Doctoral theses at NTNU, 2024:269

Hossein Nejatbakhsh Esfahani

Reinforcement Learning-based Control and State Estimation using Model Predictive Control and Moving Horizon Estimation

Doctoral thesis

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Hossein Nejatbakhsh Esfahani

Reinforcement Learning-based Control and State Estimation using Model Predictive Control and Moving Horizon Estimation

Thesis for the Degree of Philosophiae Doctor

Trondheim, July 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

ISBN 978-82-326-8130-3 (printed ver.)
ISBN 978-82-326-8129-7 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

Doctoral theses at NTNU, 2024:269

Printed by NTNU Grafisk senter

Acknowledgements

First and foremost I offer my heartfelt gratitude to the God Almighty who bestowed me with the life, the opportunity to see the everyday wonders of creation and for the strength and ability to carry out this work.

I would like to express my deepest gratitude to my supervisor, Professor Sebastien Gros for all his cooperation, support and guidance. I would also like to thank my co-supervisor, Professor Anastasios Lekkas for his great help. Further thanks also go out to my colleagues in our research group, RL-MPC for creating a great environment in which to carry out my work, as well as providing many interesting discussions. During my PhD study, I have been fortunate enough to collaborate with several people throughout this work. I would like to thank Arash Bahari Kordabad and Wenqi Cai.

Finally, my deepest gratitude goes to my beloved wife for her endless love, and encouragement, and to my family for their support. I would like to dedicate my PhD thesis to my spouse Elham and my daughter Ava.

Summary

A new Reinforcement Learning (RL) algorithm based on Model Predictive Control (MPC) has been recently proposed in which the optimal state (-action) value function and the optimal policy can be captured by a parameterized MPC scheme even if the system model underlying the MPC scheme cannot capture the real system perfectly. However, the main idea above was investigated upon the Markov Decision Process (MDP), where a full observation of the states of the real system is needed. Moreover, the idea of using the MPC-based RL can be investigated for other types of MPC schemes such as robust MPC and Linear Parameter Varying-MPC (LPV-MPC).

To investigate the above mentioned ideas and develop new frameworks in the context of MPC-based reinforcement learning, in the first part of this thesis, we investigate the use of the MPC-based RL framework in the context of Partially Observable Markov Decision Process (POMDP). We next show that the core idea of modifying the MPC scheme by RL can also be used for modifying a Moving Horizon Estimation (MHE) scheme so that the MHE performance is improved even if the system model underlying the MHE scheme is imperfect. Moreover, we propose an MHE/MPC-based RL in the context of LPV systems. In the second part of the thesis, we investigate the use of the MPC-based RL for an approximate Robust Nonlinear MPC (RNMPC). We then use a second-order Q-learning algorithm to adjust a set of parameters attached to this approximate RNMPC scheme aiming to achieve the best closed-loop performance.

In the context of POMDP, we propose an observer-based framework for solving POMDPs, where the real system is partially observable. We first propose to use a Moving Horizon Estimation-Model Predictive Control (MHE-MPC) scheme in order to provide a policy for the POMDP problem, where the states of the real

system are not fully measurable and necessarily known. We propose to parameterize both the MPC and MHE formulations, where certain adjustable parameters are regarded for tuning the policy. In this work, for the sake of tackling the unmodeled and partially observable dynamics, we leverage the RL to tune the parameters of MPC and MHE schemes jointly, with the closed-loop performance of the policy as a goal rather than model fitting or the MHE performance.

To deal with the model-based state estimation problems with imperfect models, we next present a reinforcement learning-based observer/controller using MHE and MPC schemes, where the model used in the MHE-MPC scheme cannot accurately capture the dynamics of the real system. We show how an MHE cost modification can improve the performance of the MHE scheme such that a true state estimation is delivered even if the underlying MHE model is imperfect. A compatible Deterministic Policy Gradient (DPG) algorithm is then proposed to directly tune the parameters of both the estimator (MHE) and controller (MPC) aiming to achieve the best closed-loop performance.

The LPV models use a linear structure to capture time-varying and nonlinear dynamics of complex systems. These models then facilitate the formulation of computationally efficient design algorithms for observers and controllers synthesis of nonlinear systems. In the LPV framework, we propose an MHE/MPC-based RL method for the polytopic LPV systems with inexact scheduling parameters (as exogenous signals with inexact bounds), where the Linear Time Invariant (LTI) models (vertices) captured by combinations of the scheduling parameters becomes wrong. We first propose to adopt an MHE scheme to simultaneously estimate the convex combination vector and unmeasured states based on the observations and model matching error. To tackle the wrong LTI models used in both the MPC and MHE schemes, we then exploit a Policy Gradient (PG) to learn both the estimator (MHE) and controller (MPC) so that the best closed-loop performance is achieved.

In the context of robust MPC, we present an RL-based Robust Nonlinear Model Predictive Control (RL-RNMPC) framework for controlling nonlinear dynamical systems in the presence of disturbances and uncertainties. An approximate RNMPC of low computational complexity is used in which the state trajectory uncertainty is modelled via ellipsoids. Reinforcement Learning is then used in order to handle the ellipsoidal approximation and improve the closed-loop performance of the scheme by adjusting the MPC parameters generating the ellipsoids.

Preface

This thesis is submitted in partial fulfilment of the requirements for the degree of Philosophiae Doctor (PhD) at the Norwegian University of Science and Technology (NTNU), Trondheim, Norway.

Professor Sebastien Gros has been the main supervisor of the PhD project, and Professor Anastasios M. Lekkas has been co-supervisor. The work was supported by the Research Council of Norway (RCN) (grant no. NFR 300172) project “Safe Reinforcement Learning using Model Predictive Control" (SARLEM) (project no. UV988962100) at NTNU.

Contents

List of Tables

List of Figures

List of Abbreviations

AC	Actor-Critic
ADHDP	Action-Dependent Heuristic Dynamic Programming
ADP	Approximate Dynamic Programming
ANN	Artificial Neural Network
DNN	Deep Neural Network
DP	Dynamic Programming
DPG	Deterministic Policy Gradient
FICNN	Fully Input Convex Neural Network
FIE	Full Information Estimation
GP	Gaussian Process
ICNN	Input Convex Neural Network
LMI	Linear Matrix Inequality
LPT	Law of Total Probability
LPV	Linear Parameter-Varying
LS	Least Squares
LSTDQ	Least Squares Temporal Difference Q-learning
LTI	Linear Time Invariant

MAP Maximum A Posterior

MARL Multi-Agent Reinforcement Learning

MDP Markov Decision Process

MHE Moving Horizon Estimation

ML Machine Learning

MPC Model Predictive Control

NMPC Nonlinear MPC

NN Neural Network

PG Policy Gradient

POMDP Partially Observable Markov Decision Process

RBF Radial Basis Functions

ReLU Rectified Linear Unit

RHC Receding Horizon Control

RL Reinforcement Learning

RNMPC Robust Nonlinear Model Predictive Control

RNN Recurrent Neural Network

SDP Semidefinite Programming

TD Temporal Difference

Chapter 1

Introduction

In this chapter, we briefly discuss the motivation behind the research accomplished in this thesis. The main contributions in the context of learning-based controller/observer framework are presented. We provide a list of the works published during this PhD study. Finally, an outline of the thesis is presented.

1.1 Motivation

Reinforcement Learning (RL) is a powerful tool for solving Markov Decision Processes (MDPs) without depending on a model of the probability distributions underlying the state transitions. Recently, RL-based control algorithms are gaining attention, as they can make good use of data to reduce the impact of uncertainties and disturbances, without relying on a model that captures the real system accurately. The RL algorithms typically rely on Deep Neural Networks (DNNs) as function approximators.

Unfortunately, DNN-based RL methods do not provide formal tools to discuss the closed-loop stability and constraint satisfaction. Moreover, providing the initial weights of a DNN-based policy is difficult and often done randomly, which leads to a lengthy learning process. In contrast, Model Predictive Control (MPC)-based policies benefit from a large set of theoretical tools addressing those issues, and could provide fairly effective policies by using the available system models.

MPC is a successful control strategy that employs a (possibly inaccurate) model of the real system to generate input-state sequences that minimize a certain cost, possibly under some constraints. The MPC problem is solved at every time instant, in a receding-horizon fashion, delivering a policy for the real system. For many applications, the building an MPC model able to capture the real system dynamics

accurately is very difficult, especially if the real system is stochastic. For these applications, the performance of the MPC scheme can be severely affected by this lack of accuracy. Moreover, for computational reasons, simple models are usually preferred in the MPC-scheme. Hence, the MPC model often does not have the required structure to correctly capture the real system dynamics and stochasticity. As a result, the MPC scheme can deliver a reasonable approximation of the optimal policy, but it is usually suboptimal. Nevertheless, choosing the model parameters that best fit the MPC model to the real system does not necessarily yield a policy that achieves the best closed-loop performance.

In the context of learning-based MPC, some Machine-Learning (ML)-based MPC algorithms have been recently developed to capture an accurate model of the real system to be used in the MPC schemes [2, 3, 4, 5, 6, 7]. However, a core issue with these ML-based MPC schemes is that the modelling is not directly related to the control objectives. More specifically, the ML-based models are constructed to deliver the best possible predictions, in the hope that this will turn into the best possible MPC performances.

To address the problems above and leverage the advantages of both the MPC and RL, an MPC-based RL framework was proposed in [8]. In this paper, it was shown that by adjusting not only the MPC model parameters but also the parameters in the MPC cost (terminal and stage costs) and constraints, the MPC scheme can, theoretically, generate the optimal closed-loop policy even if a simple and inaccurate MPC model is used. Instead of DNN, a parameterized MPC scheme was used as a function approximator required in both the Q-learning and policy gradient methods. The RL then helps to tune the parameters so that the long-term closed-loop performance is improved. Moreover, the combination of RL and MPC is therefore unique in the field of learning-based MPC as it does not focus on improving the MPC model for more accurate predictions, but ties the MPC tuning directly to the closed-loop optimality of the resulting policy.

In this thesis, we then use the fundamental principles behind the MPC-based RL and address several open questions upon this new RL method. We investigate the use of the main idea in the context of Partially Observable Markov Decision Processes (POMDPs) as the existing theorem is only valid for MDPs assuming that a full observation of the states of the real system is available. We additionally explore a theorem to modification of the Moving Horizon Estimation (MHE) scheme such that the state estimation performance can be improved even if a simplified and an imperfect model is used in the MHE scheme. In the Linear Parameter-Varying (LPV) framework, we show that the MPC-based RL combined with an MHE scheme can improve the closed-loop performance when an inaccurate model of the polytopic LPV is used in the LPV-MPC schemes. In the context of robust

Nonlinear MPC (NMPC), we propose an RL-based robust NMPC to adjust the ellipsoidal-MPC scheme as an approximate method to robustifying a nonlinear MPC scheme in the presence of uncertainties.

1.2 Contributions

In light of the above, the main contributions presented in this thesis can be split into two parts. The first part of the thesis investigates the use of an observer combined with the MPC-based RL, where one needs to estimate some states of the real system in a partially observable environment. As a natural choice for the MPC scheme, we propose to use an MHE scheme as an observer (estimator). We then present a cost modification for the MHE scheme with imperfect model aiming to achieve the best estimation performance. We also propose to formulate an MHE/MPC-based RL method in the LPV framework for dealing with the polytopic LPV systems with inexact scheduling parameters. Finally, in the second part of the thesis, we propose to combine an ellipsoidal nonlinear MPC as an approximate Robust Nonlinear MPC (RNMPC) with reinforcement learning. These contributions are detailed in Parts I and II.

(I) **Reinforcement Learning based on MHE-MPC**

This part of the thesis consists of three chapters. In Chapter 3, we formulate an RL method based on a combined MHE-MPC scheme when dealing with POMDPs. Chapter 4 presents the central theorem behind the MHE cost modification. In Chapter 5, we propose an MHE/MPC-based policy gradient method when dealing with LPV systems.

(II) **Reinforcement Learning based on Robust NMPC**

Chapter 6 is the only chapter of this part, which investigates the use of the MPC-based RL in the context of robust NMPC.

1.3 Publications

Ten articles in total were produced and published in peer-reviewed international conferences and one peer-reviewed journal during the PhD. The author of this thesis was the first author of five publications and contributed as a co-author of five other papers.

This thesis has been written based on the following four publications:

Conference Publications

- H. N. Esfahani, A. B. Kordabad and S. Gros, "Reinforcement Learning

based on MPC/MHE for Unmodeled and Partially Observable Dynamics," 2021 American Control Conference (ACC), New Orleans, LA, USA, 2021, pp. 2121-2126, doi: 10.23919/ACC50511.2021.9483399.

- H. N. Esfahani, A. B. Kordabad and S. Gros, "Approximate Robust NMPC using Reinforcement Learning," 2021 European Control Conference (ECC), Delft, Netherlands, 2021, pp. 132-137, doi: 10.23919/ECC54610.2021.9655129.
- H.N. Esfahani, S. Gros, "Policy gradient reinforcement learning for uncertain polytopic LPV systems based on MHE-MPC," IFAC-PapersOnLine 55 (15) (2022) 1–6. 6th IFAC Conference on Intelligent Control and Automation Sciences, ICONS 2022.

Journal Publication

- H. N. Esfahani, A. B. Kordabad, W. Cai, and S. Gros, "Learning-based state estimation and control using mhe and mpc schemes with imperfect models," European Journal of Control, p. 100880, 2023.

Publications not included in this thesis

- H. N. Esfahani, B. Aminian, E. I. Grøtli and S. Gros, "Backstepping-based Integral Sliding Mode Control with Time Delay Estimation for Autonomous Underwater Vehicles," 2021 20th International Conference on Advanced Robotics (ICAR), Ljubljana, Slovenia, 2021, pp. 682-687.
- W. Cai, H. N. Esfahani, A. B. Kordabad and S. Gros, "Optimal Management of the Peak Power Penalty for Smart Grids Using MPC-based Reinforcement Learning," 2021 60th IEEE Conference on Decision and Control (CDC), Austin, TX, USA, 2021, pp. 6365-6370.
- W. Cai, A. B. Kordabad, H. N. Esfahani, A. M. Lekkas and S. Gros, "MPC-based Reinforcement Learning for a Simplified Freight Mission of Autonomous Surface Vehicles," 2021 60th IEEE Conference on Decision and Control (CDC), Austin, TX, USA, 2021, pp. 2990-2995.
- A. B. Kordabad, H. Nejatbakhsh Esfahani and S. Gros, "Bias Correction in Deterministic Policy Gradient Using Robust MPC," 2021 European Control Conference (ECC), Delft, Netherlands, 2021, pp. 1086-1091.
- A. B. Kordabad, H. N. Esfahani, A. M. Lekkas and S. Gros, "Reinforcement Learning based on Scenario-tree MPC for ASVs," 2021 American Control Conference (ACC), New Orleans, LA, USA, 2021, pp. 1985-1990.

- A. B. Kordabad, H. Nejatbakhsh Esfahani, W. Cai and S. Gros, "Quasi-Newton Iteration in Deterministic Policy Gradient," 2022 American Control Conference (ACC), Atlanta, GA, USA, 2022, pp. 2124-2129.

1.4 Outline

Chapter 2 provides a background on the basic concepts, including reinforcement learning methods and learning-based MPC. Then, the proposed RL algorithms based on the MHE-MPC scheme and the approximate robust nonlinear MPC are detailed in Parts I (Chapters $3, 4, 5$) and II (Chapter 6), respectively. Finally, in Part III (Chapter 7), the works conducted in this thesis are concluded, and some of the possible future research directions are provided .

Chapter 2

Background

In this chapter, we first provide a brief summary on the Model Predictive Control (MPC) and Moving Horizon Estimation (MHE). We then detail the Markov Decision Processes (MDPs), which is a crucial concept to formulate the Reinforcement Learning (RL) problems. We next detail the RL algorithms, including classic Q-learning and deterministic policy gradient methods. We finally discuss the core idea of using MPC as an approximator in the context of RL as we use this idea to develop our new learning-based estimator/controller frameworks in this thesis.

2.1 Model Predictive Control and Moving Horizon Estimation

2.1.1 Model Predictive Control

Model Predictive Control (MPC), also known as Receding Horizon Control (RHC), has been widely adopted in industry as an effective optimal control approach to deal with multivariable constrained control problems. The basic concept of MPC is to use a dynamic model of the real system to predict its behavior, and optimize the forecast to produce the best decision (the control move at the current time). At each physical sampling time, an MPC scheme computes the control input and the corresponding state sequence minimizing an objective function while satisfying the constraints over a given prediction horizon [9]. In this thesis, we will mainly look at the Nonlinear MPC (NMPC) schemes in a multiple shooting context. In general, the NMPC scheme is not only related to a nonlinear dynamic but also to the presence of general nonlinear constraints or a non quadratic objective function. An NMPC scheme can be described by an Optimal Control Problem (OCP) as

follows:

$$\min_{\mathbf{x},\mathbf{u}} \quad T(\mathbf{x}_{k+N_{\text{MPC}}}) + \sum_{i=k}^{k+N_{\text{MPC}}-1} L(\mathbf{x}_i, \mathbf{u}_i) \tag{2.1a}$$

$$\text{s.t.} \quad \mathbf{x}_{i+1} = \mathbf{f}^{\text{MPC}}(\mathbf{x}_i, \mathbf{u}_i), \tag{2.1b}$$

$$\mathbf{x}_k = \hat{\mathbf{x}}_k, \tag{2.1c}$$

$$\mathbf{h}(\mathbf{x}_i, \mathbf{u}_i) \leq 0, \quad \mathbf{h}^f(\mathbf{x}_{k+N_{\text{MPC}}}) \leq 0 \tag{2.1d}$$

for a given state estimation $\hat{\mathbf{x}}_k$ (or a system state $\mathbf{s}_k$), where N_{MPC} is the prediction horizon length, L is the stage cost, T is the terminal cost, $\mathbf{f}^{\text{MPC}}$ is a model of the real system, $\mathbf{h}$ are the mixed input-state constraints and $\mathbf{h}^f$ collects the terminal constraints. Note that the initial states in the constraint (2.1c) are estimated using an observer, e.g., an MHE scheme, at each time instant k.

2.1.2 Moving Horizon Estimation

In many practical applications, some states of the real system are estimated using an observer since they can not be directly measured, and the real system is possibly not fully observable. The Moving Horizon Estimation (MHE) is a well-known model-based observer in order to estimate the states of processes. An MHE-based observer scheme at the physical time k can be formulated as the following nonlinear Least-Squares problem:

$$\{\hat{\mathbf{x}}_{k-N_{\text{MHE}},\ldots,k}, \hat{\mathbf{u}}_{k-N_{\text{MHE}},\ldots,k-1}\} = \arg\min_{\mathbf{x},\mathbf{u}} \quad Z_{k-N_{\text{MHE}}} + \sum_{i=k-N_{\text{MHE}}}^{k} \|\bar{\mathbf{y}}_i - \mathbf{y}(\mathbf{x}_i)\|_Q^2 + \sum_{i=k-N_{\text{MHE}}}^{k-1} \|\mathbf{u}_i - \bar{\mathbf{u}}_i\|_R^2 \tag{2.2a}$$

$$\text{s.t.} \quad \mathbf{x}_{i+1} = \mathbf{f}^{\text{MHE}}(\mathbf{x}_i, \mathbf{u}_i) \tag{2.2b}$$

where $\bar{\mathbf{y}}_i, \bar{\mathbf{u}}_i$ are the measurements available at the physical time k while their corresponding values obtained from the MHE model read as $\mathbf{y}(\mathbf{x}_i), \mathbf{u}_i$, respectively. Let the mismatch between the model (observer) and the real system measurements be explainable by the normal centered output noise $\boldsymbol{\nu}_k$ as follows:

$$\mathbf{y}_k = \mathbf{h}(\hat{\mathbf{x}}_k) + \boldsymbol{\nu}_k \tag{2.3}$$

The matrices Q and R then denote the inverse of the covariance matrices associated to these noises on the plant output and control input measurements, respectively.

To compute the arrival cost, one can use its approximation (approximate the information prior to $k - N_{\text{MHE}}$) in which the arrival cost takes the form of a quadratic

function weighted with inverse of the covariance matrix Π_{k-N} as follows:

$$Z_{k-N_{\text{MHE}}} = \|\mathbf{x}_{k-N_{\text{MHE}}} - \tilde{\mathbf{x}}_{k-N_{\text{MHE}}}\|^2_{\Pi^{-1}_{k-N}} \tag{2.4}$$

where $\tilde{\mathbf{x}}$ is the available estimation for the state at time $k - N_{\text{MHE}}$.

$$\tilde{\mathbf{x}} = \hat{\mathbf{x}}_{k-N_{\text{MHE}}|k-1} \tag{2.5}$$

The prior weighting Π_{k-N} is obtained from the Kalman filter covariance update rule [10]:

$$\begin{aligned}\Pi_{k+1} =& A_k \Pi_k A_k^\top \\ &- A_k \Pi_k C_k^\top \left(C_k \Pi_k C_k^\top + R\right)^{-1} C_k \Pi_k A_k^\top\end{aligned} \tag{2.6}$$

initialized with the covariance matrix of the initial state Π_0. Let $\mathbf{f}^{\text{MHE}}$ be a nonlinear model of the real system. The matrices A_k and C_k are then obtained by linearization as follows:

$$A_k = \frac{\partial \mathbf{f}^{\text{MHE}}}{\partial \hat{\mathbf{x}}}|_{\hat{\mathbf{x}}_{k|k-1}}, \qquad C_k = \frac{\partial \mathbf{h}}{\partial \hat{\mathbf{x}}}|_{\hat{\mathbf{x}}_{k|k-1}} \tag{2.7}$$

2.2 Markov Decision Process

2.2.1 Formulation

RL problems are mathematically formulated in an MDP framework, where the agent-environment interaction can be typically formulated as an MDP [11]. It is worth noting that the MDP problems are described based on a fully observable environment while the problems with a partially observable environment are formulated as Partially Observable Markov Decision Processes (POMDPs) [12].

Definition 1. *(MDP) A **Markov decision process** in a discounted setting is defined as a tuple $M = (\gamma, \mathcal{S}, \mathcal{A}, \mathbb{P}, R/L)$ where $\gamma \in (0, 1]$ is a discount factor, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathbb{P}[\mathbf{s}_+|\mathbf{s}, \mathbf{a}]$ is the transition probability (stochastic state transition dynamics) for a given state-action pair $\mathbf{s}, \mathbf{a} \in \mathcal{S} \times \mathcal{A}$ and R/L is the reward/stage cost obtained when taking action $\mathbf{a}$ so that a transition from a state $\mathbf{s}$ to a successive state $\mathbf{s}_+$ is observed.*

Definition 2. *(Policy) A policy (decision rule) $\boldsymbol{\pi}$ is a mapping from states to action, and it can be defined as deterministic $\boldsymbol{\pi} : \mathcal{S} \to \mathcal{A}$ or stochastic $\boldsymbol{\pi}[\mathbf{a}|\mathbf{s}]$ denoting the probability of taking action $\mathbf{a}$ at state $\mathbf{s}$.*

Remark 1. *In classic control, the model* $\mathbf{s}_+ = \mathbf{f}(\mathbf{s}, \mathbf{a})$ *for some function* $\mathbf{f}$ *is a special case of stochastic transition, by defining:*

$$\mathbb{P}[\mathbf{s}_+|\mathbf{s}, \mathbf{a}] = \delta(\mathbf{s}_+ - \mathbf{f}(\mathbf{s}, \mathbf{a})) \tag{2.8}$$

where δ reads as a Dirac measure. Solving MDPs is then interpreted as an optimal control problem in which the aim is to find a policy $\boldsymbol{\pi}^\star$ minimizing the total cost (a.k.a performance index) as follows:

$$J(\boldsymbol{\pi}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{s}_k, \mathbf{a}_k) \middle| \mathbf{a}_k = \boldsymbol{\pi}(\mathbf{s}_k)\right] \tag{2.9}$$

where the expectation $\mathbb{E}$ is taken over trajectories of the real system subject to policy $\boldsymbol{\pi}$.

2.2.2 Value Functions and Bellman Equations

We next provide two crucial definitions, including state value function $V^{\boldsymbol{\pi}}$ and state-action value function $Q^{\boldsymbol{\pi}}$ in an MDP framework, which express how good is to be in a given state and a given state-action pair, respectively. Note that we consider the MDP frameworks in a discounted setting assuring a well-posed MDP with a bounded optimal value function. The state (-action) value functions are then defined as follows:

$$V^{\boldsymbol{\pi}}(\mathbf{s}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{s}_k, \mathbf{a}_k) \middle| \mathbf{a}_k = \boldsymbol{\pi}(\mathbf{s}_k), \mathbf{s}_0 = \mathbf{s}\right] \tag{2.10a}$$

$$Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{s}_k, \mathbf{a}_k) \middle| \mathbf{s}_0 = \mathbf{s}, \mathbf{a}_0 = \mathbf{a}, \mathbf{a}_k = \boldsymbol{\pi}(\mathbf{s}_k), \forall k \geq 1\right] \tag{2.10b}$$

where the expectation $\mathbb{E}$ is taken over the state trajectories, which is stochastic as it follows the transition model $\mathbb{P}$.

Example 1. *Let* $s \in \mathbb{N}$ *and* $a \in \mathbb{N}$ *be the state-input pair locked on a grid [13]. The stage cost is* $L(s, a) = 0.5\left(s^2 + u^2\right)$*, and we consider the following dynamics:*

$$s_+ = s + a + \mathit{round}(e), \quad e \sim \mathcal{N}(0, 1) \tag{2.11}$$

and $a = \mathit{round}\left(-\frac{s}{10}\right)$*. The state (-action) value functions are then depicted in Figure 2.1.*

Note that the relation between (2.9) and (2.10a) can be described as:

$$J(\boldsymbol{\pi}) = \mathbb{E}_{\mathbf{s}_0 \sim p_0(\mathbf{s}0)}\left[V^{\boldsymbol{\pi}}(\mathbf{s}_0)\right] = \mathbb{E}_{\tau_{\boldsymbol{\pi}}}\left[L(\mathbf{s}, \mathbf{a})\right] \tag{2.12}$$

where p_0 is the initial state distribution and $\tau_{\boldsymbol{\pi}}$ is the MDP distribution subject to policy $\boldsymbol{\pi}$. To find an optimal policy from solving an MDP, we next introduce the Bellman's Principle of Optimality described by the optimal Bellman equations. Let $V^{\boldsymbol{\pi}^\star}(\mathbf{s}) = V^\star(\mathbf{s}) : \mathcal{S} \to \mathbb{R}$, $Q^{\boldsymbol{\pi}^\star}(\mathbf{s}, \mathbf{a}) = Q^\star(\mathbf{s}, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $A^\star(\mathbf{s}, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denote the optimal state value function, the optimal state-action value function and the optimal advantage function, respectively. The Bellman equations then read as:

$$Q^\star(\mathbf{s}, \mathbf{a}) = L(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}\left[V^\star(\mathbf{s}_+) \,|\mathbf{s}, \mathbf{a}\right], \tag{2.13a}$$

$$V^\star(\mathbf{s}) = \min_{\mathbf{a}} Q^\star(\mathbf{s}, \mathbf{a}), \tag{2.13b}$$

$$\boldsymbol{\pi}^\star(\mathbf{s}) = \arg\min_{\mathbf{a}} Q^\star(\mathbf{s}, \mathbf{a}), \tag{2.13c}$$

$$A^\star(\mathbf{s}, \mathbf{a}) = Q^\star(\mathbf{s}, \mathbf{a}) - V^\star(\mathbf{s}) \tag{2.13d}$$

where the expectation $\mathbb{E}$ is taken over the distribution $\mathbb{P}\left[\mathbf{s}_+ | \mathbf{s}, \mathbf{a}\right]$.

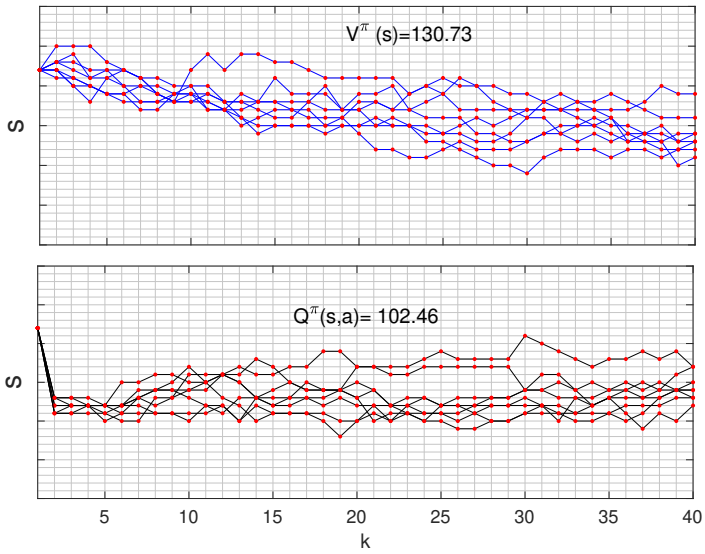


Figure 2.1: Illustration of state (-action) value functions

2.2.3 Dynamic Programming

Dynamic programming (DP) is a well-known, general-purpose method, and a fundamental to many reinforcement learning algorithms in order to find optimal control strategies based on an already given model of the real environment. Although DP will deliver the exact solutions of MDPs, the use of this method for the complex problems with higher state-action dimension is fairly challenging as an accurate model of the complex real systems is difficult to obtain. Moreover, the DP-based algorithms take a lot of memory to store the solutions of every iteration (subproblem) without ensuring whether the stored values will be exploited in the next stage. It is worth mentioning that the DP makes use of the Bellman equations discussed in the previous section in order to construct iterative algorithms for both the policy evaluation and improvement [14].

The policy evaluation is the task of computing the state value function $V^{\boldsymbol{\pi}}$ for a given policy $\boldsymbol{\pi}$. Therefore, for a given policy with any arbitrary value function $V_0(\mathbf{s})$, the following iterative procedure is used [13]:

$$V_{k+1} \leftarrow L(\mathbf{s}, \boldsymbol{\pi}(\mathbf{s})) + \gamma \mathbb{E}\left[V_k(\mathbf{s}_+) | \mathbf{s}, \boldsymbol{\pi}\right], \quad \textit{sweep over } \mathbf{s} \tag{2.14}$$

Then, for $\gamma < 1$

$$\lim_{k \to \infty} V_k(\mathbf{s}) = V^{\boldsymbol{\pi}}(\mathbf{s})$$

holds for any finite $V_0(\mathbf{s})$. Notice that the state (-action) value function can be computed jointly such that Q and V delivered from an iterative algorithm will converge to $V^{\boldsymbol{\pi}}$ and $Q^{\boldsymbol{\pi}}$, respectively. Hence, one needs to sweep over $\mathbf{s}$ and $\mathbf{a}$ in the following iteration:

$$Q(\mathbf{s}, \mathbf{a}) \leftarrow L(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}\left[V(\mathbf{s}_+) | \mathbf{s}, \mathbf{a}\right] \tag{2.15a}$$

$$V(\mathbf{s}) \leftarrow Q(\mathbf{s}, \boldsymbol{\pi}(\boldsymbol{\pi})) \tag{2.15b}$$

To find an exact solution to MDPs by solving the Bellman optimality equations, there are two DP algorithms, including policy iteration and value iteration.

Policy Iteration:

In the policy iteration approach, the policy evaluation as an iterative algorithm for a given policy $\boldsymbol{\pi}_0$ is first accomplished such that the sequences $V_0^{\boldsymbol{\pi}_0}, V_1^{\boldsymbol{\pi}_0}, \ldots, V_\infty^{\boldsymbol{\pi}_0} = V^{\boldsymbol{\pi}_0}$ are delivered. In the next stage, the policy improvement is achieved using $V_\infty^{\boldsymbol{\pi}_0} = V^{\boldsymbol{\pi}_0}$ to generate a better policy, i.e., $\boldsymbol{\pi}_1$ and $V^{\boldsymbol{\pi}_1} < V^{\boldsymbol{\pi}_0}$. The next value function $V_\infty^{\boldsymbol{\pi}_1} = V^{\boldsymbol{\pi}_1}$ is then computed and improved again to yield an even better policy $\boldsymbol{\pi}_2$. More specifically, we can obtain a sequence of improving policies and value functions as follows:

$$\boldsymbol{\pi}_0 \underset{E}{\rightarrow} V^{\boldsymbol{\pi}_0} \underset{I}{\rightarrow} \boldsymbol{\pi}_1 \underset{E}{\rightarrow} V^{\boldsymbol{\pi}_1} \underset{I}{\rightarrow} \boldsymbol{\pi}_2 \underset{E}{\rightarrow} \ldots \underset{I}{\rightarrow} \boldsymbol{\pi}^\star \underset{E}{\rightarrow} V^\star \tag{2.16}$$

where E and I denote the policy evaluation and the policy improvement, respectively. Hence, The policy evaluation and improvement are accomplished using the following updates:

$$\text{Compute} \quad V_{\infty}^{\boldsymbol{\pi}_i}: \quad V_{k+1}^{\boldsymbol{\pi}_i}(\mathbf{s}) \leftarrow L(\mathbf{s}, \boldsymbol{\pi}(\mathbf{s})) + \gamma \mathbb{E}\left[V_k^{\boldsymbol{\pi}_i}(\mathbf{s}_+) | \mathbf{s}, \boldsymbol{\pi}(\mathbf{s})\right] \tag{2.17a}$$

$$\boldsymbol{\pi}_{i+1} \leftarrow \arg\min_{\mathbf{a}} L(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}\left[V_{\infty}^{\boldsymbol{\pi}_i}(\mathbf{s}_+) | \mathbf{s}, \mathbf{a}\right] \tag{2.17b}$$

with k and i denoting the iteration number in the policy evaluation and the policy improvement stages, respectively. Note that the policy iteration will converge to an optimal policy and optimal value function in a finite number of iterations since a finite MDP has only a finite number of policies. However, the policy iteration algorithm may require multiple sweeps over the state space since each policy iteration involves policy evaluation, which itself is an iterative procedure described by equations (2.14) and (2.15).

Value Iteration:

Value iteration aims at finding the optimal action-value function for a given MDP in which an iterative algorithm is used to improve an estimate of the optimal value function until it converges to the true optimal value function. The algorithm starts with an initial arbitrary value $V_0^{\star}$ of the optimal value function and then repeatedly applies the Bellman optimality equations in order to update the estimate until it converges. Although this algorithm requires an infinite number of iterations to converge exactly to $V^{\star} = V_{\infty}^{\star}$, one can stop the iteration when no further change occurs within $\pm\epsilon$. The sequences of $V_{0,\ldots,\infty}^{\star}$ are obtained as follows:

$$Q_i^{\star}(\mathbf{s}, \mathbf{a}) \leftarrow L(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}\left[V_i^{\star}(\mathbf{s}_+) | \mathbf{s}, \mathbf{a}\right] \tag{2.18a}$$

$$V_{i+1}^{\star} \leftarrow \min_{\mathbf{a}} Q_i^{\star}(\mathbf{s}, \mathbf{a}) \tag{2.18b}$$

Then, after convergence to the optimal value functions $V^{\star}$ and $Q^{\star}$, the optimal policy reads as:

$$\boldsymbol{\pi}^{\star}(\mathbf{s}) = \arg\min_{\mathbf{a}} Q^{\star}(\mathbf{s}, \mathbf{a}) \tag{2.19}$$

2.3 Reinforcement Learning

To tackle some problems for solving MDPs with higher dimension and unknown environment, the Approximate Dynamic Programming (ADP) and RL methods are useful alternatives to DP . More specifically, they will provide an approximation of the optimal value functions and policy only by interaction with environment so that a model (an exact model) of MDP is not required. Therefore, we only need to know which parameters in our model we want to optimize while for a DP

algorithm one needs a deeper understanding of the environment. Moreover, the complexity issue upon the DP-based algorithms discussed in the previous section is addressed by using the RL techniques.

2.3.1 Classic Q-Learning

In this section, the Q-learning algorithm as a value-based reinforcement learning is detailed. The goal of Q-learning is to indirectly learn a policy, which tells the agent what action to take under what circumstances. It does not require any model of the environment, and it can handle problems with stochastic transitions and rewards, without requiring any adaptations.

In a classic Q-learning, one can use a tabular approximation where $\mathbf{s}, \mathbf{a}$ is gridded. We then learn the optimal action-value function $Q^\star$ starting from an arbitrary Q-table $\hat{Q}^\star$ as follows:

$$\delta = L\left(\mathbf{s}, \mathbf{a}\right) + \gamma \min_{\mathbf{a}_+} \hat{Q}^\star\left(\mathbf{s}_+, \mathbf{a}_+\right) - \hat{Q}^\star\left(\mathbf{s}, \mathbf{a}\right) \tag{2.20a}$$

$$\hat{Q}^\star\left(\mathbf{s}, \mathbf{a}\right) \leftarrow \hat{Q}^\star\left(\mathbf{s}, \mathbf{a}\right) + \alpha\delta \tag{2.20b}$$

where $\alpha > 0$ is a step size small enough. However, an accurate representation of the value functions and policy requires very fine grids. Moreover, in this basic version of Q-learning, one needs to generalize/extrapolate for using this method beyond the given state-action space. To generalize the tabular form above, one can parameterize the action-value function and learn the attached parameters to capture the optimal action-value function. In the next section, the Q-learning based on the generic approximator is investigated. Note that there are some well-known function approximators such as Artificial Neural Networks (ANN) , polynomials and Radial Basis Functions (RBFs) that might be used to formulate a parameterized generic approximator.

Q-Learning based on Generic Approximator

Let us consider a linear function approximation, e.g., $Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) = \boldsymbol{\phi}\left(\mathbf{s}, \mathbf{a}\right)^\top \boldsymbol{\theta}$ for some features $\boldsymbol{\phi}$. Q-learning then aims to update the parameters $\boldsymbol{\theta}$ such as to minimize the estimation error of the Q-function, which can be expressed by the following Least Squares (LS) problem:

$$\min_{\boldsymbol{\theta}} \mathbb{E}\left[\left(Q^\star(\mathbf{s}, \mathbf{a}) - Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a})\right)^2\right] \tag{2.21}$$

where the expected value $\mathbb{E}$ is taken over the system trajectories and actions $\mathbf{a}$. However, as the true action-value function $Q^\star(\mathbf{s}, \mathbf{a})$ is generally not known, it can be replaced by an approximation of the Bellman optimality equation as follows:

$$Q^\star(\mathbf{s}, \mathbf{a}) \approx L(\mathbf{s}, \mathbf{a}) + \gamma \min_{\mathbf{a}'} Q_{\boldsymbol{\theta}}(\mathbf{s}_+, \mathbf{a}') \tag{2.22}$$

where $0 < \gamma \leq 1$ is a discount factor. A classical approach to Q-learning is then parameter updates driven by the Temporal Difference (TD) learning as follows:

$$\delta = L(\mathbf{s}, \mathbf{a}) + \gamma \min_{\mathbf{a}'} Q_{\boldsymbol{\theta}}(\mathbf{s}_+, \mathbf{a}') - Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}), \tag{2.23a}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \delta \nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) \tag{2.23b}$$

where scalar $\alpha > 0$ is a step-size and δ is the TD error. In the above TD learning algorithm, a baseline stage cost $L(\mathbf{s}, \mathbf{a})$ is defined as a function of state-action pair in order to provide an evaluation signal. Indeed, the baseline cost affects the agent behavior and control policy via RL parameter updating, where the TD error is appeared. Finally, $Q_{\boldsymbol{\theta}^\star}$ reads as an approximation of the optimal action-value function $Q^\star$ such that an approximation of the optimal policy reads as:

$$\hat{\boldsymbol{\pi}}^\star(\mathbf{s}) = \arg\min_{\mathbf{a}} Q_{\boldsymbol{\theta}^\star}(\mathbf{s}, \mathbf{a}) \tag{2.24}$$

Q-Learning based on Least Squares Temporal Difference

Least squares temporal difference-based reinforcement learning methods, e.g., Least Squares Temporal Difference Q-learning (LSTDQ) make an efficient use of data and tend to converge faster than more basic temporal-difference learning methods [15]. To learn the action-value function $Q^{\boldsymbol{\pi}}$, one can form the least squares of Bellman residual error w.r.t $\boldsymbol{\theta}$ as follows:

$$\min_{\boldsymbol{\theta}} \mathbb{E}\left[\left(Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a}) - \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a})\right)^2\right] \tag{2.25}$$

Considering the Bellman equation for the action-value function, let us define the following approximation:

$$Q^{\boldsymbol{\pi}}(\mathbf{s}, \mathbf{a}) \approx L(\mathbf{s}, \mathbf{a}) + \gamma \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}}(\mathbf{s}_+, \pi(\mathbf{s}_+)) \tag{2.26}$$

where $L(\mathbf{s}, \mathbf{a})$ is the RL stage cost. By substituting (2.26) into (2.25), the least square problem (2.25) for the first-order LSTDQ-learning can be solved for some data acquired from the transitions as:

$$\mathbb{E}\left[\delta_Q \nabla_{\boldsymbol{\theta}} \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a})\right] = 0, \tag{2.27a}$$

$$\delta_Q = L(\mathbf{s}, \mathbf{a}) + \gamma \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}}(\mathbf{s}_+, \pi(\mathbf{s}_+)) - \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) \tag{2.27b}$$

where δ_Q is the temporal difference error. Let us consider a linear parameterization as follows:

$$\hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) = \boldsymbol{\phi}(\mathbf{s}, \mathbf{a})^\top \boldsymbol{\theta} \tag{2.28}$$

where $\boldsymbol{\phi}(\mathbf{s},\mathbf{a}) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n_\theta}$ are some basis functions. The LSTDQ equations then become a linear system in $\boldsymbol{\theta}$ such that:

$$\mathbb{E}\left[\boldsymbol{\phi}(\mathbf{s},\mathbf{a})\left(\boldsymbol{\phi}(\mathbf{s},\mathbf{a}) - \gamma\boldsymbol{\phi}(\mathbf{s}_+,\boldsymbol{\pi}(\mathbf{s}_+))\right)^\top\right]\boldsymbol{\theta} = \mathbb{E}\left[L(\mathbf{s},\mathbf{a})\,\boldsymbol{\phi}(\mathbf{s},\mathbf{a})\right] \tag{2.29}$$

One can also adopt a Newton method to solve (2.27) and extract the newton step for the second-order LSTDQ scheme as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha A^{-1} b, \tag{2.30a}$$

$$A = \mathbb{E}\left[\delta_Q \nabla^2_{\boldsymbol{\theta}} \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}} + \nabla_{\boldsymbol{\theta}} \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}} \left(\nabla_{\boldsymbol{\theta}} \delta_Q\right)^\top\right], \quad b = \mathbb{E}\left[\delta_Q \frac{\partial \hat{Q}^{\boldsymbol{\pi}}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}\right] \tag{2.30b}$$

where scalar $\alpha > 0$ is labelled the learning step-size.

2.3.2 Policy Gradient Method

This section presents the Policy Gradient (PG) method as a policy-based RL algorithm that attempts to optimize a policy directly, rather than indirectly via a value function. The policy is usually modeled with a parameterized function respect to $\boldsymbol{\theta}$, $\boldsymbol{\pi}_{\boldsymbol{\theta}}$. In the PG methods, the parameterized policy function is either stochastic or deterministic so that the policy type will determine whether the PG algorithm is deterministic or stochastic [16]. In a stochastic PG, the policy function $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\cdot|\mathbf{s})$ is modeled as a probability distribution over actions $\mathcal{A}$ given the current state $\mathbf{s}$ while the policy function in a deterministic case is regarded as a deterministic decision $\mathbf{a} = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})$, which dedicates a deterministic action to each state $\mathbf{s}$. However, the stochastic PG may require more data as the policy gradient is constructed by integrating over both state and action spaces [17]. In the present thesis, we then use a deterministic PG in which the parameterized policy is delivered from a parameterized MPC scheme.

Deterministic Policy Gradient

As discussed, the Q-learning methods seek the fitting of $Q_{\boldsymbol{\theta}}$ to $Q^\star$ using (2.25) such that $Q_{\boldsymbol{\theta}^\star} \approx Q^\star$ and consequently $\hat{\boldsymbol{\pi}}^\star \approx \boldsymbol{\pi}^\star$. However, this policy approximation may not hold even if the previous approximation holds. To tackle this issue, the policy gradient methods are useful as the policy is approximated directly. We will focus on the Deterministic PG (DPG) method that formally maximizes the policy performance based on the deterministic policy gradient theorem [17]. The policy parameters $\boldsymbol{\theta}$ can be directly optimized by the gradient descent steps such that the best expected closed-loop cost (a.k.a policy performance index J) can be captured by applying the policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$.

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) \tag{2.31}$$

The policy gradient then reads as:

$$\nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right) = \mathbb{E}\left[\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}\right) \nabla_{\mathbf{a}} Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}, \mathbf{a}\right)\Big|_{\mathbf{a}=\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s})}\right] \tag{2.32}$$

where the expectation $\mathbb{E}$ is taken over trajectories of the real system subject to policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$. Note that $\nabla_{\mathbf{a}} Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}, \mathbf{a}\right)$ can be replaced by $\nabla_{\mathbf{a}} A^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}, \mathbf{a}\right)$, where $A^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}, \mathbf{a}\right) = Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}, \mathbf{a}\right) - V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}\right)$ denotes the advantage function. In the next section, we will obtain an approximation of $A^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$. Then, a necessary condition of optimality to $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ reads as:

$$\nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right) = 0 \tag{2.33}$$

Note that we use an MPC scheme as an approximator for $\boldsymbol{\pi}_{\boldsymbol{\theta}} \approx \boldsymbol{\pi}^{\star}$, where $\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}$ is obtained by a sensitivity analysis on the MPC scheme [18, 8]. To capture $Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$, one can use a parameterized $Q^{\mathbf{w}}$, e.g, either a linear parameterization or a compatible function as an approximation of $Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ detailed in the next section. The TD actor-critic algorithm as a well-known approach in the context of DPG then uses the following updating rules:

$$\delta_k = L(\mathbf{s}_k, \mathbf{a}_k) + \gamma Q^{\mathbf{w}}(\mathbf{s}_{k+1}, \boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}_{k+1}\right)) - Q^{\mathbf{w}}(\mathbf{s}_k, \mathbf{a}_k), \tag{2.34a}$$

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha_w \delta_k \nabla_{\mathbf{w}} Q^{\mathbf{w}}(\mathbf{s}_k, \mathbf{a}_k) \tag{2.34b}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}_k\right) \nabla_{\mathbf{a}} Q^{\mathbf{w}}\left(\mathbf{s}_k, \boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}_k\right)\right) \tag{2.34c}$$

where $\alpha_w, \alpha_{\boldsymbol{\theta}} > 0$ are the learning step-sizes for the action-value (critic) and policy (actor) functions, respectively.

DPG using Compatible Approximator

Under some conditions detailed in [17], one can use a *compatible* approximation of the action-value function $Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s}_k, \mathbf{a}_k)$ in which a class of compatible function approximator $Q^{\mathbf{w}}(\mathbf{s}_k, \mathbf{a}_k)$ exists such that the policy gradient is preserved. Therefore, the compatible function for a deterministic policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ can be expressed as follows:

$$Q^{\mathbf{w}}(\mathbf{s}_k, \mathbf{a}_k) = \underbrace{\left(\mathbf{a}_k - \boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}\right)\right)^{\top} \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}^{\top}\left(\mathbf{s}_k\right) \mathbf{w}}_{A^{\mathbf{w}}} + V^{\boldsymbol{\nu}}\left(\mathbf{s}_k\right) \tag{2.35}$$

The first term in the above compatible function as critic part is an approximation for the advantage function $A^{\mathbf{w}} \approx A^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ and the second is a baseline function approximating the value function $V^{\boldsymbol{\nu}} \approx V^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$. Both functions can be computed by the linear function approximators as follows:

$$V^{\boldsymbol{\nu}}\left(\mathbf{s}_k\right) = \boldsymbol{\Upsilon}\left(\mathbf{s}_k\right)^{\top} \boldsymbol{\nu}, \tag{2.36a}$$

$$A^{\mathbf{w}}\left(\mathbf{s}_k, \mathbf{a}_k\right) = \boldsymbol{\Psi}\left(\mathbf{s}_k, \mathbf{a}_k\right)^{\top} \mathbf{w} \tag{2.36b}$$

where $\boldsymbol{\Upsilon}(\mathbf{s}_k)$ is labelled the feature vector, and $\boldsymbol{\Psi}(\mathbf{s}_k, \mathbf{a}_k)$ is computed as follows:

$$\boldsymbol{\Psi}(\mathbf{s}_k, \mathbf{a}_k) := (\mathbf{a}_k - \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}))^{\top} \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}^{\top}(\mathbf{s}_k) \tag{2.37}$$

The parameters $\mathbf{w}$ and $\boldsymbol{\nu}$ of the action-value function approximation then become the solutions of the following Least Squares (LS) problem:

$$\min_{\mathbf{w},\boldsymbol{\nu}} \mathbb{E}\left[\left(Q^{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s}, \mathbf{a}) - Q^{\mathbf{w}}(\mathbf{s}, \mathbf{a})\right)^2\right], \tag{2.38}$$

The problem above can be solved via an LSTD method, which belongs to *batch method*, seeking to find the best fitting state (-action) value functions, and it is more sample efficient than other methods [19]. The LSTD update rules then read as:

$$\boldsymbol{\nu} = \Omega_{\boldsymbol{\nu}}^{-1} b_{\boldsymbol{\nu}}, \tag{2.39a}$$

$$\mathbf{w} = \Omega_{\mathbf{w}}^{-1} b_{\mathbf{w}}, \tag{2.39b}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha b_{\boldsymbol{\theta}} \tag{2.39c}$$

where the matrices $\Omega_{(\cdot)}$ and the vectors $b_{(\cdot)}$ are computed by taking expectation ($\mathbb{E}_m$) over m episodes as follows:

$$\Omega_{\boldsymbol{\nu}} = \mathbb{E}_m\left[\sum_{k=1}^{T_f}\left[\boldsymbol{\Upsilon}(\mathbf{s}_k)\left(\boldsymbol{\Upsilon}(\mathbf{s}_k) - \gamma \boldsymbol{\Upsilon}(\mathbf{s}_{k+1})\right)^{\top}\right]\right] \tag{2.40a}$$

$$\Omega_{\mathbf{w}} = \mathbb{E}_m\left[\sum_{k=1}^{T_f}\left[\boldsymbol{\Psi}(\mathbf{s}_k, \mathbf{a}_k)\,\boldsymbol{\Psi}(\mathbf{s}_k, \mathbf{a}_k)^{\top}\right]\right], \tag{2.40b}$$

$$b_{\boldsymbol{\nu}} = \mathbb{E}_m\left[\sum_{k=1}^{T_f} \boldsymbol{\Upsilon}(\mathbf{s}_k)\, L(\mathbf{s}_k, \mathbf{a}_k)\right], \tag{2.40c}$$

$$b_{\mathbf{w}} = \tag{2.40d}$$

$$\mathbb{E}_m\left[\sum_{k=1}^{T_f}\left[\left(L(\mathbf{s}_k, \mathbf{a}_k) + \gamma V^{\boldsymbol{\nu}}(\mathbf{s}_{k+1}) - V^{\boldsymbol{\nu}}(\mathbf{s}_k)\right)\boldsymbol{\Psi}(\mathbf{s}_k, \mathbf{a}_k)\right]\right],$$

$$b_{\boldsymbol{\theta}} = E_m\left[\sum_{k=1}^{T_f} \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)\, \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}^{\top}(\mathbf{s}_k)\, \mathbf{w}\right] \tag{2.40e}$$

where T_f is the final time instant at the end of each episode.

2.4 MPC-based Reinforcement Learning

The main idea of using MPC as a function approximator in the context of RL was first developed in [8, 20]. Successful applications that build on this result include

[21, 22, 23, 24]. In this section, we briefly discuss the fundamental principles behind this concept aiming to use an MPC scheme in order to approximate the optimal value function and policy required in both the Q-learning and policy gradient methods. It is worth mentioning that the MPC is a well-known control algorithm in handling the state and input constraints such that this property is leveraged to provide a safe RL paradigm. However, the MPC performance can be poor in presence of some process noise or model mismatch, where this performance degradation can be tackled by using a combined RL-MPC framework. More precisely, the theorem developed in [8] showed that the terminal and stage costs of the MPC shceme can be modified such that the value function associated to the MPC can capture the optimal value function of the MDP even if the MPC model is imperfect. In practice, a parameterized MPC scheme is used in the MPC-based RL providing a parameterized state (-action) value function in the context of Q-learning $V_{\boldsymbol{\theta}}, Q_{\boldsymbol{\theta}}$ and a parameterized policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ in the context of policy gradient.

Let us formulate a parameterized MPC scheme as follows:

$$
\begin{aligned}
V_{\boldsymbol{\theta}}(\mathbf{s}) = \min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad & \gamma^N \left(V_{\boldsymbol{\theta}}^f(\mathbf{x}_{k+N}) + \mathbf{w}_f^\top \boldsymbol{\sigma}_{k+N} \right) \\
& + \sum_{i=k}^{k+N-1} \gamma^{i-k} \left(l_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{u}_i) + \mathbf{w}^\top \boldsymbol{\sigma}_i \right) && \text{(2.41a)} \\
\text{s.t.} \quad & \mathbf{x}_{i+1} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{u}_i), && \text{(2.41b)} \\
& \mathbf{x}_k = \mathbf{s}, && \text{(2.41c)} \\
& \mathbf{g}(\mathbf{u}_i) \leq 0, && \text{(2.41d)} \\
& \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{u}_i) \leq \boldsymbol{\sigma}_i, \quad \mathbf{h}_{\boldsymbol{\theta}}^f(\mathbf{x}_{k+N}) \leq \boldsymbol{\sigma}_{k+N} && \text{(2.41e)} \\
& \boldsymbol{\sigma}_{k,\ldots,k+N} \geq 0 && \text{(2.41f)}
\end{aligned}
$$

where the parameterized functions $l_{\boldsymbol{\theta}}, V_{\boldsymbol{\theta}}^f, f_{\boldsymbol{\theta}}, \mathbf{h}_{\boldsymbol{\theta}}, \mathbf{h}_{\boldsymbol{\theta}}^f$ are the stage cost, the terminal cost, the MPC model, the mixed constraints and the terminal constraints, respectively. $\mathbf{g}$ is labelled the pure input constraints. In many real processes, there are uncertainties and disturbances that may cause an MPC scheme to become infeasible. Therefore, an ℓ_1 relaxation of the mixed constraints (2.41e) is introduced. An exact penalty is then imposed on the corresponding slack variables $\boldsymbol{\sigma}_k$ with large enough weights $\mathbf{w}, \mathbf{w}_f$ such that the trajectories predicted by the MPC scheme will respect the constraints. All elements in the above MPC scheme are parameterized by $\boldsymbol{\theta}$, which will be adjusted by RL. The policy at the physical current time k then reads as:

$$
\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}) = \mathbf{u}_k^\star(\mathbf{s}, \boldsymbol{\theta}) \tag{2.42}
$$

where $\mathbf{u}_k^\star$ is the first element of the input sequence $\mathbf{u}_k^\star, \cdots, \mathbf{u}_{k+N_{\text{MPC}}-1}^\star$ solution of (2.41). We next consider this optimal policy delivered by the MPC scheme as an action $\mathbf{a}$ in the context of reinforcement learning, where it is selected according to the above policy with the possible addition of exploratory moves.

2.4.1 MPC-based Q-Learning

An MPC-based approximator for the action-value function $Q_{\boldsymbol{\theta}}$ can be formulated as:

$$
\begin{aligned}
Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) = \min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad & (2.41a) && (2.43a)\\
\text{s.t.} \quad & (2.41b) - (2.41f) && (2.43b)\\
& \mathbf{u}_k = \mathbf{a} && (2.43c)
\end{aligned}
$$

Note that the proposed approximators (2.41)-(2.43) satisfies the fundamental equalities underlying the Bellman equations such that:

$$
\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}) \in \arg\min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}), \quad V_{\boldsymbol{\theta}}(\mathbf{s}) = \min_{\mathbf{a}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) \tag{2.44}
$$

The parameter updating rule then becomes:

$$
\begin{aligned}
\delta &= L(\mathbf{s}, \mathbf{a}) + \gamma V_{\boldsymbol{\theta}}(\mathbf{s}_+) - Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}), && (2.45a)\\
\boldsymbol{\theta} &\leftarrow \boldsymbol{\theta} + \alpha\delta\nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) && (2.45b)
\end{aligned}
$$

Example 2. *(MPC-based Q-Learning) In this example [8], we investigate the MPC-based Q-learning detailed above, where the following optimization problem as a parameterized MPC scheme is solved at each time instant k:*

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad & \theta_c + \frac{\gamma^N}{2}\left(\mathbf{x}_{k+N}^\top M_\theta^f \mathbf{x}_{k+N} + \mathbf{w}_f^\top \boldsymbol{\sigma}_{k+N}\right)\\
& + \sum_{i=k}^{k+N-1} \frac{\gamma^{i-k}}{2}\left(\mathbf{c}^\top M_\theta \mathbf{c} + \mathbf{w}^\top \boldsymbol{\sigma}_i\right) + \mathbf{f}^\top \mathbf{c} && (2.46a)\\
\text{s.t.} \quad & \mathbf{x}_{i+1} = A\mathbf{x}_i + B\mathbf{u}_i + \mathbf{b}, && (2.46b)\\
& \mathbf{x}_k = \mathbf{s}, && (2.46c)\\
& -1 \le \mathbf{u}_i \le 1, && (2.46d)\\
& \begin{bmatrix} 0 \\ -1 \end{bmatrix} + \underline{\boldsymbol{\theta}} - \boldsymbol{\sigma}_i \le \mathbf{x}_i \le \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \bar{\boldsymbol{\theta}} + \boldsymbol{\sigma}_i && (2.46e)\\
& \boldsymbol{\sigma}_{k,\ldots,k+N} \ge 0 && (2.46f)
\end{aligned}
$$

where $\mathbf{c} = [\mathbf{x}_k, \mathbf{u}_k]^\top$ *and the MPC parameters subject to the RL scheme are:*

$$\boldsymbol{\theta} = \left(\theta_c, M_\theta^f, \mathbf{f}, M_\theta, A, B, \underline{\boldsymbol{\theta}}, \bar{\boldsymbol{\theta}}\right) \tag{2.47}$$

The positive semidefinite weighting matrices (M_θ^f, M_θ) *are adjusted using the constrained RL steps, e.g., a Semidefinite Programming (SDP) [22]. One can choose a baseline stage cost used in the updating rule* (2.45) *as follows:*

$$L(\mathbf{x}_k, \mathbf{u}_k) = l(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}^\top \max(0, \mathbf{h}(\mathbf{x}_k)) \tag{2.48}$$

where $l(\mathbf{x}_k, \mathbf{u}_k)$ *is adopted as a quadratic function. The second term in the above baseline is considered to penalize the constraint violations, where* $\mathbf{h} \geq 0$ *is pure inequality vector of constraints on the states and* $\mathbf{w}^\top = [100, 100]$. *The step size is* $\alpha = 10^{-7}$. *The real system has the following dynamics:*

$$\mathbf{x}_{k+1} = \begin{bmatrix} 0.9 & 0.35 \\ 0 & 1.1 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} 0.0813 \\ 0.2 \end{bmatrix} u_k + \begin{bmatrix} e_k \\ 0 \end{bmatrix} \tag{2.49}$$

and we choose an imperfect model for the MPC scheme as:

$$A = \begin{bmatrix} 1 & 0.25 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0312 \\ 0.25 \end{bmatrix} \tag{2.50}$$

where the disturbance e_k *is random, uncorrelated and uniformly distributed variable in the interval* $[-0.13, 0]$.

The first state of the real system x_1 has a reference point and lower bound at zeros shown in Figure 2.2. Hence, this state violates its constraint at the beginning stage of the Q-learning due to the disturbance e_k while the violation is tackled after some state transitions, and the state keeps its distance as close as possible to the desired point at 0.

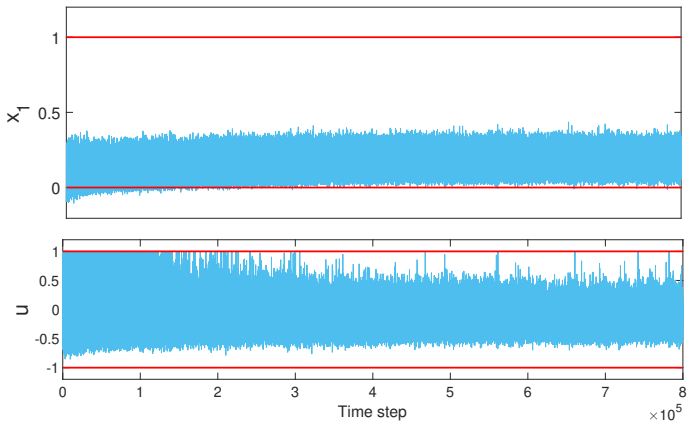
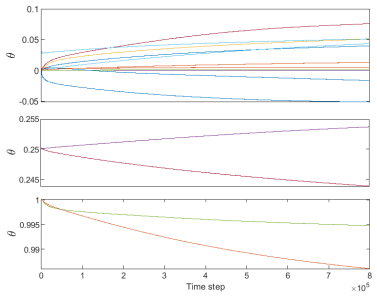
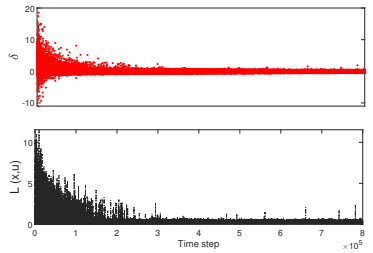


Figure 2.2: Evolution of the first state x_1 and the action u. Constraint violation is observed on the first state of the real system x_1 when the Q-learning process starts.



(a) Some MPC parameters adjusted by RL



(b) TD-error and RL stage cost

Figure 2.3: Q-Learning algorithm based on MPC-based value function approximator

2.4.2 MPC-based Deterministic Policy Gradient

In an MPC-based DPG algorithm, one needs to compute two terms, including compatible action value function $Q^{\mathbf{w}}(\mathbf{s}_k, \mathbf{a}_k)$ and policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$. These terms are constructed based on the policy sensitivity term $\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k)$, which can be computed by using a sensitivity analysis on the MPC scheme. To this end, let us define the primal-dual Karush Kuhn Tucker (KKT) conditions underlying the MPC scheme (2.41) as follows:

$$\mathbf{R} = \left[\begin{array}{c} \nabla_{\boldsymbol{\zeta}} \mathcal{L}_{\boldsymbol{\theta}}, \mathbf{G}_{\boldsymbol{\theta}}, \operatorname{diag}(\boldsymbol{\mu}) \mathbf{H}_{\boldsymbol{\theta}} \end{array} \right]^{\top} \tag{2.51}$$

where $\boldsymbol{\zeta}_k = \{\mathbf{x}, \mathbf{u}, \boldsymbol{\sigma}\}$ includes the primal decision variables of (2.41) and the term $\mathcal{L}_{\boldsymbol{\theta}}$ is the associated Lagrange function as follows:

$$\mathcal{L}_{\boldsymbol{\theta}}(\mathbf{y}_k) = \Phi_{\boldsymbol{\theta}} + \boldsymbol{\lambda}^\top \mathbf{G}_{\boldsymbol{\theta}} + \boldsymbol{\mu}^\top \mathbf{H}_{\boldsymbol{\theta}}, \tag{2.52}$$

where $\Phi_{\boldsymbol{\theta}}$ is the MPC cost (2.41a), $\mathbf{G}_{\boldsymbol{\theta}}$ gathers the equality constraints and $\mathbf{H}_{\boldsymbol{\theta}}$ collects the inequality constraints of the MPC (2.41). Let $\boldsymbol{\lambda}, \boldsymbol{\mu}$ be the associated dual variables. Argument $\mathbf{y}_k$ reads as $\mathbf{y}_k = \{\boldsymbol{\zeta}, \boldsymbol{\lambda}, \boldsymbol{\mu}\}$ and $\mathbf{y}_k^\star$ refers to the solution of the MPC (2.41). Consequently, the policy sensitivity $\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}$ can then be obtained as follows [25]:

$$\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}_k) = -\nabla_{\boldsymbol{\theta}} \mathbf{R}(\mathbf{y}_k^\star, \mathbf{s}_k, \boldsymbol{\theta}) \nabla_{\mathbf{y}_k} \mathbf{R}(\mathbf{y}_k^\star, \mathbf{s}_k, \boldsymbol{\theta})^{-1} \frac{\partial \mathbf{y}_k}{\partial \mathbf{u}_k}, \tag{2.53}$$

Example 3. *Let us consider the previous Example 2. We then use an MPC-based DPG algorithm in which the RL parameters are updated by using an LSTD method detailed by* (2.38)-(2.40)*.*

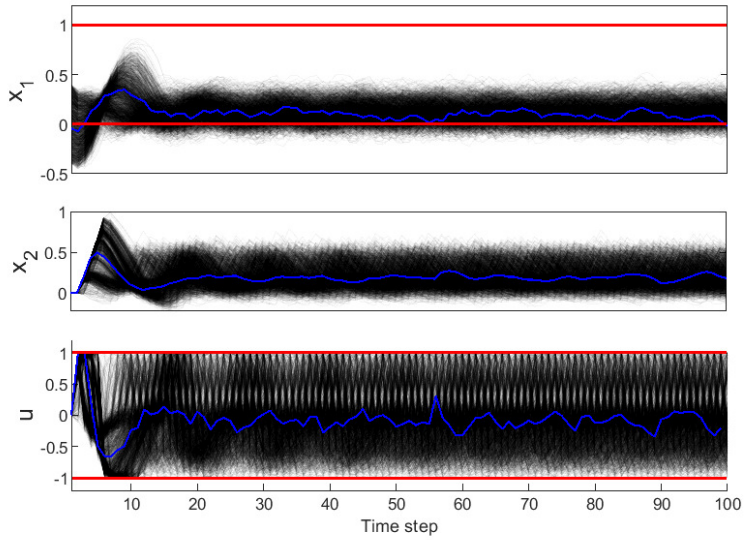
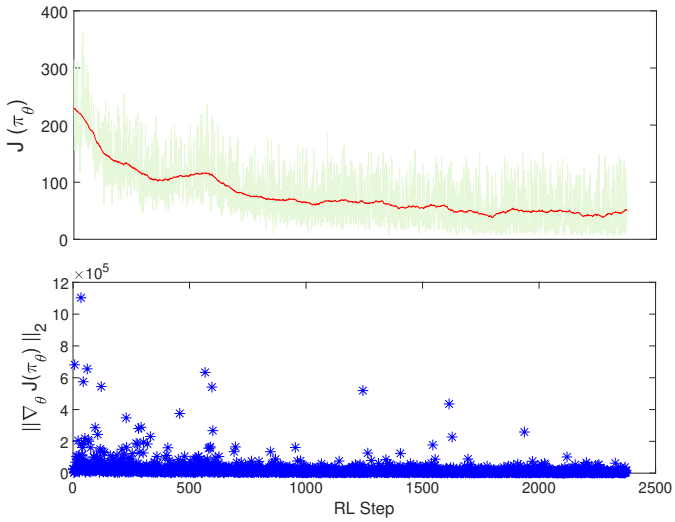
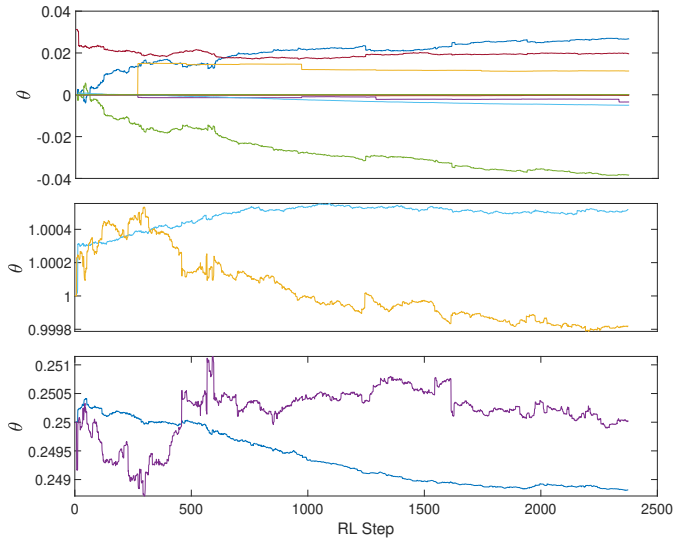


Figure 2.4: Evolution of the states and control input. The blue lines show the results after the learning progress is terminated. As observed, the constraint violation on the first state x_1 is disappeared.



(a)



(b)

Figure 2.5: MPC-based DPG algorithm using an LSTD method. (a) shows the closed-loop performance $J(\pi_\theta)$ and the norm of the policy gradient steps $||\nabla_\theta J(\pi_\theta)||_2$. (b) shows the evolution of the policy parameters.

Part I

Reinforcement Learning based on MHE-MPC

Chapter 3

Reinforcement learning based on MPC-MHE for unmodeled and partially observable dynamics

- H. N. Esfahani, A. B. Kordabad and S. Gros, "Reinforcement Learning based on MPC/MHE for Unmodeled and Partially Observable Dynamics," 2021 American Control Conference (ACC), New Orleans, LA, USA, 2021, pp. 2121-2126, doi: 10.23919/ACC50511.2021.9483399.

This chapter proposes an observer-based framework for solving Partially Observable Markov Decision Processes (POMDPs) when an accurate model is not available. We first propose to use a Moving Horizon Estimation-Model Predictive Control (MHE-MPC) scheme in order to provide a policy for the POMDP problem, where the full state of the real process is not measured and necessarily known. We propose to parameterize both MPC and MHE formulations, where certain adjustable parameters are regarded for tuning the policy. In this chapter, for the sake of tackling the unmodeled and partially observable dynamics, we leverage the Reinforcement Learning (RL) to tune the parameters of MPC and MHE schemes jointly, with the closed-loop performance of the policy as a goal rather than model fitting or the MHE performance. Illustrations show that the proposed approach can effectively increase the performance of close-loop control of systems formulated as POMDPs.

3.1 Introduction

Reinforcement Learning (RL) is a powerful tool for solving Markov Decision Processes (MDP) problems [11]. RL methods often use Deep Neural Network (DNN) to approximate either the optimal policy underlying the MDP directly or the action-value function from which the optimal policy can be indirectly extracted.

Recent publications are discussing RL for POMDPs. A neural network-based computation of belief states (posterior distributions over states) was proposed to aggregate historical information needed to estimate a belief state [26, 27]. An RL algorithm tailored to POMDPs was proposed in [12] that incorporated spectral parameter estimation within an exploration-exploitation strategy. A data-driven algorithm based on approximate Dynamic Programming (ADP) was proposed in [28] to stabilize a plant with partially observable dynamics. The authors used an Action-Dependent Heuristic Dynamic Programming (ADHDP) algorithm, including two neural networks as an actor-critic (AC) method to estimate both the unmeasured state and the performance index. The proposed ADP-based approach in [29] is similar to classic RL algorithms but requires only measurements of the input/output data and not of the full system state.

In [30], a neural network-based actor-critic structure was proposed to approximate the control policies where a full system state is not accessible. In [31], a fuzzy neural network was used to find the local optimal policy. A Recurrent Neural Network (RNN) was proposed in [32] to learn and infer the true state observations from the noisy and correlated observations in a POMDP. Most of the proposed RL-based control techniques in the above literature are based on DNN-based approximators.

Model predictive control (MPC) is a popular and widely used practical approach to optimal control. MPC is often selected for its capability to handle both input and state constraints [9]. At each time instant, MPC calculates the input and corresponding state sequence minimizing a cost function while satisfying the constraints over a given prediction horizon. The first input is applied, and the optimal solution is recalculated at the next time instant based on the latest state of the system.

In many practical applications, some states of the plant are estimated using an observer since they can not be directly measured, and the plant is possibly not fully observable. The Moving Horizon Estimation (MHE) is a well-known model-based observer in order to estimate the states of processes. In this chapter, we use this type of observer as a natural choice for the MPC scheme [33].

Recently, the integration of machine learning in model predictive control has been presented, with the aim of learning the model of the system, the cost function or even the control law directly [34, 35]. These approaches are based on DNN-based approximation. The direct combination of RL and MPC has been investigated in [36, 37, 38]. It is shown that a single MPC scheme can capture the optimal value function, action-value function, and policy of an MDP, even if the MPC model is inaccurate, hence providing a valid and generic function approximator for RL. The applications of this new MPC-based RL framework have been recently presented in [39, 40].

However, these approaches assume that the state of the process is known and can be fully measured. For many applications this assumption is not fulfilled. To address this issue, this chapter proposes to use a state observer such as an MHE combined with the MPC scheme to build a policy based on the historic of the available measurements rather than on the full state of the system. MHE delivers state estimations by fitting the process model trajectory to past measurements obtained on the real system. We adopt an MPC-based Q-learning algorithm to tune the parameters included in the MHE-MPC scheme for the closed-loop performance of the resulting policy.

This chapter is organised as follows. In Section 3.2, some background material is given. Then the MPC and MHE schemes are detailed. The implementation of the Q-learning algorithm for tuning both the MPC and MHE schemes together is detailed in Section 3.3. An illustrative example is proposed in Section 3.4. Finally, conclusions and future work are given in Section 3.5.

3.2 Preliminaries and problem formulation

In the context of reinforcement learning a partially observable real plant is described by a discrete POMDP having (possibly) stochastic state transitions as fol-

lows:

$$\mathbf{x}_{k+1} = \mathbf{f}^{\text{plant}}(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\zeta}) \tag{3.1a}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \boldsymbol{\eta}) \tag{3.1b}$$

where the full state $\mathbf{x}_k$ is not measurable or not even known and $\mathbf{x}_{k+1}$ is the next plant state vector under stochastic transition with some random disturbances $\boldsymbol{\zeta}$. The model outputs $\mathbf{y}$ are measured on the real system and delivered by the output function $\mathbf{h}$ associated with some random measurement noises $\boldsymbol{\eta}$. We present next the MHE and MPC schemes and how they can be used to create the action-value function approximation required in Q-learning.

3.2.1 Parameterized MHE Formulation

For a POMDP, the measurements available from the real process at a given time instant do not constitute a Markov state. As a result, the full history of the measurements becomes possibly relevant to the optimal policy. However, building a policy based on the complete measurement history to solve the POMDPs is not realistic. The RL community either considers a limited sequence of past observations as a sufficient history or estimates a belief state using a recurrent neural network. In this chapter, we propose a more structured solution to address this issue, by using MHE as a model-based approach to build a state from the measurement history. The complete measurement history is then transformed into a (possibly small) model state that is compatible with the selected policy. The MHE-based observer at the physical time k can be stated as the following Nonlinear Least-Squares problem:

$$\begin{aligned}\{\hat{\mathbf{x}}_{k-N_{\text{MHE}},\ldots,k}, \hat{\mathbf{u}}_{k-N_{\text{MHE}},\ldots,k-1}\} = \\ \underset{\mathbf{x},\mathbf{u}}{\arg\min} \quad \|\mathbf{x}_{k-N_{\text{MHE}}} - \tilde{\mathbf{x}}_{k-N_{\text{MHE}}}\|^2_{A_r^{\boldsymbol{\theta}}} \\ + \sum_{i=k-N_{\text{MHE}}}^{k} \|\bar{\mathbf{y}}_i - \mathbf{y}(\mathbf{x}_i)\|^2_{Q_E^{\boldsymbol{\theta}}} + \phi_{\boldsymbol{\theta}}(\mathbf{x}_i) + \sum_{i=k-N_{\text{MHE}}}^{k-1} \|\mathbf{u}_i - \bar{\mathbf{u}}_i\|^2_{R_E^{\boldsymbol{\theta}}} + \phi_{\boldsymbol{\theta}}(\mathbf{u}_i)\end{aligned} \tag{3.2a}$$

$$\text{s.t.} \quad \mathbf{x}_{i+1} = \mathbf{f}_\theta^{\text{MHE}}(\mathbf{x}_i, \mathbf{u}_i) \tag{3.2b}$$

where k is the current time instant, i is the time instant along the estimation horizon window. $\bar{\mathbf{y}}_i, \bar{\mathbf{u}}_i$ are the measurements available at the physical time k while their corresponding values obtained from the MHE model are $\mathbf{y}(\mathbf{x}_i), \mathbf{u}_i$, respectively. Let us consider the mismatch between the model (observer) and the real plant measurements is explainable by normal centered output noise, then matrices $Q_E^{\boldsymbol{\theta}}$ and $R_E^{\boldsymbol{\theta}}$ are the inverse of the covariance matrices associated to these noises on the plant output and control input measurements, respectively. The first term in

(3.2a) is an arrival cost weighted with matrix $A_r^{\boldsymbol{\theta}}$, which aims at approximating the information prior to $k - N_{\text{MHE}}$, where $\tilde{\mathbf{x}}$ is the available estimation for the state at time $k - N_{\text{MHE}}$. In practice, since the MHE fitting error is not only coming from some normal centered output noise but also model error, more intricate noise, and possibly unmodelled dynamics, it is very difficult to decide what symmetric positive semi-definite weighting matrices $Q_E^{\boldsymbol{\theta}}$, $R_E^{\boldsymbol{\theta}}$, $A_r^{\boldsymbol{\theta}}$ ought to be used to obtain the best closed-loop performance. To address this issue, we propose to adjust them using the RL algorithm. Moreover, as the Least-Squares cost as a choice of penalty in the MHE are not necessarily sufficient, we introduce a cost modification $\phi_{\boldsymbol{\theta}}$ tuned by RL. Note that we consider a gradient form of the cost modification in this chapter $\phi_{\boldsymbol{\theta}}(\mathbf{x}_i) = \mathbf{f}_1^\top \mathbf{x}_i$ and $\phi_{\boldsymbol{\theta}}(\mathbf{u}_i) = \mathbf{f}_2^\top \mathbf{u}_i$, where $\mathbf{f}_1$ and $\mathbf{f}_2$ are labeled as RL parameters $\boldsymbol{\theta}$.

3.2.2 Parameterized MPC Formulation

In this work we will consider the MPC scheme as a value function approximator that can be formulated as:

$$
\begin{aligned}
V_{\boldsymbol{\theta}}(\mathbf{x}_k) = \min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad & \gamma^{N_{\text{MPC}}} \left(V_{\boldsymbol{\theta}}^f(\mathbf{x}_{k+N_{\text{MPC}}}) + \mathbf{w}_f^\top \boldsymbol{\sigma}_{k+N_{\text{MPC}}} \right) \\
& + \sum_{i=k}^{k+N_{\text{MPC}}-1} \gamma^{i-k} \left(l_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{u}_i) + \mathbf{w}^\top \boldsymbol{\sigma}_i \right) && \text{(3.3a)} \\
\text{s.t.} \quad & \mathbf{x}_{i+1} = \mathbf{f}_{\boldsymbol{\theta}}^{\text{MPC}}(\mathbf{x}_i, \mathbf{u}_i), && \text{(3.3b)} \\
& \mathbf{x}_k = \hat{\mathbf{x}}_k, && \text{(3.3c)} \\
& \mathbf{g}(\mathbf{u}_i) \leq 0, && \text{(3.3d)} \\
& \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{u}_i) \leq \boldsymbol{\sigma}_i, \quad \mathbf{h}_{\boldsymbol{\theta}}^f(\mathbf{x}_{k+N_{\text{MPC}}}) \leq \boldsymbol{\sigma}_{k+N_{\text{MPC}}} && \text{(3.3e)} \\
& \boldsymbol{\sigma}_{k,\ldots,k+N_{\text{MPC}}} \geq 0 && \text{(3.3f)}
\end{aligned}
$$

where $l_{\boldsymbol{\theta}}$ is the stage cost, $V_{\boldsymbol{\theta}}^f$ the terminal cost, $f_{\boldsymbol{\theta}}^{\text{MPC}}$ the MPC model (possibly but not necessarily different from the MHE model), $\mathbf{h}_{\boldsymbol{\theta}}$ the mixed constraints, $\mathbf{g}$ the pure input constraints, $\mathbf{h}_{\boldsymbol{\theta}}^f$ the terminal constraints. The MPC initial conditions in (3.3c) are delivered by MHE scheme at the current time instant k. In many real processes, there are uncertainties and disturbances that may cause an MPC scheme to become infeasible. Therefore, an ℓ_1 relaxation of the mixed constraints (3.3e) is introduced. An exact penalty is imposed on the corresponding slack variables $\boldsymbol{\sigma}_k$ with large enough weights $\mathbf{w}, \mathbf{w}_f$ such that the trajectories predicted by the MPC scheme will respect the constraints. All elements in the above MPC scheme are parameterized by $\boldsymbol{\theta}$, which will be adjusted by RL, as detailed in [36].

Let us consider the policy at the physical current time k as:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{x}_k) = \mathbf{u}_k^* \tag{3.4}$$

where, $\mathbf{u}_k^\star$ is the first element of the input sequence $\mathbf{u}_k^\star, \cdots, \mathbf{u}_{k+N_{\text{MPC}}-1}^\star$ solution of (3.3). We next consider this optimal policy delivered by the MPC scheme as an action $\mathbf{a}_k$ in the context of reinforcement learning where, it is selected according to the above policy with the possible addition of exploratory moves [11]. Then, an action-value function approximation $Q_{\boldsymbol{\theta}}$ can be formulated as:

$$Q_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k) = \min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad (3.3a) \tag{3.5a}$$

$$\text{s.t.} \quad (3.3b) - (3.3f) \tag{3.5b}$$

$$\mathbf{u}_k = \mathbf{a}_k \tag{3.5c}$$

Note that the proposed approximations (3.3)-(3.5) satisfies the fundamental equalities underlying the Bellman equations [41]:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{x}_k) = \arg\min_{\mathbf{u}} Q_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k), \quad V_{\boldsymbol{\theta}}(\mathbf{x}_k) = \min_{\mathbf{u}} Q_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k) \tag{3.6}$$

3.3 MPC-MHE-based RL

In this section, we present the algorithmic details needed to implement a classic Q-learning algorithm on the combination of MPC-MHE schemes.

3.3.1 Q-Learning for MPC-MHE

A classical off-policy Q-Learning algorithm is based on the temporal-difference learning procedure [11] in which the updating rule for the RL parameters can be expressed as follows:

$$\delta_k = L(\mathbf{x}_k, \mathbf{u}_k) + \gamma V_{\boldsymbol{\theta}}(\mathbf{x}_{k+1}) - Q_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k), \tag{3.7a}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \delta_k \nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k) \tag{3.7b}$$

where scalar $\alpha > 0$ is a step size, $0 < \gamma \leq 1$ is a discount factor and δ_k is the Temporal Difference (TD) error at the physical time k. In the above TD learning algorithm, a baseline stage cost $L(\mathbf{x}_k, \mathbf{u}_k)$ (reward in the context of RL) is defined as a function of state-action pair in order to provide an evaluation signal. Indeed, the baseline cost affects the agent behavior and control policy via RL parameter updating, where the TD error is appeared.

The gradient of function $Q_{\boldsymbol{\theta}}$ needed in (3.7) requires one to compute the sensitivities of the optimal value of Nonlinear Programming (NLP) (3.5). This sensitivity ought to be computed with care since the RL parameters $\boldsymbol{\theta}$ impact $Q_{\boldsymbol{\theta}}$ both directly via the MPC scheme and indirectly via the MHE scheme, by modifying the state estimation $\hat{\mathbf{x}}_k$ at the physical current time instant k that enters as an initial condition $\mathbf{x}_k = \hat{\mathbf{x}}_k$ in the MPC scheme. The gradient $\nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}$ associated to the proposed MPC/MHE scheme is given by the following total derivative:

$$\frac{dQ_{\boldsymbol{\theta}}}{d\boldsymbol{\theta}} = \frac{\partial Q_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} + \frac{\partial Q_{\boldsymbol{\theta}}}{\partial \hat{\mathbf{x}}_k} \frac{\partial \hat{\mathbf{x}}_k}{\partial \boldsymbol{\theta}} \tag{3.8}$$

Figure 3.1 shows an overview of the proposed learning-based observer/controller.

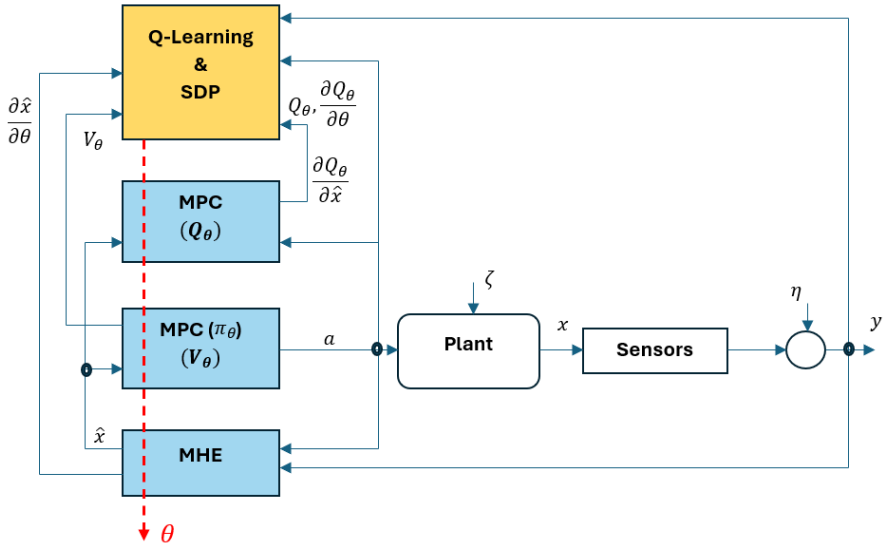


Figure 3.1: An overview of the MHE/MPC-based Q-learning. Both the MPC and MHE models are assumed to be partially observable dynamics. The state and action value functions $V_{\boldsymbol{\theta}}, Q_{\boldsymbol{\theta}}$ are approximated by (3.3) and (3.5), respectively. The action a is selected according to the policy π_θ with the possible addition of exploratory moves. The state estimation $\hat{x}$ is delivered by the MHE scheme (3.2). The SDP (3.14) is used to satisfy some requirements in the RL steps.

We detail next how to compute the above sensitivities.

3.3.2 Sensitivities of the MPC-MHE scheme

Let us define the Lagrange functions $\mathcal{L}_{\boldsymbol{\theta}}, \hat{\mathcal{L}}_{\boldsymbol{\theta}}$ associated to the MPC and MHE problems (3.2), (3.5) as follows:

$$\mathcal{L}_{\boldsymbol{\theta}} = \Phi_{\boldsymbol{\theta}} + \boldsymbol{\lambda}^{\top} G_{\boldsymbol{\theta}} + \boldsymbol{\mu}^{\top} H_{\boldsymbol{\theta}} \tag{3.9}$$

$$\hat{\mathcal{L}}_{\boldsymbol{\theta}} = \hat{\Phi}_{\boldsymbol{\theta}} + \hat{\boldsymbol{\lambda}}^{\top} \hat{G}_{\boldsymbol{\theta}} \tag{3.10}$$

where $H_{\boldsymbol{\theta}}$ gathers the inequality constraints of (3.5) and $\Phi_{\boldsymbol{\theta}}, \hat{\Phi}_{\boldsymbol{\theta}}$ are the costs of the MPC and MHE optimization problems, respectively. Variables $\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}$ are the Lagrange multipliers associated to the equality constraints $G_{\boldsymbol{\theta}}, \hat{G}_{\boldsymbol{\theta}}$ of the MPC and MHE, respectively. Variables $\boldsymbol{\mu}$ are the Lagrange multipliers associated to the inequality constraints of the MPC scheme. Let us label the primal variables as $\mathbf{p} = \{\mathbf{X}, \mathbf{U}\}$ and $\hat{\mathbf{p}} = \left\{\hat{\mathbf{X}}, \hat{\mathbf{U}}\right\}$ for the MPC and MHE, respectively. The primal-dual variables of the MPC and MHE schemes will be labeled as $\mathbf{z} = \{\mathbf{p}, \boldsymbol{\lambda}, \boldsymbol{\mu}\}$ and $\hat{\mathbf{z}} = \left\{\hat{\mathbf{p}}, \hat{\boldsymbol{\lambda}}\right\}$, respectively.

The sensitivities of the MPC scheme (3.5) required in (3.8) can be obtained by the sensitivity analysis detailed in [42] as follows:

$$\frac{\partial Q_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{L}_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{z}^{\star})}{\partial \boldsymbol{\theta}}, \quad \frac{\partial Q_{\boldsymbol{\theta}}}{\partial \mathbf{x}_k} = \frac{\partial \mathcal{L}_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{z}^{\star})}{\partial \mathbf{x}_k}, \quad \mathbf{x}_k = \hat{\mathbf{x}}_k \tag{3.11}$$

where $\mathbf{z}^{\star}$ is the primal-dual solution vector of (3.5).

The sensitivity $\frac{\partial \hat{\mathbf{x}}_k}{\partial \boldsymbol{\theta}}$ associated to the MHE scheme can be obtained via using the Implicit Function Theorem (IFT) on the Karush Kuhn Tucker (KKT) conditions underlying the parametric NLP. Assuming that Linear Independence Constraint Qualification (LICQ) and Second Order Sufficient Condition (SOSC) hold [18] at $\hat{\mathbf{z}}^{\star}$, then, the following holds:

$$\frac{\partial \hat{\mathbf{z}}^{\star}}{\partial \boldsymbol{\theta}} = -\frac{\partial R_{\boldsymbol{\theta}}}{\partial \hat{\mathbf{z}}}^{-1} \frac{\partial R_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \tag{3.12}$$

where

$$R_{\boldsymbol{\theta}} = \begin{bmatrix} \nabla_{\hat{\mathbf{p}}} \hat{\mathcal{L}}_{\boldsymbol{\theta}} \\ \hat{G}_{\boldsymbol{\theta}} \end{bmatrix} \tag{3.13}$$

are the KKT conditions associated to the MHE scheme (3.2). As $\hat{\mathbf{x}}_k$ is part of $\hat{\mathbf{z}}^{\star}$, the sensitivity of the MHE solution $\frac{\partial \hat{\mathbf{x}}_k}{\partial \theta}$ required in (3.8) can be extracted from matrix $\frac{\partial \hat{\mathbf{z}}^{\star}}{\partial \theta}$.

3.3.3 Constrained RL steps

The adjustable weighting matrices in the proposed parameterization of both MPC and MHE in (3.2), (3.3) and (3.5) are tuned using Q-learning. As a requirement, the weighting matrices $Q_E^{\boldsymbol{\theta}}$, $R_E^{\boldsymbol{\theta}}$, $A_r^{\boldsymbol{\theta}}$ must be positive semidefinite. However, the RL steps delivered by Q-learning do not necessarily respect this requirement, and we need to enforce it via constraints on the RL steps throughout the learning process. To address this requirement, we formulate a Semi-Definite Program (SDP) as a least squares optimization problem:

$$\min_{\Delta\boldsymbol{\theta}} \quad \frac{1}{2}\left\|\Delta\boldsymbol{\theta}\right\|^2 - \alpha\delta_k \nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}(\mathbf{x}_k, \mathbf{u}_k)^\top \Delta\boldsymbol{\theta} \tag{3.14a}$$

$$\text{s.t.} \quad Q_E^{\boldsymbol{\theta}}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) \geq 0 \tag{3.14b}$$

$$R_E^{\boldsymbol{\theta}}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) \geq 0 \tag{3.14c}$$

$$A_r^{\boldsymbol{\theta}}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) \geq 0 \tag{3.14d}$$

where we assume that the weighting matrices $Q_E^{\boldsymbol{\theta}}, R_E^{\boldsymbol{\theta}}, A_r^{\boldsymbol{\theta}}$ are linear functions of $\boldsymbol{\theta}$. Then, these matrices are updated in each time instant due to updating $\Delta\boldsymbol{\theta}$, which is a solution of the above SDP scheme. The proposed learning process is described in the Alg. 1.

Algorithm 1 (MPC+MHE)-Based RL

Require: $\alpha, tol > 0, \boldsymbol{\theta} = \boldsymbol{\theta}_0, \mathbf{x}_0, \mathbf{u}_0$

while Iter **do**

1. Measure output $\mathbf{y}_k$ from (3.1b) at current time k
2. Obtain $\hat{\mathbf{x}}_k, \frac{\partial \hat{\mathbf{x}}_k}{\partial \boldsymbol{\theta}}$ from (3.2) and (3.12)
3. Obtain $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{x}_k)$, $V_{\boldsymbol{\theta}}(\mathbf{x}_k)$ from (3.3)
4. Exploration: $\mathbf{u}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{x}_k) + \mathbf{d}$, $\mathbf{d} \sim \mathcal{N}\left(\mu, \sigma^2\right)$
5. Obtain $Q_{\boldsymbol{\theta}}$, $\frac{\partial Q_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ from (3.5), (3.11)
6. Assemble $\frac{dQ_{\boldsymbol{\theta}}}{d\boldsymbol{\theta}}$ from (3.8)
7. Apply $\mathbf{u}_k$ to the real plant: (3.1a)
8. Evaluate baseline $L(\mathbf{x}_k, \mathbf{u}_k)$
9. RL update:
   - Obtain $\Delta\boldsymbol{\theta}$ from (3.14)
   - $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$

$k \leftarrow k + 1$

end while

3.4 Numerical Example

In this section, we illustrate the performance of the proposed MPC/MHE-based RL scheme, which is tested on a constrained two-mass-spring-damper system shown in Figure 3.2, for which the MPC/MHE model ignores some of the dynamics.

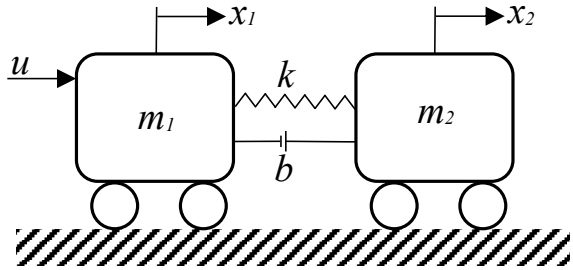


Figure 3.2: Two-Mass-Spring-Damper

The control input acts on mass 1, and the position of mass 2 is measured. Let us consider $m_1 = 0.8$ kg, $m_2 = 0.5$ kg, $k = 25\ \frac{N}{m}$, $b = 3\ \frac{Ns}{m}$ and define the plant dynamics as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{k}{m_1} & \frac{k}{m_1} & -\frac{b}{m_1} & \frac{b}{m_1} \\ \frac{k}{m_2} & -\frac{k}{m_2} & \frac{b}{m_2} & -\frac{b}{m_2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{m_1} \\ 0 \end{bmatrix} u$$

where x_1, x_2 are positions of masses 1, 2, respectively, and x_3, x_4 are corresponding velocities. Variable u is the control input applied to the first mass. In this simulation, we propose to formulate the MPC/MHE scheme based on a partially observable model. More precisely, the adopted MPC scheme is presented based on a 2-states model, capturing only the position and velocity of mass 1. The MHE scheme is based on the same model, but is fed as measurements the position of the mass 2 ($y = x_2$). Let $\mathbf{f}_{\boldsymbol{\theta}}^{\text{MPC}}$ in (3.3b) be a partially observable and inaccurate MPC model as follows:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_3 \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + A^{\text{bias}} \right) \begin{bmatrix} x_1 \\ x_3 \end{bmatrix} + \left(\begin{bmatrix} 0 \\ \frac{1}{m_1+m_2} \end{bmatrix} + B^{\text{bias}} \right) u \tag{3.15}$$

where $A^{\text{bias}}, B^{\text{bias}}$ are adjusted as model bias RL parameters $\boldsymbol{\theta}$ via Q-learning in order to tackle the inaccurate above MPC model. In this example, the following optimization problem as a parameterized MPC scheme is solved at each time

instant k:

$$\min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad \theta_c + \frac{\gamma^{N_{\text{MPC}}}}{2}\left(\mathbf{x}_{N^f}^\top M_{\boldsymbol{\theta}}^f \mathbf{x}_{N^f} + \mathbf{w}_f^\top \boldsymbol{\sigma}_{N^f}\right) + \sum_{i=k}^{k+N_{\text{MPC}}-1} \frac{\gamma^{i-k}}{2}\left((\mathbf{c}-\boldsymbol{\theta}_r)^\top M_{\boldsymbol{\theta}}(\mathbf{c}-\boldsymbol{\theta}_r) + \mathbf{w}^\top \boldsymbol{\sigma}_i\right) \tag{3.16a}$$

$$\text{s.t.} \quad \mathbf{x}_{i+1} = \mathbf{f}_{\boldsymbol{\theta}}^{\text{MPC}}(\mathbf{x}_i, \mathbf{u}_i), \tag{3.16b}$$

$$\mathbf{x}_k = \hat{\mathbf{x}}_k, \tag{3.16c}$$

$$-1 \leq \mathbf{u}_i \leq 1, \tag{3.16d}$$

$$\begin{bmatrix} 0 \\ -10 \end{bmatrix} + \underline{\boldsymbol{\theta}} - \boldsymbol{\sigma}_i \leq \mathbf{x}_i \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} + \bar{\boldsymbol{\theta}} + \boldsymbol{\sigma}_i \tag{3.16e}$$

$$\boldsymbol{\sigma}_{k,\ldots,k+N_{\text{MPC}}} \geq 0 \tag{3.16f}$$

where $N^f = k + N_{\text{MPC}}$, $\mathbf{c} = [\mathbf{x}_k, \mathbf{u}_k]^\top$ and the MPC parameters subject to the RL scheme are:

$$\boldsymbol{\theta} = \left(\theta_c, M_{\boldsymbol{\theta}}^f, \boldsymbol{\theta}_r, M_{\boldsymbol{\theta}}, A^{\text{bias}}, B^{\text{bias}}, \underline{\boldsymbol{\theta}}, \bar{\boldsymbol{\theta}}\right) \tag{3.17}$$

The positive semidefinite weighting matrices ($M_{\boldsymbol{\theta}}^f$, $M_{\boldsymbol{\theta}}$,$Q_E^{\boldsymbol{\theta}}$, $R_E^{\boldsymbol{\theta}}$, $A_r^{\boldsymbol{\theta}}$) in both MPC and MHE schemes are adjusted using the constrained RL steps in (3.14). One can choose a baseline stage cost used in the RL scheme (3.7) as:

$$L(y_k, u_k) = l(y_k, u_k) + \mathbf{w}^\top \max(0, \mathbf{h}(y_k)) \tag{3.18}$$

where $l(y_k, u_k)$ is adopted as a quadratic function of the output and action deviations from their desired values. The second term in the above baseline is considered to penalize the constraint violations, where $\mathbf{h} \geq 0$ is pure inequality vector of constraints on the states and $\mathbf{w}^\top = [10, 10]$. Note that different step sizes α were used for the different parameters based on the problem scaling. The desired values for the MPC model states (position and velocity of the first body) x_1, x_3 are chosen at $[0, 0]^T$, respectively. We apply a process noise $\zeta \sim \mathcal{N}(\mu = 0, (\sigma = 0.02)^2)$ on the velocity of the second body and a measurement noise $\eta \sim \mathcal{N}(\mu = 0, (\sigma = 0.05)^2)$ on the position of second body. In this simulation we choose $\gamma = 0.9$, and $N_{\text{MPC}} = N_{\text{MHE}} = 8$.

As this simulation is considered as a POMDP and uncertain scenario and there are both process and measurement Gaussian noises, the violations are observed on the states in Figure 3.6. We demonstrate that the proposed MPC/MHE-based RL can attenuate these violations and increase the control performance even if the controller/observer models are unmodeled and partially observable.

In this example, we consider three different scenarios.

1) *Without learning* (MPC+MHE): In the first scenario, there are large violations of the position and velocity constraints affecting the closed-loop performance (large cost J) shown in Figure 3.6 and Figure 3.7.

2) *MPC-based RL learning* (MPC-RL+MHE): In the second scenario, the learning is only performed on the MPC scheme. This MPC-based RL reduces the violations and increases the closed-loop performance shown in Figure 3.6 by reducing the discounted sum of the RL stage cost J over a receding horizon. There is also a decrease of the baseline cost shown in Figure 3.8 after starting the MPC learning while the MHE learning is not still activated.

3) *MPC/MHE-based RL learning* (MPC-RL+MHE-RL): Finally in the third scenario, the performance is improved after allowing the MHE to be adjusted using the Q-learning algorithm and there is a solid decrease in the TD-error and baseline cost and an increasing closed-loop performance (decrease of J). The evolution of MHE parameters are illustrated in Figure 3.9 and Figure 3.10. The evolution of MPC parameters are depicted in Figures 3.3, 3.4, and 3.5.

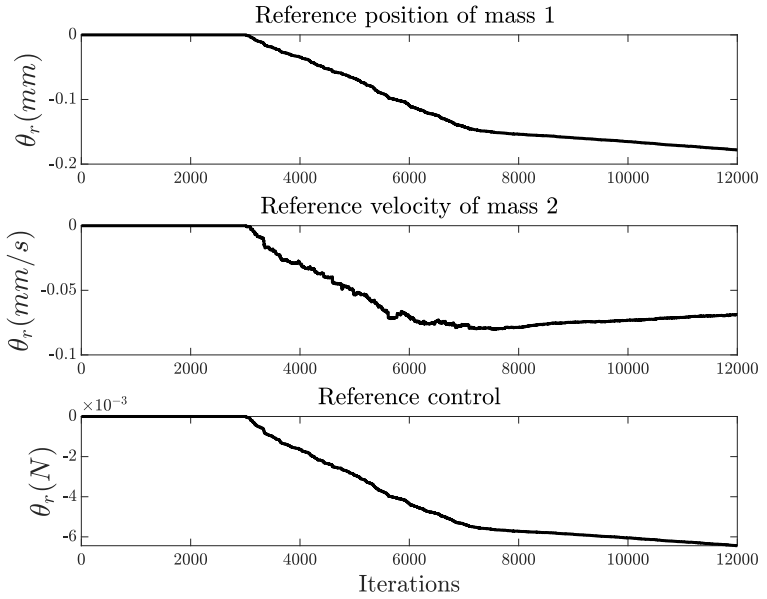


Figure 3.3: MPC adjustment: Reference signals

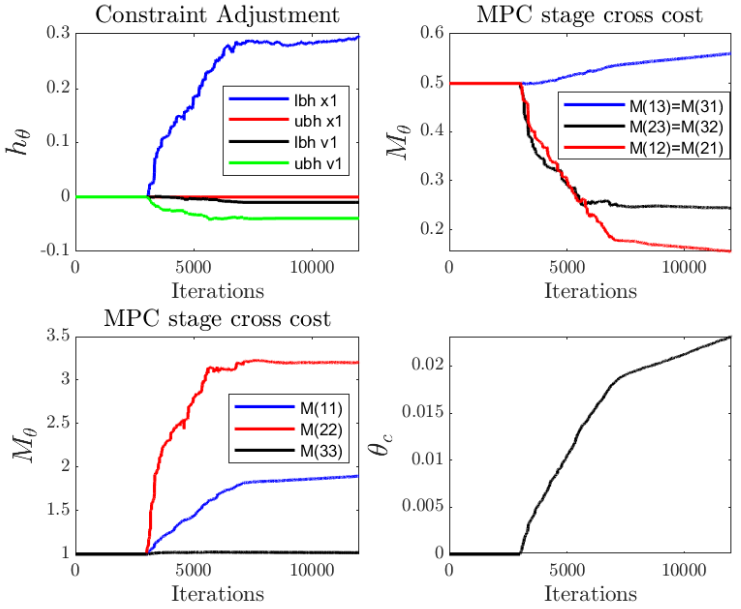


Figure 3.4: MPC adjustment: Constraints and stage cost

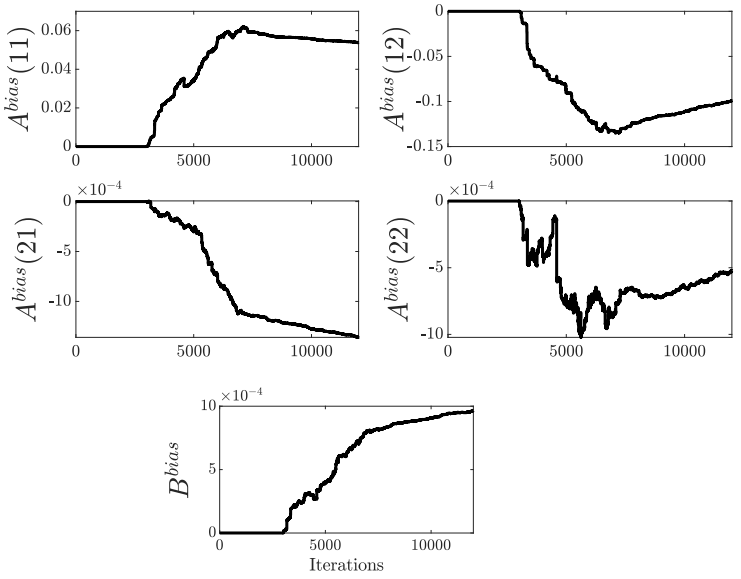


Figure 3.5: MPC adjustment: Model bias

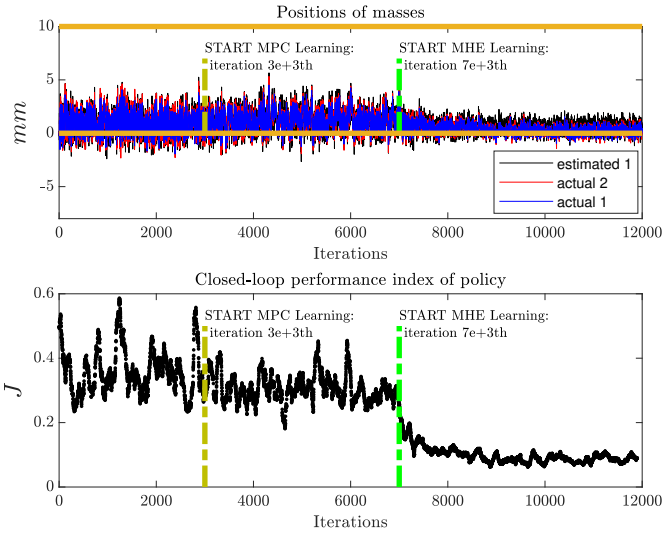


Figure 3.6: Positions of masses and closed-loop performance. The brown lines are shown as lower bound (0 mm) and upper bound (10 mm) constraints on the positions. Position references are (0 mm).

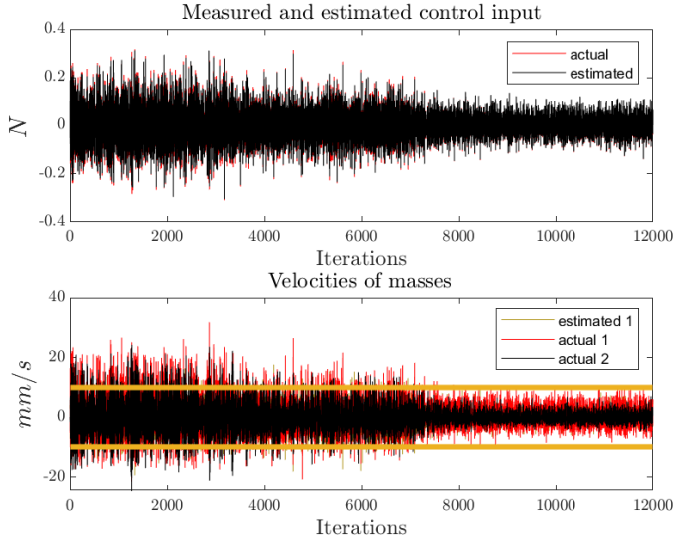


Figure 3.7: Control input and velocities of masses. The brown lines are shown as lower bound (-10 mm/s) and upper bound (10 mm/s) constraints on the velocities.

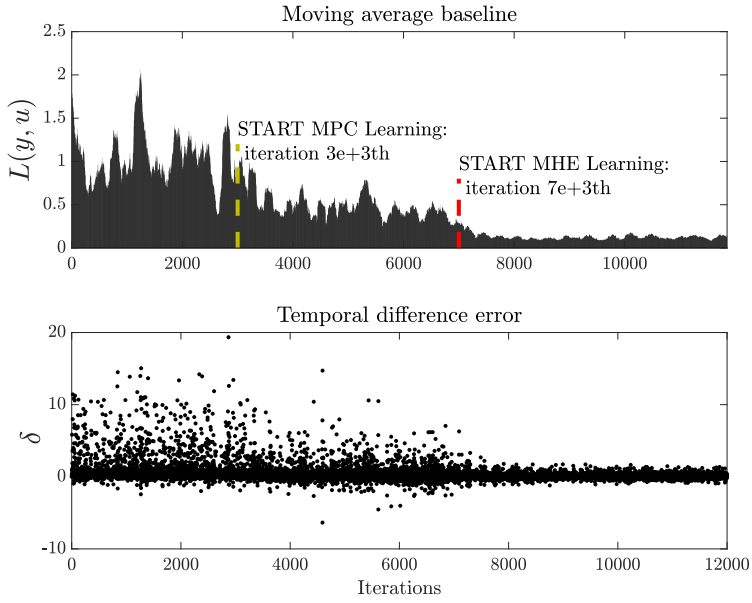


Figure 3.8: RL performance: Baseline cost (RL stage cost) and TD error

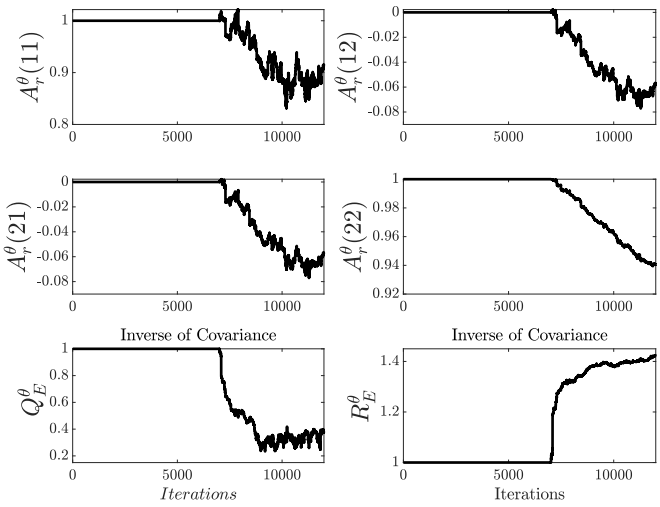


Figure 3.9: MHE adjustment: Arrival matrix and penalizing weights

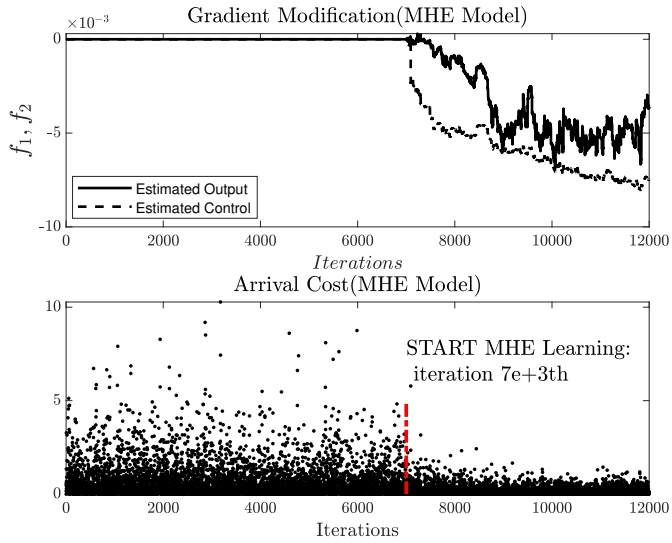


Figure 3.10: Evolution of the arrival cost and gradients in the MHE scheme.

3.5 Conclusion

This chapter proposed the combination of MPC-based Reinforcement Learning with an MHE scheme to tackle POMDPs. The introduction of an MHE scheme allows to deploy MPC-based Reinforcement Learning without a full state measurement, and without necessarily holding a correct representation of the system state in the MPC and MHE models. Furthermore, we propose to tune the MHE and MPC schemes jointly, focusing directly on the closed-loop performance, as opposed to using indirect criteria such as decreasing the MHE output error. We detail the application of Q-learning to this approach, and test it in a simulated spring mass example operating under constraints, where only a part of the real system dynamics are modelled in the MPC and MHE schemes. We show that the method manages to tune the MHE and MPC scheme to reduce the constraints violations and improve the closed-loop performance. Future work will propose an stability and feasibility analysis on the proposed MPC/MHE-based RL scheme.

Chapter 4

Learning-based State Estimation and Control using MHE and MPC Schemes with Imperfect Models

- H. N. Esfahani, A. B. Kordabad, W. Cai, and S. Gros, "Learning-based state estimation and control using mhe and mpc schemes with imperfect models," European Journal of Control, p. 100880, 2023.

This chapter presents a reinforcement learning-based observer/controller using Moving Horizon Estimation (MHE) and Model Predictive Control (MPC) schemes where the models used in the MHE-MPC cannot accurately capture the dynamics of the real system. We first show how an MHE cost modification can improve the performance of the MHE scheme such that a true state estimation is delivered even if the underlying MHE model is imperfect. A compatible Deterministic Policy Gradient (DPG) algorithm is then proposed to directly tune the parameters of both the estimator (MHE) and controller (MPC) in order to achieve the best closed-loop performance based on inaccurate MHE-MPC models. To demonstrate the effectiveness of the proposed learning-based estimator-controller, three numerical examples are illustrated.

4.1 Introduction

In the context of model-based control approaches, Model Predictive Control (MPC) is a well-known control scheme, which uses a dynamic model to predict the future behavior of the real system over a finite time horizon. At each time instant, MPC calculates the input and corresponding state sequence minimizing a given cost function while satisfying constraints over a given prediction horizon [9]. In many real applications, a state estimator (observer) is needed to provide an estimation of the current system states to the MPC scheme. In this chapter, we adopt a Moving Horizon Estimation (MHE) scheme as a state observer, which is a simple choice in combination with an MPC scheme. MHE is an optimization-based state observer that works on a horizon window covering a limited history of past measurements [33].

Accurate models of dynamical systems are often difficult to obtain due to uncertainties and unknown dynamics. It is also worth noting that even if an accurate model is available, it may be in general too complex to be used in MHE and MPC schemes. However, if the model is imperfect, the inaccuracies can significantly degrade the performance of the MHE-MPC scheme. To cope with this problem, data-driven methods can be used in order to either improve the MPC and MHE models [43, 44, 45, 46] or modify the MHE/MPC cost functions [8, 21].

The data-driven MPC/MHE schemes mentioned above often incorporate Machine Learning (ML)-based techniques such as Reinforcement Learning (RL) and Gaussian Process (GP). RL is a powerful ML method for Markov Decision Processes (MDPs), which seeks to improve the closed-loop performance of the control policy deployed on the MDPs as observations are collected [11]. Most RL methods use a Deep Neural Network (DNN) to approximate either the optimal policy underlying the MDP directly or the action-value function from which the optimal policy can be indirectly extracted [47].

The idea of using an MPC scheme as a value function/policy approximator in the RL context was proposed in [8, 48]. Specifically, the motivation was to replace the DNN-based approximators with the MPC schemes such that some challenging issues in the context of RL, including stability guarantee and safety were addressed. In an MPC-based RL, it was established that an MPC scheme can generate jointly the optimal (action-) value function and optimal policy underlying an MDP even if the MPC model does not capture the real system dynamics accurately. As a data-driven MPC, the MPC-based RL framework has shown promising results for different applications [22, 40, 23, 49, 24]. Inspired by the researches mentioned above in the context of MPC-based RL, in the present chapter, we will use an MHE-MPC scheme as a policy approximator for a deterministic policy gradient algorithm.

In some real-world control applications, the measurements available from the real system at a given time instant do not constitute a Markov state. In the context of RL, these systems are then formulated as Partially Observable MDPs (POMDPs) [50, 28]. To tackle a POMDP, one solution is to formulate a belief MDP where the information about the current state is described as a probability distribution over the state space a.k.a belief state. Hence, POMDPs can be regarded as traditional MDPs using the concept of belief states as complete observable states [51].

Most previous works in the context of POMDPs rely on training a Neural Network (NN) or a Recurrent Neural Network (RNN) to summarise past observations and learn a policy based on DNN-based approximators [32, 52, 53]. An NN-based framework (posterior distributions over states) was proposed in [26, 27] in order to estimate a belief state based on historical information. These NN-based algorithms are formulated as completely model-free approaches. Most recently, as a combined model-based/data-driven technique for dealing with POMDPs, a Q-learning method based on MHE-MPC with inaccurate models was developed in [21]. In this research, the authors proposed to integrate MHE and MPC to treat the hidden Markovian state evolution. More specifically, a structured solution by using MHE as a model-based approach was proposed to build a state from the measurement history.

In this chapter, we seek to improve the performance of MHE-MPC as a combined observer/controller based on an inaccurate model. Assuming the real system is fully observable and the MHE model has a correct state structure, we show that both the arrival cost and the stage cost of the MHE scheme can be modified such that a perfect state estimation is delivered even if the underlying model is imperfect. However, the proposed method can arguably perform well on an incomplete model structure (partially observable), which is demonstrated by a numerical example. To tackle the performance degradation of the MHE scheme due to the

use of an imperfect model, we propose to modify the MHE cost function rather than adapting the MHE model. An NN-based approximator is proposed to deliver the modified MHE cost. To achieve the best closed-loop performance even if the underlying MHE-MPC model is imperfect, we then propose to jointly tune the MHE-MPC parameters using a compatible Deterministic Policy Gradient (DPG) reinforcement learning algorithm.

The chapter is structured as follows. The central theorem upon the cost modification of the MHE scheme using an imperfect model is detailed in Section 4.2. Section 4.3 describes a tractable approach for the MHE cost modification. Section 4.4 is dedicated to the parameterization method upon the MHE cost and the MPC scheme in order to formulate an adjustable and learning-based MHE-MPC scheme. To achieve the best closed-loop performance for an MHE-MPC scheme, a policy gradient-based RL algorithm is detailed in Section 4.5 to adjust the parameterized MHE cost function and learn a policy captured from a parameterized MHE-MPC scheme. Section 4.6 provides three numerical examples: 1) a linear system with model mismatch 2) a POMDP test case in which a smart building is described as an imperfect dynamical model and its climate is controlled by the proposed approach, and finally 3) a Continuous Stirred Tank Reactor (CSTR) as a nonlinear system is investigated.

Notation. a is a scalar while $\mathbf{a}$ is a vector. For n vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ we define $\mathrm{col}(\mathbf{x}_1, \ldots, \mathbf{x}_n) := [\mathbf{x}_1^\top, \ldots, \mathbf{x}_n^\top]^\top$. $\mathbb{R}$ is the set of real numbers and $\mathbb{I}$ is the set of integers.

4.2 Modified MHE with Imperfect Model

In this section, we first consider an ideal stochastic MHE, which is formulated as a Full Information Estimation (FIE) problem. The FIE problems are fundamentally formulated based on an optimization problem in which the entire history of the measurements is used at each time instant [54]. We then formulate an MHE scheme using an imperfect model, and show that the stage cost function can be modified so that the MHE delivers the same estimation as an ideal MHE. At the end of this section, as opposed to the FIE version of the MHE, we will formulate a finite version of the modified MHE scheme in order to make it computationally tractable.

4.2.1 Stochastic MHE Scheme

To formulate an ideal stochastic MHE scheme, we consider discrete dynamical systems evolving on a continuous state space over $\mathbb{R}^n$, with stochastic states $\mathbf{s}_k \in \mathcal{S} \subseteq \mathbb{R}^n$, where k denotes the time index. Let ϱ_k be a probability measure associ-

ated with the stochastic states as follows:

$$\mathbf{s}_k \sim \varrho_k(.) \tag{4.1}$$

We will consider a measure space for $\mathbf{s}_k$, which is equipped with the Lebesgue measure as a reference measure, and the set of Lebesgue-measurable sets as σ-algebra. Let us define stochastic dynamics as a conditional probability density as follows:

$$\zeta\left[\mathbf{s}_{k+1} \,|\, \mathbf{s}_k, \mathbf{a}_k\right] \tag{4.2}$$

where $\mathbf{s}_k, \mathbf{a}_k \in \mathcal{A} \subseteq \mathbb{R}^m$ and $\mathbf{s}_{k+1}$ are the current state-input pair and subsequent state, respectively, and $\mathcal{A}$ is the set of inputs available for the system.

Let us define a transition operator $\mathcal{T}_{\mathbf{a}_k} : \mathcal{M} \times \mathcal{A} \to \mathcal{M}$ as the map from a probability measure ϱ_k to its successor ϱ_{k+1} under input $\mathbf{a}_k$, and $\mathcal{M}$ is the set of probability measures over $\mathcal{S}$ such that the sequence of probability measures $\varrho_k \in \mathcal{M}, k = 0, \cdots, \infty$. We then define the Law of Total Probability (LTP) with stochastic dynamics (4.2) as follows:

$$\varrho_{k+1}(.) = \mathcal{T}_{\mathbf{a}_k} \varrho_k(.) = \int_{\mathcal{S}} \zeta\left[. \,|\, \mathbf{s}_k, \mathbf{a}_k\right] \varrho_k(\mathrm{d}\mathbf{s}_k) \tag{4.3}$$

Let us label $\mathbb{E}_{\mathbf{s}_k \sim \varrho_k}[.]$ the expected value operator with respect to probability measure $\varrho_k \in \mathcal{M}$. To formulate a stochastic MHE scheme a.k.a Full Information Estimation (FIE), its cost function can be derived using a functional stage cost where this functional is either an expectation or the Maximum A Posterior (MAP) [55]. In the present chapter, we use an expectation to formulate a stochastic MHE under some conditions detailed in the remainder of the chapter. We then define a value functional associated with the stochastic MHE scheme as follows:

$$V\left[\varrho_k, \mathbf{o}_k\right] := \sum_{i=-\infty}^{k} \gamma^{k-i} \mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\Big[L\left(\mathbf{s}_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right)\Big], \tag{4.4}$$

where $\gamma \in (0, 1]$ is a discount factor, $\mathbf{y}_i \in \mathcal{Y} \subseteq \mathcal{S}$, $\mathbf{o}_k = \mathrm{col}\left\{\mathbf{a}_{\ldots,k-1}, \mathbf{y}_{\ldots,k}\right\} \in \mathcal{O}$ is the available history of measurements up to time k, $L : \mathcal{S} \times \mathcal{A} \times \mathcal{Y} \to \mathbb{R}$ is a fitting function. It is worth noting that the discounting above ensures the existence of the estimation problem on an infinite horizon for an MDP. However, the basic Theorem on the cost modification structure detailed in the remainder of this section also holds for the undiscounted setting, e.g., $\gamma = 1$. We assume that the forward transition operator $\mathcal{T}_{\mathbf{a}_k}$ has a backward transition operator $\mathcal{T}^{-1}_{\mathbf{a}_{i-1}}$ such that $\varrho_{i-1} = \mathcal{T}^{-1}_{\mathbf{a}_{i-1}} \varrho_i, \forall i \in \mathbb{I}_{\leq k}$. Note that we use the backward transition operator since an MHE scheme at the current time k is formulated based on past information.

Then the aim of the stochastic MHE scheme is to find the best probability measure $\rho^\star$ as a function of $\mathbf{o}_k$ that minimizes $V\left[\varrho_k, \mathbf{o}_k\right]$. More specifically:

$$\rho_k^\star(\mathbf{o}_k) \in \arg\min_{\varrho_k} V\left[\varrho_k, \mathbf{o}_k\right] \tag{4.5}$$

However, we only have access to an imperfect model of (4.2) (typically deterministic). To cope with this issue, in the remainder of this section, we first develop the central theorem on the modification of the stochastic MHE schemes with imperfect models. We then propose a more practical formulation of the modified stochastic MHE in which a deterministic state estimation can be delivered.

4.2.2 Modification of the MHE Cost Function

The main contribution of this chapter is described by the next theorem, where an MHE scheme equipped with a modified stage cost function is proposed to tackle the performance degradation due to an imperfect MHE model. It will be shown that one can, under some assumptions, find a modified MHE cost such that a probability measure equal to (4.5) is delivered even if the underlying model is inaccurate. In this chapter, we define a cost functional $\Phi : \mathcal{M} \times \mathcal{A} \times \mathcal{Y} \to \mathbb{R}$ such that this cost functional linearly depends on the stage cost function as follows:

$$\Phi\left[\varrho_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right] = \mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\left[L\left(\mathbf{s}_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right)\right] \tag{4.6}$$

Note that the above equality is valid under some conditions detailed in the next section. Then, the value functional (4.4) can be rewritten as follows:

$$V[\varrho_k, \mathbf{o}_k] = \sum_{i=-\infty}^{k} \gamma^{k-i} \Phi\left[\varrho_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right], \tag{4.7}$$

Let $\zeta_b\left[\mathbf{s}_{k-1} \,|\, \mathbf{s}_k, \mathbf{a}_{k-1}\right]$ and $\hat{\zeta}_b\left[\hat{\mathbf{s}}_{k-1} \,|\, \hat{\mathbf{s}}_k, \mathbf{a}_{k-1}\right]$ be a backward model of (4.2) and an imperfect model of ζ_b, respectively. We then label $\hat{\mathcal{T}}_{\mathbf{a}_{i-1}}^{-1}$ the corresponding imperfect backward transition operator. Now we propose to modify the MHE cost Φ detailed in the next theorem in order to cope with the performance degradation of an MHE scheme where the MHE model is imperfect. Hence, the corresponding value functional for a modified stochastic MHE scheme is formulated as follows:

$$\hat{V}[\hat{\varrho}_k, \mathbf{o}_k] := \sum_{i=-\infty}^{k} \gamma^{k-i} \hat{\Phi}\left[\hat{\varrho}_i, \mathbf{o}_i\right] \tag{4.8}$$

where $\hat{\varrho}_{i-1} = \hat{\mathcal{T}}_{\mathbf{a}_{i-1}}^{-1} \hat{\varrho}_i, \ \forall i \in \mathbb{I}_{\leq k}$ and $\hat{\Phi} : \mathcal{M} \times \mathcal{O} \to \mathbb{R}$ is an MHE cost functional based on the measurement history for the model. Note that the arguments of the stage cost Φ in (4.7) include $\mathbf{a}_{i-1}, \mathbf{y}_i$ while they are shown as a measurement

history $\mathbf{o}_i$ for the arguments of the modified stage cost $\hat{\Phi}$ in (4.8). More precisely, the modified stage cost will be a function of the measurement history at each time step i, which is detailed in the proof of the next theorem.

Analogous to the previous, we define the best probability measure resulting from the imperfect model, as follows:

$$\hat{\rho}_k^\star(\mathbf{o}_k) \in \arg\min_{\hat{\varrho}_k} \hat{V}\left[\hat{\varrho}_k, \mathbf{o}_k\right] \tag{4.9}$$

We then aim to propose $\hat{\Phi}$ such that $\hat{\rho}_k^\star(\mathbf{o}_k) = \rho_k^\star(\mathbf{o}_k)$. In the following, we make mild assumptions on the boundedness of the discounted value function.

Assumption 1. *There exists a non-empty set of probability measures $\mathcal{M}_0 \subseteq \mathcal{M}$, including $\hat{\varrho}_k^\star$, such that for all $\hat{\varrho}_k \in \mathcal{M}_0$ and for all $\gamma \in (0,1]$ it holds that*

$$\left|\gamma^N V\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right]\right| < \infty, \quad \forall N \in \mathbb{I}_{\geq 0} \tag{4.10}$$

where $\hat{\varrho}_{k-N} = \hat{\mathcal{T}}_{\mathbf{a}_{k-N}}^{-1} \ldots \hat{\mathcal{T}}_{\mathbf{a}_{k-1}}^{-1} \hat{\varrho}_k$.

Assumption 2. *For a discount factor $\gamma \in (0,1]$ and $\hat{\varrho}_{k-N} \in \mathcal{M}_0, \forall N \in \mathbb{I}_{\geq 0}$:*

$$\lim_{N\to\infty} \gamma^N V\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right] = 0 \tag{4.11}$$

Theorem 1. *Under Assumptions 1, 2, there exists a modified stage cost functional $\hat{\Phi} : \mathcal{M} \times \mathcal{O} \to \mathbb{R}$ such that the following equalities hold for all $\hat{\varrho}_k \in \mathcal{M}_0$ and all $\mathbf{o}_k \in \mathcal{O}$:*

$$\hat{V}[\hat{\varrho}_k, \mathbf{o}_k] = V[\hat{\varrho}_k, \mathbf{o}_k], \qquad \hat{\varrho}_k^\star(\mathbf{o}_k) = \varrho_k^\star(\mathbf{o}_k) \tag{4.12}$$

Proof. Let us define the modified stage cost functional $\hat{\Phi}$ as follows:

$$\hat{\Phi}\left[\hat{\varrho}_i, \mathbf{o}_i\right] = V\left[\hat{\varrho}_i, \mathbf{o}_i\right] - \gamma V\left[\hat{\mathcal{T}}_{\mathbf{a}_{i-1}}^{-1} \hat{\varrho}_i, \mathbf{o}_{i-1}\right] \tag{4.13}$$

By substituting the modified stage cost (4.13) in (4.8), the value functional then

becomes a telescoping sum as follows:

$$\begin{aligned}\hat{V}\left[\hat{\varrho}_k, \mathbf{o}_k\right] &= \sum_{i=-\infty}^{k} \gamma^{k-i} \hat{\Phi}\left[\hat{\varrho}_i, \mathbf{o}_i\right] \\ &= \sum_{i=-\infty}^{k} \gamma^{k-i} \left(V\left[\hat{\varrho}_i, \mathbf{o}_i\right] - \gamma V\left[\hat{\varrho}_{i-1}, \mathbf{o}_{i-1}\right]\right) \\ &= V\left[\hat{\varrho}_k, \mathbf{o}_k\right] - \gamma V\left[\hat{\varrho}_{k-1}, \mathbf{o}_{k-1}\right] + \\ &\quad \gamma V\left[\hat{\varrho}_{k-1}, \mathbf{o}_{k-1}\right] - \gamma^2 V\left[\hat{\varrho}_{k-2}, \mathbf{o}_{k-2}\right] + \\ &\quad \gamma^2 V\left[\hat{\varrho}_{k-2}, \mathbf{o}_{k-2}\right] - \ldots - \lim_{N\to\infty} \gamma^N V\left[\hat{\varrho}_{-N}, \mathbf{o}_{-N}\right] \\ &= V\left[\hat{\varrho}_k, \mathbf{o}_k\right] - \lim_{N\to\infty} \gamma^N V\left[\hat{\varrho}_{-N}, \mathbf{o}_{-N}\right] \end{aligned} \tag{4.14}$$

for all $\hat{\varrho}_k \in \mathcal{M}_0$. Note that under Assumption 1 all terms in (4.14) are bounded and the following equality holds:

$$\hat{V}\left[\hat{\varrho}_k, \mathbf{o}_k\right] = V\left[\hat{\varrho}_k, \mathbf{o}_k\right] \tag{4.15}$$

and under Assumption 2,

$$\arg\min_{\hat{\varrho}_k} \hat{V}\left[\hat{\varrho}_k, \mathbf{o}_k\right] = \arg\min_{\hat{\varrho}_k} V\left[\hat{\varrho}_k, \mathbf{o}_k\right] \tag{4.16}$$

implies $\hat{\varrho}_k^\star(\mathbf{o}_k) = \varrho_k^\star(\mathbf{o}_k)$ since $\hat{\varrho}_k^\star \in \mathcal{M}_0$. □

It is worth noting that the modified stage cost function (4.13) proposed as a cost modification is constructed based on a full history of the measurements. Hence, this fundamental observation can impact on the practical implementation of the modified cost. More specifically, the central Theorem 1 aims to show that there exists such a modification and to understand its structure. However, the proposed modification structure above is not tractable in terms of implementation since it is too complex to compute the modified stage cost (4.13) and apply it to the modified MHE scheme directly. To tackle this problem, we will provide a finite H-step structure of the modified stage cost in the next section. Finally, we will propose to construct an approximate structure of the modified stage cost using a Neural Network (NN) and adopt a reinforcement learning algorithm to learn the parameters of the NN in practice, which is detailed in the section 4.4.

Although Theorem 1 shows that the modified stochastic MHE scheme with the corresponding value functional (4.8) can deliver a correct estimation of the probability measure using an imperfect model, this infinite-horizon model-based fitting problem requires an infinite amount of data, which makes this full information observer

unsuitable in practice. To cope with this problem, we propose a more practical formulation detailed by the next theorem, which provides a finite-horizon stochastic MHE problem so that it delivers the same optimal density and value functional as (4.8).

It is worth mentioning that the proposed modified state cost (4.13) has been constructed based on the value functionals, and then Assumption 1 ensures that all intermediate terms (value functionals) appeared in the telescoping sum (4.14) cancel out. However, there are infinitely many intermediate terms in the telescoping sum (4.14) that must be bounded while Assumption 1 may not be satisfied for a situation with an arbitrarily large N, e.g., let us consider the case $\gamma = 1$, which then imposes the condition $\lim_{N\to\infty} V\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right] = 0$. Hence, the additional Assumption 2 is needed to establish the Theorem 1. To address this issue, one can consider a milder assumption with a specific horizon window N to be used in a finite-horizon MHE problem. We then provide the following assumption and develop the corresponding theorem.

Assumption 3. *There exists a non-empty set of probability measures $\mathcal{M}_1 \subseteq \mathcal{M}$, including $\hat{\varrho}_k^\star$, such that for all $\hat{\varrho}_k \in \mathcal{M}_1$ and for all $\gamma \in (0, 1]$ it holds that*

$$\left|\gamma^{N_0} V\left[\hat{\varrho}_{k-N_0}, \mathbf{o}_{k-N_0}\right]\right| < \infty, \quad 0 \leq N_0 \leq N \tag{4.17}$$

where $\hat{\varrho}_{k-N_0} = \hat{\mathcal{T}}^{-1}_{\mathbf{a}_{k-N_0}} \ldots \hat{\mathcal{T}}^{-1}_{\mathbf{a}_{k-1}} \hat{\varrho}_k$ and $N < \infty$ is labeled the horizon window.

Note that this assumption is weaker than Assumption 1, indeed we have $\mathcal{M}_0 \subseteq \mathcal{M}_1$.

Theorem 2. *Consider the MHE scheme with a horizon window of N steps at the current time k :*

$$\begin{aligned} \hat{V}^N[\hat{\varrho}_k, \mathbf{o}_k] :=& \gamma^N \ell[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}] \\ &+ \sum_{i=k-N+1}^{k} \gamma^{k-i} \hat{\Phi}\left[\hat{\varrho}_i, \mathbf{o}_i\right], \end{aligned} \tag{4.18a}$$

$$\hat{\rho}_k^{\star,N}(\mathbf{o}_k) \in \arg\min_{\hat{\varrho}_k} \hat{V}^N\left[\hat{\varrho}_k, \mathbf{o}_k\right] \tag{4.18b}$$

where $\ell : \mathcal{M} \times \mathcal{O} \to \mathbb{R}$ reads as an arrival cost functional. Then, under Assumption 3, the following equalities hold for all $\hat{\varrho}_k \in \mathcal{M}_1$ and all $\mathbf{o}_k \in \mathcal{O}$:

$$\hat{V}^N\left[\hat{\varrho}_k, \mathbf{o}_k\right] = V\left[\hat{\varrho}_k, \mathbf{o}_k\right], \quad \hat{\varrho}_k^{\star,N}(\mathbf{o}_k) = \varrho_k^\star(\mathbf{o}_k) \tag{4.19}$$

Proof. Let us define the modified stage cost $\hat{\Phi}$ as (4.13) and arrival cost ℓ as follows:

$$\ell\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right] = V\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right] \tag{4.20}$$

By substituting the modified stage cost (4.13) and the modified arrival cost (4.20) in (4.18a), the value functional then becomes a telescoping sum as follows:

$$\begin{aligned}
\hat{V}^N\left[\hat{\varrho}_k, \mathbf{o}_k\right] =& \gamma^N V\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right] + \sum_{i=k-N+1}^{k} \gamma^{k-i}\left(V\left[\hat{\varrho}_i, \mathbf{o}_i\right] - \gamma V\left[\hat{\varrho}_{i-1}, \mathbf{o}_{i-1}\right]\right) \\
=& \gamma^N V\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right] + V\left[\hat{\varrho}_k, \mathbf{o}_k\right] - \gamma V\left[\hat{\varrho}_{k-1}, \mathbf{o}_{k-1}\right] \\
&+ \gamma V\left[\hat{\varrho}_{k-1}, \mathbf{o}_{k-1}\right] - \gamma^2 V\left[\hat{\varrho}_{k-2}, \mathbf{o}_{k-2}\right] + \ldots \\
&+ \gamma^{N-1} V\left[\hat{\varrho}_{k-N+1}, \mathbf{o}_{k-N+1}\right] - \gamma^N V\left[\hat{\varrho}_{k-N}, \mathbf{o}_{k-N}\right] \\
=& V\left[\hat{\varrho}_k, \mathbf{o}_k\right]
\end{aligned} \tag{4.21}$$

for all $\hat{\varrho}_k \in \mathcal{M}_1$, and

$$\arg\min_{\hat{\varrho}_k} \hat{V}^N\left[\hat{\varrho}_k, \mathbf{o}_k\right] = \arg\min_{\hat{\varrho}_k} V\left[\hat{\varrho}_k, \mathbf{o}_k\right] \tag{4.22}$$

delivers $\hat{\varrho}_k^{\star,N}(\mathbf{o}_k) = \varrho_k^\star(\mathbf{o}_k)$, since $\hat{\varrho}_k^\star \in \mathcal{M}_1$. Then it delivers (4.19). □

As an observation in the proposed finite-horizon MHE scheme (4.18a), the modified stage cost still depends on the complete measurement history despite using an arrival cost. Therefore, a practical modification of the stage cost will be detailed in the next section.

4.3 Tractable Method for the MHE Cost Modification

Although Theorem 2 proposes the modified finite-horizon stochastic MHE as a more practical scheme than an infinite problem, there are still two implementation issues to address: 1) implementing a stage cost functional (4.13) is not tractable in practice because it is constructed based on time-varying value functionals in which the current distribution function $\hat{\varrho}_k$ as given probability measure at the current time k is difficult to model and calculate exactly. Then, it is reasonable to consider a function version of the cost functional in the modified MHE scheme. 2) implementing a modified stage cost based on the full measurement history is not tractable. In the rest of this section, we discuss the solutions to tackle these problems.

4.3.1 Modified Stage Cost Function

To construct a practical cost modification based on the above results, one can consider a deterministic state estimation at the physical time k such that the modified cost is then constructed based on a value function instead of a value functional. Although this choice makes the implementation more practical, the estimation of a single state rather than a probability measure will sacrifice the MHE capability in order to explicitly describe the state estimation uncertainty. More specifically, we replace a belief state with a unique state such that the MHE solution cannot incorporate any information upon the uncertainty level of the current state.

In order to form an MHE scheme with a deterministic estimation of the state at time k, the proposed structure entails significant characteristics established by the next propositions. In the next Propositions 1,2, we first show that the backward transition operator $\mathcal{T}^{-1}$ is a linear transformation and the value functional $V[\varrho_i, \mathbf{o}_i]$ is linear in the probability measure.

Proposition 1. *The inverse of a linear operator $\mathcal{T}$ is a linear backward transition operator $\mathcal{T}^{-1}$ such that:*

$$\mathcal{T}^{-1}(\varrho + \bar{\varrho}) = \mathcal{T}^{-1}\varrho + \mathcal{T}^{-1}\bar{\varrho} \tag{4.23a}$$

$$\mathcal{T}^{-1}(\alpha\varrho) = \alpha\mathcal{T}^{-1}\varrho \tag{4.23b}$$

where the probability measures $\varrho, \bar{\varrho} \in \mathcal{M}$ and $\alpha \in \mathbb{C}$.

Proof.

$$\begin{aligned} &\mathcal{T}^{-1}(\varrho + \bar{\varrho}) = \\ &\mathcal{T}^{-1}\left(\mathcal{T}\left(\mathcal{T}^{-1}\varrho\right) + \mathcal{T}\left(\mathcal{T}^{-1}\bar{\varrho}\right)\right) \\ &\mathcal{T}^{-1}\left(\mathcal{T}\left(\mathcal{T}^{-1}\varrho + \mathcal{T}^{-1}\bar{\varrho}\right)\right) = \mathcal{T}^{-1}\varrho + \mathcal{T}^{-1}\bar{\varrho} \end{aligned} \tag{4.24}$$

and

$$\begin{aligned} &\mathcal{T}^{-1}(\alpha\varrho) = \\ &\mathcal{T}^{-1}\left(\alpha\mathcal{T}\left(\mathcal{T}^{-1}\varrho\right)\right) \\ &\mathcal{T}^{-1}\left(\mathcal{T}\left(\alpha\mathcal{T}^{-1}\varrho\right)\right) = \alpha\mathcal{T}^{-1}\varrho \end{aligned} \tag{4.25}$$

Then, the backward operator $\mathcal{T}^{-1}$ fulfills the requirements of a linear transformation. □

Proposition 2. *The value functional $V[\varrho_i, \mathbf{o}_i]$ is linear in the probability measure.*

Proof. According to (4.4), the value functional $V\left[\varrho_i, \mathbf{o}_i\right]$ at time step i is defined as follows:

$$V\left[\varrho_i, \mathbf{o}_i\right] = \sum_{j=-\infty}^{i} \gamma^{i-j} \mathbb{E}_{\mathbf{s}_j \sim \varrho_j}\left[L\left(\mathbf{s}_j, \mathbf{a}_{j-1}, \mathbf{y}_j\right)\right] \tag{4.26}$$
$$= \mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\left[L\left(\mathbf{s}_i, \cdot, \cdot\right)\right] + \gamma \mathbb{E}_{\mathbf{s}_{i-1} \sim \varrho_{i-1}}\left[L\left(\mathbf{s}_{i-1}, \cdot, \cdot\right)\right] + \ldots$$

First a backward transition $\mathcal{T}^{-1}$ is a linear transformation, as established by Proposition 1. We then conclude that each ϱ_j is linear in ϱ_i. Hence, each expected stage cost term, appearing on the right-hand side of (4.26) is linear in ϱ_i. Then, the summation of discounted expectations on the right-hand side of the above equation will also be linear in the probability measure, which proves the proposition. □

Now, in the next proposition, we will show a relation between the value functional and the value function such that the following assumption must be satisfied:

Assumption 4. *Let us assume that the expected value function v is bounded for all $\varrho_i \in \mathcal{M}$ and $\mathbf{s}_i \in \mathcal{S}$:*

$$\mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\left[\left|v\left(\mathbf{s}_i, \mathbf{o}_i\right)\right|\right] < \infty \tag{4.27}$$

Note that the assumption above ensures that the expected value of the value function $v\left(\mathbf{s}_i, \mathbf{o}_i\right)$ will remain finite for all $\varrho_i \in \mathcal{M}$ and $\mathbf{s}_i \in \mathcal{S}$, a harmless restriction in practice.

Proposition 3. *Let the value function $v\left(\mathbf{s}_i, \mathbf{o}_i\right)$ be a Lebesgue measurable function (a.k.a Borel measurable) on the σ-algebra of Borel sets, and the probability measure ϱ_i be a compactly supported continuous function. A value functional then can be represented as an expected value function as follows:*

$$V\left[\varrho_i, \mathbf{o}_i\right] = \mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\left[v\left(\mathbf{s}_i, \mathbf{o}_i\right)\right] \tag{4.28}$$

Proof. Under Assumption 4 and the linearity of $V\left[\varrho_i, \mathbf{o}_i\right]$ in ϱ_i, see Proposition 2, the proof follows the Riesz-Markov theorem, see [56], chapter 9, page 105, such that:

$$V\left[\varrho_i, \mathbf{o}_i\right] = \int_{\mathcal{S}} v\left(\mathbf{s}_i, \mathbf{o}_i\right) \varrho_i(d\mathbf{s}_i) = \mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\left[v\left(\mathbf{s}_i, \mathbf{o}_i\right)\right]$$

and a value function $v\left(\mathbf{s}_i, \mathbf{o}_i\right)$ can be found so that the equality above holds. □

Now we choose the probability measure at time k as a Dirac measure centered at the current state such that $\varrho_k = \delta_{\mathbf{s}_k}(.)$. According to Proposition 3, the value functional then becomes a value function:

$$V\left[\varrho_k, \mathbf{o}_k\right] = \mathbb{E}_{\mathbf{s}_k \sim \delta_{\mathbf{s}_k}(.)}\left[v\left(\mathbf{s}_k, \mathbf{o}_k\right)\right] = v\left(\mathbf{s}_k, \mathbf{o}_k\right) \tag{4.29}$$

Then, the ideal stochastic MHE scheme with the value functional (4.4) can be rewritten as follows:

$$v\left(\mathbf{s}_k, \mathbf{o}_k\right) := \sum_{i=-\infty}^{k} \gamma^{k-i} \mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\left[L\left(\mathbf{s}_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right)\right], \tag{4.30a}$$

$$\mathbf{s}_k^{\star}(\mathbf{o}_k) \in \arg\min_{\mathbf{s}_k} v\left(\mathbf{s}_k, \mathbf{o}_k\right) \tag{4.30b}$$

where $\varrho_{i-1} = \mathcal{T}_{\mathbf{a}_{i-1}}^{-1} \varrho_i$ and $\varrho_k = \delta_{\mathbf{s}_k}(\cdot)$. We next show that the modified stage cost functional (4.13) can be rewritten as stage cost function at each time step i, which is constructed based on the value functions defined as (4.30a).

We first remind that the linear relation (4.6) between the stage cost functional and the stage cost function is valid based on the same synthesis as Proposition 3 using two underlying conditions: 1) expected stage cost function is bounded $\mathbb{E}_{\mathbf{s}_i \sim \varrho_i}\left[\left|L\left(\mathbf{s}_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right)\right|\right] < \infty$ 2) stage cost functional Φ is linear in ϱ_i. Hence, this relation also holds for $\hat{\Phi}$ in (4.13) considering the next remark.

Remark 2. *The modified stage cost functional* (4.13) *is also linear in the probability measures since it is defined based on a linear equation of the value functionals, which are linear in the probability measures, see Proposition 2.*

Now the modified stage cost functional can be described as:

$$\hat{\Phi}\left[\hat{\varrho}_i, \mathbf{o}_i\right] = \mathbb{E}_{\hat{\mathbf{s}}_i \sim \hat{\varrho}_i}\left[\hat{L}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right)\right] \tag{4.31}$$

where $\hat{L} : \mathcal{S} \times \mathcal{O} \to \mathbb{R}$ reads the modified stage cost function.

By considering Proposition 3, equality (4.31) and adopting $\hat{\varrho}_i = \delta_{\hat{\mathbf{s}}_i}(\cdot)$, one can obtain a practical cost modification of (4.13) at each time step i as follows:

$$\begin{aligned} &\mathbb{E}_{\hat{\mathbf{s}}_i \sim \delta_{\hat{\mathbf{s}}_i}(.)}\left[\hat{L}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right)\right] = \\ &\mathbb{E}_{\hat{\mathbf{s}}_i \sim \delta_{\hat{\mathbf{s}}_i}(.)}\left[v\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right)\right] - \gamma \mathbb{E}_{\hat{\mathbf{s}}_{i-1} \sim \hat{\varrho}_{i-1}}\left[v\left(\hat{\mathbf{s}}_{i-1}, \mathbf{o}_{i-1}\right)\right] \\ &\quad = v\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right) - \gamma \mathbb{E}_{\hat{\mathbf{s}}_{i-1} \sim \hat{\varrho}_{i-1}}\left[v\left(\hat{\mathbf{s}}_{i-1}, \mathbf{o}_{i-1}\right)\right] \end{aligned} \tag{4.32}$$

where $\hat{\varrho}_{i-1} = \hat{\zeta}_b\left[\cdot \,|\, \hat{\mathbf{s}}_i, \mathbf{a}_{i-1}\right]$.

Hence, the modified cost functional $\hat{\Phi}$ can then be replaced by the stage cost function $\hat{L}$ at time step i in practice:

$$\hat{L}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right) = v\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right) - \gamma \mathbb{E}_{\hat{\mathbf{s}}_{i-1} \sim \hat{\varrho}_{i-1}}\left[v\left(\hat{\mathbf{s}}_{i-1}, \mathbf{o}_{i-1}\right)\right] \tag{4.33}$$

We then adopt the above-modified stage cost function to formulate a modified MHE scheme using an imperfect model in practice. Hence, the modified stochastic MHE scheme based on the value (stage cost) function instead of the value (stage cost) functional (4.8) is formulated as follows:

$$\hat{v}\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) := \sum_{i=-\infty}^{k} \gamma^{k-i} \mathbb{E}_{\hat{\mathbf{s}}_i \sim \hat{\varrho}_i}\left[\hat{L}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right)\right] \tag{4.34a}$$

$$\hat{\mathbf{s}}_k^{\star}(\mathbf{o}_k) \in \arg\min_{\hat{\mathbf{s}}_k} \hat{v}\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) \tag{4.34b}$$

where $\hat{\varrho}_{i-1} = \hat{\mathcal{T}}_{\mathbf{a}_{i-1}}^{-1} \hat{\varrho}_i$ and $\hat{\varrho}_k = \delta_{\hat{\mathbf{s}}_k}(\cdot)$.

Now, according to the developments above, we have shown that the modified MHE scheme with a stage cost functional $\hat{\Phi}$ can be formulated as a tractable MHE (4.34) in which the modified cost function $\hat{L}$ (4.33) is practically constructed based on the value functions instead of the value functionals. The following corollary shows that the structure (4.34) can still preserve the property established by Theorem 1.

Corollary 1. *By adopting the same approach as was detailed to prove Theorem 1 and under the assumption*

$$\gamma^N \mathbb{E}_{\hat{\mathbf{s}}_{k-N} \sim \hat{\varrho}_{k-N}}\left[\left|v\left(\hat{\mathbf{s}}_{k-N}, \mathbf{o}_{k-N}\right)\right|\right] < \infty, \qquad \forall N \in \mathbb{I}_{\geq 0} \tag{4.35}$$

one can show that the following equalities hold:

$$\hat{v}\left(\cdot\right) = v\left(\cdot\right), \qquad \hat{\mathbf{s}}_k^{\star} = \mathbf{s}_k^{\star} \tag{4.36}$$

Proof. By substituting the modified stage cost function (4.33) in the value function associated to the problem (4.34) and using a telescoping sum argument, one can observe that:

$$\begin{aligned} \hat{v}\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) = \sum_{i=-\infty}^{k} \gamma^{k-i} \mathbb{E}_{\hat{\mathbf{s}}_i \sim \hat{\varrho}_i} \Big[& v\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right) \\ & - \gamma \mathbb{E}_{\hat{\mathbf{s}}_{i-1} \sim \hat{\varrho}_{i-1}}\left[v\left(\hat{\mathbf{s}}_{i-1}, \mathbf{o}_{i-1}\right)\right] \Big] = v\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) \end{aligned} \tag{4.37}$$

and

$$\arg\min_{\hat{\mathbf{s}}_k} \hat{v}\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) = \arg\min_{\hat{\mathbf{s}}_k} v\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) \tag{4.38}$$

results in $\hat{\mathbf{s}}_k^{\star}(\mathbf{o}_k) = \mathbf{s}_k^{\star}(\mathbf{o}_k)$. □

4.3.2 Tractable Modified Stage Cost

In the proposed modified stage cost function (4.33), the value functions captured from the MHE scheme (4.30) are based on the complete measurement history, and the amount of historical data is growing at each time instant. Hence, constructing the corresponding modified stage cost is intractable in practice. We then propose to formulate a finite version (H-step) of the optimization problem (4.30) so that the corresponding value function reads as:

$$v^{H}\left(\mathbf{s}_k, \mathbf{o}_k\right) := \gamma^{H}\mathbb{E}_{\mathbf{s}_{k-H}\sim\varrho_{k-H}}\left[Z_{k-H}\left(\mathbf{s}_{k-H}, \mathbf{o}_{k-H}\right)\right] + \sum_{i=k-H+1}^{k}\gamma^{k-i}\mathbb{E}_{\mathbf{s}_i\sim\varrho_i}\left[L\left(\mathbf{s}_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right)\right] \tag{4.39}$$

where $\varrho_{i-1} = \mathcal{T}^{-1}_{\mathbf{a}_{i-1}}\varrho_i$ and $\varrho_k = \delta_{\mathbf{s}_k}(\cdot)$.

Notice that the cost term $Z_{k-H}\left(\mathbf{s}_{k-H}, \mathbf{o}_{k-H}\right)$ is labeled the exact arrival cost function, which summarizes the effects of past information before time $k-H$. Then, under an exact arrival cost, the stochastic MHE scheme based on the value function (4.39) can be regarded as an ideal MHE scheme, i.e.,

$$v^{H}\left(\mathbf{s}_k, \mathbf{o}_k\right) = v\left(\mathbf{s}_k, \mathbf{o}_k\right), \tag{4.40}$$

Now the modified stage cost (4.33) can be rewritten based on the value function (4.39) as follows:

$$\hat{L}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right) = v^{H}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right) - \gamma\mathbb{E}_{\hat{\mathbf{s}}_{i-1}\sim\hat{\varrho}_{i-1}}\left[v^{H}\left(\hat{\mathbf{s}}_{i-1}, \mathbf{o}_{i-1}\right)\right] \tag{4.41}$$

Note that the expectation above is taken over the imperfect model whereas the expected values appeared in the definition of the value functions $v^{H}(\cdot)$ in (4.39) are on the real system. Although the value function (4.39) is based on the full measurement history, the implementation of the modified stage cost (4.41) will be finally tractable for a finite MHE scheme with a horizon N proposed in the next theorem. More specifically, the full history of the measurements due to the arrival cost term of v^{H} is transferred to the arrival cost Z_{k-N}, which can be approximated in practice. A practical implementation based on the mentioned argument above will be discussed in detail in section 4.4. Now we develop the next theorem for a modified MHE scheme based on the above-modified stage cost function. To this end, let us consider the following assumption:

Assumption 5. *There exists a non-empty set $\mathcal{S}_0 \subseteq \mathcal{S}$ such that for all $\hat{\mathbf{s}} \in \mathcal{S}_0$ and for all $\gamma \in (0, 1]$ it holds that*

$$\left|\gamma^{N_0}\mathbb{E}_{\hat{\mathbf{s}}_{k-N_0}\sim\hat{\varrho}_{k-N_0}}\left[v^{H}\left(\hat{\mathbf{s}}_{k-N_0}, \mathbf{o}_{k-N_0}\right)\right]\right| < \infty, \quad 0 \leq N_0 \leq N \tag{4.42}$$

where N is labeled a specific horizon window. Note that the expectation in the above inequality is taken over the imperfect model density

$$\hat{\varrho}_{k-N-1} = \hat{\mathcal{T}}^{-1}_{\mathbf{a}_{k-N-1}} \hat{\varrho}_{k-N}$$

Then, the following theorem is defined under the above-mentioned assumption:

Theorem 3. *There exists an exact arrival cost function, including some prior information as available observation $Z_{k-N} : \mathcal{S} \times \mathcal{O} \to \mathbb{R}$ and a modified stage cost function $\hat{L} : \mathcal{S} \times \mathcal{O} \to \mathbb{R}$. We then formulate the following finite stochastic MHE scheme at the current time k:*

$$\hat{v}^N(\hat{\mathbf{s}}_k, \mathbf{o}_k) := \gamma^N \mathbb{E}_{\hat{\mathbf{s}}_{k-N} \sim \hat{\varrho}_{k-N}} \left[Z_{k-N}(\hat{\mathbf{s}}_{k-N}, \mathbf{o}_{k-N}) \right] + \sum_{i=k-N+1}^{k} \gamma^{k-i} \mathbb{E}_{\hat{\mathbf{s}}_i \sim \hat{\varrho}_i} \left[\hat{L}(\hat{\mathbf{s}}_i, \mathbf{o}_i) \right] \tag{4.43a}$$

$$\hat{\mathbf{s}}_k^{\star,N} \in \arg\min_{\hat{\mathbf{s}}_k} \hat{v}^N(\hat{\mathbf{s}}_k, \mathbf{o}_k) \tag{4.43b}$$

where $\hat{\varrho}_{i-1} = \hat{\mathcal{T}}^{-1}_{\mathbf{a}_{i-1}} \hat{\varrho}_i$, $\hat{\varrho}_k = \delta_{\hat{\mathbf{s}}_k}(\cdot)$.

Then under Assumption 5, the MHE scheme above will deliver the following equalities for all $\hat{\mathbf{s}}_k \in \mathcal{S}_0$:

$$\hat{v}^N(\hat{\mathbf{s}}_k, \mathbf{o}_k) = v^H(\hat{\mathbf{s}}_k, \mathbf{o}_k), \quad \hat{\mathbf{s}}_k^{\star,N}(\mathbf{o}_k) = \mathbf{s}_k^{\star}(\mathbf{o}_k) \tag{4.44}$$

Proof. Let us select the modified stage cost (4.41) and define the arrival cost Z_{k-N} as follows:

$$Z_{k-N} = v^H(\hat{\mathbf{s}}_{k-N}, \mathbf{o}_{k-N}) \tag{4.45}$$

By substituting the modified stage cost function (4.41) and the arrival cost function (4.45) in the value function (4.43a), it then becomes a telescoping sum as follows:

$$\begin{aligned}
\hat{v}^N(\hat{\mathbf{s}}_k, \mathbf{o}_k) &= \gamma^N \mathbb{E}_{\hat{\mathbf{s}}_{k-N} \sim \hat{\varrho}_{k-N}} \left[v^H(\hat{\mathbf{s}}_{k-N}, \mathbf{o}_{k-N}) \right] \\
&+ \sum_{i=k-N+1}^{k} \gamma^{k-i} \mathbb{E}_{\hat{\mathbf{s}}_i \sim \hat{\varrho}_i} \left[v^H(\hat{\mathbf{s}}_i, \mathbf{o}_i) - \gamma \mathbb{E}_{\hat{\mathbf{s}}_{i-1} \sim \hat{\varrho}_{i-1}} \left[v^H(\hat{\mathbf{s}}_{i-1}, \mathbf{o}_{i-1}) \right] \right] \\
&= \gamma^N \mathbb{E}_{\hat{\mathbf{s}}_{k-N} \sim \hat{\varrho}_{k-N}} \left[v^H(\hat{\mathbf{s}}_{k-N}, \mathbf{o}_{k-N}) \right] + v^H(\hat{\mathbf{s}}_k, \mathbf{o}_k) \\
&- \gamma \mathbb{E}_{\hat{\mathbf{s}}_{k-1} \sim \hat{\varrho}_{k-1}} \left[v^H(\hat{\mathbf{s}}_{k-1}, \mathbf{o}_{k-1}) \right] + \gamma \mathbb{E}_{\hat{\mathbf{s}}_{k-1} \sim \hat{\varrho}_{k-1}} \left[v^H(\hat{\mathbf{s}}_{k-1}, \mathbf{o}_{k-1}) \right] + \dots \\
&- \gamma^N \mathbb{E}_{\hat{\mathbf{s}}_{k-N} \sim \hat{\varrho}_{k-N}} \left[v^H(\hat{\mathbf{s}}_{k-N}, \mathbf{o}_{k-N}) \right] = v^H(\hat{\mathbf{s}}_k, \mathbf{o}_k)
\end{aligned} \tag{4.46}$$

for all $\hat{\mathbf{s}}_k \in \mathcal{S}_0$, and

$$\arg\min_{\hat{\mathbf{s}}_k} \hat{v}^N\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) = \arg\min_{\hat{\mathbf{s}}_k} v^H\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) \tag{4.47}$$

delivers $\hat{\mathbf{s}}_k^{\star,N}(\mathbf{o}_k) = \mathbf{s}_k^{\star}(\mathbf{o}_k)$. □

Note that the horizon H is the length of the measurement history used in the modified stage cost in the MHE scheme (4.43) with a horizon of length N. In the next section, we will describe how this measurement history of length H can be used in the proposed convex neural network to modify the MHE stage cost in practice. It is worth noting that the horizon H may be selected larger than N to capture the modified stage cost accurately. However, one can choose a small length of H in order to provide an acceptable trade-off between the computational effort and the approximate value captured from the neural network.

4.4 Proposed Learning-based MHE-MPC Scheme

4.4.1 Practical Implementation

In what follows, we provide a practical version of the modified MHE scheme (4.43). We then discuss the approaches in order to approximate both the arrival cost Z_{k-N} and the modified stage cost $\hat{L}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i\right)$.

Learning-based Arrival Cost

To approximate the arrival cost, we adopt a common approach so that the arrival cost takes the form of a quadratic function as follows:

$$\hat{Z}_{k-N} = \|\hat{s}_{k-N} - \tilde{\mathbf{s}}_{k-N}\|^2_{\Pi^{-1}_{k-N}} \tag{4.48}$$

where $\tilde{\mathbf{s}}_{k-N}$ is obtained as:

$$\tilde{\mathbf{s}}_{k-N} = \mathbf{s}^{\star}_{k-N|k-1} \tag{4.49}$$

Note that $\mathbf{s}^{\star}_{k-N|k-1}$ is the first element of the horizon window at the previous physical time $k-1$. The prior weighting Π_{k-N} is obtained from the Kalman filter covariance update rule [10]:

$$\begin{aligned}\Pi_{k+1} =& A_k \Pi_k A_k^{\top} \\ &- A_k \Pi_k C_k^{\top}\left(C_k \Pi_k C_k^{\top} + R\right)^{-1} C_k \Pi_k A_k^{\top}\end{aligned} \tag{4.50}$$

initialized with the covariance matrix of the initial state Π_0. Let $\mathbf{f}(\hat{\mathbf{s}}, \mathbf{a})$ be a nonlinear model as a deterministic approximation of (4.2). The matrices A_k and C_k are then obtained by linearization as follows:

$$A_k = \frac{\partial \mathbf{f}}{\partial \hat{\mathbf{s}}}|_{\hat{\mathbf{s}}_{k|k-1}}, \qquad C_k = \frac{\partial \mathbf{h}}{\partial \hat{\mathbf{s}}}|_{\hat{\mathbf{s}}_{k|k-1}} \tag{4.51}$$

and R is the covariance of the output noise $\boldsymbol{\nu}_k$ where the measurements are delivered as $\mathbf{y}_k = \mathbf{h}(\hat{\mathbf{s}}_k) + \boldsymbol{\nu}_k$. However, the approach detailed above is based on classic Kalman filtering that may not be the best choice from a parameterization standpoint where the model is imperfect. More specifically, the update rule (4.50) cannot deliver a perfect approximation of Π since the matrices A, C captured, respectively, from the dynamical model f and the measurement model h are imperfect. To tackle this problem, we propose to adopt reinforcement learning in order to adjust the entries of the matrices $A_k^{\boldsymbol{\theta}}, C_k^{\boldsymbol{\theta}}$ and the covariance matrix $R_{\boldsymbol{\theta}}$ used in (4.50), where $\boldsymbol{\theta}$ will be parameters that can be adjusted via RL. Then, the parameterized covariance update rule reads as:

$$\begin{aligned}\Pi_{k+1} =& A_k^{\boldsymbol{\theta}} \Pi_k \left(A_k^{\boldsymbol{\theta}}\right)^\top \\ &- A_k^{\boldsymbol{\theta}} \Pi_k \left(C_k^{\boldsymbol{\theta}}\right)^\top \left(C_k^{\boldsymbol{\theta}} \Pi_k \left(C_k^{\boldsymbol{\theta}}\right)^\top + R_{\boldsymbol{\theta}}\right)^{-1} C_k^{\boldsymbol{\theta}} \Pi_k \left(A_k^{\boldsymbol{\theta}}\right)^\top\end{aligned} \tag{4.52}$$

It is worth noting that the policy $\boldsymbol{\pi}$ captured from the MHE-MPC scheme will have an extra state Π_k, which is obtained from the above dynamics. More specifically, Π_k has its own dynamics in the MHE scheme such that the state estimation and the policy delivered, respectively, from the MHE and MHE-MPC will depend on Π_k. We then consider the effect of Π_k on the policy gradient in an MHE/MPC-based reinforcement learning detailed in the next section.

Learning-based MHE Stage Cost

According to Theorem 3, a finite stochastic MHE scheme can deliver a true state estimation using an imperfect model of the real system by adopting a cost modification.

Remark 3. *In this chapter, we denote the true state estimation (perfect estimation) by the estimation captured from the FIE problems with the correct model in* (4.30) *and* (4.39).

In this theorem, we have proposed to construct a modified stage cost (4.41) based on the H-step value function (4.39), and we practically propose to approximate this modified stage cost. To this end, let us consider the MHE scheme (4.43) and the

value function (4.39) where $\hat{\varrho}_i = \delta_{\hat{\mathbf{s}}_i}(\cdot)$. We then observe that all expected arrival cost functions $Z_{i-H}, Z_{i-H-1}, i = k-N+1, \cdots, k$, including entire history can be transferred to the arrival cost Z_{k-N}. More precisely, the time step i used in the modified stage cost of the MHE scheme (4.43) is in the interval $i = k-N+1, \cdots, k$, and this stage cost defined in (4.41) is basically constructed based on the value functions $v^H(\hat{\mathbf{s}}_i, \mathbf{o}_i)$ and $v^H(\hat{\mathbf{s}}_{i-1}, \mathbf{o}_{i-1})$ captured from (4.39). The corresponding arrival costs then read as Z_{i-H} and Z_{i-H-1} for $i = k-N+1, \cdots, k$. Hence, the modified stage cost can practically be constructed based on a finite history as follows:

$$\hat{L}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i^H\right) = L\left(\hat{\mathbf{s}}_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right) + L_h\left(\hat{\mathbf{s}}_{i-H,\cdots,i-1}, \mathbf{o}_{i-1}^H\right) \tag{4.53}$$

where the cost term L_h is constructed based on a finite history and

$$\mathbf{o}_i^H = \operatorname{col}\left\{\mathbf{y}_{i-H+1}, \ldots, \mathbf{y}_i, \mathbf{a}_{i-H}, \ldots, \mathbf{a}_{i-1}\right\}$$

We then propose to somehow approximate this part of the modified stage cost $\hat{L}$ in practice.

As one practical solution to approximate L_h, one can use a Neural Network (NN) as follows:

$$\hat{L}_{\boldsymbol{\theta}}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i^H\right) \approx L_{\boldsymbol{\theta}_0}\left(\hat{\mathbf{s}}_i, \mathbf{a}_{i-1}, \mathbf{y}_i\right) + L_{\mathrm{NN}}\left(\mathbf{Y}_i, \boldsymbol{\theta}_{\mathrm{NN}}\right) \tag{4.54}$$

where $L_{\boldsymbol{\theta}_0}$ is a parameterized least-squares cost at the current time step i used for an *output noise MHE* scheme [33] and

$$\begin{aligned} \mathbf{Y}_i &= \operatorname{col}\left\{\hat{\mathbf{s}}_{i-H,\cdots,i-1}, \mathbf{o}_{i-1}^H\right\} \\ \mathbf{o}_{i-1}^H &= \operatorname{col}\left\{\mathbf{y}_{i-H}, \ldots, \mathbf{y}_{i-1}, \mathbf{a}_{i-H-1}, \ldots, \mathbf{a}_{i-2}\right\} \end{aligned} \tag{4.55}$$

Note that $\mathbf{o}_{i-1}^H \in \mathcal{O}^H \subset \mathcal{O}$ is regarded as a finite history of the measurements and $\mathbf{Y}_i \in \mathbb{R}^{n_y}$ is labeled the Neural Network (NN) input. To adjust the parameters θ_0, θ_{NN}, we will use a reinforcement learning algorithm based on the policy gradient method.

Neural networks are well-known universal function approximators so an NN, including three layers is capable to approximate any continuous multivariate function down to prescribed accuracy, if there are no constraints on the number of neurons [57].

We then propose to use a convex class of NNs to approximate L_{NN} in the MHE stage cost (4.54). While enforcing convexity in the MHE stage cost does not imply that the overall MHE problem is convex, using a non-convex stage cost will often

require significantly more caution in providing an initial guess for the NLP solver tackling the MHE scheme than if a convex stage cost is used. To preserve the convexity of the MHE stage cost function (4.54), we then propose to compute L_{NN} using an Input Convex Neural Network (ICNN). In this type of neural network, the partial weights meet certain constraints such that the output of the ICNN is a convex function of the input [58, 59]. Compared to building conventional neural networks, ICNN structures must meet two additional requirements: 1) activation functions are convex and non-decreasing 2) the weights of NN are constrained to be non-negative. As a form of ICNN, we choose a Fully Input Convex NN (FICNN) architecture since the scalar output of the network is convex with respect to all inputs.

Let us consider a l-layer FICNN over $\mathbf{Y}_i$ in order to estimate $L_{\text{NN}}\left(\mathbf{Y}_i, \boldsymbol{\theta}_{\text{NN}}\right)$ as follows:

$$z_{j+1} = g_j\left(W_j^{(z)} z_j + W_j^{(y)} \mathbf{Y}_i + b_j\right), \tag{4.56a}$$

$$L_{\text{NN}}\left(\mathbf{Y}_i, \theta\right) = c \cdot z_l \tag{4.56b}$$

$$s.t. \quad W_{1:l-1}^{(z)} \geq 0, W_0^{(z)} \equiv 0, z_0 \equiv 0 \tag{4.56c}$$

where $j = 0, \ldots, l-1$ and $z_j \in \mathbb{R}^{n_y \times 1}$ denotes the middle layers (layer activations). The neural network weights are $W_j^{(z)} \in \mathbb{R}^{n_y \times n_y}, W_j^{(y)} \in \mathbb{R}^{n_y \times n_y}, b_j \in \mathbb{R}^{n_y \times 1}, c \in \mathbb{R}^{1 \times n_y}$. Note that c is considered as the connection weight between the output layer and the last middle layer. Then, $\boldsymbol{\theta}_{\text{NN}} = \left\{W_{1:l-1}^{(z)}, W_{0:l-1}^{(y)}, b_{0:l-1}, c\right\}$ are the modifiable weights, and g_j are nonlinear activation functions (convex and non-decreasing, e.g., Rectified Linear Unit *ReLU*).

Then, the reformulated MHE scheme (4.43) reads as:

$$\hat{v}^N\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) := \gamma^N \mathbb{E}_{\hat{\mathbf{s}}_{k-N} \sim \hat{\varrho}_{k-N}}\left[\hat{Z}_{k-N}\left(\hat{\mathbf{s}}_{k-N}, \mathbf{o}_{k-N}^H\right)\right] + \sum_{i=k-N+1}^{k} \gamma^{k-i} \mathbb{E}_{\hat{\mathbf{s}}_i \sim \hat{\varrho}_i}\left[\hat{L}_{\boldsymbol{\theta}}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i^H\right)\right] \tag{4.57a}$$

$$\hat{\mathbf{s}}_k^{\star,N} \in \arg\min_{\hat{\mathbf{s}}_k} \hat{v}^N\left(\hat{\mathbf{s}}_k, \mathbf{o}_k\right) \tag{4.57b}$$

where $\hat{\varrho}_{i-1} = \hat{\mathcal{T}}_{\mathbf{a}_{i-1}}^{-1} \hat{\varrho}_i, \quad \hat{\varrho}_k = \delta_{\hat{\mathbf{s}}_k}(\cdot)$.

In the remainder of this section, we practically formulate a deterministic version of the above MHE scheme (4.57) with a fully parameterized cost function, including arrival and stage cost. We then propose a parameterization for the MPC scheme to deliver a policy approximation required in the context of policy gradient RL.

4.4.2 Deterministic MHE Scheme with Adjustable Cost

As a result of theorem 3, a finite optimization-based state estimation scheme with an imperfect model can deliver a true state estimation by modifying the stage and arrival costs. As a practical approach, we proposed to leverage NN in approximating the modified stage cost function (4.54). One can also choose a parameterization method on the arrival cost detailed in the previous section. Finally, a reinforcement learning algorithm is used to adjust all parameters.

Note that the theory proposed is very generic so that the acquired result holds e.g., for the states of a stochastic dynamical model being estimated by an MHE scheme based on an inaccurate deterministic model. Hence, the transition model $\hat{\zeta}\left[\hat{\mathbf{s}}_{i+1} \,|\, \hat{\mathbf{s}}_i, \mathbf{a}_i\right]$ trivially includes deterministic models as:

$$\hat{\zeta}\left[\hat{\mathbf{s}}_{i+1} \,|\, \hat{\mathbf{s}}_i, \mathbf{a}_i\right] = \delta\left(\hat{\mathbf{s}}_{i+1} - \hat{\mathbf{f}}^{\text{MHE}}\left(\hat{\mathbf{s}}_i, \mathbf{a}_i\right)\right) \tag{4.58}$$

We then propose to formulate the following parameterized MHE scheme:

$$\begin{aligned}
\mathbf{s}^{\star}_{k-N_{\text{MHE}},\ldots,k} = \arg\min_{\hat{\mathbf{s}}} \quad & \gamma^{N_{\text{MHE}}} \hat{Z}_{\boldsymbol{\theta}}\left(\hat{\mathbf{s}}_{k-N_{\text{MHE}}}, \tilde{\mathbf{s}}\right) \\
& + \sum_{i=k-N_{\text{MHE}}+1}^{k} \gamma^{k-i} \hat{L}_{\boldsymbol{\theta}}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i^H\right) && \text{(4.59a)} \\
\text{s.t.} \quad & \hat{\mathbf{s}}_{i+1} = \hat{\mathbf{f}}^{\text{MHE}}_{\boldsymbol{\theta}}(\hat{\mathbf{s}}_i, \mathbf{a}_i) && \text{(4.59b)}
\end{aligned}$$

where $\hat{Z}_{\boldsymbol{\theta}} = \|\hat{\mathbf{s}}_{k-N_{\text{MHE}}} - \tilde{\mathbf{s}}\|^2_{\Pi_k}$ and Π_k is obtained from the parameterized updating rule (4.52) and $\tilde{\mathbf{s}}$ is the available estimation $\mathbf{s}^{\star}_{k-N_{\text{MHE}}}$ at time $k-1$. Note that in the case of linear systems, another solution to adjust the arrival cost is to directly modify the arrival cost by adjusting the corresponding positive weight matrix $\Pi_{\boldsymbol{\theta}}$ using e.g., Semi-Definite Programming (SDP).

4.4.3 Parameterized MPC Scheme

Let us define the closed-loop performance of a parameterized policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ delivered from an MHE-MPC scheme for a given stage cost $L\left(\mathbf{y}_k, \mathbf{a}_k\right)$ as the following total expected cost:

$$J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right) = \mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left[\sum_{k=0}^{\infty} \gamma^k L\left(\mathbf{y}_k, \mathbf{a}_k\right) \,\middle|\, \mathbf{a}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}^{\star}_k)\right] \tag{4.60}$$

where the expectation $\mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ is taken over the distribution of the Markov chain in closed-loop with policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$. The cost $L\left(\mathbf{y}_k, \mathbf{a}_k\right)$ reads as a baseline cost (RL stage cost), which is a function of measurable states and actions at the current time k.

It is worth noting that the initial conditions are defined by the environment (real MDP), e.g., in the simulation section, the MDPs for test cases 1 and 2 are defined with fixed initial conditions while the third test case has an MDP with random initial conditions. We then seek the optimal policy parameters as follows:

$$\boldsymbol{\theta}_\star = \arg\min_{\boldsymbol{\theta}} J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right) \tag{4.61}$$

As a learning-based control approach in this chapter, we propose to use a parameterized MPC scheme as a policy approximation in order to deliver $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ required in a policy gradient method. The MPC-based reinforcement learning then allows us to leverage the capability of MPC in handling the state-input constraints. Although a constraint violation may occur due to an imperfect MPC model, the constraints are finally satisfied by letting RL adjust the whole MPC scheme, e.g., the constraints can be adjusted in the case of constraint violation.

For a given estimated state $\mathbf{s}_k^\star$ obtained from the MHE scheme, the policy delivered by a parameterized MPC scheme is

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}_k^\star\right) = \mathbf{u}_0^\star\left(\mathbf{s}_k^\star, \boldsymbol{\theta}\right) \tag{4.62}$$

where $\mathbf{u}_0^\star$ is the first element of the control input sequence $\mathbf{u}^\star$ delivered by the following parameterized MPC scheme:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad & \gamma^{N_{\text{MPC}}}\left(T_{\boldsymbol{\theta}}(\mathbf{x}_{k+N_{\text{MPC}}}) + \mathbf{w}_f^\top \boldsymbol{\sigma}_{k+N_{\text{MPC}}}\right) \\
& + \sum_{i=k}^{k+N_{\text{MPC}}-1} \gamma^{i-k}\left(l_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{u}_i) + \mathbf{w}^\top \boldsymbol{\sigma}_i\right) && \text{(4.63a)} \\
\text{s.t.} \quad & \mathbf{x}_{i+1} = \hat{\mathbf{f}}_{\boldsymbol{\theta}}^{\text{MPC}}(\mathbf{x}_i, \mathbf{u}_i), && \text{(4.63b)} \\
& \mathbf{x}_k = \mathbf{s}_k^\star, && \text{(4.63c)} \\
& \mathbf{g}(\mathbf{u}_i) \leq 0, && \text{(4.63d)} \\
& \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{u}_i) \leq \boldsymbol{\sigma}_i, \quad \mathbf{h}_{\boldsymbol{\theta}}^f(\mathbf{x}_{k+N_{\text{MPC}}}) \leq \boldsymbol{\sigma}_{k+N_{\text{MPC}}} && \text{(4.63e)} \\
& \boldsymbol{\sigma}_{k,\ldots,k+N_{\text{MPC}}} \geq 0 && \text{(4.63f)}
\end{aligned}$$

where $l_{\boldsymbol{\theta}}$ and $T_{\boldsymbol{\theta}}$ are the parameterized stage cost and terminal cost, respectively. Note that the imperfect MPC model $\hat{\mathbf{f}}^{\text{MPC}}$ is possibly but not necessarily different from the MHE model. We label $\mathbf{h}_{\boldsymbol{\theta}}$ the mixed constraints, $\mathbf{g}$ the pure input constraints, and $\mathbf{h}_{\boldsymbol{\theta}}^f$ the terminal constraints. The MPC initial conditions in (4.63c) are delivered by the MHE scheme at the current time instant k. To relax the inequality constraints, an ℓ_1 relaxation of the mixed constraints (4.63e) is introduced. An

exact penalty is then imposed on the corresponding slack variables $\boldsymbol{\sigma}_k$ with large enough weights $\mathbf{w}, \mathbf{w}_f$ such that the MPC scheme will not be infeasible under some constraint violations, which appears due to inaccurate MPC model, uncertainties and disturbances.

4.5 Policy Gradient RL with MHE-MPC

In this section, we propose a new observer-based RL framework based on Deterministic Policy Gradient (DPG), MPC, and MHE to deal with the partially observable and imperfect dynamics.

4.5.1 Compatible Deterministic Actor-Critic

In the context of DPG-based RL algorithms, the policy parameters $\boldsymbol{\theta}$ can be directly optimized by the gradient descent step such that the best-expected closed-loop cost (a.k.a policy performance index J) can be captured by applying the policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$. More specifically, the policy parameters $\boldsymbol{\theta}$ can be updated as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) \tag{4.64}$$

for some $\alpha > 0$ small enough as the step size. In the context of hybrid controller/observer scheme MHE-MPC, the input signal can be interpreted as a sequence of measurements $\bar{\mathbf{o}}_k = \operatorname{col}\left\{\mathbf{a}_{k-N_{\text{MHE}},\ldots,k-1}, \mathbf{y}_{k-N_{\text{MHE}},\ldots,k}\right\} \in \mathcal{O}$ at the physical time k. Then, the intermediate variable $\mathbf{s}_k^\star$ is delivered by the MHE scheme based on the history of the measurements and fed to the MPC scheme to deliver the control policy. Let us assume that the measurement history $\bar{\mathbf{o}}_k$ of length $N_{\mathbf{o}}$ is sufficient to determine the statistics of the next output $\mathbf{y}_{k+1}$ such that it remains unaffected for any $\bar{N}_{\mathbf{o}} > N_{\mathbf{o}}$. It follows that $\bar{\mathbf{o}}_k$ is a Markov state. We then consider an input-output MDP based on $\bar{\mathbf{o}}_k$ rather than on the state of the real system, and consequently, this MDP can be described based on the state estimation $\mathbf{s}_k^\star$ as an implicit function of $\bar{\mathbf{o}}_k$. Therefore, the state estimation $\mathbf{s}_k^\star$ also reads as a Markov state, and one can use it in the state (-action) value functions. Let us define the policy performance index by the following expected value:

$$J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right) = \mathbb{E}_{\bar{\mathbf{o}}_k \sim p_k}\left[Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}_k^\star, \boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}_k^\star\right)\right)\right] = \mathbb{E}_{\bar{\mathbf{o}}_k \sim p_k}\left[V_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}_k^\star\right)\right] \tag{4.65}$$

where p_k is the measurement distribution at the current physical time k, e.g., a Gaussian distribution. Note that we remove $\bar{\mathbf{o}}_k$ from the arguments of the state (-action) value functions $Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ and $V_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ above as $\mathbf{s}_k^\star$ is implicitly constructed (delivered from the MHE scheme (4.59)) based on the history $\bar{\mathbf{o}}_k$. It also follows that the control policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\mathbf{s}_k^\star\right) = \mathbf{u}_0^\star\left(\mathbf{s}_k^\star, \boldsymbol{\theta}\right)$ captured from the MHE-MPC reads as an implicit function of the measurement history. The action-value function $Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ is then defined as follows:

$$Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}_k^\star, \mathbf{a}_k\right) = L\left(\mathbf{y}_k, \mathbf{a}_k\right) + \gamma \mathbb{E}_\zeta\left[V_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\mathbf{s}_{k+1}^\star\right) | \mathbf{s}_k^\star, \mathbf{a}_k\right] \tag{4.66}$$

where the expectation $\mathbb{E}_{\zeta}$ is taken over the distribution of the Markov chain (4.2). Based on the proposed DPG theorem by [17] and the fact that both the $\boldsymbol{\pi_\theta}$ and $Q_{\boldsymbol{\pi_\theta}}$ are functions of $\mathbf{s}_k^\star$, the policy gradient equation is described as follows:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi_\theta}) = \mathbb{E}_{\boldsymbol{\pi_\theta}} \left[\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi_\theta}(\mathbf{s}_k^\star) \nabla_{\mathbf{a}_k} Q_{\boldsymbol{\pi_\theta}}(\mathbf{s}_k^\star, \mathbf{a}_k)|_{\mathbf{a}_k = \boldsymbol{\pi_\theta}} \right] \tag{4.67}$$

where the expectation $\mathbb{E}_{\boldsymbol{\pi_\theta}}$ is taken over the distribution of the Markov chain resulting from the real system in closed-loop with $\boldsymbol{\pi_\theta}$. To represent the effect of the parameterized MHE upon the policy gradient, the sensitivity of the policy w.r.t $\boldsymbol{\theta}$ can be updated such that the new policy gradient is described by the following expectation:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi_\theta}) = \mathbb{E}_{\boldsymbol{\pi_\theta}} \left[\Xi \nabla_{\mathbf{a}_k} Q_{\boldsymbol{\pi_\theta}}(\mathbf{s}_k^\star, \mathbf{a}_k)|_{\mathbf{a}_k = \boldsymbol{\pi_\theta}} \right] \tag{4.68}$$

where the Jacobian matrix Ξ is obtained by the following chain rule:

$$\Xi = \nabla_{\boldsymbol{\theta}} \boldsymbol{\pi_\theta} + \left(\nabla_{\boldsymbol{\theta}} \mathbf{s}_k^\star + \nabla_{\boldsymbol{\theta}} \Pi_k \nabla_{\Pi_k} \mathbf{s}_k^\star \right) \nabla_{\mathbf{s}_k^\star} \boldsymbol{\pi_\theta} \tag{4.69}$$

Hence, the Jacobian matrix above is constructed based on both the MHE and MPC sensitivities where the optimal policy is delivered by a combined MHE-MPC scheme. In this chapter, we adopt a *compatible deterministic actor-critic* algorithm [17] in which the action-value function $Q_{\boldsymbol{\pi_\theta}}(\mathbf{s}_k^\star, \mathbf{a}_k)$ can be replaced by a class of compatible function approximator $Q^{\mathbf{w}}(\mathbf{s}_k^\star, \mathbf{a}_k)$ such that the policy gradient is preserved. Therefore, the compatible function for a deterministic policy $\boldsymbol{\pi_\theta}$ delivered by the parameterized MHE-MPC scheme can be expressed as follows:

$$Q^{\mathbf{w}}(\mathbf{s}_k^\star, \mathbf{a}_k) = (\mathbf{a}_k - \boldsymbol{\pi_\theta})^\top \Xi^\top \mathbf{w} + V^{\boldsymbol{\nu}}(\mathbf{s}_k^\star) \tag{4.70}$$

The first term in the above compatible function as the critic part is an estimation for the advantage function and the second term estimates a value function for the history of the measurements delivered as a summarized variable $\mathbf{s}_k^\star$ by the MHE scheme. Both functions can be computed by the linear function approximators as follows:

$$V^{\boldsymbol{\nu}}(\mathbf{s}_k^\star) = \boldsymbol{\Upsilon}(\mathbf{s}_k^\star)^\top \boldsymbol{\nu}, \tag{4.71a}$$

$$A^{\mathbf{w}}(\mathbf{s}_k^\star, \mathbf{a}_k) = \boldsymbol{\Psi}(\mathbf{s}_k^\star, \mathbf{a}_k)^\top \mathbf{w} \tag{4.71b}$$

where $\boldsymbol{\Upsilon}(\mathbf{s}_k^\star)$ is the summarized measurement feature vector in order to constitute all monomials of the history of the measurements with degrees less than or equal to 2. The vector $\boldsymbol{\Psi}(\mathbf{s}_k^\star, \mathbf{a}_k) := \Xi(\mathbf{a}_k - \boldsymbol{\pi_\theta}(\mathbf{s}_k^\star))$ includes the state-action features. Considering (4.70), the policy gradient (4.68) is then rewritten as follows:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi_\theta}) = \mathbb{E}_{\boldsymbol{\pi_\theta}} \left[\Xi \Xi^\top \mathbf{w} \right] \tag{4.72}$$

Figure 4.1 shows an overview of the proposed learning-based MHE-MPC.

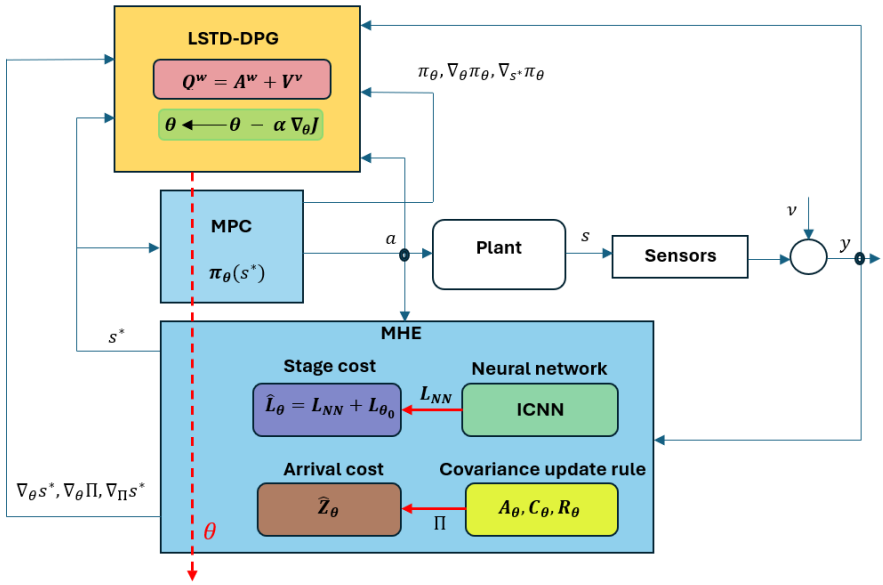


Figure 4.1: An overview of the MHE/MPC-based deterministic policy gradient. Both the MPC and MHE models are assumed to be inaccurate (they cannot capture the real plant perfectly). The deterministic MHE scheme (4.59) delivers the state estimation $\mathbf{s}^\star$, and the corresponding sensitivities are used in the LSTD-DPG method. The MPC scheme (4.63) combined with the MHE scheme delivers the parameterized policy $\boldsymbol{\pi_\theta}$. The action a is then selected according to the policy π_θ with the possible addition of exploratory moves.

In this chapter, the parameterized policy in the context of policy gradient RL is proposed to be captured by the MPC scheme (4.63). To evaluate the policy gradient (4.72), one needs to calculate some sensitivities upon the MPC and MHE schemes in order to compute the Jacobian matrix Ξ. Hence, the Jacobian matrices $\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi_\theta}$ and $\nabla_{\mathbf{s}_k^\star} \boldsymbol{\pi_\theta}$ can be computed by the sensitivity analysis for the parameterized MPC scheme while the gradient $\nabla_{\boldsymbol{\theta}} \mathbf{s}_k^\star$ and the Jacobian matrices $\nabla_{\boldsymbol{\theta}} \Pi_k$, $\nabla_{\Pi_k} \mathbf{s}_k^\star$ are obtained as sensitivity terms for the parameterized MHE scheme.

4.5.2 Sensitivity Analysis and LSTD-based DPG

Sensitivity Computation

We describe next how to compute the sensitivities (gradients) needed in the proposed policy gradient RL framework based on MHE-MPC. To that end, let us define the Lagrange functions $\hat{\mathcal{L}}_{\boldsymbol{\theta}}, \mathcal{L}_{\boldsymbol{\theta}}$ associated to the MHE and MPC schemes

(4.59), (4.63) as follows:

$$\hat{\mathcal{L}}_{\boldsymbol{\theta}}\left(\hat{\mathbf{z}}\right) = \hat{\Lambda}_{\boldsymbol{\theta}} + \hat{\boldsymbol{\lambda}}^{\top}\hat{G}_{\boldsymbol{\theta}} \tag{4.73}$$

$$\mathcal{L}_{\boldsymbol{\theta}}\left(\mathbf{z}\right) = \Lambda_{\boldsymbol{\theta}} + \boldsymbol{\lambda}^{\top}G_{\boldsymbol{\theta}} + \boldsymbol{\mu}^{\top}H_{\boldsymbol{\theta}} \tag{4.74}$$

where $\Lambda_{\boldsymbol{\theta}}$ and $\hat{\Lambda}_{\boldsymbol{\theta}}$ are the total parameterized costs of the MPC and MHE schemes, respectively. The inequality constraints of (4.63) are collected by $H_{\boldsymbol{\theta}}$ while $G_{\boldsymbol{\theta}}$ and $\hat{G}_{\boldsymbol{\theta}}$ gather, respectively, the equality constraints in the MPC and MHE schemes. We then label $\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}$ the Lagrange multipliers associated with the equality constraints $G_{\boldsymbol{\theta}}, \hat{G}_{\boldsymbol{\theta}}$ of the MPC and MHE, respectively. Variables $\boldsymbol{\mu}$ are the Lagrange multipliers associated with the inequality constraints of the MPC scheme. Let us label $\boldsymbol{\Gamma} = \{\mathbf{x}, \mathbf{u}, \boldsymbol{\sigma}\}$ and $\hat{\boldsymbol{\Gamma}} = \hat{\mathbf{s}}$ the primal variables for the MPC and MHE, respectively. The associated primal-dual variables then read as $\mathbf{z} = \{\boldsymbol{\Gamma}, \boldsymbol{\lambda}, \boldsymbol{\mu}\}$ and $\hat{\mathbf{z}} = \left\{\hat{\boldsymbol{\Gamma}}, \hat{\boldsymbol{\lambda}}\right\}$.

The sensitivity of the policy delivered by the MPC scheme (4.63) w.r.t policy parameters and the sensitivity of the estimated state associated with the MHE scheme (4.59) can be obtained via using the Implicit Function Theorem (IFT) on the Karush Kuhn Tucker (KKT) conditions underlying the parametric NLP. Assuming that Linear Independence Constraint Qualification (LICQ) and Second Order Sufficient Condition (SOSC) hold [18] at $\mathbf{z}^{\star}$ and $\hat{\mathbf{z}}^{\star}$, then, the following holds:

$$\frac{\partial \mathbf{z}^{\star}}{\partial \boldsymbol{\theta}} = -\frac{\partial \kappa_{\boldsymbol{\theta}}}{\partial \mathbf{z}}^{-1}\frac{\partial \kappa_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}, \tag{4.75a}$$

$$\frac{\partial \hat{\mathbf{z}}^{\star}}{\partial \boldsymbol{\theta}} = -\frac{\partial \hat{\kappa}_{\boldsymbol{\theta}}}{\partial \hat{\mathbf{z}}}^{-1}\frac{\partial \hat{\kappa}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \tag{4.75b}$$

where

$$\kappa_{\boldsymbol{\theta}} = \begin{bmatrix} \nabla_{\boldsymbol{\Gamma}}\mathcal{L}_{\boldsymbol{\theta}} \\ G_{\boldsymbol{\theta}} \\ \operatorname{diag}\left(\boldsymbol{\mu}\right)\mathbf{H}_{\boldsymbol{\theta}} \end{bmatrix}, \quad \hat{\kappa}_{\boldsymbol{\theta}} = \begin{bmatrix} \nabla_{\hat{\boldsymbol{\Gamma}}}\hat{\mathcal{L}}_{\boldsymbol{\theta}} \\ \hat{G}_{\boldsymbol{\theta}} \end{bmatrix} \tag{4.76}$$

are the KKT conditions associated with the MPC and MHE schemes, respectively. As $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ and $\mathbf{s}_k^{\star}$ are, respectively, part of $\mathbf{z}^{\star}$ and $\hat{\mathbf{z}}^{\star}$. Then, the sensitivity of the MPC policy $\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi}_{\boldsymbol{\theta}}$ and the sensitivity of the MHE solution $\nabla_{\boldsymbol{\theta}}\mathbf{s}_k^{\star}$ required in (4.72) can be extracted from gradients $\frac{\partial \mathbf{z}^{\star}}{\partial \boldsymbol{\theta}}$ and $\frac{\partial \hat{\mathbf{z}}^{\star}}{\partial \boldsymbol{\theta}}$, respectively.

LSTD-based Policy Gradient

In the context of compatible DPG, one can evaluate the optimal parameters $\mathbf{w}$ and $\boldsymbol{\nu}$ of the action-value function approximation (4.70) as solutions of the following

Least Squares (LS) problem:

$$\min_{\mathbf{w},\boldsymbol{\nu}} \mathbb{E}\left[\left(Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\mathbf{s}_k^\star, \mathbf{a}_k) - Q^{\mathbf{w}}(\mathbf{s}_k^\star, \mathbf{a}_k)\right)^2\right], \tag{4.77}$$

In the context of RL, the Least-Squares Temporal Difference (LSTD) algorithms offer efficient use of data and tend to converge faster than other methods [22]. The LSTD update rules for a policy gradient RL are then obtained as follows:

$$\boldsymbol{\nu} = \Omega_{\boldsymbol{\nu}}^{-1} b_{\boldsymbol{\nu}}, \tag{4.78a}$$

$$\mathbf{w} = \Omega_{\mathbf{w}}^{-1} b_{\mathbf{w}}, \tag{4.78b}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha b_{\boldsymbol{\theta}} \tag{4.78c}$$

where the matrices $\Omega_{(\cdot)}$ and the vectors $b_{(\cdot)}$ are calculated by taking expectation ($\mathbb{E}_m$) over m episodes as follows:

$$\Omega_{\boldsymbol{\nu}} = \mathbb{E}_m\left[\sum_{k=1}^{T_f}\left[\boldsymbol{\Upsilon}\left(\mathbf{s}_k^\star\right)\left(\boldsymbol{\Upsilon}\left(\mathbf{s}_k^\star\right) - \gamma\boldsymbol{\Upsilon}\left(\mathbf{s}_{k+1}^\star\right)\right)^\top\right]\right] \tag{4.79a}$$

$$\Omega_{\mathbf{w}} = \mathbb{E}_m\left[\sum_{k=1}^{T_f}\left[\boldsymbol{\Psi}\left(\mathbf{s}_k^\star, \mathbf{a}_k\right)\boldsymbol{\Psi}\left(\mathbf{s}_k^\star, \mathbf{a}_k\right)^\top\right]\right], \tag{4.79b}$$

$$b_{\boldsymbol{\nu}} = \mathbb{E}_m\left[\sum_{k=1}^{T_f}\boldsymbol{\Upsilon}\left(\mathbf{s}_k^\star\right)L(\mathbf{y}_k, \mathbf{a}_k)\right], \tag{4.79c}$$

$$b_{\mathbf{w}} = \tag{4.79d}$$

$$\mathbb{E}_m\left[\sum_{k=1}^{T_f}\left[\left(L(\mathbf{y}_k, \mathbf{a}_k) + \gamma V^{\boldsymbol{\nu}}\left(\mathbf{s}_{k+1}^\star\right) - V^{\boldsymbol{\nu}}\left(\mathbf{s}_k^\star\right)\right)\boldsymbol{\Psi}(\mathbf{s}_k^\star, \mathbf{a}_k)\right]\right],$$

$$b_{\boldsymbol{\theta}} = E_m\left[\sum_{k=1}^{T_f}\Xi\Xi^\top\mathbf{w}\right] \tag{4.79e}$$

where T_f is the final time instant at the end of each episode.

4.6 Simulation Results

In this section, we illustrate the performance of the proposed learning-based control and estimation algorithm to deal with three types of problems. In the first test case, we consider a linear system evaluating a model mismatch problem where the MHE model in a combined MHE-MPC scheme is wrong and cannot capture the real system. In the second test case, we show that the proposed framework achieves a better closed-loop performance for the control of systems using inaccurate models where a reduced model is used for both the control and estimation

goals. We implement our algorithm for a smart building in order to maintain the room temperature in its comfort range even if there is no sufficient knowledge about the building dynamics. Finally, we investigate the proposed learning-based framework applied to a Continuous Stirred Tank Reactor (CSTR) as an example with nonlinear dynamics.

4.6.1 Test Case 1

In this case study, we consider a model mismatch upon the MHE and evaluate a set-point tracking using an MHE-MPC for a two states linear system $\mathbf{x}_{k+1} = A\mathbf{x}_k + Bu_k$ where x_1 is selected as measurement. The real system and MPC model are chosen as:

$$A = \begin{bmatrix} 1 & 0.25 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0312 \\ 0.25 \end{bmatrix}, \tag{4.80}$$

while the MHE model is selected as:

$$\hat{\mathbf{x}}_{k+1} = \begin{bmatrix} 0.9 & 0.35 \\ 0 & 1.1 \end{bmatrix} \hat{\mathbf{x}}_k + \begin{bmatrix} 0.0813 \\ 0.2 \end{bmatrix} u_k \tag{4.81}$$

We then use the MHE scheme (4.59) where the arrival cost is adjusted based on the updating rule (4.52) and the stage cost is approximated based on the parameterization (4.54). The input convex NN has two hidden layers with 15 neurons and both the MHE and MPC horizons are set to 8. We use a smooth version of ReLU as an activation function g_j in ICNN (4.56).

$$g_j(x) = \log\left(1 + \exp(x)\right) \tag{4.82}$$

Note that in this example only the MHE scheme is learned by RL and the MPC scheme is not parameterized. Figure 4.2 shows that the model mismatch on the MHE scheme can affect both the estimation performance and the set-point tracking performance. Indeed, the MHE model mismatch causes a large estimation error on x_2 and the set-point 0.8 on x_1 cannot be tracked. As it is shown in Figure 4.3, the mentioned problems due to model mismatch have been solved and a correct state estimation is delivered where the proposed modification of the MHE cost is implemented. Figure 4.4 shows the learning progress, including the system states x_1, x_2 and their estimations during 60 RL steps such that the closed-loop performance $J(\boldsymbol{\pi}_{\boldsymbol{\theta}})$ is improved by the MHE cost modification, and the correct state estimations shown in Figure 4.3 are delivered.

4.6.2 Test Case 2

Building Model

Let us select a model of the real system of a house floor heating system connected to a ground source-based heat pump shown in Figure 4.5. We consider a dynamical

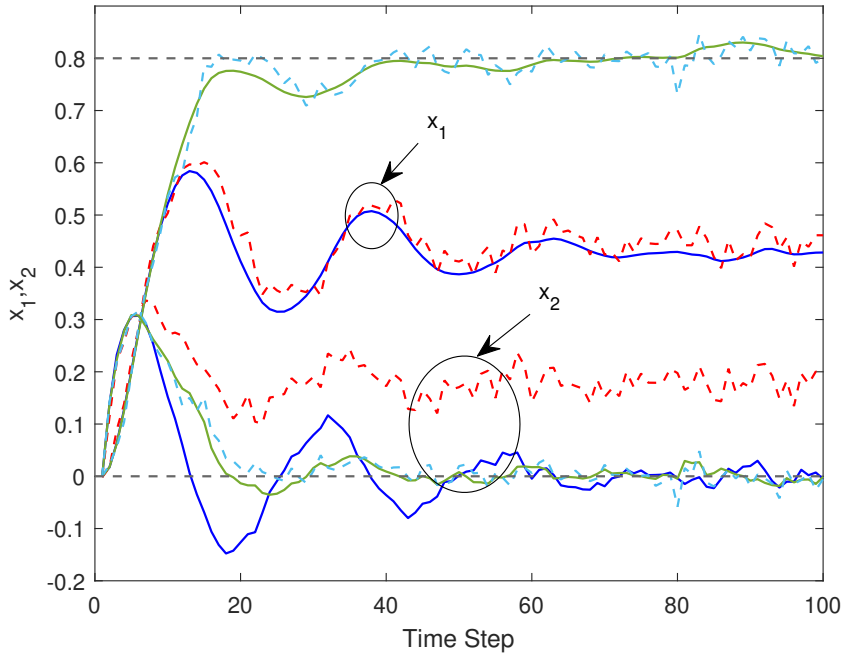


Figure 4.2: Test Case 1: Real system behavior and state estimations for a set-point tracking ($x_{d_1} = 0.8$ and $x_{d_2} = 0$) in the presence of the model mismatch on the MHE. The solid lines of blue color indicate the states while the estimations are indicated as dashed lines of red color. The correct states and estimations without model mismatch are shown in green.

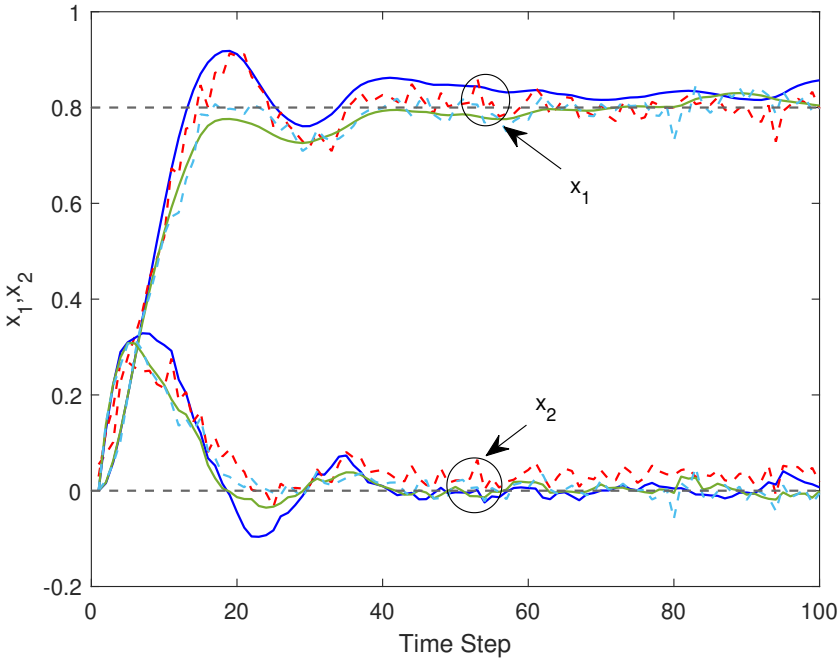


Figure 4.3: Test Case 1: Real system behavior and state estimations for a set-point tracking ($x_{d_1} = 0.8$ and $x_{d_2} = 0$) where the MHE scheme is modified. The solid lines of blue color indicate the states while the estimations are indicated as dashed lines of red color. The correct states and estimations without model mismatch are shown in green.

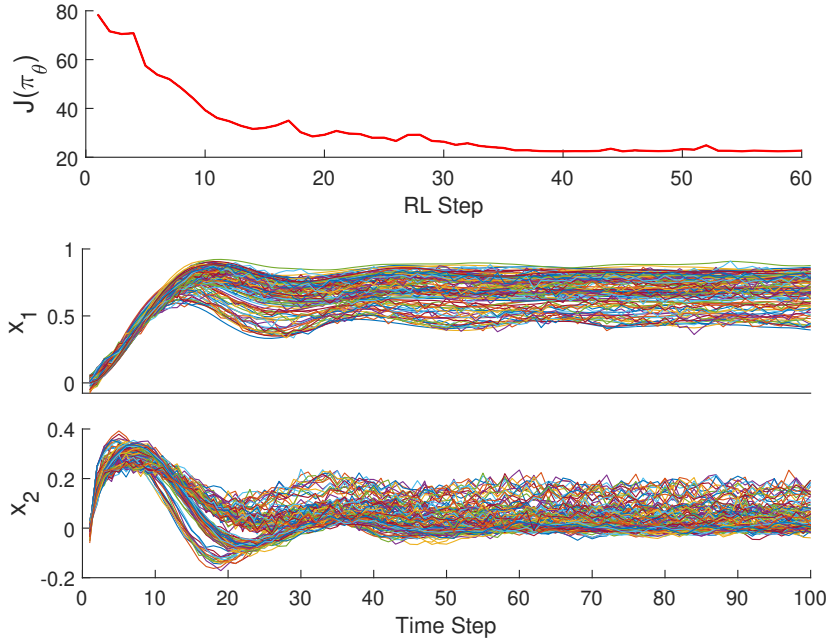


Figure 4.4: Test Case 1: Closed-loop performance and evolution of states and their estimations during reinforcement learning.

model with four states for the building as the real system under control, which is described by a set of ordinary differential equations as follows [60]:

$$C_{wa}\frac{dT_{wa}}{dt} = K_{wa,a}\left(T_a - T_{wa}\right) + K_{wa,r}\left(T_r - T_{wa}\right) \tag{4.83a}$$

$$C_r\frac{dT_r}{dt} = K_{wa,r}\left(T_{wa} - T_r\right) + K_{f,r}\left(T_f - T_r\right) \tag{4.83b}$$

$$C_f\frac{dT_f}{dt} = K_{f,r}\left(T_r - T_f\right) + K_b\left(T_w - T_f\right) \tag{4.83c}$$

$$C_w\frac{dT_w}{dt} = K_b\left(T_f - T_w\right) + \eta W_c \tag{4.83d}$$

where the control input $u = W_c$ is the power used by the heat pump. The states of the real system $\mathbf{x}^r = [T_{wa}, T_r, T_f, T_w]^\top$ are labeled the wall temperature, the room temperature, the floor (pavement) temperature, and the water pipeline temperature, respectively. The coefficients C_{wa}, C_r, C_f, and C_w read as the corresponding heat capacities of the above-mentioned temperatures. We label $K_{wa,a}$, $K_{wa,r}$, $K_{f,r}$ and K_b the overall heat transfer coefficients between the $\{T_{wa}, T_a\}$ wall-ambient, $\{T_{wa}, T_r\}$ wall-room, $\{T_f, T_r\}$ floor-room and $\{T_f, T_w\}$ floor-water pipeline, respectively. The Coefficient of Performance (COP) η for heat pumps varies with

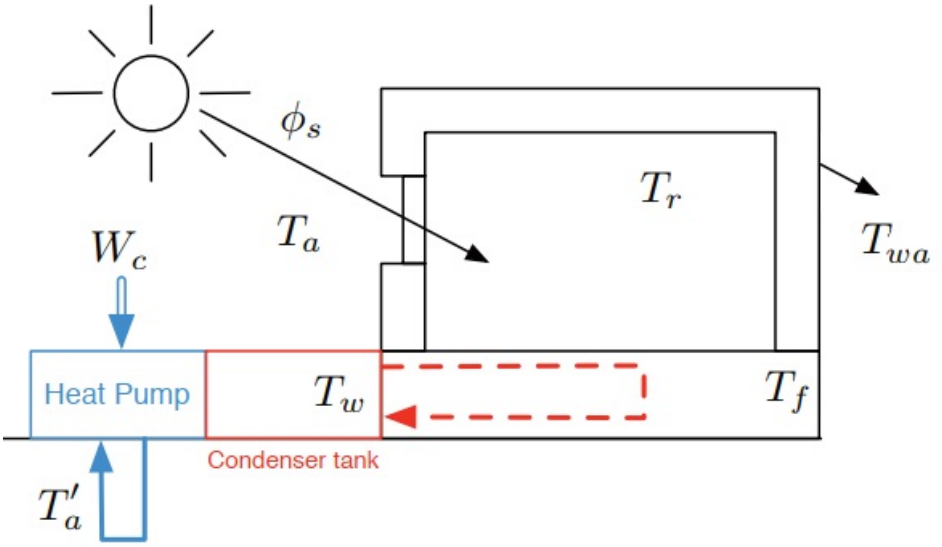


Figure 4.5: Test Case 2: Building climate control [1] using a heat pump floor heating system. The dashed line represents the floor heating pipelines.

type, outdoor ground temperature, and condenser temperature. In this chapter, we

then adopt a stochastic COP shown in Figure 4.6 to make the simulations more realistic. To implement a POMDP scenario, we assume that the building dynam-

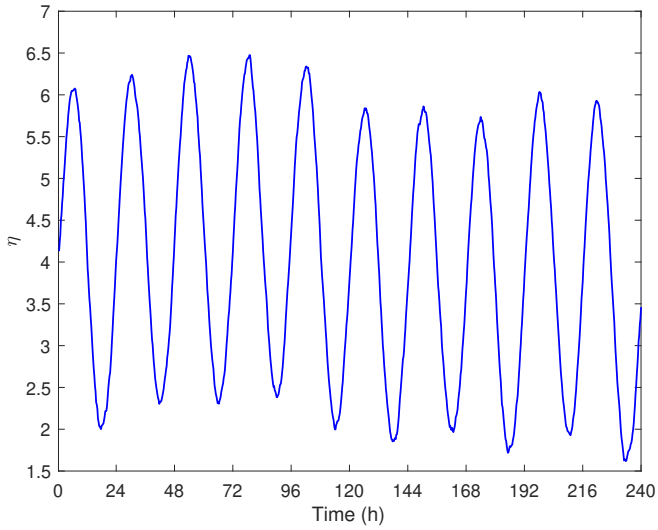


Figure 4.6: Test Case 2: Stochastic COP of heat pump sampled from the last RL step.

ics can be modeled by a reduced model considering the room and water pipeline temperatures (T_r, T_w) as the only measurable states used in the state-space model. Hence, the dynamics of wall inertia and floor are removed from the real system (4.83), and then a partially observable model with two states is adopted for both the MHE and MPC schemes. To reduce the order of a state-space model captured from the real model of the building (4.83), we propose to use ¨*modred*¨ with option ¨*MatchDC*¨ as a built-in function in MATLAB. By eliminating the states T_{wa}, T_f from the real system, the frequency response of the reduced model is affected so that it can no longer follow the real response shown in Figure 4.7.

The parameters of the building model adopted in this simulation are given in the following table.

Table 4.1: Building Parameters

C_{wa}	$24.2 \times 10^6[\frac{J}{K}]$	$K_{wa,a}$	$56[\frac{W}{K}]$
C_r	$6 \times 10^6[\frac{J}{K}]$	$K_{wa,r}$	$386[\frac{W}{K}]$
C_f	$24.8 \times 10^6[\frac{J}{K}]$	$K_{f,r}$	$594[\frac{W}{K}]$
C_w	$20.7 \times 10^6[\frac{J}{K}]$	K_b	$506[\frac{W}{K}]$

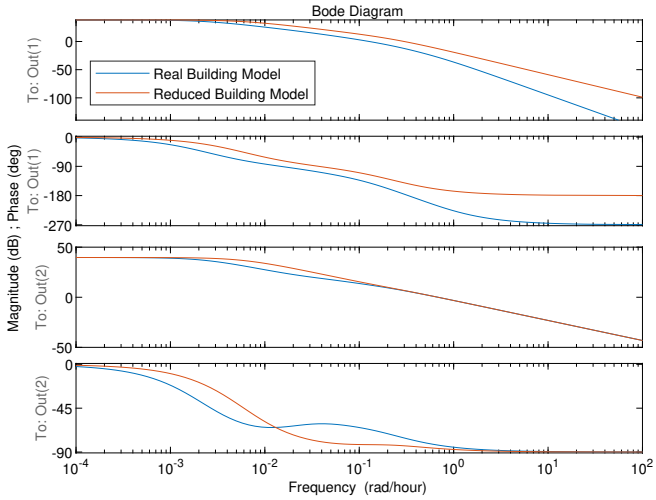


Figure 4.7: Test Case 2: Bode plot of the frequency response.

Simulation Settings

As we propose to adopt a reduced model of the real system (4.83) as a POMDP scenario, we label $\mathbf{x}^m = [T_r, T_w]^\top$ the model states (measurements) used in both the MHE and MPC schemes. We then use a parameterized MHE scheme as (4.59) to estimate the model states from the noisy measurements $\bar{\mathbf{y}} = \mathbf{x}^m$. The stage cost $\hat{L}_{\boldsymbol{\theta}}\left(\hat{\mathbf{s}}_i, \mathbf{o}_i^H\right)$ in this MHE scheme consists of two cost terms expressed in (4.54) so that $L_{\text{NN}}\left(\mathbf{Y}_i, \boldsymbol{\theta}_{\text{NN}}\right)$ is approximated using an input convex neural network defined in (4.56). This NN consists of two hidden layers and each layer has 26 neurons where a smooth version of ReLU is used. The cost term $L_{\boldsymbol{\theta}_0}$ is selected as a least square problem parameterized as follows:

$$L_{\boldsymbol{\theta}_0} = \|\bar{\mathbf{y}}_i - \mathbf{h}\left(\hat{\mathbf{x}}_i^m\right)\|_{Q_{\boldsymbol{\theta}}}^2 + \mathcal{G}_{\boldsymbol{\theta}}^\top \hat{\mathbf{x}}_i^m \tag{4.84}$$

Note that the second term in the cost above reads as a gradient modification term. The adjustable weighting matrix $Q_{\boldsymbol{\theta}}$ in the equation above is tuned using RL. As a requirement, this weighting matrix must be symmetric and positive semidefinite. However, the RL steps delivered by the LSTD-based DPG do not necessarily respect this requirement, and we need to enforce it via constraints on the RL steps throughout the learning process. To address this requirement, we then formulate a Semidefinite Programming (SDP) as the following least squares optimization

problem:

$$\min_{\Delta\boldsymbol{\theta}} \quad \frac{1}{2}\left\|\Delta\boldsymbol{\theta}\right\|^2 - d^\top \Delta\boldsymbol{\theta} \tag{4.85a}$$

$$\text{s.t.} \quad Q_{\boldsymbol{\theta}}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) \geq 0, \tag{4.85b}$$

$$W^{(z)}_{1:l-1} \geq 0 \tag{4.85c}$$

where $\boldsymbol{\theta} = \{Q_\theta, W^{(z)}_{1:l-1}\}$ and $d = -\alpha b_{\boldsymbol{\theta}}$. We assume that the weighting matrix $Q_{\boldsymbol{\theta}}$ is a linear function of $\boldsymbol{\theta}$. Then, it is updated at every RL step (epoch) due to updating $\Delta\boldsymbol{\theta}$, which is a solution of the above SDP scheme. Note that the second term in the objective function (4.85a) ensures that $\Delta\boldsymbol{\theta}$ is obtained in the direction of the policy gradient at every RL step.

To keep the room temperature in a comfortable range, we formulate an economic MPC scheme as follows:

$$\begin{aligned}\min_{\mathbf{x}^m,u,\sigma} \quad & \gamma^{N_{\text{MPC}}}\left(w_f \sigma_{k+N_{\text{MPC}}}\right) \\ & + \sum_{i=k}^{k+N_{\text{MPC}}-1} \gamma^{i-k}\left(p_{u,i}u_i + w\sigma_i\right) && (4.86a)\end{aligned}$$

$$\text{s.t.} \quad \mathbf{x}^m_{i+1} = \hat{\mathbf{f}}^{\text{MPC}}(\mathbf{x}^m_i, u_i), \tag{4.86b}$$

$$\mathbf{x}^m_k = \hat{\mathbf{x}}^{\star,m}_k, \tag{4.86c}$$

$$\underline{\theta} + T^{\min}_{r,i} - \sigma_i \leq T_{r,i} \leq \bar{\theta} + T^{\max}_{r,i} + \sigma_i, \tag{4.86d}$$

$$\Delta u_{\min} \leq \Delta u_i \leq \Delta u_{\max}, \tag{4.86e}$$

$$u_{\min} \leq u_i \leq u_{\max}, \tag{4.86f}$$

$$\sigma_{k,\ldots,k+N_{\text{MPC}}} \geq 0 \tag{4.86g}$$

where $\hat{\mathbf{x}}^{\star,m}_k$ is the current state estimation delivered by the approximate MHE scheme, p_u is the cost coefficient for the electricity prices, and $\hat{\mathbf{f}}^{\text{MPC}}$ is captured from a model reduction approach. To adjust the constraints upon the room temperature, we consider two parameters $(\underline{\theta}, \bar{\theta})$ and let RL tune them. As a result of the theorems developed in this chapter, we propose to modify the stage cost of the MHE scheme with a reduced model to tackle POMDPs. To that end, we let RL adjust the NN weights $\boldsymbol{\theta}_{\text{NN}}$ and some parameters of the first stage cost term $L_{\boldsymbol{\theta}_0}$, including inverse of the covariance matrix $Q_{\boldsymbol{\theta}}$ and gradient term $\mathcal{G}_{\boldsymbol{\theta}}$ in (4.84). Hence, all RL parameters $\boldsymbol{\theta} = \left\{\boldsymbol{\theta}_{\text{NN}}, Q_{\boldsymbol{\theta}}, \mathcal{G}_{\boldsymbol{\theta}}, \underline{\theta}, \bar{\theta}\right\}$ are adjusted by the proposed LSTD-based DPG reinforcement learning. We adopt a baseline stage cost in the proposed LSTD-based RL algorithm as follows:

$$L(y_k, a_k) = p_{u,k}.a_k + w.\max\left(0, h\left(T_{r,k}\right)\right) \tag{4.87}$$

where $a_k = \pi_{\boldsymbol{\theta}}\left(\hat{\mathbf{x}}_k^{\star,m}\right) = u_0^{\star}\left(\hat{\mathbf{x}}_k^{\star,m}, \boldsymbol{\theta}\right)$ with the possible addition of occasional random exploratory moves. Note that $\mathbf{u}_0^{\star}$ is the first element of the control input sequence $\mathbf{u}^{\star}$ delivered by the MPC scheme (4.86). We use the weight $w = 100$ where $h\left(T_{r,k}\right)$ collects the inequality constraints upon indoor temperature $T_{r,k}^{\min} \leq T_{r,k} \leq T_{r,k}^{\max}$.

We choose a sampling time $15min$ and a forecast $24h$ for the ambient disturbances and electricity prices. Therefore, the prediction and estimation horizons $(N_{\text{MPC}}, N_{\text{MHE}})$ are set to 96. The ambient temperature and electricity prices are forecasted for 10 days starting from the first day of January 2021 in Trondheim, Norway where the data used in this simulation is provided by Nord Pool Spot as an electricity market operator.

Discussion

In practice, it is very difficult to make an accurate model of a building for the model-based control approaches, i.e., an MPC scheme since there are some complex dynamics and uncertainties that may not be captured. To address this complexity, a common solution is to adopt some simplified and reduced models in this context. Although these simplified models are useful to be used in an MPC scheme in order to reduce computational complexity, they can affect the control performance in a building climate control system. In this simulation, we use a super-simplified and realistic building model where its dynamics include only two measurable states T_r, T_w while the aim is to control the indoor temperature in a real model (4.83). As it is shown in Figure 4.8, the first evolution (No learning is used) of T_r in blue color cannot perfectly respect the lower variable constraint and there is a heavy violation since the model is not truly captured.

The evolution of estimated T_r is depicted in Figure 4.9 in red color and it can be observed that the first evolution of this estimation is not able to follow the first evolution of the real T_r in blue color. This estimation error, where there is still no adopted learning mechanism upon MHE and MPC, can be clearly observed in Figure 4.11. To address these problems induced by adopting a reduced model, we let an LSTD-based DPG reinforcement learning adjust the parameters of both the MHE (cost modification) and MPC (constraint adjustment) schemes shown in Figure 4.10 in order to capture a correct state estimation and deliver a learned policy to tackle this model inaccuracy.

To conclude the proposed learning-based state estimation and control, we can observe that the proposed theorem of cost modification in an MHE scheme with imperfect model works since we achieve a perfect closed-loop performance by applying that theorem in order to modify the MHE cost depicted in Figure 4.10. It is

worth noting that, the learned policy is optimally captured from the MPC scheme so that the heat pump power has its highest value in lower electricity prices and it has a minimum peak for times that the electricity is expensive shown in Figure 4.8.

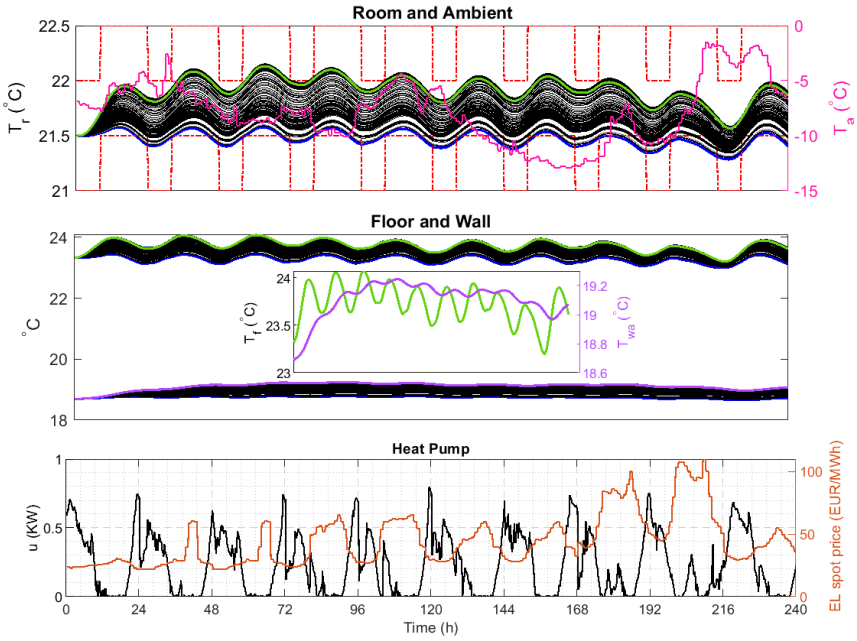


Figure 4.8: Test Case 2: Evolution of the building temperatures T_r, T_f, T_{wa} (black color) and trained optimal policy u where both the estimator (MHE) and controller (MPC) use an imperfect model. The comfort T_r is captured (green color) after 185 learning steps (epoch) for the adjustment of MHE and MPC schemes.

4.6.3 Test Case 3

CSTR Nonlinear Model

In this section, the proposed learning-based MHE-MPC scheme is applied to a Continuous Stirred Tank Reactor (CSTR), where the dynamical system is nonlinear and may not be modeled accurately. In this chemical reactor, the reaction $(A \rightarrow B)$ is accomplished by means of an irreversible and exothermic chemical reaction, and the aim is to control the concentration of A, C_a, and the reaction volume, V, by manipulating the output process flow rate, q_s, and the coolant flow

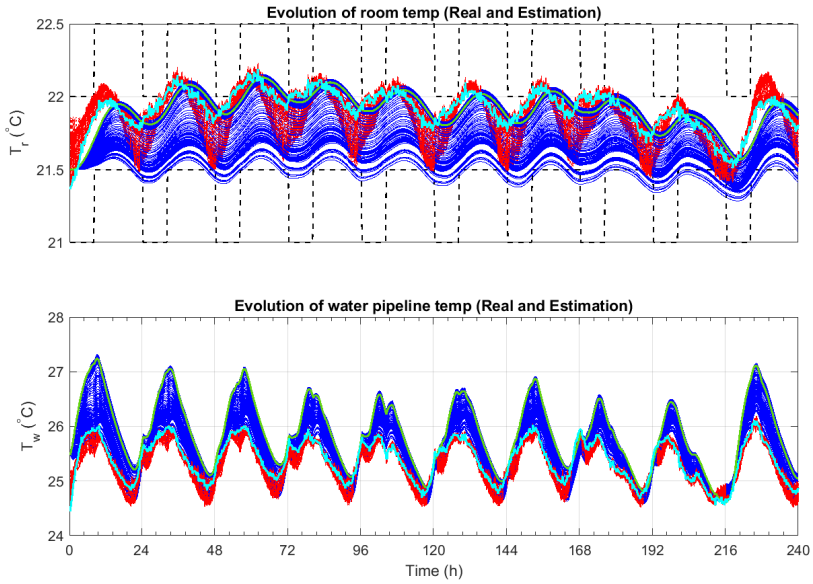


Figure 4.9: Test Case 2: Evolution of the real states as measurements (temperatures T_r, T_w in blue color) and their estimations (red color) used in the inaccurate models of MHE and MPC as POMDPs. The estimations in light blue color are captured from the trained MHE estimator.

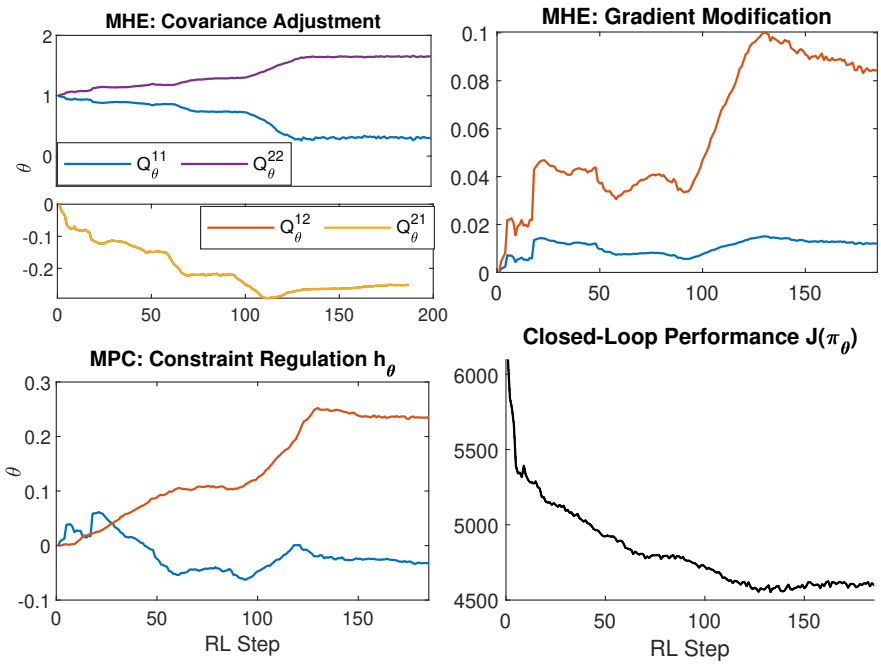


Figure 4.10: Test Case 2: Some parameters of the MHE/MPC and the closed-loop performance $J(\pi_\theta)$ over reinforcement learning steps.

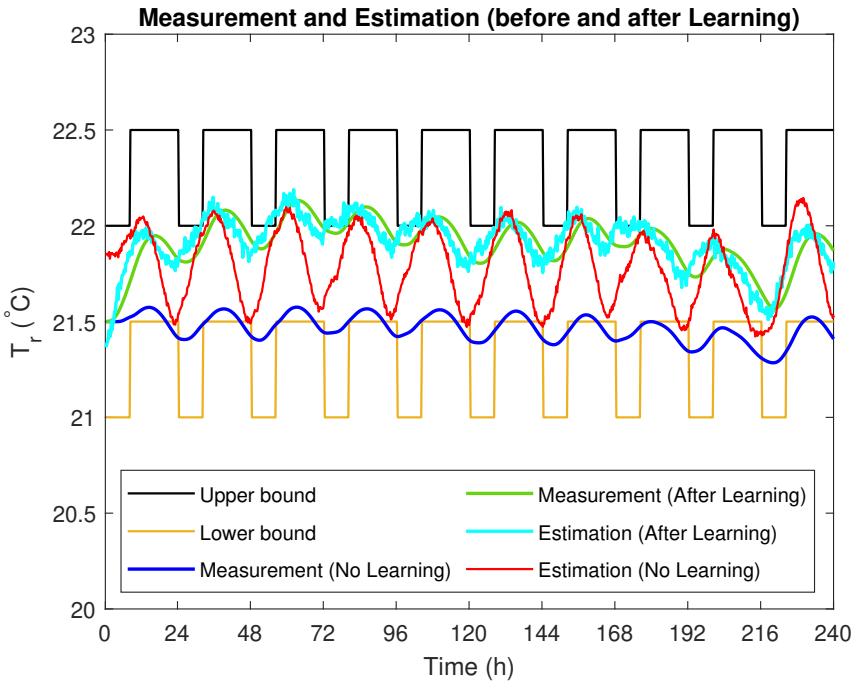


Figure 4.11: Test Case 2: The building indoor temperature before and after learning the estimator and controller.

rate, q_c, see [61]. The CSTR dynamics are described as follows:

$$\dot{V}(t) = q_o - q_s(t), \tag{4.88a}$$

$$\dot{C}_a(t) = \frac{q_o}{V(t)}\left(C_{a_o} - C_a(t)\right) - k_0 \mathrm{e}^{\frac{-E}{RT(t)}} C_a(t), \tag{4.88b}$$

$$\dot{T}(t) = \frac{q_o}{V(t)}\left(T_o - T(t)\right) + k_1 \mathrm{e}^{\frac{-E}{RT(t)}} C_a(t) + k_2 \frac{q_c(t)}{V(t)}\left(1 - \mathrm{e}^{\frac{-k_3}{q_c(t)}}\right)\left(T_{co} - T(t)\right), \tag{4.88c}$$

$$k_1 = \frac{-\Delta H k_0}{\rho C_p}, \quad k_2 = \frac{\rho_c C_{pc}}{\rho C_p}, \quad k_3 = \frac{hA}{\rho_c C_{pc}} \tag{4.88d}$$

where $V(t), C_a(t), T(t)$ are the reaction volume, the concentration of A, and the reactor temperature, respectively. We consider $V(t)$ and $T(t)$ as measurable states since it is not usually easy to measure the concentration C_a directly. The measurement noises are selected as $\mathcal{N}(0, Q)$ with $Q = \mathrm{diag}(2.5^2, 2.5^2)$. The control inputs are q_s and q_c. The constant parameters given in Table 2 are the process flow rate q_o, the feed concentration C_{a_o}, the reaction rate k_0, the activation energy term E/R, the feed temperature T_o, the inlet coolant temperature T_{co}, the heat of reaction ΔH, the heat transfer term hA, the liquid densities ρ, ρ_c and the specific heats C_p, C_{pc}. To investigate the performance of the proposed modification of the MHE

Table 4.2: CSTR Model Parameters

q_o	$100[\frac{l}{min}]$	C_{a_o}	$1[\frac{mol}{l}]$
T_o	$350[K]$	T_{co}	$350[K]$
ΔH	$-2 \times 10^5[\frac{cal}{mol}]$	ρC_p	$1000[\frac{cal}{lK}]$
k_0	$7.2 \times 10^{10}[\frac{1}{min}]$	E/R	$1 \times 10^4[K]$
$\rho_c C_{pc}$	$1000[\frac{cal}{lK}]$	hA	$7 \times 10^5[\frac{cal}{minK}]$

scheme, we adopt the correct model (4.88) in the MPC scheme while the MHE scheme is formulated using an imperfect model of the real system as follows:

$$\dot{V}(t) = q_o - q_s(t), \tag{4.89a}$$

$$\dot{C}_a(t) = 0.93\frac{q_o}{V(t)}\left(C_{a_o} - C_a(t)\right) - 1.2k_0 \mathrm{e}^{\frac{-E}{RT(t)}} C_a(t), \tag{4.89b}$$

$$\dot{T}(t) = 0.93\frac{q_o}{V(t)}\left(T_o - T(t)\right) + 1.3k_1 \mathrm{e}^{\frac{-E}{RT(t)}} C_a(t) + 0.8k_2 \frac{q_c(t)}{V(t)}\left(1 - \mathrm{e}^{\frac{-0.8k_3}{q_c(t)}}\right)\left(T_{co} - T(t)\right) \tag{4.89c}$$

The constraints on the states and control inputs are $90 \leq V \leq 110$, $0 \leq C_a \leq 0.35$, $400 \leq T \leq 480$, $80 \leq q_s \leq 120$ and $75 \leq q_c \leq 140$.

Simulation Settings

In this simulation, both the modification step H and the horizon N are set to 10. The number of neurons in the hidden layers of the ICNN is set to 18, and we consider a sampling time 0.1min. In the reinforcement learning setting, the parameterized MHE scheme is adjusted during 800 episodes (RL steps), where each episode includes a 4min (40 time steps) of running the real system. Hence, in this simulation scenario, we use a total of 3.2×10^4 learning samples to modify the MHE scheme with the imperfect model (4.89), and improve the closed-loop performance. Note that the initial conditions are randomly selected for episodes of length 40. The set-point tracking goal is set as $V^d = 105\ l$, $C_a^d = 0.12\ mol/l$, $T^d = 433.72\ K$, $q_s^d = 100\ l/min$ and $q_c^d = 110\ l/min$.

Discussion

Figure 4.12 depicts the evolution of the system states and their estimation during the learning progress. As it is observed, the correct state estimations (orange circles) are delivered after 800 RL steps so that the system states (blue lines) can track the corresponding set points accurately. Figure 4.13 shows the evolution of the control inputs in black color during the learning progress, and it is observed that the optimal control inputs shown as stairs in blue color track the corresponding set-points while the constraints are guaranteed. The results depicted in Figure 4.14 provide a comparative analysis between the proposed learning-based MHE-MPC and one without learning. It can be observed that the system states in black color are struggling to track the references since the imperfect model (4.89) is used in the MHE scheme, and the wrong estimation is delivered, shown as blue circles. The proposed learning-based modification of the MHE scheme then adjusts the MHE stage cost function so that the state estimations (red circles) perfectly match the correct estimations. Note that the correct estimations are those captured from the MHE scheme with the perfect model (4.88). The closed-loop performance $J(\pi_\theta)$ is illustrated in Figure 4.15, and it is observed that the best performance is achieved after 500 episodes, and the norm of policy gradient steps $\nabla_\theta J$ moves towards zero since the policy parameters converge to the optimal parameters.

4.6.4 Computation time

To investigate the computation time of solving the proposed modified MHE-MPC scheme, the choice of the modification step H and the number of neuron used in the hidden layers of ICNN in the modified MHE scheme are regarded as crucial issues since they determine the number of the parameters required in the proposed modification. Although, the horizon H could be larger than N to approximate the modified stage cost accurately, we may choose a small size of H even smaller than

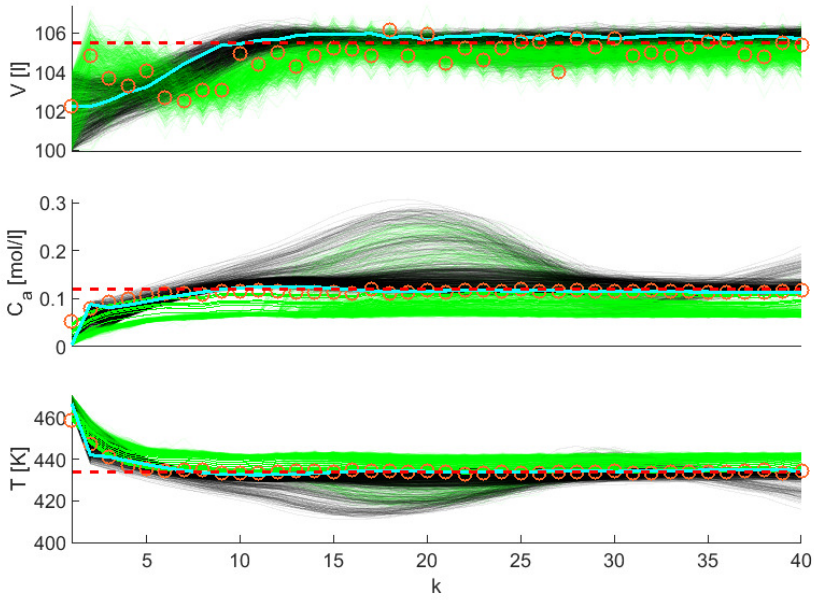


Figure 4.12: Test Case 3: Evolution of the CSTR states and their estimations during the learning progress. The system states and their estimations at the learning stage are shown as black and green lines, respectively. The orange circles and blue lines, respectively, represent the correct state estimations and the system states delivered after 800 RL steps. The set points are shown as red dashed lines.

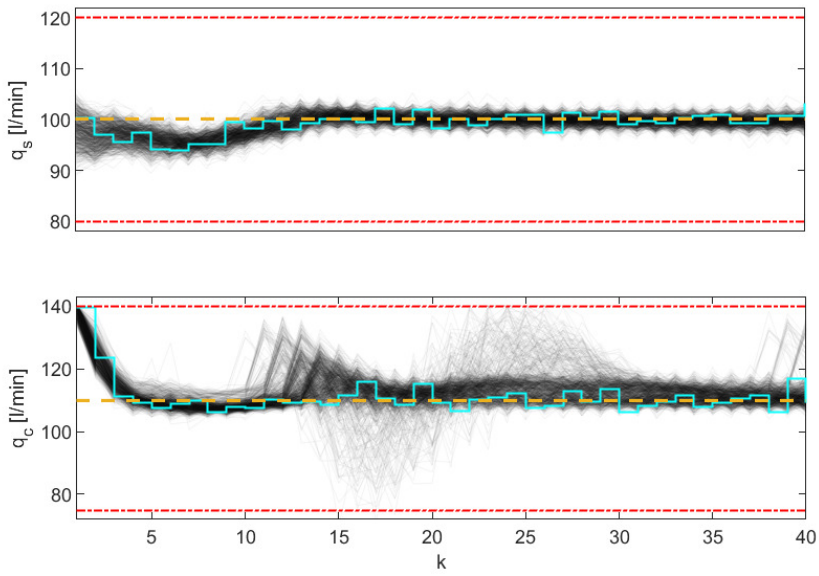


Figure 4.13: Test Case 3: Evolution of the CSTR control inputs during the learning progress. The control inputs at the learning stage are shown as black lines while the optimal control inputs delivered from the MHE-MPC after 800 RL steps are shown as blue stairs. The constraints and set points are shown as red-dashed and yellow-dashed lines, respectively.

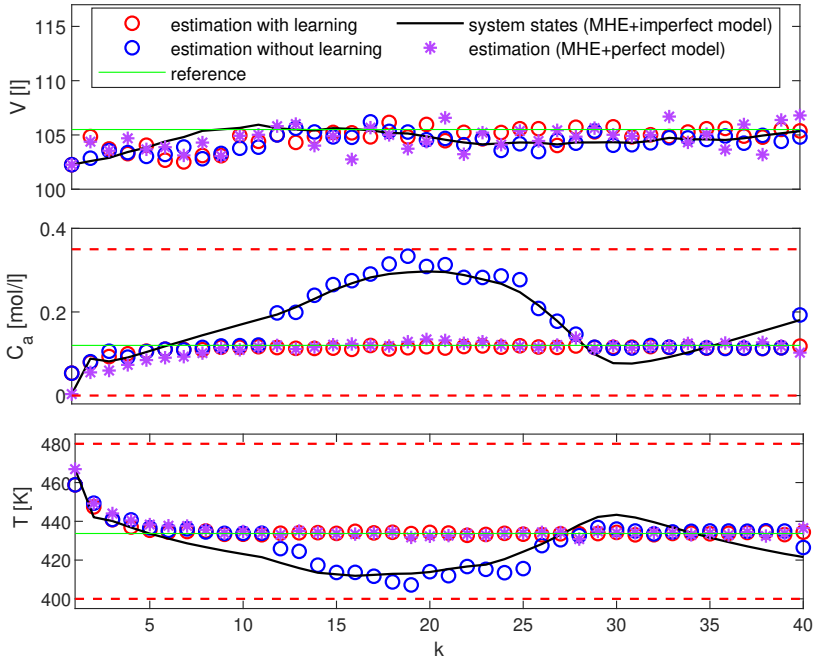


Figure 4.14: Test Case 3: Comparative analysis between the conventional MHE-MPC with an imperfect model and learning-based MHE-MPC.

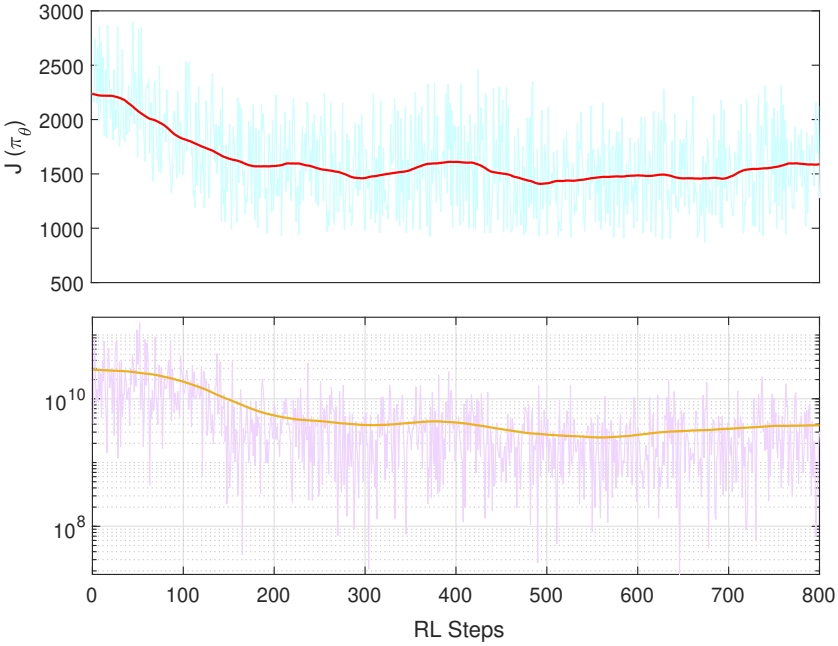


Figure 4.15: Test Case 3: Closed-loop performance and the norm of the policy gradient steps $||\nabla_\theta J(\pi_\theta)||_2$ in logarithmic scale. The noisy closed-loop performance is shown as a blue line while its moving average is shown as a red line. The noisy logarithmic scale of $||\nabla_\theta J(\pi_\theta)||_2$ is shown as a purple line while its moving average is shown as a yellow line.

N in order to provide an acceptable trade-off between the computational effort and the approximate value captured from the NN. To mitigate the computational efforts for all three numerical examples, the horizon length of the modification step H has been set to the same value as the prediction/estimation horizon N. Fortunately, the training stage of the proposed MHE/MPC-based RL can be accomplished offline, and it is worth mentioning that the MPC parameterization is quite flexible so that the MPC cost can be parameterized from a numerical perspective that makes the MPC implementation as tractable and effective as possible. However, the MHE/MPC-based RL combined with NN in the loop may struggle a bit in the real-time applications, in particular those cases with very small sampling times and large estimation/prediction horizons. Nonetheless, the progress in the optimization algorithms and in the computational hardware makes the deployment of real-time MHE/MPC possible for most of the real applications. The computation times for three test cases above are provided in the following table. Note that we do not use real-time solvers in the present chapter.

Table 4.3: Computation time

Time	Baseline MHE-MPC	Modified MHE-MPC
Test Case 1	0.89 sec	1.68 sec
Test Case 2	10.2 min	19.47 min
Test Case 3	1.23 sec	2.15 sec

4.7 Conclusion

In this chapter, we have shown how an MHE scheme can be modified such that its performance degradation due to using an imperfect MHE model is tackled. The stage cost modification in both versions of the stochastic and deterministic MHE schemes is proposed so that a correct probability measure and state estimation can be delivered even if the underlying model cannot capture the real system. A practical implementation of the proposed approach upon the MHE cost modification is discussed. To achieve the best closed-loop performance for a combined MHE-MPC scheme using an imperfect model of the real system, we detail a parameterization method for both the deterministic MHE and MPC schemes and develop an MHE/MPC-based policy gradient reinforcement learning algorithm. The effectiveness of the proposed learning-based estimator/controller has been established for three examples, including a model mismatch problem, a climate control of smart building where the building model used in the MHE-MPC is simplified and different from the real dynamics, and a CSTR estimation/control problem. Further work will aim to implement a modified stochastic MHE scheme proposed in this chapter.

Chapter 5

Policy Gradient Reinforcement Learning for Uncertain Polytopic LPV Systems based on MHE-MPC

- H.N. Esfahani, S. Gros, "Policy gradient reinforcement learning for uncertain polytopic LPV systems based on MHE-MPC," IFAC-PapersOnLine 55 (15) (2022) 1–6. 6th IFAC Conference on Intelligent Control and Automation Sciences, ICONS 2022.

In this chapter, we propose a learning-based Model Predictive Control (MPC) approach for the polytopic Linear Parameter-Varying (LPV) systems with inexact scheduling parameters (as exogenous signals with inexact bounds), where the Linear Time Invariant (LTI) models (vertices) captured by combinations of the scheduling parameters becomes wrong. We first propose to adopt a Moving Horizon Estimation (MHE) scheme to simultaneously estimate the convex combination vector and unmeasured states based on the observations and model matching error. To tackle the wrong LTI models used in both the MPC and MHE schemes, we then adopt a Policy Gradient (PG) Reinforcement Learning (RL) to learn both the estimator (MHE) and controller (MPC) so that the best closed-loop performance is achieved. The effectiveness of the proposed RL-based MHE/MPC design is demonstrated using an illustrative example.

5.1 Introduction

Model predictive control (MPC) is an advanced control approach and widely used to many control problems due to its capability to handle both input and state constraints by making explicit use of the process model to predict the future evolution of the system [9]. However, in most applications some of the model parameters are uncertain at design time so that the prediction model may not be accurate enough to capture the true dynamics of a real system that this issue can affect the closed-loop performance [62].

In many real applications of MPC, an accurate model of the (possibly nonlinear) real system may not be available so that the simplified linear models are commonly adopted as the prediction models, which are captured at a specific operation point of the real system. However, these identified models may not work for some uncertain systems where some parameters of the dynamics vary or not be known perfectly at different operation points. Therefore, the Linear Parameter-Varying (LPV) framework is usually adopted to address the issue above in which the linear models with parametric uncertainties (i.e, exogenous scheduling parameters) are represented by the LPV models [63].

The use of polytopic LPV prediction models has been proven to be an effective approach to develop the MPC schemes for both the linear and nonlinear systems, e.g., see [64, 65, 66]. In most of the MPC-based polytopic LPVs, a multi-model approach is used to represent a convex combination of the vertices as Linear Time Invariant (LTI) models in which the uncertainty (exogenous/endogenous scheduling parameter with known bounds) is associated to the convex combination vector. However, if the scheduling parameters are not known and constantly change, the robust MPC schemes, i.e, a tube MPC can be used [67].

Several solutions were proposed to incorporate the LPV model scheduling parameter dynamics within the MPC-LPV prediction, e.g. by the analysis of parameters' rate or by a parameter prediction policy [68, 69]. Because of the unavailability of the future scheduling parameters, the MPC design can be challenging since the recursive feasibility and closed-loop stability of the moving horizon requires an MPC which is robust against all possible future scheduling variations [70]. To address this issue, authors in [71] proposed to benefit from a recursive extrapolation method in order to estimate the future trajectory of the quasi-LPV (qLPV) scheduling parameters. In [72], authors modeled the uncertainties that arise due to the unavailability of the scheduling trajectory along the prediction horizon as a bounded interconnection in the form of a Linear Fractional Transformation (LFT). An optimal online tuning scheme for design-related scheduling parameters of adaptive LPV systems was proposed in [73], where an MPC scheme was leveraged to generate the optimal sequence of the scheduling parameters. In [74], authors proposed a data-driven LPV predictor based on the behavioral system theory in order to generate the future scheduling parameters. The proposed predictor only requires reasonably short recorded trajectories of the system input, output and scheduling signals, to represent the complete dynamic behavior of the system. The MPC-LPV scheme then exploits this predictor model as an equality constraint.

As an alternative approach, as we consider in this chapter, the combination vector of the scheduling parameters is assumed to be constant or changing slowly, which can be estimated by a parameter estimation scheme [75, 76]. As an interesting method to concurrently estimate this vector and some unmeasured states, a Moving Horizon Estimation (MHE) was adopted in [77]. However, the LTI models used as vertices of the polytopic LPV may not be exact due to the inexact selection of the scheduling parameter bounds and then, these wrong models used in both the MHE and MPC schemes can decrease the closed-loop performance. As a solution to address this problem, a learning mechanism, i.e, a Reinforcement Learning (RL) can be leveraged to learn these schemes with the aim of increasing the closed-loop performance.

Reinforcement Learning (RL) is a powerful tool for solving Markov Decision Processes (MDP) problems [11], which has been recently combined with the MPC and MHE schemes in order to improve their performance where the adopted prediction/estimation models may not capture the real system dynamics perfectly, e.g., see [21, 40, 23].

Motivated by this observation in the context of the MPC-based RL, in this chapter, we propose to use an MHE scheme to deliver the convex combination vector needed in the MPC scheme and deal with the polytopic LPV systems with possibly inexact vertices (wrong LTI models) by adopting a Deterministic Policy Gradi-

ent (DPG) RL to adjust a certain number of the parameters in the parameterized MHE and MPC schemes. It is worth highlighting that the proposed *compatible deterministic policy gradient reinforcement learning based on a combined MHE-MPC scheme* is a central contribution in this chapter. Although this learning-based (state-parameter) estimator/controller scheme is developed and employed in the context of the LPV systems in this chapter, the proposed algorithm can be readily used to control many other applications, i.e, partially observable systems in which a combined estimator/observer-MPC is needed.

It is noted that the proposed MHE-based MPC-LPV is not a pure data-driven MPC-LPV as we do not exploit any identification method to fit the model to the data. We rather propose a data-driven assisted MHE-based MPC-LPV framework using RL aiming at improving the closed-loop performance even if the scheduling parameters are inexact. However, the proposed method can be used in the context of data-driven MPC-LPV where the LPV model is constructed based on any identification method. Then, the proposed learning-based control approach can be leveraged to indirectly compensate the effect of the possible model-mismatch in the identified LPV model. For example, the Koopman operator theory and Artificial Neural Networks (ANNs) can be used to identify an LPV model in state-space form [78, 79].

In what follows, the polytopic LPV model is introduced in Section 5.2, and the parameterized MHE and MPC schemes are detailed in Section 5.3. The MHE/MPC-based policy gradient reinforcement learning is then developed in Section 5.4. Simulations on a mass-spring-damper case study are reported in Section 5.5, and finally, conclusions and future work are given in Section 5.6.

5.2 Uncertain Polytopic LPV

In this chapter, we focus on the classic (pure) LPV systems, in which the uncertain dynamical model of the system can be described as follows:

$$\mathbf{x}_{k+1} = A\left(\boldsymbol{\rho}_k\right)\mathbf{x}_k + B\left(\boldsymbol{\rho}_k\right)\mathbf{u}_k \tag{5.1a}$$

$$\mathbf{y}_k = C(\boldsymbol{\rho}_k)\mathbf{x}_k \tag{5.1b}$$

where the scheduling parameters $\boldsymbol{\rho} \in \mathcal{P} \in \mathbb{R}^p$ are considered as some exogenous signals applied to the real system. The vectors $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$ are states and control inputs, respectively. The system measurements is labeled by $\mathbf{y}_k \in \mathbb{R}^e$. Hence, the system (5.1) can be represented by a polytopic approximation $[A\left(\boldsymbol{\rho}_k\right), B\left(\boldsymbol{\rho}_k\right), C\left(\boldsymbol{\rho}_k\right)] \in \Omega$, where a polytope Ω is defined as a convex hull upon the Linear Time Invariant (LTI) models as follows:

$$\Omega = Co\left\{[A_1, B_1, C_1], [A_2, B_2, C_2], \cdots, [A_\ell, B_\ell, C_\ell]\right\} \tag{5.2}$$

where $\ell = 2^p$ is the number of vertices and $Co\{\cdot\}$ reads a convex hull. Indeed, the LTI matrices are delivered by combinations of the scheduling parameters at their bounds $\underline{\boldsymbol{\rho}}, \bar{\boldsymbol{\rho}}$. We then evaluate the variable matrices in (5.1) as a convex combination of the LTI models (a.k.a vertices) as follows:

$$A\left(\boldsymbol{\rho}_k\right) = \sum_{i=1}^{\ell} \beta_i A_i, \quad B\left(\boldsymbol{\rho}_k\right) = \sum_{i=1}^{\ell} \beta_i B_i, \tag{5.3a}$$

$$C\left(\boldsymbol{\rho}_k\right) = \sum_{i=1}^{\ell} \beta_i C_i \tag{5.3b}$$

where A_i, B_i, C_i are known matrices of appropriate size and $\beta_i, i = 1 \cdots \ell$ is the unknown weight of each vertex (a.k.a convex combination vector) such that:

$$\chi = \left\{ \boldsymbol{\beta} \in \mathbb{R}^{\ell} : \sum_{i=1}^{\ell} \beta_i = 1, \quad 0 \leq \beta_i \leq 1 \right\} \tag{5.4}$$

Assumption 6. *By [80], the convex combination vector $\boldsymbol{\beta}_k$ is assumed to be constant or changing slowly with respect to the dynamics and there exists an estimator to compute the estimate of this vector such that $\hat{\boldsymbol{\beta}}_k \in \chi$ for all $k \in \mathbb{Z}_{0+}$.*

Note that the stability and constraint satisfaction of the MPC schemes for the polytopic LPV systems under Assumption 1 was detailed and established in [75] and by Lemma 2 and Theorem 1 in [80]. We then focus on this kind of the LPV system where its vertices may not be known and exact. As discussed, the vertices in a polytopic LPV may have a wrong model because of the inexact bounds of the scheduling parameters, some parametric uncertainties and the linearization approximation effects. Then, the wrong LTI models in the polytope Ω can be introduced by the inexact vertices as follows:

$$\hat{\Omega} = Co\left\{ [\hat{A}_1, \hat{B}_1, \hat{C}_1], [\hat{A}_2, \hat{B}_2, \hat{C}_2], \cdots, [\hat{A}_{\ell}, \hat{B}_{\ell}, \hat{C}_{\ell}] \right\} \tag{5.5}$$

In the context of MPC schemes for the polytopic LPVs, under Assumption 1, one needs to evaluate an estimation of $\beta_i, i = 1 \cdots \ell$ [75, 77]. In this chapter we propose to adopt an MHE scheme to deliver these unknown weights while some unmeasured states are estimated. However, the inexact vertices in $\hat{\Omega}$ can affect parameter/state estimations, the policy captured from the MPC and the closed-loop performance, subsequently. In addition to this inexact polytope issue, the partially measurable states can be challenging since the MHE scheme as an estimator of $\beta_{i,k}$ needs to capture all states as measurements and minimize a distance as

a model matching error between these measurements and those states delivered by the uncertain LPV model.

To cope with these problems, we propose to parameterize both the MHE and MPC schemes and adopt a reinforcement learning to adjust them in order to achieve the best closed-loop performance even when using some wrong LTI models in a polytopic LPV system. More precisely, it is possible to generate the optimal policy based on a wrong model by learning the MHE and MPC schemes.

5.3 Adjustable MHE-MPC in a LPV Framework

Under Assumption 1, an obvious choice for the combination vectors used in the MPC scheme would be $\boldsymbol{\beta}_{j|k} = \boldsymbol{\beta}_k$, for all $j \in \mathbb{Z}_{k,\dots,k+N}$. As discussed in [80], this choice could not be proper because if the entire β-parameter prediction vector suddenly changes, the MPC value function may not be decreasing. Hence, under Assumption 1, one can select a N-step delay in this parameter prediction as follows:

$$\boldsymbol{\beta}_{j|k} = \hat{\boldsymbol{\beta}}_{k-N+j}, \quad \forall j \in \mathbb{Z}_{0,\dots,N} \tag{5.6}$$

We then adopt an MHE-based estimator to deliver these weights needed in an MPC scheme. Let us formulate a parameterized MHE scheme as β-parameter and state estimator, where the vertices of the polytopic LPV are inexact and the real system states are not fully measurable.

$$\begin{aligned}
\left\{\hat{\boldsymbol{\beta}}_{k-N\dots k}, \hat{\mathbf{x}}_{k-N\dots k}\right\} = \underset{\boldsymbol{\beta},\mathbf{x}}{\arg\min}\, & \gamma^N \left\|\mathbf{x}_{k-N} - \tilde{\mathbf{x}}_{k-N}\right\|^2_{A_{\boldsymbol{\theta}}} \\
& + \sum_{j=k-N+1}^{k} \gamma^{k-j} \left(\left\|\boldsymbol{\mu}_j\right\|^2_{R_{\boldsymbol{\theta}}} + \left\|\Delta\boldsymbol{\beta}_{j|k}\right\|^2 \right) \\
\text{s.t.} \quad & \mathbf{x}_{j+1} = \hat{A}\left(\boldsymbol{\rho}_j\right)\mathbf{x}_j + \hat{B}\left(\boldsymbol{\rho}_j\right)\bar{\mathbf{u}}_j, && \text{(5.7a)} \\
& \hat{A}\left(\boldsymbol{\rho}_j\right) = \sum_{i=1}^{\ell} \beta_{j,i}\hat{A}_i, \quad \hat{B}\left(\boldsymbol{\rho}_j\right) = \sum_{i=1}^{\ell} \beta_{j,i}\hat{B}_i, && \text{(5.7b)} \\
& \bar{\mathbf{y}}_j = \left(\sum_{i=1}^{\ell} \beta_{j,i}\hat{C}_i\right)\mathbf{x}_j + \boldsymbol{\mu}_j, && \text{(5.7c)} \\
& \sum_{i=1}^{\ell} \beta_{j,i} = 1, \quad 0 \le \beta_{j,i} \le 1 && \text{(5.7d)} \\
& \Delta\boldsymbol{\beta}_{j|k} = \boldsymbol{\beta}_{j|k} - \boldsymbol{\beta}_{j|k-1}, \quad \boldsymbol{\beta}_j = Col\left\{\beta_{j,1}, \dots, \beta_{j,\ell}\right\} && \text{(5.7e)}
\end{aligned}$$

where $\gamma \in \mathbb{Z}_{(0,1)}$ is a discount factor and $\bar{\mathbf{y}}_j, \bar{\mathbf{u}}_j$ are the measurements available at the physical time k. We label $R_{\boldsymbol{\theta}}$ as an adjustable weight matrix penalizing the

deviation between some estimated states captured from the LPV model and their corresponding measurements. The first term in the above least squares problem is an arrival cost weighted with matrix $A_{\boldsymbol{\theta}}$, which aims at approximating the information prior to $k - N_{\text{MHE}}$, where $\tilde{\mathbf{x}}$ is the available estimation $\hat{\mathbf{x}}_{k-N}$ at time $k-1$. We then detail a policy gradient RL in order to adjust the combination vectors β_i, where the LTI models (vertices of the polytope) are wrong.

This chapter proposes a learning-based estimation and control for a polytopic LPV system with possibly inexact LTI models at the vertices, where the control policy is delivered by a parameterized MPC scheme as follows:

$$
\begin{aligned}
&\min_{\mathbf{x},\mathbf{u},\boldsymbol{\sigma}} \quad \gamma^N \left(T_{\boldsymbol{\theta}}(\mathbf{x}_{k+N}, \boldsymbol{\beta}_{k+N}) + \mathbf{w}_f^\top \boldsymbol{\sigma}_{k+N} \right) \\
&+ \sum_{j=k}^{k+N-1} \mathcal{G}_{\boldsymbol{\theta}}(\mathbf{x}_j, \mathbf{u}_j) + \sum_{j=k}^{k+N-1} \gamma^{j-k} \left(l_{\boldsymbol{\theta}}(\mathbf{x}_j, \mathbf{u}_j) + \mathbf{w}^\top \boldsymbol{\sigma}_j \right) && \text{(5.8a)} \\
&\text{s.t.} \quad \mathbf{x}_{j+1} = \hat{A}\left(\boldsymbol{\rho}_j\right) \mathbf{x}_j + \hat{B}\left(\boldsymbol{\rho}_j\right) \mathbf{u}_j, && \text{(5.8b)} \\
&\quad \mathbf{x}_k = \hat{\mathbf{x}}_k, \quad \boldsymbol{\beta}_{j|k} = \hat{\boldsymbol{\beta}}_{j-N} && \text{(5.8c)} \\
&\quad \hat{A}\left(\boldsymbol{\rho}_j\right) = \sum_{i=1}^{\ell} \beta_{j,i} \hat{A}_i, && \text{(5.8d)} \\
&\quad \hat{B}\left(\boldsymbol{\rho}_j\right) = \sum_{i=1}^{\ell} \beta_{j,i} \hat{B}_i, && \text{(5.8e)} \\
&\quad \mathbf{g}(\mathbf{u}_j) \leq 0, && \text{(5.8f)} \\
&\quad \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_j, \mathbf{u}_j) \leq \boldsymbol{\sigma}_j, \quad \mathbf{h}_{\boldsymbol{\theta}}^f(\mathbf{x}_{k+N}) \leq \boldsymbol{\sigma}_{k+N} && \text{(5.8g)} \\
&\quad \boldsymbol{\sigma}_{k,\ldots,k+N} \geq 0 && \text{(5.8h)}
\end{aligned}
$$

where $T_{\boldsymbol{\theta}}$ and $l_{\boldsymbol{\theta}}$ are the parameterized terminal and stage costs, respectively. The function $\mathcal{G}_{\boldsymbol{\theta}}$ is labeled the cost modification term, which can be, i.e, in a form of gradient $\mathbf{f}^\top [\mathbf{x}_j, \mathbf{u}_j]^\top$. However, the polytopic LPV model is uncertain with wrong vertices that could cause some violations in its constraints and bring the MPC in an infeasible region. To avoid such an infeasibility, we introduce some slack variables $\boldsymbol{\sigma}$ penalized by enough large weights $\mathbf{w}, \mathbf{w}_f$. We also propose to have some adjustable parameters upon inequality constraints $h_{\boldsymbol{\theta}}, h_{\boldsymbol{\theta}}^f$. Then, we let RL to tune the MPC parameters (a.k.a policy parameters). Note that, the proposed parameterization of the MHE and MPC is a general form of a rich parameterization and one may take into account some of these parameters to be adjusted by RL.

Note that the parameterized terminal cost in the MPC scheme (5.8) is defined as a

function of the terminal states and the convex combination vector as follows:

$$T_{\boldsymbol{\theta}} = \mathbf{x}_{k+N}^{\top} \left(\sum_{i=1}^{\ell} \beta_{k+N,i} P_i \right) \mathbf{x}_{k+N} \tag{5.9}$$

where the matrices $P_i > 0, i \in \mathbb{Z}_{1,\ldots,\ell}$ are adjusted by RL. However, the initial values for these matrices can be computed offline via a Linear Matrix Inequality (LMI) based on a parameter-dependent Lyapunov function and a parameter-dependent linear control law detailed in [75].

5.4 MHE/MPC-based Policy Gradient RL

Let us define the closed-loop performance of a parameterized policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ for a given stage cost $L(\mathbf{x}, \mathbf{a})$ as the following total expected cost:

$$J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) = \mathbb{E}_{\varrho_s} \left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{x}_k, \mathbf{a}_k) \,\middle|\, \mathbf{a}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{x}_k) \right] \tag{5.10}$$

where the expectation $\mathbb{E}_{\varrho_s}[\cdot]$ is taken over the distribution of the Markov chain ϱ_s in the closed-loop under policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$. We then seek the optimal policy parameters as follows:

$$\boldsymbol{\theta}_{\star} = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) \tag{5.11}$$

As a learning-based control approach in this chapter, we propose to use a parameterized MPC scheme as a policy approximation in order to deliver $\boldsymbol{\pi}_{\boldsymbol{\theta}}$. As an advantage to use a MPC-based policy approximation instead of DNNs, it can offer a learning-based controller so that all the state-input constraints are satisfied and the closed-loop performance is improved by adjusting the policy parameters.

For a given estimated state $\hat{\mathbf{x}}_k$ and parameter $\hat{\boldsymbol{\beta}}_k$ delivered by the MHE scheme, the policy captured by a parameterized MPC scheme is

$$\boldsymbol{\pi}_{\boldsymbol{\theta}} \left(\hat{\mathbf{x}}_k, \hat{\boldsymbol{\beta}}_k \right) = \mathbf{u}_0^{\star} \left(\hat{\mathbf{x}}_k, \hat{\boldsymbol{\beta}}_k, \boldsymbol{\theta} \right) \tag{5.12}$$

where $\mathbf{u}_0^{\star}$ is the first element of the control input sequence $\mathbf{u}^{\star}$ delivered by (5.8).

In this section, we propose a policy gradient RL framework based on MPC and MHE to deal with the polytopic LPV systems with wrong LTI models.

5.4.1 Compatible Deterministic Actor-Critic

In the context of DPG-based RL algorithms, the policy parameters $\boldsymbol{\theta}$ can be directly optimized by the gradient descent steps such that the best expected closed-loop cost (a.k.a policy performance index J) can be captured by applying the

policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$.

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) \tag{5.13}$$

for some $\alpha > 0$ small enough as the step size. In the context of hybrid controller/observer scheme MHE-MPC, the input signal can be interpreted as a sequence of measurements $\mathbf{y} = \left[y^1_{0,\cdots,k}, \cdots, y^{n_y}_{0,\cdots,k}\right]^\top$ at the physical time k, where n_y is the number of system states selected as measurable outputs. Then, the intermediate variables $\hat{\mathbf{x}}_k, \hat{\boldsymbol{\beta}}_k$ are delivered by the MHE scheme based on the history of the measurements and fed to the MPC scheme to deliver the control policy. Assuming that this history constitutes a Markov state, one can define a policy performance index by the following expected value:

$$J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) := \mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left[Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\hat{\mathbf{x}}_k, \boldsymbol{\pi}_{\boldsymbol{\theta}}\left(\hat{\mathbf{x}}_k, \hat{\boldsymbol{\beta}}_k\right)\right)\right] \tag{5.14}$$

where the expectation $\mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ is taken over the distribution of the Markov chain resulting from the real system in closed-loop with $\boldsymbol{\pi}_{\boldsymbol{\theta}}$. The action-value function $Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ is then defined as follows:

$$Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\hat{\mathbf{x}}_k, \mathbf{a}_k\right) = L\left(\mathbf{y}_k, \mathbf{a}_k\right) + \gamma \mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left[V_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left(\hat{\mathbf{x}}_{k+1}\right) | \hat{\mathbf{x}}_k, \mathbf{a}_k\right] \tag{5.15}$$

where, $L\left(\mathbf{y}_k, \mathbf{a}_k\right)$ reads as baseline cost (RL stage cost), which is a function of measurable states and actions at the current time k. Based on the proposed DPG theorem by [17] and the fact that both the $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ and $Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}$ are functions of $\hat{\mathbf{x}}_k$, the policy gradient equation is described as follows:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) = \mathbb{E}\left[\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_k) \nabla_{\mathbf{u}} Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\hat{\mathbf{x}}_k, \mathbf{a}_k)|_{\mathbf{a}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}}\right] \tag{5.16}$$

To represent the effect of the parameterized MHE upon the policy gradient, the sensitivity of the policy w.r.t $\boldsymbol{\theta}$ can be updated such that the new policy gradient is described by the following expectation:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi}_{\boldsymbol{\theta}}) = \mathbb{E}\Big[\Xi \nabla_{\mathbf{u}} Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\hat{\mathbf{x}}_k, \mathbf{a}_k)|_{\mathbf{a}_k = \boldsymbol{\pi}_{\boldsymbol{\theta}}}\Big] \tag{5.17}$$

where the Jacobian matrix Ξ is obtained by the following chain rule:

$$\Xi = \left(\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}} + \nabla_{\boldsymbol{\theta}} \hat{\mathbf{x}}_k \nabla_{\hat{\mathbf{x}}_k} \boldsymbol{\pi}_{\boldsymbol{\theta}} + \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{\beta}}_k \nabla_{\hat{\boldsymbol{\beta}}_k} \boldsymbol{\pi}_{\boldsymbol{\theta}}\right) \tag{5.18}$$

In this chapter, we adopt a *compatible deterministic actor-critic* algorithm [17] in which the action-value function $Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\hat{\mathbf{x}}_k, \mathbf{a}_k)$ can be replaced by a class of compatible function approximator $Q^{\mathbf{w}}(\hat{\mathbf{x}}_k, \mathbf{a}_k)$ such that the policy gradient is preserved.

Therefore, the compatible function for a deterministic policy $\boldsymbol{\pi_\theta}$ delivered by the parameterized MHE-MPC scheme can be expressed as follows:

$$Q^{\mathbf{w}} = (\mathbf{a}_k - \boldsymbol{\pi_\theta})^\top \Xi^\top \mathbf{w} + V^{\boldsymbol{\nu}}(\hat{\mathbf{x}}_k) \tag{5.19}$$

The first term in the above compatible function as critic part is an estimation for the advantage function and the second term estimates a value function for the history of the measurements delivered as a summarized variable $\hat{\mathbf{x}}_k$ by the MHE scheme. Both functions can be computed by the linear function approximators as follows:

$$V^{\boldsymbol{\nu}}(\hat{\mathbf{x}}_k) = \boldsymbol{\Upsilon}(\hat{\mathbf{x}}_k)^\top \boldsymbol{\nu}, \tag{5.20a}$$

$$A^{\mathbf{w}}(\hat{\mathbf{x}}_k, \mathbf{a}_k) = \boldsymbol{\Psi}(\hat{\mathbf{x}}_k, \mathbf{a}_k)^\top \mathbf{w} \tag{5.20b}$$

where $\boldsymbol{\Upsilon}(\hat{\mathbf{x}}_k)$ is the summarized measurement feature vector in order to constitute all monomials of the history of the measurements with degrees less than or equal to 2. The vector $\boldsymbol{\Psi}(\hat{\mathbf{x}}_k, \mathbf{a}_k) := \Xi\left(\mathbf{a}_k - \boldsymbol{\pi_\theta}(\hat{\mathbf{x}}_k, \hat{\boldsymbol{\beta}}_k)\right)$ includes the state-action features. Considering (5.19), the policy gradient (5.17) is then rewritten as follows:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\pi_\theta}) = \mathbb{E}\left[\Xi\Xi^\top \mathbf{w}\right] \tag{5.21}$$

In this chapter, the parameterized policy in the context of policy gradient RL is proposed to be captured by the MPC scheme (5.8). Figure 5.1 shows an overview of the proposed learning-based control algorithm.

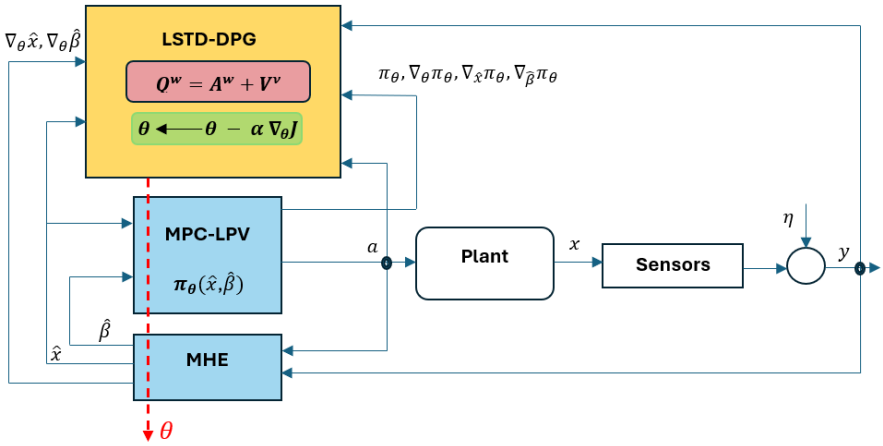


Figure 5.1: An overview of the MHE/MPC-based deterministic policy gradient method where the MPC model is represented in the polytopic LPV framework. The MHE scheme (5.7) delivers both the state estimation $\hat{x}$ and the combination vector $\hat{\beta}$. The corresponding sensitivities are then used in the LSTD-DPG method. The MPC-LPV scheme (5.8) combined with the MHE scheme delivers the parameterized policy π_θ. The action a is then selected according to the policy π_θ with the possible addition of exploratory moves.

To evaluate the policy gradient (5.21), one need to calculate some sensitivities upon the MPC and MHE schemes in order to compute the Jacobian matrix Ξ. Hence, the Jacobian matrices $\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi}_{\boldsymbol{\theta}}$, $\nabla_{\hat{\mathbf{x}}_k}\boldsymbol{\pi}_{\boldsymbol{\theta}}$ and $\nabla_{\hat{\boldsymbol{\beta}}_k}\boldsymbol{\pi}_{\boldsymbol{\theta}}$ can be computed by the sensitivity analysis for the parameterized MPC scheme while the gradients $\nabla_{\boldsymbol{\theta}}\hat{\mathbf{x}}_k$ and $\nabla_{\boldsymbol{\theta}}\hat{\boldsymbol{\beta}}_k$ are obtained as a sensitivity term for the parameterized MHE scheme.

5.4.2 Sensitivity Analysis and LSTD-based DPG

Sensitivity Computation

We describe next how to compute the sensitivities (gradients) needed in the proposed policy gradient RL framework based on MHE-MPC. To that end, let us define the Lagrange functions $\hat{\mathcal{L}}_{\boldsymbol{\theta}}, \mathcal{L}_{\boldsymbol{\theta}}$ associated to the MHE and MPC schemes (5.7), (5.8) as follows:

$$\hat{\mathcal{L}}_{\boldsymbol{\theta}}\left(\hat{\mathbf{z}}\right) = \hat{\Lambda}_{\boldsymbol{\theta}} + \hat{\boldsymbol{\lambda}}^{\top}\hat{G}_{\boldsymbol{\theta}} \tag{5.22}$$

$$\mathcal{L}_{\boldsymbol{\theta}}\left(\mathbf{z}\right) = \Lambda_{\boldsymbol{\theta}} + \boldsymbol{\lambda}^{\top}G_{\boldsymbol{\theta}} + \boldsymbol{\mu}^{\top}H_{\boldsymbol{\theta}} \tag{5.23}$$

where $\Lambda_{\boldsymbol{\theta}}$ and $\hat{\Lambda}_{\boldsymbol{\theta}}$ are the total parameterized costs of the MPC and MHE schemes, respectively. The inequality constraints of (5.8) are collected by $H_{\boldsymbol{\theta}}$ while $G_{\boldsymbol{\theta}}$ and $\hat{G}_{\boldsymbol{\theta}}$ gather, respectively, the equality constraints in the MPC and MHE schemes. We then label $\boldsymbol{\lambda}, \hat{\boldsymbol{\lambda}}$ the Lagrange multipliers associated to the equality constraints $G_{\boldsymbol{\theta}}, \hat{G}_{\boldsymbol{\theta}}$ of the MPC and MHE, respectively. Variables $\boldsymbol{\mu}$ are the Lagrange multipliers associated to the inequality constraints of the MPC scheme. Let us label $\boldsymbol{\Gamma} = \{\mathbf{x}, \mathbf{u}, \boldsymbol{\sigma}\}$ and $\hat{\boldsymbol{\Gamma}} = \left\{\hat{\mathbf{x}}, \hat{\boldsymbol{\beta}}\right\}$ the primal variables for the MPC and MHE, respectively. The associated primal-dual variables then read as $\mathbf{z} = \{\boldsymbol{\Gamma}, \boldsymbol{\lambda}, \boldsymbol{\mu}\}$ and $\hat{\mathbf{z}} = \left\{\hat{\boldsymbol{\Gamma}}, \hat{\boldsymbol{\lambda}}\right\}$.

The sensitivity of the policy delivered by the MPC scheme (5.8) w.r.t policy parameters and the sensitivity of the estimated state associated to the MHE scheme (5.7) can be obtained via using the Implicit Function Theorem (IFT) on the Karush Kuhn Tucker (KKT) conditions underlying the parametric NLP. Assuming that Linear Independence Constraint Qualification (LICQ) and Second Order Sufficient Condition (SOSC) hold [18] at $\mathbf{z}^{\star}$ and $\hat{\mathbf{z}}^{\star}$, then, the following holds:

$$\frac{\partial \mathbf{z}^{\star}}{\partial \boldsymbol{\theta}} = -\frac{\partial \kappa_{\boldsymbol{\theta}}}{\partial \mathbf{z}}^{-1}\frac{\partial \kappa_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}, \tag{5.24a}$$

$$\frac{\partial \hat{\mathbf{z}}^{\star}}{\partial \boldsymbol{\theta}} = -\frac{\partial \hat{\kappa}_{\boldsymbol{\theta}}}{\partial \hat{\mathbf{z}}}^{-1}\frac{\partial \hat{\kappa}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \tag{5.24b}$$

where

$$\kappa_{\boldsymbol{\theta}} = \begin{bmatrix} \nabla_{\mathbf{\Gamma}} \mathcal{L}_{\boldsymbol{\theta}} \\ G_{\boldsymbol{\theta}} \\ \operatorname{diag}(\boldsymbol{\mu})\, \mathbf{H}_{\boldsymbol{\theta}} \end{bmatrix}, \quad \hat{\kappa}_{\boldsymbol{\theta}} = \begin{bmatrix} \nabla_{\hat{\mathbf{\Gamma}}} \hat{\mathcal{L}}_{\boldsymbol{\theta}} \\ \hat{G}_{\boldsymbol{\theta}} \end{bmatrix} \tag{5.25}$$

are the KKT conditions associated to the MPC and MHE schemes, respectively. As $\boldsymbol{\pi}_{\boldsymbol{\theta}}$ and $(\hat{\mathbf{x}}, \hat{\boldsymbol{\beta}})$ are, respectively, part of $\mathbf{z}^\star$ and $\hat{\mathbf{z}}^\star$. Then, the sensitivity of the MPC policy $\nabla_{\boldsymbol{\theta}} \boldsymbol{\pi}_{\boldsymbol{\theta}}$ and the sensitivity of the MHE solution $(\nabla_{\boldsymbol{\theta}} \hat{\mathbf{x}}, \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{\beta}})$ required in (5.21) can be extracted from gradients $\frac{\partial \mathbf{z}^\star}{\partial \boldsymbol{\theta}}$ and $\frac{\partial \hat{\mathbf{z}}^\star}{\partial \boldsymbol{\theta}}$, respectively.

LSTD-based Policy Gradient

In the context of compatible DPG, one can evaluate the optimal parameters $\mathbf{w}$ and $\boldsymbol{\nu}$ of the action-value function approximation (5.19) as solutions of the following Least Squares (LS) problem:

$$\min_{\mathbf{w}, \boldsymbol{\nu}} \mathbb{E}\left[\left(Q_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}(\hat{\mathbf{x}}_k, \mathbf{a}_k) - Q^{\mathbf{w}}(\hat{\mathbf{x}}_k, \mathbf{a}_k) \right)^2 \right], \tag{5.26}$$

In the context of RL, the LSTD-based algorithms offer an efficient use of data and tend to converge faster than other methods, e.g., see [22]. The LSTD update rules for a policy gradient RL are then obtained as follows:

$$\boldsymbol{\nu} = A_{\boldsymbol{\nu}}^{-1} b_{\boldsymbol{\nu}}, \tag{5.27a}$$

$$\mathbf{w} = A_{\mathbf{w}}^{-1} b_{\mathbf{w}}, \tag{5.27b}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha b_{\boldsymbol{\theta}} \tag{5.27c}$$

where the matrices $\Omega_{(\cdot)}$ and the vectors $b_{(\cdot)}$ are calculated by taking expectation ($\mathbb{E}_m$) over m episodes as follows:

$$A_{\boldsymbol{\nu}} = \mathbb{E}_m \left[\sum_{k=1}^{T_f} \left[\boldsymbol{\Upsilon} \left(\hat{\mathbf{x}}_k \right) \left(\boldsymbol{\Upsilon} \left(\hat{\mathbf{x}}_k \right) - \gamma \boldsymbol{\Upsilon} \left(\hat{\mathbf{x}}_{k+1} \right) \right)^\top \right] \right] \tag{5.28a}$$

$$A_{\mathbf{w}} = \mathbb{E}_m \left[\sum_{k=1}^{T_f} \left[\boldsymbol{\Psi} \left(\hat{\mathbf{x}}_k, \mathbf{a}_k \right) \boldsymbol{\Psi} \left(\hat{\mathbf{x}}_k, \mathbf{a}_k \right)^\top \right] \right], \tag{5.28b}$$

$$b_{\boldsymbol{\nu}} = \mathbb{E}_m \left[\sum_{k=1}^{T_f} \boldsymbol{\Upsilon} \left(\hat{\mathbf{x}}_k \right) L(\mathbf{y}_k, \mathbf{a}_k) \right], \tag{5.28c}$$

$$b_{\mathbf{w}} = \tag{5.28d}$$

$$\mathbb{E}_m \left[\sum_{k=1}^{T_f} \left[\left(L(\mathbf{y}_k, \mathbf{a}_k) + \gamma V^{\boldsymbol{\nu}} \left(\hat{\mathbf{x}}_{k+1} \right) - V^{\boldsymbol{\nu}} \left(\hat{\mathbf{x}}_k \right) \right) \boldsymbol{\Psi}(\hat{\mathbf{x}}_k, \mathbf{a}_k) \right] \right],$$

$$b_{\boldsymbol{\theta}} = E_m \left[\sum_{k=1}^{T_f} \Xi \Xi^\top \mathbf{w} \right] \tag{5.28e}$$

where T_f is the final time instant at the end of each episode.

5.5 Illustrative Example

To demonstrate the effectiveness of the proposed learning-based control approach, we choose a stabilization problem for a mass-spring-damper system where the spring and damper coefficients are considered as the exogenous scheduling parameters.

$$\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -d/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u \tag{5.29}$$

Then, the LPV model can be represented as a linear system with polytopic uncertainty, including four vertices (LTI models), which are constructed based on the scheduling bounds $1 \leq k \leq 2, 0 \leq b \leq 0.5$ shown in Figure 5.2.

$$A_1 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & 1 \\ -1 & -0.5 \end{bmatrix} \tag{5.30}$$
$$A_3 = \begin{bmatrix} 0 & 1 \\ -2 & 0 \end{bmatrix}, \quad A_4 = \begin{bmatrix} 0 & 1 \\ -2 & -0.5 \end{bmatrix}$$

and $B_{1,\ldots,4} = [0, 1]^\top$. On the other hand, we select the scheduling bounds used in the real system different from those adopted in four LTI models $0.5 \leq k \leq$

$2,0 \leq b \leq 0.2$. The position of the mass (x_1) is selected as measurement while the velocity ($x_2 = \dot{x}_1$) is estimated. We consider a constraint on the control input $-1 \leq u \leq 1$. As it can be observed in Figure 5.3, the stabilization scenario is af-

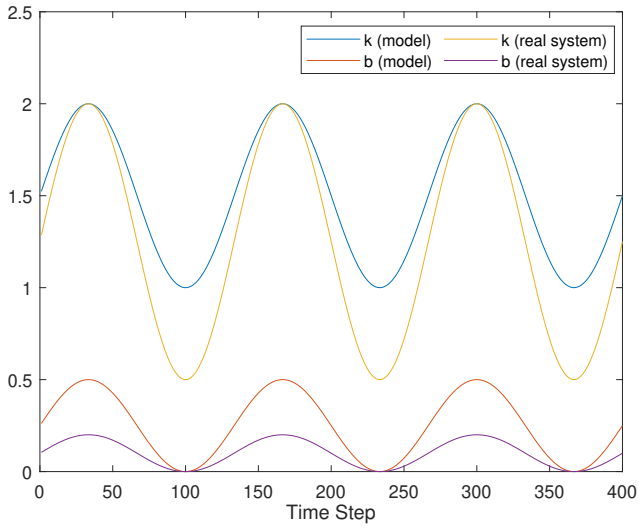


Figure 5.2: Mass-Spring-Damper with variable spring constant and damping factor as scheduling parameters.

fected since the scheduling parameters violate their correct values used in four LTI models. We then adopt a policy gradient RL to adjust the MHE-MPC parameters in order to cope with this problem and improve the closed-loop performance $J(\pi_\theta)$. The evolution of some RL parameters and the closed-loop performance are shown in Figure 5.4. The evolution during learning shows that the optimal policy (Figure 5.3: π_θ in black color) can be delivered after 35 RL step and the stabilization goal is perfectly achieved. Note that the system's behavior in terms of β-parameters and control policy without learning is highlighted in red color shown in Figure 5.3. It is obvious that the estimated combination vector $\boldsymbol{\beta}$ is improved by adjusting the MHE scheme under wrong LTI models. As a comparative study shown in Figure 5.5, the performance of the stabilization has been dramatically improved after learning the MHE-MPC scheme where the LTI models cannot capture the correct polytope based on the scheduling bounds used in the real system.

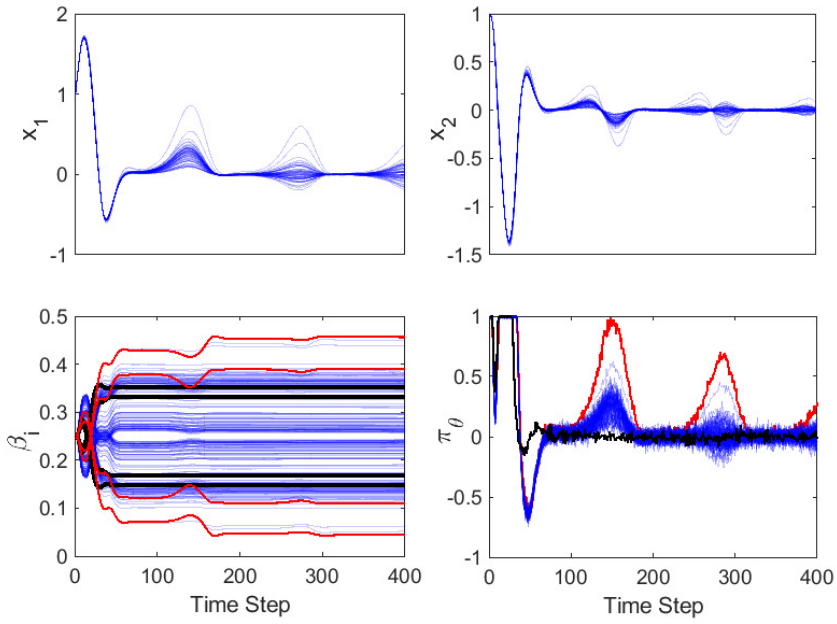


Figure 5.3: The blue lines show the evolution of the states, policy and β-parameters during reinforcement learning. The red lines show the evolution of the combination vectors $\beta_{1,\dots,4}$ and the policy π_θ without learning while the black lines show the results obtained by the proposed method when the learning progress is terminated after 35 RL steps.

5.6 Conclusion

In this chapter, an MHE-MPC scheme was proposed to deal with the linear systems with polytopic uncertainty in the context of the LPV systems with exogenous scheduling parameters. However, the performance of the adopted multi-model approach can be affected where the bounds of the scheduling parameters are different from their real values applied to the real system. To address a data-driven MHE-MPC approach for the polytopic LPV systems with inexact LTI models, we proposed to adopt a policy gradient reinforcement learning to capture the optimal policy and achieve the best closed-loop performance by adjusting the MHE and MPC schemes.

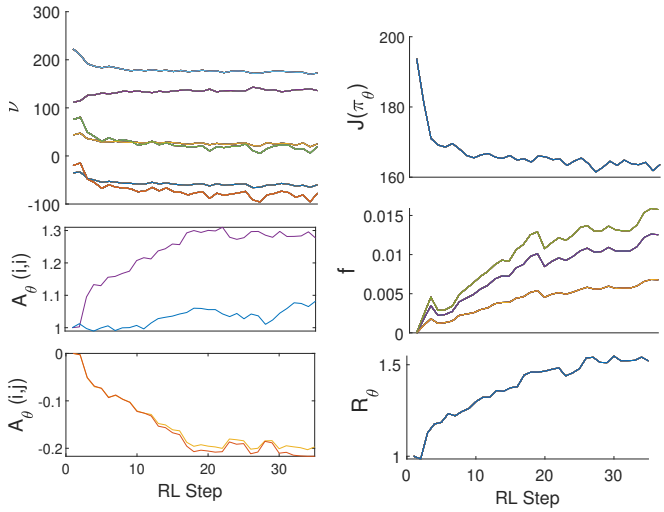


Figure 5.4: As observed, the closed-loop performance $J(\pi_\theta)$ is improved by learning the combined MHE-MPC scheme. Some parameters of the DPG method are: the weight matrix A_θ of the arrival cost, the weight matrix R_θ, the parameters of the MPC cost modification f and the parameters of the value function ν.

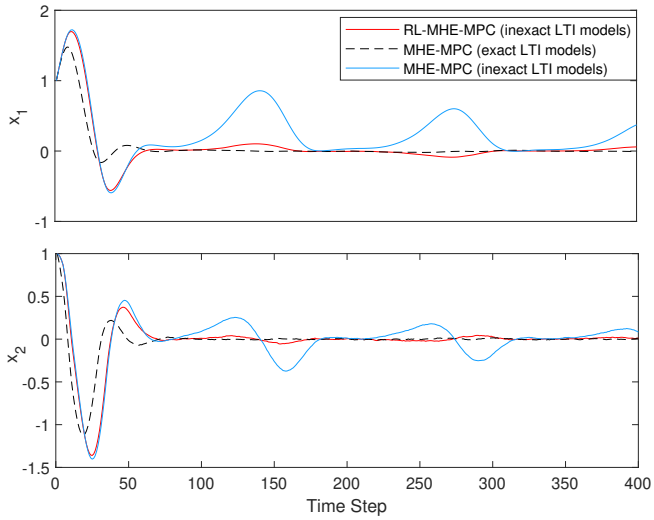


Figure 5.5: Comparative analysis. The blue lines show the evolution of the sates where the LTI models used in the MPC-LPV scheme are inexact. The black dashed lines show the results obtained from the MHE-MPC with exact LTI models used in the MPC-LPV scheme. The red lines depict the evolution of the states using the proposed MHE/MPC-based DPG where the wrong LTI models are used in the MPC-LPV scheme.

Part II

Reinforcement Learning based on Robust NMPC

Chapter 6

Approximate Robust Nonlinear MPC using RL

- H. N. Esfahani, A. B. Kordabad and S. Gros, "Approximate Robust NMPC using Reinforcement Learning," 2021 European Control Conference (ECC), Delft, Netherlands, 2021, pp. 132-137, doi: 10.23919/ECC54610.2021.9655129.

We present a Reinforcement Learning-based Robust Nonlinear Model Predictive Control (RL-RNMPC) framework for controlling nonlinear systems in the presence of disturbances and uncertainties. An approximate Robust Nonlinear Model Predictive Control (RNMPC) of low computational complexity is used in which the state trajectory uncertainty is modelled via ellipsoids. Reinforcement Learning is then used in order to handle the ellipsoidal approximation and improve the closed-loop performance of the scheme by adjusting the MPC parameters generating the ellipsoids. The approach is tested on a simulated Wheeled Mobile Robot (WMR) tracking a desired trajectory while avoiding static obstacles.

6.1 Introduction

Nonlinear Model Predictive Control (NMPC) is an optimization based control approach operating in a receding horizon [9], which is often adopted for its ability to handle linear/nonlinear state and input constraints. MPC offers rigorous theoretical properties (such as recursive feasibility, constraint satisfaction and stability), assuming that an accurate model of the plant is available. There are many autonomous systems (such as ground vehicles, marine robots and unmanned aerial vehicles) for which NMPC-based algorithms have been adopted [81], [82], [83].

Reinforcement Learning (RL) is a powerful tool for tackling Markov Decision Processes (MDP) without depending on a model of the probability distributions underlying the state transitions of the real system [11]. Indeed, most RL methods rely purely on observed state transitions, and realizations of the stage cost in order to increase the performance of the control policy.

Recently, the integration of machine learning in MPC has been invesstigated, with the aim of learning the model of the system, the cost function or even the control law directly [34, 35]. For computational reasons, simple models are usually preferred in the MPC scheme. Hence, the MPC model often does not have the structure required to correctly capture the real system dynamics and stochasticity. As a result, MPC delivers a reasonable but suboptimal approximation of the optimal policy. Choosing the MPC parameters that maximises the closed-loop performance for the selected MPC formulation is a difficult problem. Indeed, e.g. selecting the MPC model parameters that best fit the model to the real system is not guaranteed to yield the best closed-loop performance that the MPC scheme can achieve [25]. In [25, 84], it is shown that adjusting the MPC model, cost and constraints can be beneficial to achieve the best closed-loop performances, and Reinforcement Learning is proposed as a possible approach to perform that adjustment in practice. Further recent research have focused on MPC-based policy approximation for RL [2, 85, 86, 40, 21], where it is shown that a single Model Predictive Control (MPC) scheme can capture the optimal value function, action-

value function, and policy of an MDP, even if the MPC model is inaccurate, hence providing a valid and generic function approximator for RL.

Model-plant mismatch and disturbances can be treated via Robust NMPC (RNMPC) techniques. For linear MPC models and polytopic disturbance models and constraints, tube-based MPC techniques provides computationally effective techniques [87], [88]. Treating nonlinear MPC models or generic disturbances and constraints is more challenging [89]. Researchers in [90] proposed to use a tube-based MPC with a Min-Max differential inequality. Multi-stage or Scenario-tree NMPC scheme was proposed in [91, 92] as a real-time NMPC that accounts for the uncertain influence and generates decisions to control a nonlinear plant in a robust sense. These approaches remain challenging for problems that are not of small scale.

In this chapter, we model the propagation of perturbations in the state dynamics via ellipsoids, based on the linearization of the system dynamics and constraints on the nominal trajectories and using a Gaussian disturbance model. We propose to adjust this scheme using the RL method in order to tailor this inaccurate uncertainty model to the real system and achieve a better closed-loop performance. A fast convergence of the adjustable parameters of RNMPC is achieved via a second-order Least Square Temporal Difference Q-learning (LSTDQ).

This chapter is organised as follows. In Section 6.2, the proposed approximate robust NMPC is formulated. The combination of the RL algorithm and RNMPC is detailed in Section 6.3. A mobile robot under uncertainties is adopted in Section 6.4 to be controlled by the proposed RL-RNMPC for a trajectory tracking scenario associated to elliptic static obstacle avoidance in presence of model uncertainty. Finally, conclusions and future work are given in Section 6.5.

6.2 Ellipsoidal-based Robust NMPC

We consider a model based on the nonlinear dynamics

$$\mathbf{x}_+ = \mathbf{f}\left(\mathbf{x}, \mathbf{u}, \mathbf{d}\right) \tag{6.1}$$

where $\mathbf{d}$ is a stochastic variables affecting the state evolution. We assume that the real system is unknown and imperfectly represented by (6.1). A nominal NMPC scheme based on (6.1) reads e.g. as:

$$\min_{\bar{\mathbf{x}},\bar{\mathbf{u}}} \quad T\left(\bar{\mathbf{x}}_N\right) + \sum_{k=0}^{N-1} L\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) \tag{6.2a}$$

$$\text{s.t.} \quad \bar{\mathbf{x}}_{k+1} = \mathbf{f}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k, \mathbf{d}_k\right), \quad \mathbf{x}_0 = \mathbf{s} \tag{6.2b}$$

$$\mathbf{h}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) \leq 0, \quad \mathbf{h}_{\mathrm{f}}\left(\bar{\mathbf{x}}_N\right) \leq 0 \tag{6.2c}$$

where $L\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right)$ and $T\left(\bar{\mathbf{x}}_N\right)$ are the stage and terminal costs, respectively. $\mathbf{s}$ is the current system state while the nominal NMPC predicted trajectories are $\bar{\mathbf{x}}$ and $\bar{\mathbf{u}}$. We label $\mathbf{h}$ and $\mathbf{h}_f$ as the nominal stage and terminal inequality constraints, respectively. We will model the model error and real system stochasticity via the uncertain sequence $\mathbf{d}_k$ of disturbances, and we will approximate its effect on the state dynamics in a computationally effective way. More specifically, we will model the sequence $\mathbf{d}_k$ via an i.i.d Gaussian model with mean $\bar{\mathbf{d}}_k$ and covariance Λ i.e.

$$\mathbf{d}_k \sim \mathcal{N}\left(\bar{\mathbf{d}}_k, \Lambda\right) \tag{6.3}$$

The first-order deviation of the model state $\mathbf{x}_k$ from its nominal trajectory $\bar{\mathbf{x}}_k$ is then given by:

$$\begin{aligned}\Delta\mathbf{x}_{k+1} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_k,\mathbf{u}_k} \Delta\mathbf{x}_k + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_k,\mathbf{u}_k} \Delta\mathbf{u}_k \\ + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{d}}\right|_{\mathbf{x}_k,\mathbf{u}_k} \Delta\mathbf{d}_k\end{aligned} \tag{6.4}$$

where $\Delta\mathbf{x}_0$ is a given deviation of the initial state and $\Delta\mathbf{d}_k = \mathbf{d}_k - \bar{\mathbf{d}}_k$. We observe that for $\mathbb{E}\left[\Delta\mathbf{x}_0\right] = 0$, $\bar{\mathbf{d}}_k = \mathbb{E}\left[\mathbf{d}_k\right]$ and $\boldsymbol{\Delta}\mathbf{u}_k = 0$. The expected value of the state deviation is $\mathbb{E}\left[\Delta\mathbf{x}_k\right] = 0$. We consider a linear feedback over the state deviations for the input deviation $\Delta\mathbf{u}_k$ as:

$$\Delta\mathbf{u}_k = -K_k\Delta\mathbf{x}_k \tag{6.5}$$

where K_k is a control gain matrix typically based on Linear Quadratic Regulator (LQR) techniques. Then the state deviation dynamics read as:

$$\Delta\mathbf{x}_{k+1} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} - \frac{\partial \mathbf{f}}{\partial \mathbf{u}}K_k\right)\Bigg|_{\mathbf{x}_k,\mathbf{u}_k} \Delta\mathbf{x}_k + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{d}}\right|_{\mathbf{x}_k,\mathbf{u}_k} \Delta\mathbf{d}_k$$

Let us define the time-varying matrices as follows:

$$A_k = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} - K_k\left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_k,\mathbf{u}_k}, \quad B_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{d}}\right|_{\mathbf{x}_k,\mathbf{u}_k}$$

Then, the covariance of the state deviation reads as:

$$\begin{aligned}\Sigma_{k+1} &= \mathbb{E}\left[\Delta\mathbf{x}_{k+1}\Delta\mathbf{x}_{k+1}^\top\right] \\ &= \mathbb{E}\left[\left(A_k\Delta\mathbf{x}_k + B_k\Delta\mathbf{d}_k\right)\left(A_k\Delta\mathbf{x}_k + B_k\Delta\mathbf{d}_k\right)^\top\right] \\ &= \mathbb{E}\left[A_k\Delta\mathbf{x}_k\Delta\mathbf{x}_k^\top A_k^\top + B_k\Delta\mathbf{d}_k\Delta\mathbf{d}_k^\top B_k^\top\right] \\ &= A_k\Sigma_k A_k^\top + B_k\Lambda B_k^\top\end{aligned}$$

where Λ is the covariance of the process noise. We observe that the matrices Σ_k describe ellipsoids of state uncertainty propagations defined by the confidence regions:

$$R_k := \left\{ \Delta \mathbf{x}_k \;\middle|\; \frac{1}{2} \Delta \mathbf{x}_k^\top \Sigma_k^{-1} \Delta \mathbf{x}_k \leq \sigma_k \right\} \tag{6.6}$$

where σ_k is a Mahalanobis distance (the radius of the ellipsoids as the confidence regions). More specifically, we observe that for a Gaussian disturbance $\mathbf{d}_k$ as in (6.3) with n-dimensional mean vector $\bar{\mathbf{d}}_k$, the state deviation approximation Δx_k is also Gaussian, given by the following density:

$$\rho\left(\Delta x_k\right) = (2\pi)^{-\frac{n}{2}} \left(\det\left(\Sigma_k\right)\right)^{-\frac{1}{2}} e^{-\frac{1}{2} \Delta \mathbf{x}_k^\top \Sigma_k^{-1} \Delta \mathbf{x}_k} \tag{6.7}$$

such that at time k, a state deviations Δx_k has a probability density larger or equal to $(2\pi)^{-\frac{n}{2}} \left(\det\left(\Sigma_k\right)\right)^{-\frac{1}{2}} e^{-\sigma_k}$ to belong to R_k in (6.6).

The probability of the state deviation to belong to the ellipsoid described by (6.6) is given by:

$$\mathbb{P}[\Delta x_k \in R_k] = \int_{R_k} \rho\left(\Delta x_k\right) \mathrm{d}\Delta x_k = \frac{\gamma_k\left(\frac{n}{2}, \frac{\sigma_k^2}{2}\right)}{\Gamma_k\left(\frac{n}{2}\right)} \tag{6.8}$$

where, Γ_k is the Gamma function and γ_k is the lower incomplete Gamma function describing the probability content for a n-dimensional multinormal distribution [93]. Imposing that the ellipsoids described by (6.6) are contained within the MPC feasible set for some sequence σ_k then becomes a way to approximately satisfy the MPC constraints with probability (6.8). In the next section, we detail how to build an inexpensive Robust MPC scheme that ensures the ellipsoids described by (6.6) are contained within the MPC feasible set.

6.2.1 Inclusion constraint

Consider the constraint

$$\mathbf{h}_i\left(\mathbf{x}_k, \mathbf{u}_k\right) \leq 0 \tag{6.9}$$

for some index i and suppose that we want to formulate a computationally tractable constraint requiring that (6.9) is satisfied for all state deviations in the ellipsoid (6.6). For nonlinear constraints, this is unfortunately very difficult. As a result, in line with the philosophy above, we will consider the approximation resulting from linearizing the constraint on the nominal trajectory $\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k$ of the MPC scheme.

More specifically, we will consider imposing the constraint:

$$\mathbf{h}_i\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) + \left(\frac{\partial \mathbf{h}_i}{\partial \mathbf{x}} - \frac{\partial \mathbf{h}_i}{\partial \mathbf{u}} K_k\right)\Big|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k} \Delta \mathbf{x}_k \leq 0, \quad \forall\, \Delta \mathbf{x}_k \in R_k \tag{6.10}$$

Fortunately, this requirement takes a closed form. Indeed, we can simply seek the $\Delta \mathbf{x}_k \in$ (6.6) that gives the worst (highest) value to $\mathbf{h}_i$. Let us label

$$\mathbf{b}_i = \left(\frac{\partial \mathbf{h}_i}{\partial \mathbf{x}} - \frac{\partial \mathbf{h}_i}{\partial \mathbf{u}} K_k\right)^\top$$

Then (6.10) can be (tightly) enforced by imposing that:

$$\mathbf{h}_i + \sqrt{2\sigma_k}\left(\mathbf{b}_i^\top \Sigma_k \mathbf{b}_i\right)^{\frac{1}{2}} \leq 0 \tag{6.11}$$

which is a constraint mixing $\mathbf{h}$ and its derivative, the dynamics of Σ_k and the given σ_k.

6.2.2 Robust MPC formulation

We can now formulate an approximate robust NMPC scheme:

$$\begin{aligned}
V_{\boldsymbol{\theta}}(\mathbf{s}) = \min_{\bar{\mathbf{x}}, \bar{\mathbf{u}}, \boldsymbol{\zeta}, \Sigma} \quad & \gamma^N \left(T_{\boldsymbol{\theta}}\left(\bar{\mathbf{x}}_N\right) + \mathbf{w}_f^\top \boldsymbol{\zeta}_N\right) \\
& + \sum_{k=0}^{N-1} \gamma^k \left(l_{\boldsymbol{\theta}}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) + \mathbf{w}^\top \boldsymbol{\zeta}_k\right) && \text{(6.12a)} \\
\text{s.t.} \quad & \bar{\mathbf{x}}_{k+1} = \mathbf{f}_{\boldsymbol{\theta}}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k, \mathbf{d}_k\right), \quad \bar{\mathbf{x}}_0 = \mathbf{s} && \text{(6.12b)} \\
& \Sigma_{k+1} = A_k \Sigma_k A_k^\top + B_k \Lambda B_k^\top, \quad \Sigma_0 = S_0 && \text{(6.12c)} \\
& \mathbf{h}_i\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) + \sqrt{2\sigma_k}\left(\mathbf{b}_i^\top \Sigma_k \mathbf{b}_i\right)^{\frac{1}{2}}\Big|_{\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k} \leq \boldsymbol{\zeta}_k && \text{(6.12d)} \\
& \mathbf{h}_i^{\mathrm{f}}\left(\bar{\mathbf{x}}_N\right) + \sqrt{2\sigma_N}\left(\mathbf{b}_i^\top \Sigma_N \mathbf{b}_i\right)^{\frac{1}{2}}\Big|_{\bar{\mathbf{x}}_N} \leq \boldsymbol{\zeta}_N && \text{(6.12e)} \\
& \boldsymbol{\zeta}_{0,\ldots,N} \geq 0 && \text{(6.12f)}
\end{aligned}$$

where, $0 < \gamma \leq 1$ is a discount factor and $T_{\boldsymbol{\theta}}$ is an adjustable terminal cost function. The slack variables and initial guess for the covariance matrix of the state deviation are labeled by $\boldsymbol{\zeta}$ and S_0, respectively. Although we initialize S_0 at zero in this chapter, One can use an observer scheme to estimate this initial

matrix. The adjustable stage cost function in the above RNMPC scheme is defined as follows:

$$l_{\boldsymbol{\theta}}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) = L\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right) + \varphi_{\boldsymbol{\theta}}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k, \Sigma_k\right), \tag{6.13}$$

where, $\varphi_{\boldsymbol{\theta}}\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k, \Sigma_k\right) = \mathrm{Tr}\left(\frac{\partial^2 L(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k)}{\partial \bar{\mathbf{x}}_k^2} M \Sigma_k\right)$ is proposed to adopt as a cost modification term in the stage cost $L\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k\right)$ of the RNMPC scheme. This term is considered to deliver the impact of the uncertainty on the adopted stage cost. We propose to use a quadratic form for this term, which includes the adjustable matrix M as an RL parameter. $\mathbf{f}_{\boldsymbol{\theta}}$ is the nonlinear dynamics, which could be adjusted by RL through the model bias parameters. Because the robust NMPC scheme (6.12) uses approximations and a possibly inaccurate model of the system dynamics and disturbances, it may become infeasible. To address that issue, an ℓ_1 relaxation of the inequality constraints is introduced in (6.12f). If the weights $\mathbf{w}, \mathbf{w}_f$ are chosen large enough, then the solution of the RNMPC scheme (6.12) respects the constraints when it is feasible to do so.

We ought to specify here that the proposed approximate RNMPC may be interpreted either as a genuine approximate robust NMPC scheme or an approximate stochastic NMPC scheme depending on the nature of the real system disturbances. Indeed, even if the real system does not yield any approximation in (6.12), i.e. the disturbances are Gaussian and the system dynamics and constraints are linear, then this scheme ensures the constraint satisfaction in the sense of the probability (6.8), and shall therefore be seen as a stochastic NMPC scheme. If the disturbances are bounded and properly covered by the ellipsoidal model used in (6.12), then the proposed scheme can be seen as a robust NMPC scheme. In both cases, the choice of parameters in (6.12) to model the ellipsoids so as to achieve the given control objectives is difficult.

The NMPC parameters $\boldsymbol{\theta} = \left\{M, \bar{\mathbf{d}}_k, \Lambda, \sigma_k\right\}$ will be adjusted using LSTDQ-learning. We propose to use (6.12) as an approximator for the true value function $V_\star$, i.e. we will seek the robust MPC parameters $\boldsymbol{\theta}$ that best achieve $V_{\boldsymbol{\theta}} \approx V_\star$. Let us define the control policy as:

$$\boldsymbol{\pi}_{\boldsymbol{\theta}}(\mathbf{s}) = \bar{\mathbf{u}}_0^\star \tag{6.14}$$

where, $\bar{\mathbf{u}}_0^\star$ is the first element of the input sequence $\bar{\mathbf{u}}_0^\star, \cdots, \bar{\mathbf{u}}_{N-1}^\star$ solution of (6.12). We next consider this optimal policy delivered by the RNMPC scheme as an action $\mathbf{a}$ in the context of reinforcement learning where it is selected according to the above policy with the possible addition of exploratory moves [11]. As a result in [25], a parameterized NMPC scheme can be used as an (action)-value function approximator in the context of the reinforcement learning. Then,

we propose to use the parameterized RNMPC scheme (6.12) to deliver the value function needed in the proposed second-order LSTDQ-learning algorithm. Then, the action-value function in (6.15) results from solving the same RNMPC scheme with its first input constrained to the delivered action $\mathbf{a}$.

$$Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{a}) = \min_{\bar{\mathbf{x}},\bar{\mathbf{u}},\boldsymbol{\zeta},\Sigma} \quad (6.12a) \tag{6.15a}$$

$$\text{s.t.} \quad (6.12b) - (6.12f) \tag{6.15b}$$

$$\bar{\mathbf{u}}_0 = \mathbf{a} \tag{6.15c}$$

6.3 RL-based Robust NMPC

In this section, we propose to use RL to tune the approximate robust NMPC scheme. Indeed, the proposed robust NMPC scheme is based on coarse approximations of the state uncertainty propagation, which are likely to yield suboptimality and even constraints violations in practice. Finding the parameters that yield the best closed-loop performance for the real system in terms of cost and constraints violations is a difficult problem. We propose to use RL to find these parameters. More specifically, RL is used to tune the sequences of σ_k and the expected value of the noise $\bar{\mathbf{d}}_k$, the process noise Λ and the matrix M in the cost modification $\varphi_{\boldsymbol{\theta}}$ in order to improve the Robust NMPC scheme. RL allows one to perform this tuning based on observed state transition, and without any knowledge of the real system stochasticity.

In this section, we first present the algorithmic details needed to implement a second order LSTDQ-learning algorithm on the ellipsoidal RNMPC scheme. Then the sensitivity analysis needed in the reinforcement learning scheme is described.

6.3.1 Second-Order LSTDQ Learning

LSTDQ-based reinforcement learning methods make an efficient use of data and tend to converge faster than more basic temporal-difference learning methods. Let us form the least squares of Bellman residual error w.r.t $\boldsymbol{\theta}$ as:

$$\min_{\boldsymbol{\theta}} \mathbb{E}\left[\left(Q_\star(\mathbf{s},\mathbf{a}) - Q_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{a})\right)^2\right] \tag{6.16}$$

where $Q_{\boldsymbol{\theta}}$ obtained from the RNMPC scheme in (6.15) is an approximation of the true action-value function $Q_\star$. Let us consider the following approximation of the Bellman optimality equation, which is used in the Temporal Difference (TD)-based learning approaches.

$$Q_\star(\mathbf{s},\mathbf{a}) \approx L(\mathbf{s},\mathbf{a}) + \gamma \underbrace{\min_{\mathbf{a}'} Q_{\boldsymbol{\theta}}(\mathbf{s}_+,\mathbf{a}')}_{V_{\boldsymbol{\theta}}(\mathbf{s}_+)} \tag{6.17}$$

where $\mathbf{a}' = \boldsymbol{\pi}(\mathbf{s}_+)$ and $L(\mathbf{s}, \mathbf{a})$ is the baseline cost in the RL scheme. Substituting (6.17) into (6.16), the least square problem (6.16) for the first-order LSTDQ-learning can be solved for some data acquired from the transitions $\mathbf{s}, \mathbf{a} \rightarrow \mathbf{s}_+$ as:

$$\mathbb{E}\left[\delta \nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a})\right] = 0, \tag{6.18a}$$

$$\delta = L(\mathbf{s}, \mathbf{a}) + \gamma V_{\boldsymbol{\theta}}(\mathbf{s}_+) - Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}) \tag{6.18b}$$

where δ is the temporal difference error. Both value and action-value functions, respectively, $V_{\boldsymbol{\theta}}(\mathbf{s})$ and $Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a})$ are obtained from the RNMPC schemes (6.12) and (6.15). In this chapter, we adopt a Newton method to solve (6.18) and extract the newton step for the second-order LSTDQ scheme as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha A^{-1} b, \tag{6.19a}$$

$$A = \mathbb{E}\left[\delta \nabla^2_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}} + \nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}} \nabla^\top_{\boldsymbol{\theta}} \delta\right], \quad b = \mathbb{E}\left[\delta \frac{\partial Q_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}\right] \tag{6.19b}$$

where, the scalar $\alpha > 0$ is the step size. Figure 6.1 shows an overview of the proposed RNMPC-based RL.

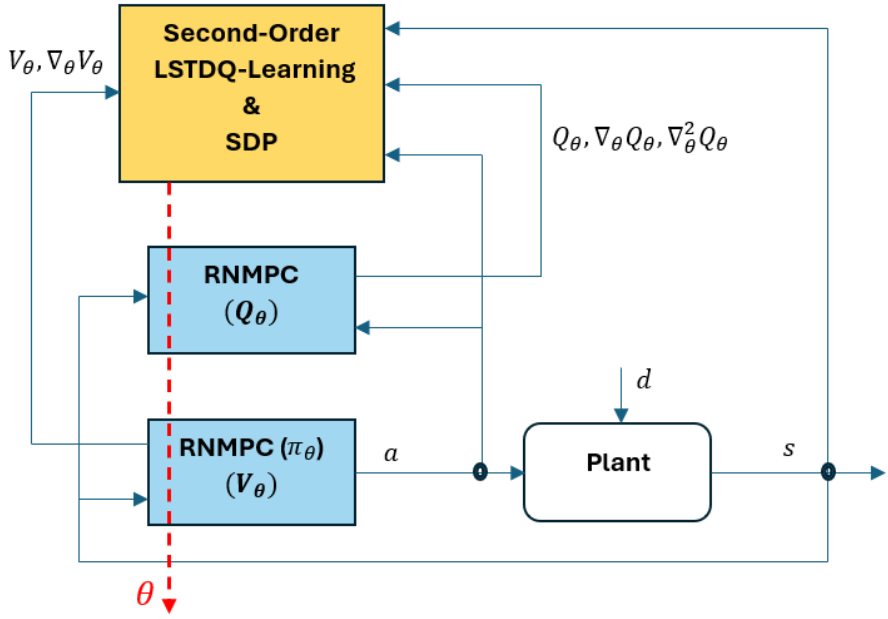


Figure 6.1: An overview of the RNMPC-based LSTDQ-learning. The state and action value functions $V_{\boldsymbol{\theta}}, Q_{\boldsymbol{\theta}}$ are approximated by (6.12) and (6.15), respectively. The action a is selected according to the policy π_θ with the possible addition of exploratory moves. The SDP (6.22) is used to satisfy some requirements in the RL steps.

6.3.2 Sensitivity Analysis

The gradient and Hessian of the function $Q_{\boldsymbol{\theta}}$ needed in (6.19) require one to compute the sensitivities of the optimal value of NLP (6.15). Let us define the Lagrange function $\mathcal{L}_{\boldsymbol{\theta}}$ associated to the RNMPC problem (6.15) as follows:

$$\mathcal{L}_{\boldsymbol{\theta}} = \Phi_{\boldsymbol{\theta}} + \boldsymbol{\lambda}^\top G_{\boldsymbol{\theta}} + \boldsymbol{\mu}^\top H_{\boldsymbol{\theta}} \tag{6.20}$$

where $H_{\boldsymbol{\theta}}$ gathers the inequality constraints of (6.15) and $\Phi_{\boldsymbol{\theta}}$ is the cost of the RNMPC optimization problem. Variable $\boldsymbol{\lambda}$ is the Lagrange multiplier vector associated to the equality constraints $G_{\boldsymbol{\theta}}$ of the RNMPC. Variable $\boldsymbol{\mu}$ is the Lagrange multiplier vector associated to the inequality constraints of the RNMPC scheme. Let us label the primal variables for the RNMPC scheme as $\mathbf{p} = \{\mathbf{X}, \mathbf{U}\}$, where $\mathbf{X}, \mathbf{U}$ are the state and control input trajectories predicted by (6.15). Then, the primal-dual variables read as $\mathbf{z} = \{\mathbf{p}, \boldsymbol{\lambda}, \boldsymbol{\mu}\}$.

The sensitivity analysis of optimization problems detailed in [42] delivers the Gradient $\nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}$ and Hessian $\nabla^2_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}$ ($H(Q_{\boldsymbol{\theta}})$) terms needed in (6.19) as follows:

$$\frac{\partial Q_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{L}_{\theta}(\mathbf{s}, \mathbf{a}, \mathbf{z}^\star)}{\partial \boldsymbol{\theta}}, \tag{6.21a}$$

$$H(Q_{\boldsymbol{\theta}}) = \frac{D}{D\boldsymbol{\theta}} \left(\frac{\partial \mathcal{L}_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}, \mathbf{z}^\star)}{\partial \boldsymbol{\theta}} \right) \approx \frac{\partial^2 \mathcal{L}_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{a}, \mathbf{z}^\star)}{\partial \boldsymbol{\theta}^2} \tag{6.21b}$$

where $\mathbf{z}^\star$ is the primal-dual solution of (6.15) and D is the total derivative. The Hessian term $H(Q_{\boldsymbol{\theta}})$ can improve the search direction using the curvature information and thus make more progress per step in comparison with the first-order LSTDQ learning relying only on the gradient calculation.

6.3.3 Constrained RL steps

In the proposed RL-RNMPC scheme, there is a matrix Λ in the dynamics (6.12c), which is adjusted using the reinforcement learning method. As a requirement, this disturbance covariance matrix must be positive semidefinite. However, the RL steps delivered by second-order LSTDQ learning do not necessarily respect this requirement, and we need to enforce it via constraints on the RL steps throughout the learning process. To address this requirement, we formulate a Semi-Definite Program (SDP) as a least squares optimization problem:

$$\min_{\Delta\boldsymbol{\theta}} \quad \frac{1}{2} \|\Delta\boldsymbol{\theta}\|^2 - F^\top \Delta\boldsymbol{\theta} \tag{6.22a}$$

$$\text{s.t.} \quad \Lambda(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) \geq 0 \tag{6.22b}$$

where we assume that the above matrix is linear function of $\boldsymbol{\theta}$ and the gradient term $F = -\alpha A^{-1} b$ is delivered by the second-order LSTDQ-learning algorithm (6.19).

6.4 Numerical Example

To illustrate proposed RL-RNMPC, we consider a simulated Wheeled Mobile Robot (WMR) tracking trajectories while avoiding static obstacles avoidance, and affected by model uncertainties and disturbances. Let us define the WMR model as:

$$\mathbf{f}(\mathbf{x},\mathbf{u}) = \begin{bmatrix} \cos(\psi) & 0 \\ \sin(\psi) & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u} \tag{6.23}$$

where $\mathbf{x} = [x, y, \psi]^\top$ and $\mathbf{u} = [v, \omega]^\top$ are the state and control input vectors, respectively. The position coordinates of the WMR are labeled x, y and ψ is the robot orientation angle. The initial position of the mobile robot is $\mathbf{x}_0 = [-1, 2, 0]^\top$. The control inputs v and ω are the linear and angular velocities, respectively. To discretize the above continuous model, we adopt a fourth-order Runge-Kutta (RK4) integrator providing discretized function $\mathbf{f}_d$ of the WMR model. We will consider that the real system evolves according to the dynamics:

$$\mathbf{x}(k+1) = \mathbf{f}_d\left(\mathbf{x}(k), \mathbf{u}(k) + \mathbf{\Gamma}_1(k)\right) + \mathbf{\Gamma}_2(k) \tag{6.24}$$

where the two variables $\mathbf{\Gamma}_1, \mathbf{\Gamma}_2$ in the above equations model the uncertainties as:

$$\mathbf{\Gamma}_1(k) = v(k) \begin{bmatrix} d_1(k) \\ d_2(k) \end{bmatrix}, \quad \mathbf{\Gamma}_2(k) = T_s v(k) \begin{bmatrix} 0 \\ 0 \\ d_2(k) \end{bmatrix} \tag{6.25}$$

where $d_1(k) \sim \mathcal{N}\left(0, \Sigma_1^2\right)$ and $d_2(k) \sim \mathcal{N}\left(0, \Sigma_2^2\right)$. The sampling time is T_s= 0.2 s and disturbance variances Σ_1 and Σ_2 are set as 0.2 and 0.4, respectively. We initialize the adjustable process noise covariance matrix (3×3) as $\Lambda = diag(\Sigma_1^2, \Sigma_2^2, \Sigma_2^2)$. The expected value of the process noise $\bar{d}$ and the matrix M as an RL parameter in the cost modification term φ_θ are initialized at zero. According to (6.8), the probability of the uncertainties in the ellipsoids can be obtained from the Gamma functions. Since the simulated mobile robot has three states, there is a 3-dimensional multinormal distribution $n = 3$. To ensure a large enough probability of the uncertainty propagation in the ellipsoids (confidence regions), we set $\sigma_k = 2.65$. Therefore, with pair $n = 3, \quad \sigma_k = 2.65$ we will have the probability of the state deviations to belong to the ellipsoids $\mathbb{P}[\Delta x_k \in R_k] \approx 93\%$.

The RMPC model will be based on the real system dynamics, assuming that we know them perfectly. More specifically, the nominal model used in the RNMPC scheme is based on the expected value of the exact distribution of the disturbances d_1, d_2, and the structure according to which the disturbances enter the system is

assumed known. We further observe that by assuming the disturbances are Gaussian, we are considering an ideal setup for the approximate robust MPC to perform well. Hence any gain of performance achieved by RL in this simulation setup is done through handling the approximations introduced in the robust MPC scheme. As it is proposed to adopt the RL approach in order to adjust the RNMPC parameters, we formulate a second-order LSTDQ scheme with a batch size equal to 100 data samples and let RL to update the parameters based on the collected data in the batch. In this simulation, the number of transitions (iterations) is 8000 and thus the number of RL steps is 80. The step size is set as $\alpha = 10^{-6}$. We adopt a baseline stage cost used in the LSTDQ scheme as:

$$L(\mathbf{x}_k, \mathbf{u}_k) = l(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}^\top \max(0, \mathbf{h}(\mathbf{x}_k)) \tag{6.26}$$

where $l(\mathbf{x}_k, \mathbf{u}_k)$ can be expressed as a quadratic function of the state and action deviations from their desired values. The second term in the above baseline is considered to cope with the violations, where $\mathbf{h} \geq 0$ is pure inequality vector of constraints induced by the obstacles. The penalty weights are $\mathbf{w}^\top = [30, 30, 30]$.

The obstacles are represented as ellipsoids (see Fig. 6.2). The reference trajectory shown in Fig. 6.2 is an eight-shaped path, which intersects the obstacles or comes very close to them.

In Fig. 6.3, we compare the closed-loop performance of the nominal NMPC scheme (6.2), the approximate robust NMPC without learning, and with learning. The second-order LSTDQ-learning algorithm improves the performance of the robust NMPC scheme in comparison with the classic NMPC and RNMPC without learning.

Since the approximate RNMPC scheme is built based on an exact knowledge of the disturbance statistics and structure, and system model, the improvement of performance observed in Fig. 6.3 results purely on improving the approximation performed in the RNMPC scheme via modifying the robust NMPC parameters $\boldsymbol{\theta} = \{M, \bar{\mathbf{d}}_k, \Lambda, \sigma_k\}$. For majority of the ellipsoids, the dimension of the ellipsoid-shaped confidence regions R_k is changed by RL to be modelled as larger than the originally selected σ_k in order to reduce the risk of constraints violation, see Fig. 6.6. For some situations that the risk of hitting is not high, this dimension is decreasing using the RL. Indeed, there is a trade-off made by the RL between the avoidance and tracking accuracy in this proposed RL-RNMPC. The adjustment of the RNMPC can also contribute the mobile robot to avoid hitting the obstacles 1,2 due to possible drifting (as an uncertainty) in the road bend shown in Fig. 6.2. The RL scheme additionally modifies the uncertainty model via $\bar{d}$ and Λ, see Fig. 6.5, in order to gain performance.

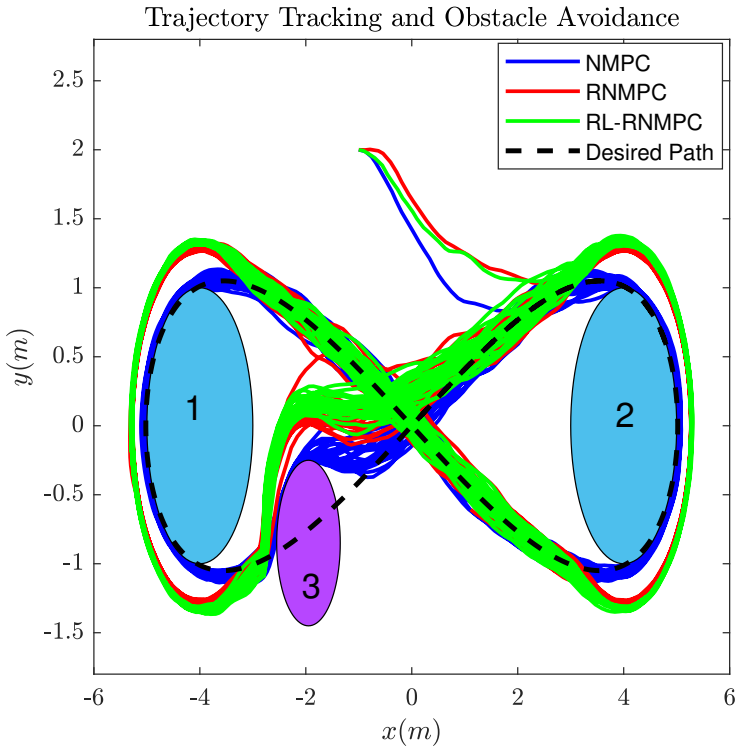


Figure 6.2: A comparative study: Trajectory tracking and obstacle avoidance for a WMR under uncertainties.

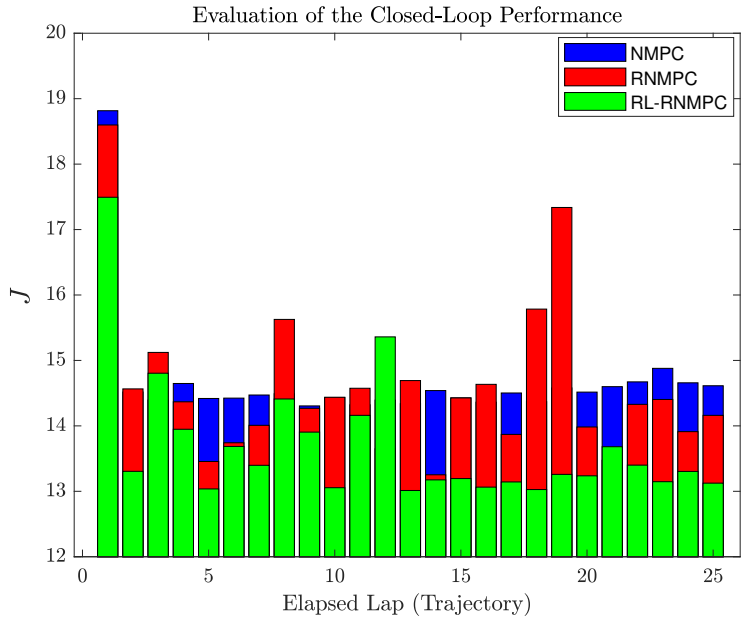


Figure 6.3: Average Closed-Loop performance index for 25 elapsed trajectories (laps).

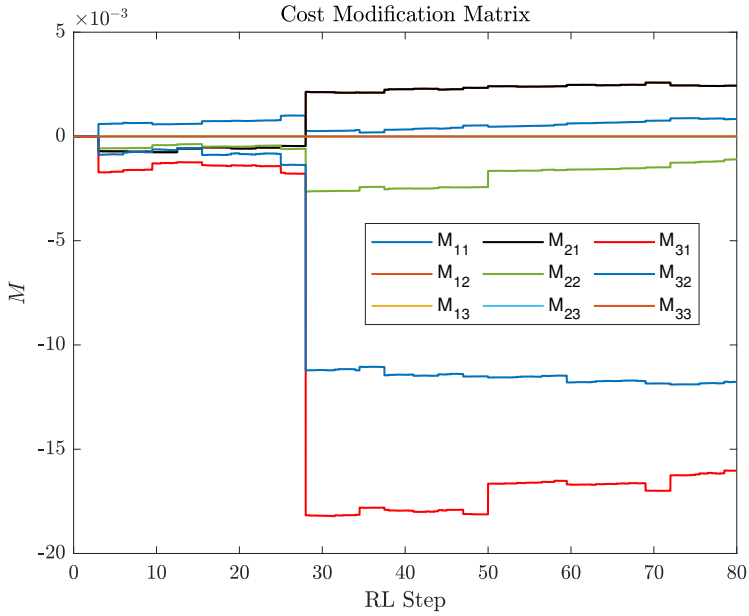


Figure 6.4: Matrix M is adjusted as a RL parameter $\boldsymbol{\theta}$ in the cost modification term $\varphi_\theta\left(\bar{\mathbf{x}}_k, \bar{\mathbf{u}}_k, \Sigma_k\right)$.

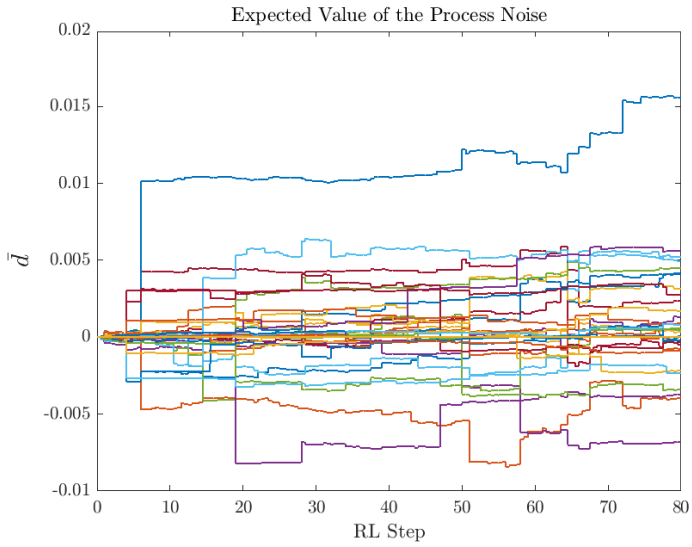


Figure 6.5: Tuning the expected values of the process noises applied to the three states for a prediction horizon $N = 15$. The RL then modifies the uncertainty model via $\bar{d}$.

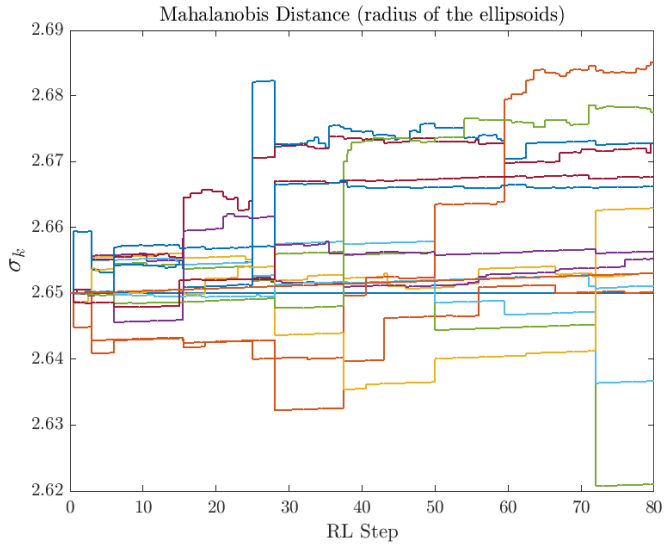


Figure 6.6: Tuning the ellipsoidal confidence region (adjustment of the radius σ_k). For majority of the ellipsoids, the confidence regions (R_k in (6.6)) is changed by RL to be modelled as larger than the originally selected σ_k in order to reduce the risk of constraints violation. For some situations that the risk of hitting is not high, this dimension is decreasing using the RL.

6.5 Conclusion

In the context of the Robust Nonlinear Model Predictive Control (RNMPC), the formal RNMPC techniques are difficult to implement on nonlinear systems, and thus it is common to use approximate RNMPC methods instead. In this chapter we proposed to describe the propagation of the uncertainties using the ellipsoidal tubes in which the disturbances and state deviations are modeled as a Gaussian noise. However, the approximated models and constraints used in this kind of RNMPC can affect the closed-loop performance and thus we adopted an LSTDQ learning as a fast RL algorithm to adjust some crucial parameters of the approximate RNMPC. In this chapter, we used the proposed RNMPC as a value function approximator for the LSTDQ algorithm. AS a future work, we will propose to embed a Linear Quadratic Regulator (LQR) in the proposed RNMPC in order to deliver the first guess of the control gain matrix for the adopted linear feedback over the state deviation. Furthermore, we will investigate the proposed RL-RNMPC for a large-scale application.

Part III

Concluding Remarks

Chapter 7

Conclusions, Limitations and Future Possibilities

7.1 Conclusion

The combination of MPC and RL has shown promising results for a wide range of applications such as autonomous ships, mobile robots, smart buildings, chemical reactors, and smart grids. For Autonomous Surface Vessels (ASVs) and mobile robots, the objective of the MPC-based RL is to find an optimal policy that minimizes the closed-loop performance of a mission such as collision-free path following, autonomous docking, and a skillful transition between them. To control the indoor temperature of smart buildings, an accurate building model plays a crucial role in the the model-based control approaches. However, it is very difficult to provide such an accurate model required in, i.e., an MPC scheme since there are some complex dynamics, uncertainties and external disturbances that may not be captured. Hence, the MPC-based RL allows one to use a simplified model while the degradation of control performance can be compensated by letting RL adjust the entire MPC scheme. Given the difficulties posed by highly uncertain user demand and stochastic local power consumption/production in smart grids, the MPC-based RL method is adopted to seek an optimal smart-grid policy that minimizes the long-term economic costs, including the spot-market cost and the peak-power cost.

In practice, all the states of a dynamical system may not be measurable for the applications above, and one needs to use an observer (estimator) to capture those unmeasurable states. These problems then cannot be formulated as MDPs so that the classic RL methods are no longer able to solve these problems. These prob-

lems are then formulated as POMDPs, where a policy is constructed based on the historic of the available measurements rather than on the full state of the system. To provide a framework in the context of MPC-based RL to solve POMDPs, we then use an MHE scheme combined with the MPC-based RL. The MHE scheme is a well-known state estimator that optimizes a moving finite-horizon cost based on a model of the real system in order to find the best estimations of the unmeasurable states. Similar to the concept that RL can adjust the MPC parameters, RL methods can also be used to adjust a parameterized MHE scheme to compensate for the possible model mismatch.

In Part I of the thesis, we showed that a combined MHE/MPC-based RL can be used to solve POMDPs even if the underlying models used in both the MHE and MPC schemes cannot capture the real system perfectly. Moreover, we showed that the proposed learning-based observer-controller framework is able to deal with the reduced models, where the dimension of the state space model of the real system does not match the real system (all the states of the real system are not available for the MHE-MPC models). To deeply investigate the learning-based state estimation problems, we studied the case that the MHE scheme cannot deliver the best state estimation due to an imperfect model of the real system. We then presented the novel idea of modifying the MHE cost function so that the estimation performance is enhanced even if the MHE model is imperfect. We practically established that the proposed modification of the MHE cost can be achieved using RL. Finally, we used the proposed MHE/MPC-based RL algorithm for the systems formulated in the LPV framework with inexact scheduling parameters (as exogenous signals with unknown bounds). We first proposed to adopt an MHE scheme to simultaneously estimate the convex combination vector and unmeasured states based on the observations and model matching error. To tackle the wrong Linear Time Invariant (LTI) models (vertices) used in both the MPC and MHE schemes, we then proposed a policy gradient method based on the MHE-MPC scheme in order to learn both the estimator (MHE) and controller (MPC) schemes.

In Part II of the thesis, we proposed to combine an approximate robust NMPC with RL to achieve the best closed-loop performance by adjusting the parameters used in the robust MPC. More specifically, the proposed RNMPC is constructed based on an approximate model of the propagation of perturbations in the state dynamics. This approximate model describes ellipsoids of state uncertainty propagation, which are constructed based on the linearization of the system dynamics and constraints on the nominal trajectories and using a Gaussian disturbance model. We then proposed to adjust this scheme using the RL method in order to tailor this inaccurate uncertainty model to the real system and achieve a better closed-loop performance. We also proposed to use a second-order Least Square Temporal

Difference Q-learning (LSTDQ) to achieve a faster convergence of the adjustable parameters of RNMPC.

7.2 Limitations

In this section, we discuss the restrictions of the proposed learning-based control/estimation approaches and provide some potential solutions for addressing them.

- In this thesis, we adopted the core idea of using MPC as function approximators to develop an MHE/MPC-based reinforcement learning method. The central theorem in [36] showed that the MPC scheme can deliver the optimal policy and value functions even if the underlying MPC model is inaccurate. In this thesis, we also showed that a true state estimation can be obtained by modifying the MHE state cost function using an imperfect MHE model. However, these observations raise the natural question about the role of the models used in both the MHE and MPC schemes if they do not need to be accurate. Hence, this issue restricts the use of any model in the MHE-MPC scheme so that one needs to take some requirements into consideration for exploiting the simplified models. As discussed in [94], Assumption (13) of Theorem 1, the most obvious insight lies in the finite value function for all the model trajectories such that it resembles the stability of the model under the optimal policy. Analogously, we made the mild Assumption (1) in Chapter 4, in equation (4.10), to address the requirement above for the modified MHE scheme with imperfect model. Furthermore, one can also use the Robust MPC schemes to build safe policies in RL. However, more work is required to investigate the concept of a robust estimation/control approach for the combined MHE-MPC scheme aiming at providing a safe RL framework.

- In Chapter 3, we proposed an MHE/MPC-based RL method for POMDPs. Although the results establish the efficacy of the proposed approach in improving the closed-loop performance, it is not trivial to mathematically explain the role of the modified MHE scheme, where a full state-space model is not used in the MHE and MPC schemes. However, in Chapter 4, we showed that a true state estimation can be obtained by modifying the MHE stage cost function, but more work is necessary to interpret a modification of the MHE scheme for POMDPs. Furthermore, it should be investigated how the estimation error due to reduced model can be quantified and incorporated into the proposed RL framework for POMDPs aiming at improving both the state estimation and the closed-loop performance.

- In this thesis, the proposed MHE/MPC-based RL methods cannot be used for the large-scale networked and multi-agent system in a cooperative or distributed manner. More specifically, the proposed method should be extended to Multi-Agent Reinforcement Learning (MARL) framework to deal with multi-agent systems in particular for cooperative control systems with coupling constraints. One potential solution is to develop a cooperative MARL based on Distributed MPC (DMPC) scheme, which can be formulated in both the Q-learning and Policy Gradient (PG) frameworks. More precisely, the structure of the DMPC can be leveraged to introduce some local MPC-based value function approximators required in a cooperative reinforcement learning. In the context of PG, one can readily use the same structure to capture the local optimal policies, which are delivered from the local MPC schemes.

- The proposed MHE/MPC-based RL methods are gradient-based methods that require frequent evaluations of computationally expensive MPC schemes. Hence, these methods may struggle a bit in the real-time application, in particular for those cases with very short sampling times. More precisely, the real-time computational burden comes mostly from solving the MPC (computing the sensitivity is inexpensive). Nonetheless, the progress in the optimization algorithms and in the computational hardware makes the deployment of real-time MPC possible for most of the real applications. Moreover, in the proposed algorithms, we verified the effectiveness of our proposed approach where a simplified model is used in both the MHE and MPC schemes such that the corresponding optimization problems are no longer computationally expensive. In the context of learning from existing big data, it is a challenging issue to use RL methods based on the MHE-MPC scheme. More specifically, performing RL for MHE-MPC on big data can be impractical due to amount of computational time required. This issue was partially addressed in [95], where the task of learning a high-performing MPC scheme from data was reduced to a supervised learning problem and it is solved at a low computational expense as compared with the existing MPC-based RL methods. However, more work is required to fully address the computational complexity of RL based on the combined MHE-MPC, in particular for those cases that neural networks are incorporated into the MHE-MPC scheme.

7.3 Future Works

Some of the possible future research directions are provided below:

- **MPC-based RL in the LPV Framework using Data-Driven Models:**

In this thesis, we used the MHE/MPC-based RL in the LPV framework for a simple example, where an imperfect model of the real system was used in both the MHE and MPC schemes. However, providing an even simplified LPV-State Space (LPV-SS) model of the underlying system may be challenging in particular for large-scale nonlinear systems affected by uncertainties and external disturbances. Despite decades of research to acquire comprehensive knowledge of real-world systems, uncertainty remains and observed phenomena still cannot be sufficiently described by knowledge-based models. To address this problems, the data-driven methods have been recently used for the LPV-SS model identification. However, the distribution of the data in the application environment can be different from the distribution of the training data, and thus the accuracy of the model may decrease for applications. Moreover, the data may not represent the behaviors of the system due to the limitation of data collection, which results in the model uncertainty. To tackle the closed-loop performance degradation due the model mismatch (the mismatch between a data-driven model of a system and its actual dynamics), one can then use the MPC-based RL in order to efficiently tune the parameters of observer/controllers.

- **MPC-based RL in the LPV Framework using Tube-based MPC:**

  In the LPV framework, the current scheduling parameters $\boldsymbol{\rho}_k$ can be measured for all times k while the future behavior of $\boldsymbol{\rho}$ is basically not known exactly at time k. Solving a predictive control problem under uncertainty then requires the on-line optimization over feedback policies, leading to a so-called min–max feedback control problem. Another different paradigm devised to reduce complexity with respect to the min–max solution, is tube-based MPC. This robust MPC scheme has been recently used in the LPV framework. However, to be able to construct tube-based controllers for LPV systems that can achieve better complexity/performance trade-offs, new approaches for designing parameterized tube can be adopted. Then, one can investigate the use of the MPC-based RL for adjusting this parameterized tube.

- **Data-Driven Ellipsoidal MPC Combined with RL:**

  In the context of ellipsoidal MPC as an approximate robust MPC, a meaningful extension could be to complement RL with some more classic identification methods, i.e. use the data to refine the ellipsoidal model and account for the fact that the construction is based on a linearization of the trajectories.

- **Safety-Critical Stochastic MPC-bsed RL using Control Barrier Function:**

As another extension for the ellipsoidal robust MPC, one can use this scheme in a stochastic setting combined with a stochastic Control Barrier Function (CBF) introduced as a chance-constraint in the MPC scheme. Then this stochastic CBF can be approximated by a deterministic ellipsoidal constraint so that we can cope with the problems induced by the uncertainties in both the MPC and CBF models. However, the proposed stochastic framework is constructed based on the approximate model of the uncertainty propagation via a deterministic model of the stochastic chance-constraint CBF. Moreover, one can assume that the CBF model used in the MPC scheme cannot capture the true CBF in the real environment. To tackle these problems, a parameterized version of the stochastic CBF and MPC scheme is then introduced, where the corresponding parameters are adjusted by RL to improve the closed-loop performance in the presence of model uncertainties and unknown CBFs.

Bibliography

[1] R. Halvgaard, N. K. Poulsen, H. Madsen, and J. B. Jørgensen, "Economic model predictive control for building climate control in a smart grid," in *2012 IEEE PES Innovative Smart Grid Technologies (ISGT)*, pp. 1–6, 2012.

[2] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, "Learning-based model predictive control for safe exploration," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 6059–6066, IEEE, 2018.

[3] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-time neural MPC: Deep learning model predictive control for quadrotors and agile robotic platforms," *IEEE Robotics and Automation Letters*, vol. 8, pp. 2397–2404, apr 2023.

[4] S. A. Munoz, J. Park, C. M. Stewart, A. M. Martin, and J. D. Hedengren, "Deep transfer learning for approximate model predictive control," *Processes*, vol. 11, no. 1, 2023.

[5] Z. Wu, D. Rincon, Q. Gu, and P. D. Christofides, "Statistical machine learning in model predictive control of nonlinear processes," *Mathematics*, vol. 9, no. 16, 2021.

[6] J. Kocijan, R. Murray-Smith, C. Rasmussen, and A. Girard, "Gaussian process model based predictive control," in *Proceedings of the 2004 American Control Conference*, vol. 3, pp. 2214–2219 vol.3, 2004.

[7] M. Maiworm, D. Limon, and R. Findeisen, "Online learning-based model predictive control with gaussian process models and stability guarantees," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8785–8812, 2021.

[8] S. Gros and M. Zanon, "Data-driven economic NMPC using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2020.

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

[10] M. Tenny and J. Rawlings, "Efficient moving horizon estimation and nonlinear model predictive control," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, vol. 6, pp. 4475–4480 vol.6, 2002.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[12] K. Azizzadenesheli, *Reinforcement Learning in Structured and Partially Observable Environments*. PhD thesis, University of California, Irvine, 2019.

[13] S. Gros, "Reinforcement learning in optimal control: Current and future research," *Institute of Flight System Dynamics, Technische Universität München (TUM)*, October 2019.

[14] R. A. Howard, *Dynamic programming and Markov processes*. John Wiley, 1960.

[15] M. G. Lagoudakis, R. Parr, and M. L. Littman, "Least-squares methods in reinforcement learning for control," in *Methods and Applications of Artificial Intelligence* (I. P. Vlahavas and C. D. Spyropoulos, eds.), (Berlin, Heidelberg), pp. 249–260, Springer Berlin Heidelberg, 2002.

[16] D. P. Bertsekas, *Reinforcement Learning and Optimal Control*. Belmont: Athena Scientific, Athena Scientific Optimization and Computation Ser. 1., 2019.

[17] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proceedings of the 31st International Conference on International Conference on Machine Learning*, p. I–387–I–395, JMLR.org, 2014.

[18] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2 edition, 2006.

[19] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *Journal of machine learning research*, vol. 4, pp. 1107–1149, 2003.

[20] M. Zanon, S. Gros, and M. Palladino, "Stability-constrained markov decision processes using MPC," *Automatica*, vol. 143, p. 110399, 2022.

[21] H. N. Esfahani, A. B. Kordabad, and S. Gros, "Reinforcement learning based on MPC/MHE for unmodeled and partially observable dynamics," in *2021 American Control Conference (ACC)*, pp. 2121–2126, 2021.

[22] H. N. Esfahani, A. B. Kordabad, and S. Gros, "Approximate robust NMPC using reinforcement learning," in *2021 European Control Conference (ECC)*, pp. 132–137, 2021.

[23] W. Cai, A. B. Kordabad, H. N. Esfahani, A. M. Lekkas, and S. Gros, "MPC-based reinforcement learning for a simplified freight mission of autonomous surface vehicles," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 2990–2995, 2021.

[24] H. N. Esfahani and S. Gros, "Policy gradient reinforcement learning for uncertain polytopic LPV systems based on MHE-MPC," *IFAC-PapersOnLine*, vol. 55, no. 15, pp. 1–6, 2022. 6th IFAC Conference on Intelligent Control and Automation Sciences ICONS 2022.

[25] S. Gros and M. Zanon, "Data-driven economic NMPC using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2019.

[26] Z. D. Guo, M. G. Azar, B. Piot, B. A. Pires, T. Pohlen, and R. Munos, "Neural predictive belief representations," *CoRR*, vol. abs/1811.06407, 2018.

[27] T. Gangwani, J. Lehman, Q. Liu, and J. Peng, "Learning belief representations for imitation learning in POMDPs," in *35th Conference on Uncertainty in Artificial Intelligence*, pp. 1–14, 2019.

[28] X. Zhong, Z. Ni, Y. Tang, and H. He, "Data-driven partially observable dynamic processes using adaptive dynamic programming," in *in Proc. IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pp. 1–8, 2014.

[29] F. Lewis and K. Vamvoudakis, "Reinforcement learning for partially observable dynamic processes: adaptive dynamic programming using measured output data," *IEEE Trans. Syst. Man, Cybern. B, Cybern.,*, vol. 41, no. 1, pp. 14–25, 2011.

[30] H. Jiang and H. He, "Data-driven distributed output consensus control for partially observable multiagent systems," *IEEE Trans. Cybern.*, vol. 49, no. 3, pp. 848–858, 2019.

[31] A. Bahari Kordabad and M. Boroushaki, "Emotional learning based intelligent controller for mimo peripheral milling process," *Journal of Applied and Computational Mechanics*, vol. 6, no. 3, pp. 480–492, 2020.

[32] Y. Wang, K. Velswamy, and B. Huang, "A novel approach to feedback control with deep reinforcement learning," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 31 – 36, 2018. 10th IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2018.

[33] P. Kuhl, M. Diehl, T. Kraus, J. P. Schloder, and H. G. Bock, "A real-time algorithm for moving horizon state and parameter estimation," *Computers and Chemical Engineering*, vol. 35, no. 1, pp. 71 – 83, 2011.

[34] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *arXiv preprint arXiv:1806.10644*, 2018.

[35] B. Karg and S. Lucia, "Learning-based approximation of robust nonlinear predictive control with state estimation applied to a towing kite," in *2019 18th European Control Conference (ECC)*, pp. 16–22, 2019.

[36] S. Gros and M. Zanon, "Data-driven economic NMPC using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2019.

[37] M. Zanon, S. Gros, and A. Bemporad, "Practical reinforcement learning of stabilizing economic MPC," in *2019 18th European Control Conference (ECC)*, pp. 2258–2263, IEEE, 2019.

[38] S. Gros and M. Zanon, "Reinforcement learning for mixed-integer problems based on MPC," *arXiv preprint arXiv:2004.01430*, 2020.

[39] A. Bahari Kordabad, W. Cai, and S. Gros, "MPC-based reinforcement learning for economic problems with application to battery storage," in *2021 20th European Control Conference (ECC) (Accepted)*, IEEE, 2020.

[40] A. B. Kordabad, H. N. Esfahani, A. M. Lekkas, and S. Gros, "Reinforcement learning based on scenario-tree MPC for ASVs," in *2021 American Control Conference (ACC)*, pp. 1985–1990, 2021.

[41] D. Bertsekas, "Dynamic programming and optimal control," *Athena Scientific, 3rd edition*, 2005.

[42] C. Buskens and H. Maurer., *Online Optimization of Large Scale Systems, chapter Sensitivity Analysis and Real-Time Optimization of Parametric Nonlinear Programming Problems, pages 3–16*. Berlin Heidelberg, Berlin, Heidelberg, 2001.

[43] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, "Data-driven model predictive control: closed-loop guarantees and experimental results," *at - Automatisierungstechnik*, vol. 69, no. 7, pp. 608–618, 2021.

[44] S. Muntwiler, K. P. Wabersich, and M. N. Zeilinger, "Learning-based moving horizon estimation through differentiable convex optimization layers," 2021.

[45] B. Wang, Z. Ma, S. Lai, L. Zhao, and T. H. Lee, "Differentiable moving horizon estimation for robust flight control," 2021.

[46] B. Karg and S. Lucia, "Approximate moving horizon estimation and robust nonlinear model predictive control via deep learning," *Computers and Chemical Engineering*, vol. 148, p. 107266, 2021.

[47] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[48] M. Zanon and S. Gros, "Safe reinforcement learning using robust MPC," *IEEE Transactions on Automatic Control*, vol. 66, no. 8, pp. 3638–3652, 2021.

[49] W. Cai, H. N. Esfahani, A. B. Kordabad, and S. Gros, "Optimal management of the peak power penalty for smart grids using MPC-based reinforcement learning," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 6365–6370, 2021.

[50] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1, pp. 99–134, 1998.

[51] X. Xiang and S. Foo, "Recent advances in deep reinforcement learning applications for solving partially observable markov decision processes (pomdp) problems: Part 1—fundamentals and applications in games, robotics and natural language processing," *Machine Learning and Knowledge Extraction*, vol. 3, no. 3, pp. 554–581, 2021.

[52] M. J. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," *CoRR*, vol. abs/1507.06527, 2015.

[53] X. Nian, A. A. Irissappane, and D. Roijers, "Dcrac: Deep conditioned recurrent actor-critic for multi-objective partially observable environments," in *International Foundation for Autonomous Agents and Multiagent Systems*, AAMAS '20, p. 931–938, 2020.

[54] J. B. Rawlings and L. Ji, "Optimization-based state estimation: Current status and some new results," *Journal of Process Control*, vol. 22, no. 8, pp. 1439–1444, 2012.

[55] C. V. Rao and J. B. Rawlings, "Constrained process monitoring: Moving-horizon approach," *AIChE Journal*, vol. 48, no. 1, pp. 97–109, 2002.

[56] S. Bezuglyi and P. E. T. Jorgensen, *Transfer Operators, Endomorphisms, and Measurable Partitions*, pp. 105–111. Springer International Publishing, 2018.

[57] A. S. Zamzam, X. Fu, and N. D. Sidiropoulos, "Data-driven learning-based optimization for distribution system state estimation," *IEEE Transactions on Power Systems*, vol. 34, no. 6, pp. 4796–4805, 2019.

[58] B. Amos, L. Xu, and J. Z. Kolter, "Input convex neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70 of *Proceedings of Machine Learning Research*, pp. 146–155, PMLR, 2017.

[59] F. Bünning, A. Schalbetter, A. Aboudonia, M. H. de Badyn, P. Heer, and J. Lygeros, "Input convex neural networks for building MPC," 2020.

[60] S. Rastegarpour, L. Ferrarini, and S. Gros, "Economic NMPC for multiple buildings connected to a heat pump and thermal and electrical storages," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 17089–17094, 2020. 21st IFAC World Congress.

[61] H. A. Pipino, C. A. Cappelletti, and E. J. Adam, "Adaptive multi-model predictive control applied to continuous stirred tank reactor," *Computers & Chemical Engineering*, vol. 145, p. 107195, 2021.

[62] S. D. Cairano, "An industry perspective on MPC in large volumes applications: Potential benefits and open challenges," *IFAC Proceedings Volumes*, vol. 45, no. 17, pp. 52–59, 2012. 4th IFAC Conference on Nonlinear Model Predictive Control.

[63] D. J. Leith and W. E. Leithead, "Survey of gain-scheduling analysis and design," *International Journal of Control*, vol. 73, no. 11, pp. 1001–1025, 2000.

[64] F. K. Pour, V. Puig, and C. Ocampo-Martinez, "Comparative assessment of LPV-based predictive control strategies for a pasteurization plant," in *2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 0821–0826, 2017.

[65] B. Ding and H. Pan, "Output feedback robust MPC for LPV system with polytopic model parametric uncertainty and bounded disturbance," *International Journal of Control*, vol. 89, no. 8, pp. 1554–1571, 2016.

[66] H. A. Pipino and E. J. Adam, "MPC for linear systems with parametric uncertainty," in *2019 XVIII Workshop on Information Processing and Control (RPIC)*, pp. 42–47, 2019.

[67] W. Langson, I. Chryssochoos, S. Raković, and D. Mayne, "Robust model predictive control using tubes," *Automatica*, vol. 40, no. 1, pp. 125–133, 2004.

[68] P. Zheng, D. Li, Y. Xi, and J. Zhang, "Improved future model prediction and robust MPC design for LPV systems with bounded rates of parameter variations," in *2013 9th Asian Control Conference (ASCC)*, pp. 1–6, 2013.

[69] H. Suzukia and T. Sugie, "MPC for LPV systems with bounded parameter variation using ellipsoidal set prediction," in *2006 American Control Conference*, pp. 6 pp.–, 2006.

[70] M. M. Morato, J. E. Normey-Rico, and O. Sename, "Model predictive control design for linear parameter varying systems: A survey," *Annual Reviews in Control*, vol. 49, pp. 64–80, 2020.

[71] M. M. Morato, V. M. Cunha, T. L. Santos, J. E. Normey-Rico, and O. Sename, "A robust nonlinear tracking MPC using qLPV embedding and zonotopic uncertainty propagation," *Journal of the Franklin Institute*, vol. 361, no. 6, p. 106713, 2024.

[72] M. M. Morato, J. E. Normey-Rico, and O. Sename, "Robustness analysis of gain-scheduled model predictive control for linear parameter varying systems: An integral quadratic constraints approach," *International Journal of Robust and Nonlinear Control*, vol. 33, no. 18, pp. 10953–10972, 2023.

[73] A. Medero, M. M. Morato, V. Puig, and O. Sename, "MPC-based optimal parameter scheduling of LPV controllers: Application to lateral adas control," in *2022 30th Mediterranean Conference on Control and Automation (MED)*, pp. 951–956, 2022.

[74] C. Verhoek, H. S. Abbas, R. Tóth, and S. Haesaert, "Data-driven predictive control for linear parameter-varying systems," *IFAC-PapersOnLine*, vol. 54, no. 8, pp. 101–108, 2021. 4th IFAC Workshop on Linear Parameter Varying Systems LPVS 2021.

[75] S. Di Cairano and C. Danielson, "Indirect adaptive model predictive control and its application to uncertain linear systems," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8678–8702, 2021.

[76] J. Zhou, S. Di Cairano, and C. Danielson, "Indirect adaptive MPC for output tracking of uncertain linear polytopic systems," in *2017 American Control Conference (ACC)*, pp. 3054–3059, 2017.

[77] H. A. Pipino, M. M. Morato, E. Bernardi, E. J. Adam, and J. E. Normey-Rico, "Nonlinear temperature regulation of solar collectors with a fast adaptive polytopic LPV MPC formulation," *Solar Energy*, vol. 209, pp. 214–225, 2020.

[78] P. S. Cisneros, A. Datar, P. Göttsch, and H. Werner, "Data-driven quasi-LPV model predictive control using koopman operator techniques," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6062–6068, 2020. 21st IFAC World Congress.

[79] Y. Bao, J. M. Velni, A. Basina, and M. Shahbakhti, "Identification of state-space linear parameter-varying models using artificial neural networks," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 5286–5291, 2020. 21st IFAC World Congress.

[80] S. Di Cairano, "Indirect adaptive model predictive control for linear systems with polytopic uncertainty," in *2016 American Control Conference (ACC)*, pp. 3570–3575, 2016.

[81] P. Ru, and K. Subbarao, "Nonlinear model predictive control for unmanned aerial vehicles," *Journal of Aerospace, MDPI*, vol. 4, no. 2, p. 31, 2017.

[82] L. Hewing and A. Liniger and M. N. Zeilinger, "Cautious NMPC with Gaussian process dynamics for autonomous miniature race cars," in *2018 European Control Conference (ECC)*, pp. 1341–1348, 2018.

[83] H. Zheng, R. R. Negenborn, and G. Lodewijks, "Trajectory tracking of autonomous vessels using model predictive control," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8812 – 8818, 2014.

[84] S. Gros and M. Zanon, "Reinforcement learning for mixed-integer problems based on MPC," *arXiv preprint arXiv:2004.01430*, 2020.

[85] T. Mannucci, E.-J. van Kampen, C. de Visser, and Q. Chu, "Safe exploration algorithms for reinforcement learning controllers," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 4, pp. 1069–1081, 2017.

[86] M. Zanon, S. Gros, and A. Bemporad, "Practical reinforcement learning of stabilizing economic MPC," in *2019 18th European Control Conference (ECC)*, pp. 2258–2263, IEEE, 2019.

[87] K. Wang, Y. Jiang, J. Oravec, M. E. Villanueva, and B. Houska, "Parallel explicit tube model predictive control," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 7696–7701, 2019.

[88] Q. Mayne, E. C. Kerrigan, E. J. van Wyk, and P. Falugi, "Tube-based robust nonlinear model predictive control," *International Journal of Robust and Nonlinear Control*, vol. 21, no. 11, pp. 1341–1353, 2011.

[89] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[90] M E. Villanueva, R Quirynen, M Diehl, B Chachuat, and B Houska, "Robust MPC via min–max differential inequalities," *Automatica*, vol. 77, pp. 311–321, 2017.

[91] Z. J. Yu and L. T. Biegler, "Advanced-step multistage nonlinear model predictive control: Robustness and stability," *Journal of Process Control*, vol. 84, pp. 192–206, 2019.

[92] S. Lucia, A. Ttulea-Codrean, C. Schoppmeyer, andS. Engell, "Rapid development of modular and sustainable nonlinear model predictive control solutions," *Control EngineeringPractice*, vol. 60, pp. 51–52, 2017.

[93] C. Walck, "Hand-book on statistical distributions for experimentalists," 1996.

[94] A. B. Kordabad, D. Reinhardt, A. S. Anand, and S. Gros, "Reinforcement learning for MPC: Fundamentals and current challenges," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 5773–5780, 2023. 22nd IFAC World Congress.

[95] S. Sawant, A. S. Anand, D. Reinhardt, and S. Gros, "Learning-based MPC from big data using reinforcement learning," 2023.

ISBN 978-82-326-8130-3 (printed ver.)
ISBN 978-82-326-8129-7 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

NTNU
Norwegian University of
Science and Technology