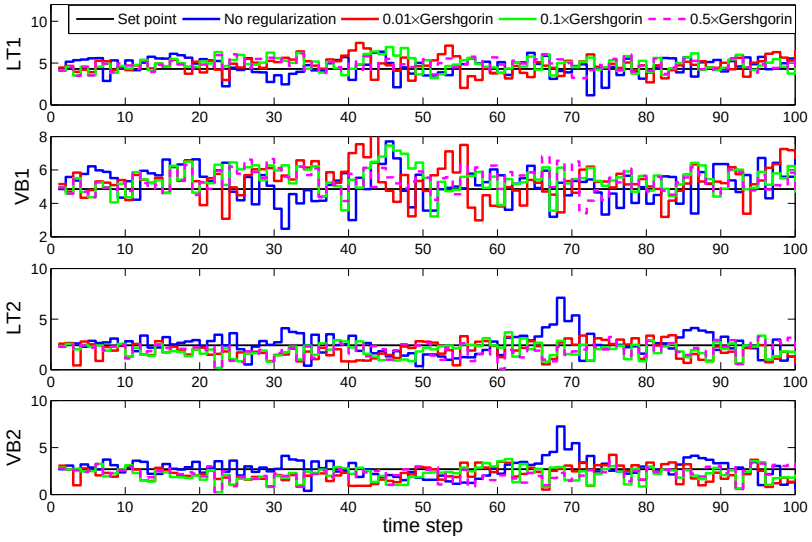



(b) Control profiles

Figure 7.9: 5% additive noise. The results without regularization terms are shown in blue; the results with regularization terms calculated with Gershgorin Theorem are shown in red; the results with only tracking terms are shown in green.




(a) State profiles




(b) Control profiles

Figure 7.10: 5% disturbance. Results with different partial values of Gershgorin weights.

need $(z^*, v^*)$. Also we plan to apply the advanced-multi-step approach to Eonomic NMPC to solve large-scale Economic NMPC problems.

# Chapter 8

# Conclusions

MPC has been widely applied in industry. For processes that are strongly nonlinear and dynamic models are available, we could use NMPC instead. However, solution time of NMPC problem is usually comparable to the sampling time of the process or even longer due to the large size of the system. In this dissertation we propose the amsNMPC strategy to handle this situation, analyze its stability properties and use case studies to evaluate its performance.

It has been shown in [7], Economic NMPC may lead to significant improvements in process performance compared to the two-layer RTO approach. However, stability of Economic NMPC remains a problem. We analyze nominal stability of Economic NMPC and propose a strategy to stabilize unstable Economic NMPC.

In this chapter we summarize the results in each chapter and propose some future directions that would be of interest.

## 8.1 Thesis Summary

Chapter 1 briefly introduces the hierarchy planning and operations structure of the operation of a chemical process. The combination of RTO and advanced control leads to the discussion of Economic NMPC in Chapter 7. Our research problems are stated in this chapter.

Chapter 2 serves as literature review and introduces the background, applications, variations, general methodology, pros and cons of setpoint tracking NMPC.

Chapter 3 discusses methodologies to solve NLPs that contain DAEs. In this dissertation, orthogonal collocation on finite elements is used to discretize DAEs and IPOPT, a solver that uses interior point method, is used to to solve NLPs. Interior point method and NLP sensitivity are also discussed in this chapter.

Chapter 4 briefly discusses some fast NMPC methods and then studies the reformulation of NLPs with state constraints and terminal constraints. The reformulated NLP is then used throughout this dissertation. The formulation, implementation and stability of asNMPC are then discussed. Clipping is introduced and could be used to prevent the bounds from being violated. asNMPC lays the theoretical foundation for amsNMPC. The concepts, assumptions and proofs of nominal and robust stabilities of setpoint tracking NMPC are then presented and Lyapunov stability analysis of ideal NMPC and asNMPC is conducted.

Chapter 5 proposes the amsNMPC scheme to avoid computational delay for NLP problems whose solutions require multiple sampling times. Two variants of amsNMPC, the parallel and the serial, are developed. For the parallel approach multiple processors are applied and an NLP problem is solved at every sampling time. For the serial approach only one processor is applied and an NLP problem is solved every $N_s$ sampling times. Both approaches predict the initial condition of the moving horizon NLP $N_s$ sampling times in advance, and update the manipulated variables at every sampling time using NLP sensitivity information. We also prove nominal stability of both approaches. Moreover, in practice, disturbances such as measurement noise and plant-model mismatch will affect the robustness of amsNMPC. We also prove robust stability when the uncertainty is give by additive disturbance.

Chapter 6 demonstrate the performance of amsNMPC compared to ideal NMPC and asNMPC with examples of a CSTR and a distillation column. Both examples illustrate that

in the nominal case, or with small level of measurement noise or plant-model mismatch, amsNMPC with $N_s \leq 3$ behaves identically to ideal NMPC and asNMPC for all values of $N_s$. However, performance deteriorates with increasing measurement noise, plant-model mismatch and $N_s$ caused by memory effect, and large measurement noise and plant-model mismatch will lead both approaches to fail. We also apply amsNMPC to a less nonlinear case, and show that nonlinearity strongly affects the performance of amsNMPC. This conclusion is observed from the distillation column example. Moreover, since the scale of the distillation model is very large, the importance of amsNMPC is emphasized, and its performance is also illustrated on such problems. In this chapter we also analyze the three nonrobust MPC examples in [24]. We show that robustness could be obtained by either using a longer horizon or reformulating the NLP with soft constraints and $\ell_1$ penalty terms in the objective function. Moreover, robustness is preserved when we apply the advanced step or advanced multi step strategy.

Chapter 7 studies Economic NMPC. For Economic NMPC, economic stage costs are used to drive the controller in the presence of disturbances. Economic NMPC has been applied on energy intensive applications with volatile electricity price [28], cyclic processes [29] and other areas [13, 4]. However, since stage costs are not quadratic terms, the objective function no longer satisfies Assumption 2 in Chapter 4 and conditions for Lyapunov functions and the stability analysis becomes more difficult.

We revisit the NMPC stability analysis through the constructions from [13] and state that if the rotated stage costs and rotated terminal costs of the transformed system, whose optimal steady state is at the origin, are strongly convex, then the transformed system satisfies Assumption 2 and thus is nominally stable. Therefore the original system, which has the same stability property as the transformed system, is nominally stable as well. If the rotated transformed stage costs and terminal costs are not convex, they should be regularized

by adding setpoint tracking terms, whereas the setpoints are the optimal solution of the original steady state problem. We also propose to use the Gershgorin Theorem to find the "minimum" regulating weights that are sufficient for Lyapunov stability.

For case studies we include a CSTR example and a double-column example. Starting with no regularization terms, we increase the regularization weights and show that the states converge faster and the cost becomes lower as weights increase.

## 8.2 Contributions

The major contributions of this dissertation is listed as follows:

1. In Chapter 4 we show that for non-robust (N)MPC controllers, their robustness could be obtained by selecting a long enough horizon or reformulating the NLP, replacing state constraints or terminal constraints by soft constraints, and adding $\ell_1$ penalty terms to the stage costs or terminal costs. We study the three examples in [24] and show that their robustness could be obtained using the above strategy. Moreover, in Chapter 6 we show that when the advanced-step strategy is applied to those examples, robustness is preserved.

2. In Chapter 4 we introduce the "clipping in first interval" method to prevent bounds on manipulated variables from begin violated after sensitivity update. In Chapter 7 we use a toy CSTR example to show that with clipping the control bounds are not violated. Moreover, since clipping is easy and fast, it is used in all the case studies in this dissertation. Moreover, we also prove robust stability for asNMPC with clipping.

3. In Chapter 5 and Chapter 6 we propose the serial approach and the parallel approach of amsNMPC and use case studies to evaluate its performance. When the solution

time of the NLP (4.3) exceeds one sampling time, amsNMPC is used to avoid computation delay and get real-time suboptimal result. We show that for a less nonlinear system, with small level of uncertainty, amsNMPC behaves almost identically to ideal NMPC and asNMPC. For the serial approach this is a great advantage because NLPs are solved less frequently and computational cost is reduced. When the system is strongly nonlinear and uncertainty level is large, memory effect leads to worse performance than ideal NMPC and asNMPC.

4. In Chapter 5 we prove the nominal stability for amsNMPC, including the serial approach and the parallel approach with $N \to \infty$. We prove the robust stability for amsNMPC under assumptions that limit memory effect.

5. In Chapter 7 we analyze criteria for Economic NMPC to be stable based on Lyapunov stability analysis. If the Economic NMPC is unstable, we propose to add regularization terms to objective function to stabilize it. Moreover, we use the Gershgorin theorem to calculate the weights on regularization terms that leads to the optimal economic performance while stability is guaranteed. We also prove the nominal and robust stability for Economic NMPC therefore follows from Huang et al [29].

## 8.3 Recommendations for Future Work

In this dissertation we propose the amsNMPC algorithm to avoid computational delay when the solution time is $N_s$ sampling times. We apply the parallel approach to a large-scale C3 splitter and use sIPOPT as the solver to get fast updated NLP solutions. We propose to stabilize Economic NMPC by adding tracking terms to the objective where the setpoint is obtained by solving an economic optimal steady state problem. There are more interesting topics worth looking into in the future.

1. Due to the size of the distillation column problem, instead of updating manipulated variables manually using MATLAB, sIPOPT, the sensitivity extension to IPOPT, is applied [48], and only the parallel approach is considered here. In future the serial approach will be realized using the distillation column model with a Schur complement extension added to sIPOPT.

2. So far $N_s$ is chosen as the upper bound of the solution time and the NLP problems are solved at a fixed frequency. However, amsNMPC could be extended to the case where actual solution time can vary, so that the frequency to solve the NLP problems could be adjusted automatically. However, the predicted state should still be $N_s$ sampling times ahead. Otherwise it can not be assured that the optimal solution is always available when the current actual state is obtained.

3. Economic NMPC has been shown to lead to better economic performance than the two-layer RTO advanced control strategy. It will be interesting to study whether the fast NMPC methods could be applied to Economic NNPC as well. Applying the advanced-multi-step algorithm to Economic NMPC is not necessarily the same as applying it to setpoint tracking NMPC because of stability issues. It is possible that the bound on $w$ in Assumption 4, Chapter 4 is tighter.

4. With our case studies it is assumed that the plant states are immediately available at the beginning of each sampling time, and every state could be measured. This is rarely true in practice, and a state observer is needed. Furthermore, a state observer could also eliminate set point offsets in the presence of plant-model mismatch. Moreover, we would like to use state and parameter estimators to update $(z^*, v^*)$ for each NMPC problem to account for the influence that unknown inputs and parameters have on the steady state values in regularization terms. Specifically we would like to use moving horizon estimation (MHE) as the estimator and combine MHE with

amsNMPC and Economic NMPC.

**Moving Horizon Estimation ([53])** Since the initial condition of NMPC is the state estimate, state estimation is required for NMPC. State estimation uses limited input and output information as well as the process model to infer the state of the process. MHE is an optimization based state estimation and has the advantage to handle variable bounds. It is very similar to NMPC, but instead of predicting states in the future, it uses outputs in history.

For MHE, other than the plant model, we introduce the output mapping into the plant dynamics as the following:

$$\begin{aligned} x(k+1) &= f(x(k),u(k)), x(k) \in \mathbb{X}, u(k) \in \mathbb{U} \\ y(k) &= h(x)(k) \end{aligned} \tag{8.1}$$

where $y(k)$ is the output at time $t_k$ and $h(\cdot) : \Re^{n_x} \mapsto \Re^{n_y}$ is a nonlinear function that maps states to outputs. At $t_k$ the following NLP is solved to get the state estimate:

$$\begin{aligned} \min_{\hat{x}_{k-N_e+l}} \quad & \sum_{l=0}^{N_e} [(y_{k-N_e+l} - \hat{y}_{k-N_e+l})^T \Pi_y (y_{k-N_e+l} - \hat{y}_{k-N_e+l})] \\ & + (\hat{x}_{k-N_e} - \bar{x}_{k-N_e})^T \Pi_0 (\hat{x}_{k-N_e} - \bar{x}_{k-N_e}) \\ \text{s.t.} \quad & \hat{x}_{k-N_e+l+1} = f(\hat{x}_{k-N_e+l}, u_{k-N_e+l})) \\ & \hat{y}_{k-N_e+l} = h(\hat{x}_{k-N_e+l}) \\ & \hat{x}_{k-N_e+l} \in \mathbb{X}, \quad l = 0, ..., N_e. \end{aligned} \tag{8.2}$$

where $N_e$ is the estimation horizon length, $\hat{x}_{k-N_e+j}, j = 0, \ldots, N_e$ is the estimation of the state within the estimation horizon, $\Pi_y, \Pi_0$ are symmetric positive definite tuning parameters. $\Pi_0$ is called arrival cost and it corresponds to the terminal cost in the formulation of NMPC problems. $\bar{x}_{k-N_e}$ is the most likely prior value of $x_{k-N_e}$ and we

choose $\hat{x}_{k-N_e+1}$ from the previous MHE problem as $\bar{x}_{k-N_e}$ of the current MHE. By solving this NLP (8.2) with the current measured output, the current state estimate is obtained.

Moreover, MHE has computational delay as well. So advanced-step strategy has been applied to MHE in [26]. Moreover, we could apply the advanced-multi-step approach to MHE in case the solution time of (8.2) exceeds one sampling time. Finally, it will be interesting to combine amsMHE with amsNMPC to solve Economic NMPC problems and see if the application of advanced-multi-step strategy on both MHE and NMPC will deteriorate stability of Economic NMPC.

# Bibliography


[1] Alamir, M. [2009], A framework for monitoring control updating period in real-time NMPC schemes, *in* L. Magni, D. Raimondo and F. Allögwer, eds, 'Nonlinear Model Predictive Control', Vol. 384 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 433–445.

[2] Amrit, R., Rawlings, J. B. and Biegler, L. T. [2013], 'Optimizing process economics online using model predictive control', *Comp. Chem. Engr.* **58**, 334–343.

[3] Angeli, D., Amrit, R. and Rawlings, J. B. [2009], Receding horizon cost optimization for overly constrained nonlinear plants, *in* 'Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference'.

[4] Angeli, D., Amrit, R. and Rawlings, J. B. [2012], 'On average performance and stability of economic model predictive control', *IEEE Transactions on Automatic Control* **57**(7), 1615–1626.

[5] Bartusiak, R. [2007], NLMPC: A platform for optimal control of feed- or product-flexible manufacturing, *in* R. Findeisen, F. Allgöwer and L. T. Biegler, eds, 'Assessment and Future Directions of Nonlinear Model Predictive Control', Vol. 358 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 367–381.

[6] Biegler, L. T. [2010], *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes*, SIAM.

[7] Biegler, L. T. and Zavala, V. M. [2009], 'Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization', *Computers and Chemical Engineering* **33**, 575–582.

[8] Chen, H. and Allgöwer, F. [1996], A quasi-infinite horizon predictive control scheme for constrained non-linear systems, *in* 'Proceedings of the 16th Chinese Control Conference', Qingdao, China, pp. 309–316.

[9] Chen, H. and Allgöwer, F. [1998], 'A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability', *Automatica* **34**, 1205–1217.

[10] Cutler, C. R. and L., R. B. [1980], Dynamic matrix controla computer control algorithm, *in* 'Joint Automatic Control Conference', Sanfrancisco, CA.

[11] Cutler, C. R. and Ramaker, B. L. [1979], Dynamic matrix controla computer control algorithm, *in* 'AICHE 86th National Meeting', Houston, TX.

[12] de Oliveira, N. M. C. and Biegler, L. T. [1998], 'Constraint handling and stability properties of model-predictive control', *Process Systems Engineering* **40**, 1138–1155.

[13] Diehl, M., Amrit, R. and Rawlings, J. B. [2011], 'A Lyapunov function for economic optimizing model predictive control', *IEEE Transactions on Automatic Control* **56**(3), 703–707.

[14] Diehl, M., Bock, H. D. and Schlöder, J. P. [2005], 'A real-time iteration scheme for nonlinear optimization in optimal feedback control', *SIAM J. Control and Optimization* **43**, 1714–1736.

[15] Diehl, M., Bock, H., Schlöder, J. P., Findeisen, R., Nagy, Z. and Allgöwer, F. [2002], 'Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations', *Journal of Process Control* **12**(4), 577 – 585.

[16] Duff, I. [2004], 'MA57 - a code for the solution of sparse symmetric definite and indefinite systems', *ACM Transactions on Mathematical Software* **30**, 118–144.

[17] Engell, S. [2007], 'Feedback control for optimal process operation', *J. Proc. Cont.* **17**, 203–219.

[18] Fiacco, A. V. [1983], *Introduction to sensitivity and stability analysis in nonlinear programming*, Vol. 226, Academic press New York.

[19] Findeisen, R. and Allgöwer, F. [2004], Computational delay in nonlinear model predictive control, *in* 'Proc. Int. Symp. Adv. Control of Chemical Processes (ADCHEM), Hong Kong'.

[20] Fischer, G. A. G. and Biegler, L. T. [2012], Fast online optimization applied to high purity propylene distillation, Technical report, Carnegie Mellon University and Petroleo Brasileiro S.A.

[21] Franke, R. and Doppelhamer, J. [2007], Integration of advanced model based control with industrial IT, *in* R. Findeisen, F. Allgöwer and L. T. Biegler, eds, 'Assessment and Future Directions of Nonlinear Model Predictive Control', Vol. 358 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 399–406.

[22] Ganesh, N. and Biegler, L. T. [1987], 'A reduced hessian strategy for sensitivity analysis of optimal flowsheets', *AICHE J.* **33**(2), 282–296.

[23] Gauvin, J. [1977], 'A necessary and sufficient regularity condition to have bounded multipliers in nonconvex programming', *Mathematical Programming* **12**(1), 136–138.

[24] Grimm, G., Messina, M. J., Tuna, S. and Teel, A. [2004], 'Examples when nonlinear model predictive control is nonrobust', *Automatica* **40**, 523–533.

[25] Hicks, G. A. and Ray, W. H. [1971], 'Approximation methods for optimal control synthesis', *Can. J. Chem. Eng.* **49**, 522–528.

[26] Huang, R. [2010], Nonlinear Model Predictive Control and Dynamic Real-time Optimization for Large-scale Processes, PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA.

[27] Huang, R. and Biegler, L. T. [2011], Stability of economically-oriented nmpc with periodic constraint, *in* '18th IFAC World Congress'.

[28] Huang, R. and Biegler, L. T. [2012], Economic NMPC for energy intensive applications with electricity price prediction, *in* I. Karimi and R. Srinivasan, eds, 'Intl. Symp. on Process Systems Engineering, Computer Aided Chemical Engineering', Vol. 31, Elsevier , Amsterdam, pp. 1612–1616.

[29] Huang, R., Biegler, L. T. and Harinath, E. [2012], 'Robust stability of economically oriented infinite horizon NMPC that include cyclic processes', *Journal of Process Control* **22**(1), 51–59.

[30] Janin, R. [1984], Directional derivative of the marginal function in nonlinear programming, *in* A. Fiacco, ed., 'Sensitivity, Stability and Parametric Analysis', Vol. 21 of *Mathematical Programming Studies*, Springer Berlin Heidelberg, pp. 110–126.

[31] Jäschke, J., Yang, X. and Biegler, L. T. [2014], 'Fast economic model predictive control based on nlp-sensitivities', *Journal of Process Control* **24**, 1260–1272.

[32] Jiang, Z. and Wang, Y. [2001], 'Input-to-state stability for discrete-time nonlinear systems', *Automatica* **37**, 857–869.

[33] Kadam, J. and Marquardt, W. [2004], Sensitivity-based solution updates in closed-loop dynamic optimization, *in* 'DYCOPS 7, Cambridge, USA'.

[34] Kadam, J. and Marquardt, W. [2007], Integration of economical optimization and control for intentionally transient process operation, *in* R. Findeisen, F. Allgöwer and L. Biegler, eds, 'Assessment and Future Directions of Nonlinear Model Predictive Control', Vol. 358 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 419–434.

[35] Keerthi, S. and Gilbert, E. [1988], 'Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations', *J. of Optimization Theory and Applications* **57**, 265–293.

[36] Kojima, M. [1980], Strongly stable stationary solutions in nonlinear programming, *in* S. M. Robinson, ed., 'Analysis and Coputation of Fized points', Academic Press, New York.

[37] Leer, R. B. [2012], Self-optimizing control structures for active constraint regions of a sequence of distillation columns, Master's thesis, Norweign University of Science and Technology. Url: http://www.diva-portal.org/smash/get/diva2:629241/FULLTEXT01.pdf.

[38] Li, W. and Biegler, L. T. [1989], 'Multistep, Newton-type control strategies for constrained, nonlinear processes', *Chemical Engineering Research and Design* **67**, 562–577.

[39] Limon, D., Alamo, T., Raimondo, D., la Peña, D., Bravo, J. and Camacho, E. [n.d.], Input-to-state stability: a unifying framework for robust model predictive control, *in* 'Assessment and Future Directions of Nonlinear Model Predictive Control', Springer. In press.

[40] Magni, L. and Scattolini, R. [2007], Robustness and robust design of MPC for nonlinear discrete-time systems, *in* R. Findeisen, F. Allgöwer and L. Biegler, eds, 'Assessment and Future Directions of Nonlinear Model Predictive Control', Vol. 358 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 239–254.

[41] Michalska, H. and Mayne, D. [1993], 'Robust receding horizon control of constrained nonlinear systems', *IEEE Transactions on Automatic Control* **38**, 1623–1633.

[42] Morari, M. and Lee, J. [1999], 'Model predictive control: past, present and future', *Computer & Chemical Engineering* **23**, 667–682.

[43] Nagy, Z., Mahn, B., Franke, R. and Allgöwer, F. [2007], Real-time implementation of nonlinear model predictive control of batch processes in an industrial framework, *in* R. Findeisen, F. Allgöwer and L. T. Biegler, eds, 'Assessment and Future Directions of Nonlinear Model Predictive Control', Vol. 358 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 465–472.

[44] Nocedal, J. and Wright, S. [2006], *Numerical Optimization*, Springer.

[45] Ohtsuka, T. [2004], 'A continuation/GMRES method for fast computation of nonlinear receding horizon control', *Automatica* **40**(4), 563 – 574.

[46] Pannocchia, G., Rawlings, J. B. and Wright, S. J. [2007], 'Fast, large-scale model predictive control by partial enumeration', *Automatica* **43**, 852–860.

[47] Pannocchia, G., Rawlings, J. and Wright, S. [2011], 'Conditions under which supboptimal nonlinear mpc is inherently robust', *Systems & Control Letters* **60**, 747–755.

[48] Pirnay, H., López-Negrete, R. and Biegler, L. T. [2012], 'Optimal sensitivity based on IPOPT', *Mathematical Program. Computation* **4**, 307–331.

[49] Qin, S. J. and Badgwell, T. A. [2003], 'A survey of industrial model predictive control technology', *Control Engineering Pactice* **11**.

[50] Ralph, D. and Dempe, S. [1995], 'Directional derivatives of the solution of a parametric nonlinear program', *Mathematical Programming* **70**(1-3), 159–172.

[51] Rawlings, J. and Amrit, R. [2009], Optimizing process economic performance using model predictive control, *in* L. Magni, D. M. Raimondo and F. Allgöwer, eds, 'Assessment and Future Directions of Nonlinear Model Predictive Control', Springer, pp. 119–138.

[52] Rawlings, J. B., Bonné, D., B.Jørgensen, J., Venkat, A. N. and Jørgensen, S. B. [2008], 'Unreachable setpoints in model predictive control', *IEEE Transactions on Automatic Control* **53**, 2209–2215.

[53] Rawlings, J. B. and Mayne, D. Q. [2009], *Model predictive control: theory and design*, Nob Hill Publishing, Madison.

[54] Rawlings, J., Bonné, D., Jørgensen, J., Venkat, A. and Jørgensen, S. [2008], 'Unreachable setpoint in model predictive control', *IEEE Trans. on Auto. Cont.* **53**, 2209–2215.

[55] Wächter, A. and Biegler, L. T. [2006], 'On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming', *Mathematical Programming* **106**(1), 25–57.

[56] Wolf, I., Würth, L. and Marquardt, W. [2011], Rigorous solution vs. fast update: Acceptable computational delay in NMPC, *in* 'Decision and Control and European Control Conference, Orlando, Florida, USA', pp. 5230–5235.

[57] Würth, L., Hannemann, R. and Marquarde, W. [2009], 'Neighboring-extremal updates for nonlinear model-predictive control and dynamic real-time optimization', *Journal of Process Control* **19**, 1277–1288.

[58] Yang, X. and Biegler, L. T. [2013], 'Advanced-multi-step nonlinear model predictive control', *J. Process Control* **23**, 1116–1128.

[59] Zavala, V. M. and Biegler, L. [2009*a*], 'Optimization-based strategies for the operation of low-density polyethylene tubular reactors: Nonlinear model predictive control', *Computers and Chemical Engineering* **33**, 1735–1746.

[60] Zavala, V. M. and Biegler, L. T. [2009*b*], 'The advanced step NMPC controller: Optimality, stability and robustness', *Automatica* **45**, 86–93.

[61] Zavala, V. M. and Biegler, L. T. [2009*c*], Nonlinear programming strategies for state estimation and model predictive control, *in* L. Magni, D. Raimondo and F. Allgöwer, eds, 'Nonlinear Model Predictive Control', Vol. 384 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 419–432.