Fast Algorithms for Stochastic Model Predictive Control with Chance Constraints via Policy Optimization

Jingyu Zhang
Department of Systems Science, Graduate School of Informatics
Kyoto University

A thesis submitted for the degree of
Doctor of Philosophy in Informatics

2023

Supervisor:

Prof. Toshiyuki Ohtsuka

Examination committee:

Prof. Toshiyuki Ohtsuka

Prof. Manabu Kano

Prof. Shun'ichi Azuma

Abstract

Stochastic model predictive control (SMPC) is currently receiving increasing attention in the control of stochastic systems because it provides a systematic way to incorporate probabilistic descriptions of uncertainties in a stochastic optimal control problem. The major challenge of SMPC comes from finding an optimal policy at each time instant. The goal of this thesis is to find novel policy optimization methods for SMPC with chance constraints, which can handle different types of problem settings, and obtain good closed-loop control effects and computational efficiency.

For linear Gaussian systems, we first present an efficient parameterization method called simplified affine disturbance feedback parameterization, which significantly reduces the computation cost while maintaining good control performance. The simulation results show that the proposed method can provide a trade-off between computation cost and control performance. We also present a fast algorithm for the numerical solution of stochastic dynamic programming. This algorithm is proved to globally converge to the optimal solution with a local Q-superlinear convergence rate. Numerical experiments show that this algorithm can achieve ideal closed-loop control performance with a very fast calculation speed.

For output-feedback nonlinear SMPC, we present a constrained stochastic approximate dynamic programming algorithm, which finds the numerical solution of the stochastic constrained Bellman equation. The information state propagation is obtained by the extended Kalman filter with a Gaussian assumption. The algorithm is proved to have a Q-superlinear local convergence rate. Numerical experiments show that our proposed algorithm can attain good control performance and reasonable chance-constraint satisfaction and is computationally efficient owing to its dynamic programming structure.

For systems with unknown models, we present a sample-based Bayesian reinforcement learning method. The Gaussian process dynamic model is used to model the system. We calculate the posterior belief update by a particle filter and solve the chance constrained optimization by a sample-based method. Numerical experiments show that our proposed method can achieve good learning efficiency and control performance in both linear Gaussian and non-linear non-Gaussian systems.

Acknowledgements

PhD study is an unforgettable journey in my life, in which there is sadness, joy, loss and growth. During the PhD study, the global outbreak of the COVID-19 also cast a shadow over the already difficult journey. I am very lucky to devote myself to such an interesting and meaningful research topic during the four years, and I am grateful for all the warmth and kindness I have received.

First and foremost, I want to thank my supervisor, Professor Toshiyuki Ohtsuka, who provided me many valuable opportunities and constant support and encouragement during my PhD and acted as a role model not only in academics but also in life. There are many things Ohtsuka sensei has taught me, not only rich experience and knowledge, but also the meticulousness and enthusiasm in research, and the patience and kindness with people, which can all benefit me for a lifetime. My examination committee members Prof. Ohtsuka, Prof. Kano (also my co-supervisor) and Prof. Azuma deserve my special thanks for their valuable advices and suggestions.

I would like to thank Associate Professor Kazunori Sakurama and Assistant Professor Kenta Hoshino for valuable discussions and suggestions in all the seminars in Ohtsuka lab, Secretary Ms. Hata for the support in affairs. Furthermore, I also would like to thank Dr. Deng, Assis. Prof. Iori, Dr. Katayama, Mr. Lin for the pleasant discussions and conversations. I also thank all my friends who are near and far, for their constant company and joy.

I would also like to thank the JST CREST and JSPS KAKENHI for the financial support of my PhD study.

Last but not least, I want to thank my parents who always care about and believe in me. They give me unconditional love and support, and make me full of confidence to face difficulties and challenges all the time.

Contents

Notation

For a vector $x \in \mathbb{R}^{n_x}$, a matrix $A \in \mathbb{R}^{n \times n}$, and a differentiable vector-valued function $g(x) : \mathbb{R}^n \to \mathbb{R}^m$, we have the following notation:

Notation	Meaning
x^i	i-th iteration of x
x_k	k-th stage of MPC
x^*	Optimal value of x
$\lvert x \rvert$	Element-wise absolute value of x
x^{T}	Vector transpose of x
$\lVert x \rVert_2$	ℓ_2 norm of x
$\lVert x \rVert$	Uniform norm of x
$A \succ 0$	A is positive-definite
$A \succeq 0$	A is positive-semidefinite
$\mathbf{E}(\cdot)$	Expectation of a random variable
$\mathrm{P}[\cdot]$	probability of a random event
$p(\cdot)$	probability density function of a random variable
$\mathbb{R}$	Set of non-negative real numbers
I	Identity matrix
0	Zero or zero matrix

Abbreviations

Abbreviation	Meaning
ADF	Affine disturbance feedback
ADP	Approximate dynamic programming
BDP	Bayesian dynamic programming
BRL	Bayesian reinforcement learning
CDF	Cumulative distribution function
CSTR	Continuous-time stirred tank reactor
DP	Dynamic programming
EKF	Extended Kalman filter
gPC	Generalized polynomial chaos expansion
GPDM	Gaussian process dynamic model
KKT	Karush–Kuhn–Tucker
LQR	Linear quadratic regulator
LQG	Linear quadratic Gaussian
MAP	Maximum a posteriori
MDP	Markov decision process
MLE	Maximum likelihood estimation
MPC	Model predictive control
OCP	Optimal control problem
PDF	Probability density function
PF	Particle filter
POMDP	Partially observable Markov decision process
RBF	Radial basis function
RL	Reinforcement learning
RRIPM	Recursive Riccati interior-point method
SADF	Simplified affine disturbance feedback
SDP	Stochastic dynamic programming
SIS	Sequential importance sampling
SKKT	Stage-wise Karush–Kuhn–Tucker
SMPC	Stochastic model predictive control
SOCP	Stochastic optimal control problem

Chapter 1

Introduction

1.1 Background and Motivation

Research on the control of stochastic systems has always been an important topic in academia and industry. In the past few decades, with the development of the times, the research on stochastic systems has changed from the simplest linear Gaussian system to complicated stochastic system. On the one hand, simple models can no longer satisfy the requirement of increasingly complex systems, such as robots with uncertain parameters and the control of complex power systems. On the other hand, modern technologies (e.g., increased computing power, availability of low-cost sensors, high-quality observation instruments, and advanced analysis techniques) provide more possibilities for the analysis of complex systems, while also putting forward higher requirements for the design of control algorithms.

Model predictive control (MPC) has received great attention among various advanced control technologies (Mayne (2014)), owing to its ability to effectively cope with the complex dynamics of systems with multiple inputs and outputs and to intuitively consider input or state constraints. In MPC literature, there are two ways to deal with uncertainty: robust (Bemporad and Morari (1999)) and stochastic approaches (Mesbah (2016)). Robust MPC keeps systems stable and allows them to operate under the worst-case perturbations analyzed under bounded uncertainties, whereas stochastic MPC (SMPC) uses probabilistic descriptions of objective values and constraints and accounts for acceptable levels of constraint violations during system operation, called chance constraints (Charnes and Cooper (1959)). The probability of the worst-case perturbations can be very small in the real world, which leads to conservative control performance in robust MPC (Garatti and Campi (2013)); therefore, SMPC is more attractive for better control performance.

The SMPC together with chance constrained optimization has been considered a promising solution in many applications, such as building climate control (Oldewurtel, Jones, Parisio, and Morari (2013)), process control (Buehler, Paulson, and Mesbah (2016); Mesbah, Streif, Findeisen, and Braatz (2014); Van Hessem and Bosgra (2006)), portfolio optimization and finance (Dombrovskii and Obyedko (2015); Herzog, Dondi, and Geering (2007)), air traffic control (Visintini, Glover, Lygeros, and Maciejowski (2006)), power production, management, and supply in systems with renewable energy sources (Hooshmand, Poursaeidi, Mohammadpour, Malki, and Grigoriads (2012); Patrinos, Trimboli, and Bemporad (2011)), robot and autonomous driving (J. Chen, Shimizu, Sun, Tomizuka, and Zhan (2021); Farrokhsiar and Najjaran (2012)).

The classification of the SMPC algorithms can be quite difficult due to the large variety of problem formulations and solution methods (Mesbah (2016)). For example, from the problem formulation point of view, design methods have been developed for linear or nonlinear, with additive, multiplicative or parameter uncertainties, state feedback or output feedback, finite or infinite horizon cost function, linear or nonlinear chance constraints, and totally or partially known systems. Solution methods can also be divided into stochastic programming-based or sampling-based approaches, open-loop or closed-loop optimization approaches, and so on.

The aim of this thesis is to provide fast algorithms for SMPC with chance constraints under various problem settings and can provide good closed-loop performance.

1.2 Overview of SMPC with Chance Constraints

The existing algorithms for solving SMPC can be roughly classified into two types, based on how to solve the underlying chance-constrained optimization problem: the first one, i.e., the *analytic approximation* methods (Geletu, Klöppel, Zhang, and Li (2013)) (also known as *probabilistic approximation* methods in Zhou and Cogill (2013)), is based on the reformulation of probabilistic-type constraints and cost function in terms of some deterministic quantities, e.g., sufficient statics of the random variables. The second approach is based on the *randomized* (Schildbach, Calafiore, Fagiano, and Morari (2012)), or *scenario generation* (Schildbach, Fagiano, Frei, and Morari (2014)) methods, i.e., generating a sufficient number of random variable realizations, and combine the solution of resulting constrained optimization problems.

Next, we provide an overview of commonly used formulation methods in SMPC with chance constraints.

1.2.1 Stochastic tube approaches

Tube-based methods are widely used in linear stochastic systems with additive disturbance. The basic idea of the stochastic tube approach is from robust MPC (Blanchini (1990)), which is to control the state of the system in a set called “tube” through a certain control policy. This set of tube is a subset of the system constraint, and then guide the entire tube to the desired position. This kind of operation on tube allows designers to directly deal with the impact of uncertainty on system dynamics and constraints. A major advantage of this method is that it can separate the deterministic and uncertain parts of the system, making the controller can complete a lot of work offline calculations, which is called pre-stabilizing techniques (Cannon, Kouvaritakis, and Wu (2009)).

Many researchers have discussed the form and offline computation of stochastic tubes. Cannon et al. (2009) uses variable polytopic cross-sections as well as several tube layers. In Cannon, Kouvaritakis, Raković, and Cheng (2010), many-layered tubes are used which can considerably reduce the conservativeness compared with Cannon et al. (2009), but still a conservative performance. This shortcoming is addressed in Kouvaritakis, Cannon, Raković, and Cheng (2010) by using the probabilistically constrained states to construct recursively feasible stochastic tubes.

In summary, stochastic tube approaches compute a state-feedback control law with pre-computed constant feedback gain. This pre-stabilizing control law can guarantee recursive feasibility as well as closed-loop stability in a mean-square sense. However, the pre-stabilizing control law is just a conservative approximation of optimal control law, since it loses a lot of design freedom.

1.2.2 Pre-parameterization of control policy

On the basis of tube-based approaches, the affine parameterization of the control policy is proposed to increase the design freedom. A natural way is to search for the solution of SOCP over an affine state feedback control law, i.e., treating feedback gains and offsets as decision variables. However, the set of constraint-admissible policies of this form is non-convex (Lofberg (2003)).

Inspired by disturbance feedback techniques in robust MPC (Ben-Tal, Goryashko, Guslitzer, and Nemirovski (2004); Goulart, Kerrigan, and Maciejowski (2006)), an affine disturbance feedback control law for SMPC is presented in Oldewurtel, Jones, and Morari (2008), where the feedback control law is defined in terms of an affine function of disturbance. There exists a one-to-one mapping between affine state feedback control law and affine disturbance feedback control law, which is proved in Goulart et al. (2006). In Oldewurtel et al. (2008), the disturbance in state equation is assumed Gaussian and bounded in a polytopic set, which enables the analyzing of input-to-state stability for the closed-loop system. Oldewurtel et al. (2010) extends the situation to unbounded disturbances while relaxing the hard input bounds considered in Oldewurtel et al. (2008) so that the inputs are lying in a subset with a certain probability level. The advantage of affine disturbance feedback control parameterization is that the set of its decision variables is guaranteed to be convex. However, the number of decision variables in affine disturbance feedback policy grows quadratically with the prediction horizon, so the real-time calculation is disastrous when the prediction horizon grows.

1.2.3 Stochastic dynamic programming approaches

The most well-known research in stochastic optimal control via dynamic programming (Bellman (1966)) is linear-quadratic-Gaussian (LQG) control (Lindquist (1973)). In the linear Gaussian case, the separation principle, which involves separation of parameter estimation/system identification and control design (Patchell and Jacobs (1971)), is exact and the stochastic optimal control law is equivalent to a certainty equivalence controller (Bar-Shalom and Tse (1974)). However, the separation principle is not optimal for general stochastic systems, and the LQG controller also cannot deal with model uncertainty.

Bar-Shalom and Tse (1974) proposed the concept of dual control, which provides a unified framework for stochastic optimal control and model uncertainty handling, but it relies on the exact solution of stochastic dynamic programming, which is computationally intractable, particularly in large-scale optimization (Mesbah (2018)). To address the computational complexity of stochastic dynamic programming, various approximate solutions have been proposed, which can be broadly categorized into two approaches (Filatov and Unbehauen (2000)): One adopts a heuristic reformulation of SMPC into a tractable optimal control problem, which can artificially include a certain degree of closed-loop control effects such as the affine parameterization of control policy we mentioned in the previous subsection, and covariance control method (D. Li, Qian, and Fu (2002)). Another approach involves the direct approximation of the Bellman equation called approximate dynamic programming (ADP), which yields a suboptimal solution to the Bellman equation (Powell (2007)).

Bayard and Eslami (1985) proposed a dual control method for stochastic nonlinear systems based on a single-stage lookahead approximation of cost-to-go, and the approximated cost-to-go functions are evaluated via averaging the cost incurred over several sample trajectories obtained from Monte Carlo simulations. Bayard and Schumitzky (2010) presented a fully sample-based approach that combines the sample-based forward dynamic programming with the particle filter, the advantage of using a particle filter includes the applicability to general nonlinear systems as well as its good performance in non-Gaussian systems.

The success of ADP-based algorithms such as differential dynamic programming (Mayne (1966)) and iterative linear quadratic regulator (W. Li and Todorov (2004)) in the trajectory optimization and machine learning fields also inspired the development of analytic approximation methods in stochastic MPC. Todorov and Li (2005) proposed an iterative LQG algorithm for locally-optimal feedback control of nonlinear stochastic systems subject to control constraints. J. Chen et al. (2021) generalized the iterative LQG algorithm to the chance-constrained problem by using the penalty methods, and tested the algorithm in autonomous driving planning tasks with static and dynamic obstacles.

1.3 Challenges in SMPC with Chance Constraints

For the optimal control of stochastic systems (see Figure 1.1), policy optimization is generally preferred rather than optimizing a trajectory through the state and control space, whereas the latter approach refers to the so-called “open-loop” controller (Mayne, Rawlings, Rao, and Scokaert (2000)). However, policy optimization is still challenging due to stochastic settings and chance constraints. This section introduces these difficulties, which are tackled in this thesis.

1.3.1 Policy optimization

In principle, Bellman’s principle of optimality (Bellman (1966)) can be used to obtain the control policy; however, the closed-form solution of the Bellman equation is always intractable, particularly in large-scale or constrained optimization.

Pre-parameterization of control policy is commonly used in the reinforcement learning field. In Sutton, McAllester, Singh, and Mansour (1999), the policy was represented by a neural network whose input is a representation of the state, whose output is action selection probabilities, and whose weights are the policy parameters, and convergence of the policy gradient algorithm was also given. There have been many works based on the policy gradient (Ghavamzadeh and Engel (2006); Kakade (2001); Silver et al. (2014)).

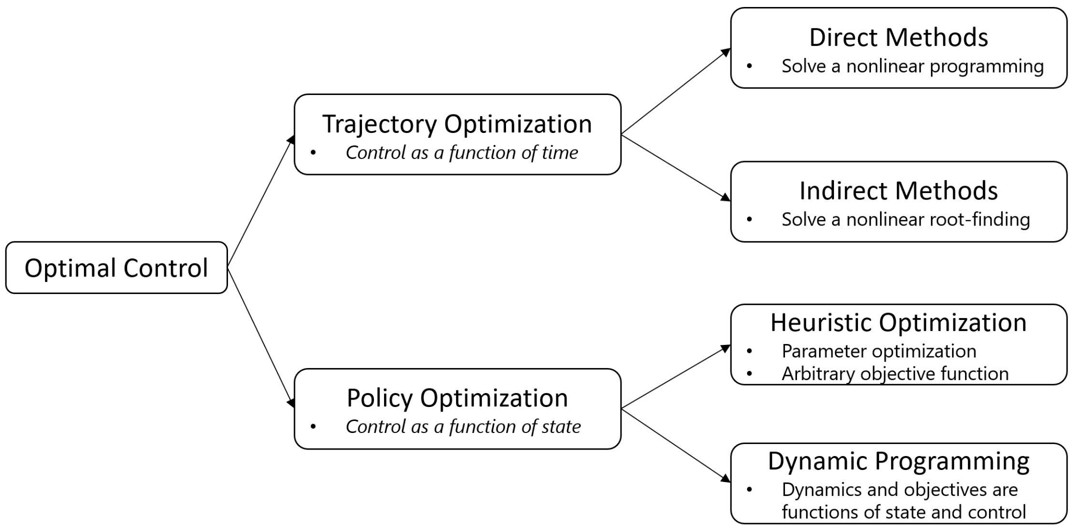
Optimal Control
Trajectory Optimization
Control as a function of time
Policy Optimization
Control as a function of state
Direct Methods
Solve a nonlinear programming
Indirect Methods
Solve a nonlinear root-finding
Heuristic Optimization
Parameter optimization
Arbitrary objective function
Dynamic Programming
Dynamics and objectives are functions of state and control

Figure 1.1: Optimal control: a high level view

When dealing with MPC problems that require real-time computing, the gradient method cannot meet such requirements. In linear SMPC, the optimal policy can be parameterized as the affine function of state or disturbance as we mentioned in Section 1.2.2, but the affine parameterization either leads to non-convex optimization or introduce too many extra decision variables, which prevents its application to real-time computation. In nonlinear cases, the optimal policy is no longer an affine function of state or disturbance, but optimizing over arbitrary feedback law also can not meet the requirements of real-time computing.

1.3.2 Uncertainties propagation in stochastic system

In linear Gaussian systems, the uncertainty propagation can be easily obtained by a linear transformation of Gaussian random variables. However, the efficient propagation method for stochastic uncertainties through the system dynamics is a key challenge in SMPC of nonlinear systems.

In Weissel, Huber, and Hanebeck (2009), the Gaussian-mixture approximation (Maz'ya and Schmidt (1996)) is used to describe the transition probability distributions of states. However, the online calculation of the coefficients of the Gaussian-mixture model brings a lot of computation burdens. Generalized polynomial chaos expansion (gPC, Xiu and Karniadakis (2002)) is also widely used in nonlinear SMPC. In Mayne (2014), a sample-based method was adopted to adapt the coefficients of gPC based on the history data. Streif, Karl, and Mesbah (2014) used Galerkin projection (Ghanem and Spanos (2003)) for adapting the coefficients of gPC, and efficiently constructing the probability distributions of state through Monte Carlo methods to evaluate the chance constraints. The disadvantages of the gPC are caused by the complexity of reconstructing the probability distribution by the statistic moments information.

1.4 Outline and Contributions

This thesis presents efficient numerical algorithms for SMPC of various stochastic systems with chance constraints. The aforementioned challenges and our approaches are summarized in Figure 1.2. Chapter 2 introduces the preliminaries of this thesis: basic knowledge of numerical optimization and the main features of SMPC. Chapter 3 presents an efficient parameterization method that combines the related simplification techniques of affine disturbance feedback to create a concise controller form. Chapter 4 presents a recursive Riccati interior-point method for directly finding the stochastic dynamic programming solution for linear SMPC. Chapter 5 presents a general stochastic output-feedback MPC scheme for chance-constrained nonlinear systems, and an approximate dynamic programming algorithm for solving the resulting optimization problem. Chapter 6 presents a Bayesian reinforcement learning scheme for unknown systems with chance constraints. We conclude the thesis and present an outlook in Chapter 7. The main contributions of each chapter are briefly discussed as follows.

Chapter 2 — Preliminaries. This chapter introduces preliminary knowledge of solving SMPC which is used throughout this thesis. The first part of this chapter introduces the basic knowledge of numerical optimization. We discuss the optimality conditions of constrained optimization and introduce the primal-dual interior-point method for a numerical solution of constrained optimization. In particular, we provide the formulation of optimal control problem to be a constrained optimization problem and the corresponding optimality conditions. The second part of this chapter introduces some important features of SMPC. We first give the general formulation of the stochastic optimal control problem with chance constraints. Then we discuss the information state update via recursive Bayesian filter, some approximation methods such as Kalman filter, extended Kalman filter, and particle filter are also given. Finally, we introduce the concept of individual and joint chance constraints, and a simple analytic reformulation method of chance constraints is given.

Chapter 3 — Efficient Control Parameterization Method for Linear SMPC. This chapter presents an efficient control parameterization method for linear

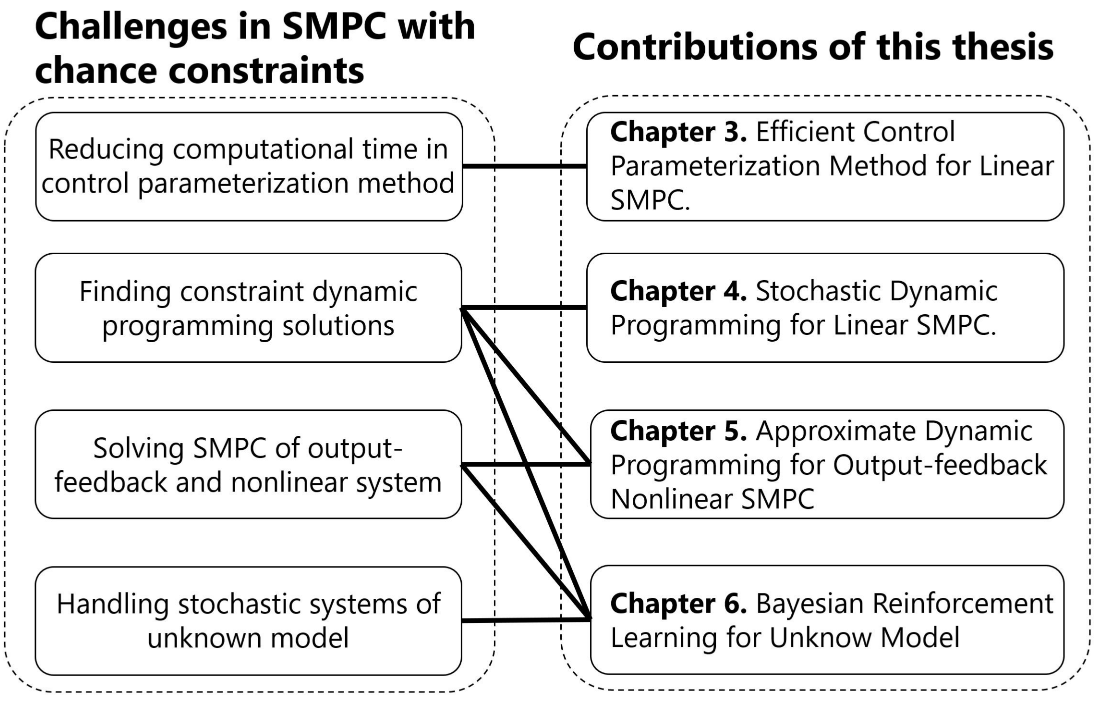
Challenges in SMPC with chance constraints
Contributions of this thesis
Reducing computational time in control parameterization method
Finding constraint dynamic programming solutions
Solving SMPC of output-feedback and nonlinear system
Handling stochastic systems of unknown model
Chapter 3. Efficient Control Parameterization Method for Linear SMPC.
Chapter 4. Stochastic Dynamic Programming for Linear SMPC.
Chapter 5. Approximate Dynamic Programming for Output-feedback Nonlinear SMPC
Chapter 6. Bayesian Reinforcement Learning for Unknow Model

Figure 1.2: Contributions overview of this thesis

SMPC with chance constraints. This parameterization combines the related simplification techniques of affine disturbance feedback to create a simplified affine disturbance feedback parameterization. The number of decision variables is decreased to grow linearly with respect to the horizon length compared with quadratic growth of the original affine disturbance feedback control law, resulting in a preferable trade-off between real-time calculation and control performance. This parameterization is shown to be equivalent to a state feedback control law, and the closed-loop stability of the SMPC problem can also be guaranteed under mild assumptions. The simulation results show that the proposed control parameterization method provides a desirable control performance with low computation cost, and achieves the expected result.

Chapter 4 — Stochastic Dynamic Programming for Linear SMPC. This chapter presents a numerical algorithm for linear SMPC with chance constraints via solving the Bellman equation. The proposed approach reformulates the SMPC problem in a stochastic programming fashion. A recursive Riccati interior-point method is proposed to solve the ensuing inequality-constrained dynamic programming. The proposed method eliminates active sets in conventional explicit MPC and does not suffer from the curse of dimensionality because it finds the value function and feedback policy only for a given state using the interior-point method. Moreover, the proposed method is proven to converge globally to a stationary solution Q-superlinearly. The numerical experiment reveals that the proposed method achieves a less conservative performance with low computational complexity compared to existing methods.

Chapter 5 — Approximate Dynamic Programming for Output-feedback Nonlinear SMPC. This chapter presents an efficient numerical algorithm for output-feedback nonlinear SMPC with chance constraints. The stochastic optimal control problem is also solved in a stochastic dynamic programming fashion like Chapter 4, and the output-feedback control is performed with the extended Kalman filter. The information state is summarized as a dynamic Gaussian belief model. Thus, the stochastic Bellman equation is transformed into a deterministic equation using this model. A novel constrained approximate dynamic programming algorithm is proposed to solve the resulting constrained Bellman equation. The proposed algorithm was proven to exhibit a Q-superlinear local convergence rate. The numerical experiment shows that the proposed method achieves good control performance and a reasonable level of constraint violation and is computationally efficient owing to the Riccati-type structure.

Chapter 6 — Bayesian Reinforcement Learning for Unknown Model. This chapter presents a sample-based Bayesian reinforcement learning method for dealing with systems of unknown model. The unknown model is learned by a Gaussian process dynamic model in a model-based Bayesian reinforcement learning framework. The POMDP is reformulated as a belief MDP by the particle filter. By using a sample-based method, the stochastic dynamic programming is transformed into several deterministic constrained dynamic programming problems, where each deterministic problem is solved by the algorithm proposed in Chapter 5. The chance constraint is treated by a sample-removal algorithm. The numerical experiment shows that the proposed method can get good control performance with a reasonable level of constraint violation, and compared with the conventional Bayesian reinforcement learning method, the learning efficiency is significantly improved.

Chapter 2

Preliminaries

This chapter introduces the preliminaries of numerical optimization and SMPC with chance constraints. The first section introduces the optimality conditions of a constrained optimization problem and the primal-dual interior point method. The second section introduces the basic idea of SMPC with chance constraint, in particular, various estimation methods for information state propagation and how to handle chance constraints in SMPC

2.1 Numerical Optimization

2.1.1 Optimality conditions for nonlinear programming

Consider a general nonlinear programming problem

$$\begin{aligned} \min_{x} \ & f(x) \\ \text{s.t. } & h(x) = 0 \\ & g(x) \leq 0 \end{aligned} \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $h : \mathbb{R}^n \to \mathbb{R}^m$, and $g : \mathbb{R}^n \to \mathbb{R}^l$. The Lagrangian associated with (2.1) is

$$L(x, \lambda, s) = f(x) + \lambda^{\mathrm{T}} h(x) + s^{\mathrm{T}} g(x) \tag{2.2}$$

where $\lambda \in \mathbb{R}^m$ and $s \in \mathbb{R}^l$ are Lagrange multipliers for equality and inequality constraints, respectively.

The Karush–Kuhn–Tucker (KKT) conditions for problem (2.1) are

$$\nabla_x L(x,\lambda,s) = 0 \tag{2.3a}$$
$$h(x) = 0 \tag{2.3b}$$
$$g(x) \leq 0 \tag{2.3c}$$
$$Sg(x) = 0 \tag{2.3d}$$
$$s \geq 0 \tag{2.3e}$$

where $S = \text{diag}[s_1, ..., s_l]$ and ∇ is the gradient operator

$$\nabla_x L(x,\lambda,s) = \nabla f(x) + \nabla h(x)\lambda + \nabla g(x)s$$

Let $\mathbb{A}(x)$ denotes the set of indices of active inequality constraints at x, i.e.,

$$\mathbb{A}(x) = \{i : g_i(x) = 0, i = 1, .., p, p \leq l\} \tag{2.4}$$

The standard Newton's method assumptions for problem (2.1) are as follows:

Assumption 2.1 (Existence). *There exists solution to problem (2.1)* (x^*, λ^*, s^*) *satisfies the KKT conditions (2.3).*

Assumption 2.2 (Smoothness). *The Hessian matrices* $\nabla^2 f(x), \nabla^2 h(x), \nabla^2 g(x)$ *exist and locally Lipschitz continuous at* x^*.

Assumption 2.3 (Regularity). *The set* $\{\nabla h_1(x^*), ..., \nabla h_m(x^*)\} \cup \{\nabla g_i(x^*) : i \in \mathbb{A}(x^*)\}$ *is linearly independent.*

Assumption 2.4 (Second-order sufficiency). *For all* $\eta \neq 0$ *satisfying* $\nabla h_i(x^*)^{\mathrm{T}}\eta = 0, i = 1, ..., m,$ *and* $\nabla g_i(x^*)^{\mathrm{T}}\eta = 0, i \in \mathbb{A}(x^*)$, *we have* $\eta^{\mathrm{T}}\nabla_x^2 L(x^*)\eta > 0$.

Assumption 2.5 (Strict complementarity). *For all* i, $g_i(x^*) < s_i^*$.

The KKT conditions (2.3) can be written in a slack variable form as

$$F(x,\lambda,s,y) = \begin{bmatrix} \nabla_x L(x,\lambda,s) \\ h(x) \\ g(x)+y \\ Sy \end{bmatrix} = 0, \ s \geq 0, \ y \geq 0 \tag{2.5}$$

where $y \in \mathbb{R}^l$ is the slack variable to transfer the inequality constraint $g(x) \leq 0$ into an equality constraint $g(x) + y = 0$ by set $y \geq 0$. The KKT system (2.5) gives the first-order necessary conditions for a local minimum for optimization problem (2.1). Note that the KKT conditions are the necessary and sufficient conditions (Boyd, Boyd, and Vandenberghe (2004)) for the global minimum if (2.1) is convex.

2.1.2 Primal-dual interior-point method

In this subsection, we introduce the primal-dual interior-point method to solve the inequality constraint optimization problem and show how the KKT system is solved.

The derivation of the primal-dual interior point method can be explained from the barrier function or the perturbed KKT conditions (El-Bakry, Tapia, Tsuchiya, and Zhang (1996)). We use the relatively simple perturbed KKT condition to derive the algorithm. Let us consider the perturbed KKT conditions

$$F_\mu(x,\lambda,s,y) = \begin{bmatrix} \nabla_x L(x,\lambda,s) \\ h(x) \\ g(x)+y \\ Sy-\mu \end{bmatrix} = 0, \ (s,y) \geq 0 \tag{2.6}$$

where $\mu \in \mathbb{R}^l$ is the current duality measure (central parameter) used to smooth the non-smooth equation $Sy = 0$. In the iterative process, the central parameter will gradually decrease to converge to the original solution of the KKT system (2.5).

We now describe the primal-dual interior-point method for the general nonlinear optimization problem (2.1). At the ith iteration, let

$$v^i = (x^i, \lambda^i, s^i, y^i). \tag{2.7}$$

We define the perturbed Newton correction from ith iteration to $(i+1)$-th iteration

$$\delta v^i = (\delta x^i, \delta\lambda^i, \delta s^i, \delta y^i). \tag{2.8}$$

The search direction δv^i is given by the solution of the perturbed Newton linear system

$$F'_\mu(v^i)\delta v^i = -F_\mu(v^i). \tag{2.9}$$

More specifically, we can write it in the matrix form

$$\begin{bmatrix} \nabla^2_{xx} L(x^i,\lambda^i,s^i) & \nabla h(x^i) & \nabla g(x^i) & 0 \\ \nabla h(x^i)^{\mathrm{T}} & 0 & 0 & 0 \\ \nabla g(x^i)^{\mathrm{T}} & 0 & 0 & I \\ 0 & 0 & S^i & Y^i \end{bmatrix} \begin{bmatrix} \delta x^i \\ \delta\lambda^i \\ \delta s^i \\ \delta y^i \end{bmatrix} = - \begin{bmatrix} \nabla_x L(x^i,\lambda^i,s^i) \\ h(x^i) \\ g(x^i)+y^i \\ S^i y^i - \mu^i \end{bmatrix} \tag{2.10}$$

After the search direction δv^i has been calculated by solving (2.10), the new iteration v^{i+1} is calculated by

$$v^{i+1} = v^i + \boldsymbol{\alpha}^i \delta v^i \tag{2.11}$$

Algorithm 2.1: Primal-dual interior-point method

```
Parameters:
  $\sigma \in [0, 1]$ reduce parameter;
  $\alpha_f = 0.995$ fraction-to-the-boundary parameter;
  $\epsilon$ terminal tolerance;
Initialization:
  iteration counter $i = 0$ ;
  initial guess $(x^0, \lambda^0, s^0, y^0)$ ;
  central parameter $\mu^0$ ;
  initial optimality measurement $(\|\xi^0\|)$ ;
while $\|\xi^i\| > \epsilon$ do
    Solve linear system (2.10) for $\delta v^i$;
    Compute step size $\boldsymbol{\alpha}^i$ by (2.12);
    Calculate new iterate $(x^{i+1}, \lambda^{i+1}, s^{i+1}, y^{i+1})$ by (2.11);
    Reduce the barrier parameter $\mu^i = \sigma\mu^i$;
    Update the iteration counter $i = i + 1$;
    Compute optimality measurements $\xi^i$.
end
```

where $\boldsymbol{\alpha}^i = \text{diag}[\alpha_x, \alpha_\lambda, \alpha_s, \alpha_y]$, which allows the flexibility of choosing different step sizes for the optimization variables. One simple method for choosing the step size is the fraction-to-the-boundary rule (Nocedal and Wright (1999))

$$\begin{aligned} \alpha_s &= \max\{\alpha \in (0, 1]) : s^i + \alpha\delta s^i \geq 0\} \\ \alpha_y &= \max\{\alpha \in (0, 1]) : y^i + \alpha\delta y^i \geq 0\} \\ \alpha_x &= \alpha_y, \quad \alpha_\lambda = \alpha_s \end{aligned} \tag{2.12}$$

with $\alpha_f \in (0, 1)$ (A typical value of α_f is 0.995), and the condition (2.12) can prevent the variables s and y from approaching their lower bounds of 0 too quickly.

The above simple method for choosing the step sizes provides the basis of modern primal-dual interior-point methods, though various line search technics are needed to deal with non-linearity and non-convexity. The other major problem is how to choose the sequence of central parameters $\{\mu^i\}$. In Fiacco and McCormick (1990), the central parameter is held fixed for a series of iterations until the perturbed KKT conditions (2.6) are satisfied to some accuracy. A more common approach is to update the central parameter at each iteration by a reduce parameter $\sigma \in (0, 1)$.

The optimality measure is defined by the current KKT measurement

$$\xi^i = \begin{bmatrix} \nabla_x L(x^i, \lambda^i, s^i) \\ h(x^i) \\ g(x^i) + y^i \\ S^i y^i \end{bmatrix} \tag{2.13}$$

and the algorithm converges when

$$\|\xi^i\| \leq \epsilon \tag{2.14}$$

where $\epsilon > 0$ is the terminal tolerance.

We summarize the primal-dual interior-point method in Algorithm 2.1.

2.1.3 Optimality conditions for optimal control problem

Consider a discrete-time system of the following form:

$$x_{k+1} = f(x_k, u_k) \tag{2.15}$$

subject to inequality constraint

$$c(x_k, u_k) \leq 0, \tag{2.16}$$

where $x_k \in \mathbb{R}^{n_x}$ and $u_k \in \mathbb{R}^{n_u}$ are state and control inputs. $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$ and $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_c}$ are the system equation and constraint function.

Consider a finite-horizon constrained optimal control problem at a known initial state x_0

$$\begin{aligned} \min_{\mathbf{x},\mathbf{u}} \ & \sum_{k=0}^{N-1} l(x_k, u_k) + l_N(x_N) \\ \text{s.t. } & x_{k+1} = f(x_k, u_k) \\ & c(x_k, u_k) \leq 0, \end{aligned} \tag{2.17}$$

where $l : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ and $l_N : \mathbb{R}^{n_x} \rightarrow \mathbb{R}$ are the stage-wise and terminal costs, respectively, N is the prediction horizon, $\mathbf{x}$ and $\mathbf{u}$ are the vectors of state and control input sequences, i.e., $\mathbf{x} = [x_0^{\mathrm{T}}, ..., x_N^{\mathrm{T}}]^{\mathrm{T}}$ and $\mathbf{u} = [x_0^{\mathrm{T}}, ..., x_{N-1}^{\mathrm{T}}]^{\mathrm{T}}$.

The Lagrangian for this constrained optimal control is

$$L(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \mathbf{s}) = \sum_{k=0}^{N-1} (l(x_k, u_k) + s_k^{\mathrm{T}} c(x_k, u_k)) + l_N(x_N, u_N) + \sum_{k=1}^{N} \lambda_k^{\mathrm{T}} (f(x_{k-1}, u_{k-1}) - x_k) \tag{2.18}$$

where $s_k \in \mathbb{R}^{n_c}$ and $\lambda_k \in \mathbb{R}^{n_x}$ are Lagrange multipliers for equality and inequality constraints respectively.

Let $\mathbf{c}(\mathbf{x}, \mathbf{u}) = (c(x_0, u_0)^{\mathrm{T}}, ..., c(x_{N-1}, u_{N-1})^{\mathrm{T}})^{\mathrm{T}}$, $\boldsymbol{\lambda} = [\lambda_0^{\mathrm{T}}, ..., \lambda_N^{\mathrm{T}}]^{\mathrm{T}}$, $\mathbf{s} = [s_0^{\mathrm{T}}, ..., s_{N-1}^{\mathrm{T}}]^{\mathrm{T}}$ and $\mathbf{S} = \mathrm{diag}[\mathbf{s}]$. The Karush–Kuhn–Tucker (KKT) conditions for the optimal control problem is

$$\begin{aligned}
&\nabla_{\mathbf{x}} L(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \mathbf{s}) = 0 \\
&\nabla_{\mathbf{u}} L(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \mathbf{s}) = 0 \\
&\nabla_{\boldsymbol{\lambda}} L(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \mathbf{s}) = 0 \\
&\mathbf{S}\mathbf{c}(\mathbf{x}, \mathbf{u}) + \boldsymbol{\lambda} = 0 \\
&\mathbf{c}(\mathbf{x}, \mathbf{u}) \leq 0, \ \mathbf{s} \geq 0,
\end{aligned} \tag{2.19}$$

where ∇ is the gradient operator. The KKT system (2.19) gives the first-order necessary conditions for a local minimum for optimal control problem (2.17), and thus the numerical solution of the optimal control problem can be given by Algorithm 2.1.

2.2 Stochastic Model Predictive Control

A general SMPC problem is introduced in this section. This formulation provides a very general framework for a wide variety of problems not only from control engineering but also from basically any field imaginable since uncertainty always exists in reality.

2.2.1 Stochastic optimal control problem

As necessary ingredients, a model of the system is needed, most commonly in the form of a discrete-time stochastic system:

$$\begin{aligned}
x_{k+1} &= f(x_k, u_k, w_k) \\
y_k &= h(x_k, v_k),
\end{aligned} \tag{2.20}$$

where k denotes the time index; $x_k \in \mathbb{R}^{n_x}$ denotes the system states; $u_k \in \mathbb{R}^{n_u}$ denotes the control inputs; $y_k \in \mathbb{N}^{n_y}$ denotes the outputs; $w_k \in \mathbb{R}^{n_w}$ denotes stochastic process noise; $v_k \in \mathbb{N}^{n_v}$ denotes measurement noise; and $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ and $h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_y}$ denote the system state and output equations, respectively. The random disturbance w_k and noise v_k are generated based on known probability distributions.

Define an N-horizon cost function:

$$J := \mathbf{E}[\sum_{k=0}^{N-1} l(x_k, u_k) + l_N(x_N)], \tag{2.21}$$

where $\mathbf{E}$ denotes the expected value, $l : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ and $l_N : \mathbb{R}^{n_x} \to \mathbb{R}$ are the stage-wise and terminal costs, respectively.

In addition to minimizing cost (2.21), the system is also subject to input and state constraints

$$g_1(x_k) \leq 0, \quad g_2(u_k) \leq 0. \tag{2.22}$$

Normally a chance constraint is imposed on the state, i.e.,

$$\mathrm{P}[g_1(x_k) \leq 0] \geq 1 - p_{vio}, \tag{2.23}$$

where P denotes the probability and $p_{vio} \in (0, 0.5]$ denotes the maximum allowed probability of state constraint violation.

Remark 2.1. *The chance constraints on input variable are still problematic in the context on SMPC. More specifically, the issue is whether to reformulate input constraints as probabilistic ones or not. In the case of noise is bounded (Cannon et al. (2010); Korda, Gondhalekar, Oldewurtel, and Jones (2014)) or recursive feasibility requirements are relaxed (Prandini, Garatti, and Lygeros (2012); Schildbach et al. (2014)), the hard constraints on control input are always considered. In the case of control parameterization method, it is possible to consider chance constraints on control input (Farina, Giulioni, Magni, and Scattolini (2013)).*

Given system equations (2.20) and a prediction horizon of length N, we can define the finite-horizon stochastic optimal control problem as follows:

Problem 2.1 (SOCP)**.**

$$\begin{aligned}
\min_{(u_k)_{k=0}^{N-1}} \quad & \mathbf{E}[\sum_{k=0}^{N-1} l(x_k, u_k) + l_N(x_N)] \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k, w_k) \\
& \mathrm{P}[g_1(x_k) \leq 0] \geq 1 - p_{vio} \\
& g_2(u_k) \leq 0 \\
& x_0 \sim p(x_0).
\end{aligned}$$

The uncertain initial states x_0 are described by known probability density function (PDF) $p(x_0)$.

The main differences between deterministic OCP and SOCP are stochastic nature of system variables and their propagation through the system equation, and how to handle chance constraints. Next we will discuss in details of these two aspects.

2.2.2 Information state

In the case of partially observable systems, we cannot directly obtain state propagation but only can be estimated by the past information. In this section, we will introduce the concept of information state, and discuss how to update the information state.

Let $\mathcal{I}_k$ denote the matrix of available system information at time k:

$$\begin{aligned} \mathcal{I}_k &:= [y_k, ..., y_0, u_{k-1}, ..., u_0] \\ \mathcal{I}_0 &:= [y_0]. \end{aligned} \tag{2.24}$$

The information state (Bertsekas (2012)) is the conditional probability density function of the state x_m given $\mathcal{I}_n$, that is, $\xi_{m|n} = p[x_m|\mathcal{I}_n]$, which describes the system uncertainty with the currently available information. The information state describes the system uncertainty under currently known information and can be computed by *recursive Bayesian estimation* (Z. Chen et al. (2003))

$$p[x_{k+1}|\mathcal{I}_k] = \int p[x_{k+1}|x_k, u_k]p[x_k|\mathcal{I}_k]dx_k \tag{2.25a}$$

$$p[x_{k+1}|\mathcal{I}_{k+1}] = \frac{p[y_{k+1}|x_{k+1}]p[x_{k+1}|\mathcal{I}_k]}{p[y_{k+1}|\mathcal{I}_k]}, \tag{2.25b}$$

and $p[x_0|\mathcal{I}_{-1}] := p[x_0]$. Recursion (2.25a) indicates that the information $\xi_{k+1|k}$ is determined by

$$\xi_{k+1|k} = \mathcal{G}(\xi_{k|k}, u_k), \tag{2.26}$$

where $\mathcal{G}$ is a mapping obtained from the Bayesian estimation. Recursion (2.25b) implies that the update of $\xi_{k+1|k+1}$ is

$$\xi_{k+1|k+1} = \mathcal{H}(\xi_{k+1|k}, I_{k+1}) \tag{2.27}$$

where the mapping $\mathcal{H}$ is derived from (2.25b) using knowledge from (2.25a).

From the definition of the information state, $\xi_{k|k}$ is a PDF with infinite dimensions, which makes the general Bayesian filter (2.25) intractable. To derive a practically efficient algorithm, it is possible to represent the information state by its sufficient statistics, that is, mean, variance, and even high-order moments. Next, we will introduce some commonly used filters to estimate the information state.

2.2.2.1 Kalman filter

Kalman filter is a famous algorithm for estimating states based on linear Gaussian dynamic systems in state space format (Kalman (1960)).

Consider the linear system model which defines the evolution of the state from time k to time $k+1$ as:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + v_k \end{aligned} \tag{2.28}$$

where $A \in \mathbb{R}^{n_x \times n_x}$ is the state transition matrix applied to the previous state vector $x_k \in \mathbb{R}^{n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$ is the control-input matrix applied to the control vector $u_k \in \mathbb{R}^{n_u}$, $y_k \in \mathbb{R}^{n_y}$ is the measurement vector, $C \in \mathbb{R}^{n_y \times n_x}$ is the measurement matrix, $w_k \in \mathbb{R}^{n_x}$ and $v_k \in \mathbb{R}^{n_y}$ are the noise vector on system and measurement equations respectively, both of them are assumed to be zero-mean Gaussian with the covariance Σ_w and Σ_v, i.e., $w_k \sim \mathcal{N}(0, \Sigma_w)$ and $v_k \sim \mathcal{N}(0, \Sigma_v)$.

Kalman filter algorithm consists of two stages: prediction and update. Note that the terms "prediction" and "update" are often called "propagation" and "correction," respectively, in different literature. The prediction equations of Kalman filter algorithm is:

$$\begin{aligned} \hat{x}_{k+1|k} &= A\hat{x}_{k|k} + Bu_k \\ \Sigma_{k+1|k} &= A\Sigma_{k|k}A^{\mathrm{T}} + \Sigma_w \end{aligned} \tag{2.29}$$

and the measurement update equations:

$$\begin{aligned} &L_{k+1} = \Sigma_{k+1|k}C^{\mathrm{T}}(C\Sigma_{k+1|k}C^{\mathrm{T}} + \Sigma_v)^{-1} \\ &\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + L_{k+1}(y_{k+1} - h(\hat{x}_{k+1|k}, 0)) \\ &\Sigma_{k+1|k+1} = (I - L_{k+1}C)\Sigma_{k+1|k}, \end{aligned} \tag{2.30}$$

In the above equations, $\hat{(\cdot)}$ means an estimate of a variable. That is, $\hat{x}_{k+1|k}$ denotes the estimate of x at time $k+1$ given observations up to and including at time k. Σ denotes the state error covariance. It encrypts the error covariance that the filter thinks the estimate error has. L_{k+1} is the Kalman filter gain.

The information state update (2.25a) and (2.25b) in linear Gaussian system can be represented by (2.29) and (2.30), where $\xi_{k|k} \sim \mathcal{N}(\hat{x}_{k|k}, \Sigma_{k|k})$.

2.2.2.2 Extended Kalman filter

The extended Kalman filter (EKF) can be viewed as a nonlinear version of the Kalman filter that linearized the models about a current estimate. Suppose we have the following models for state transition and measurement equation

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w_k) \\ y_k &= h(x_k, v_k) \end{aligned} \tag{2.31}$$

where $f(\cdot)$ and $h(\cdot)$ are the state transition and measurement equation. The noise w_k and v_k are still assumed to be zero-mean Gaussian with the covariance Σ_w and Σ_v, i.e., $w_k \sim \mathcal{N}(0, \Sigma_w)$ and $v_k \sim \mathcal{N}(0, \Sigma_v)$.

The EKF adapted techniques from calculus, namely multivariate Taylor series expansions, to linearize a model about a working point. The prediction of EKF is:

$$\begin{aligned} \hat{x}_{k+1|k} &= f(\hat{x}_{k|k}, u_k, 0) \\ \Sigma_{k+1|k} &= f_{x,k}\Sigma_{k|k}f_{x,k}^{\mathrm{T}} + f_{w,k}\Sigma_w f_{w,k}^{\mathrm{T}}, \end{aligned} \tag{2.32}$$

and the measurement update equations:

$$\begin{aligned} L_{k+1} &= \Sigma_{k+1|k}h_{x,k}^{\mathrm{T}}(h_{x,k}\Sigma_{k+1|k}h_{x,k}^{\mathrm{T}} + h_{v,k}\Sigma_v h_{x,k}^{\mathrm{T}})^{-1} \\ \hat{x}_{k+1|k+1} &= \hat{x}_{k+1|k} + L_{k+1}(y_{k+1} - h(\hat{x}_{k+1|k}, 0)) \\ \Sigma_{k+1|k+1} &= (I - L_{k+1}h_{x,k})\Sigma_{k+1|k}, \end{aligned} \tag{2.33}$$

where $f_{x,k}$ and $f_{w,k}$ denote the derivatives of f with respect to x and w at time k, $h_{x,k}$ and $h_{v,k}$ denote the derivatives of h with respect to x and v at time k.

A notable point is, unlike its linear counterpart, the EKF in general is *not* an optimal estimator (it is optimal if the measurement and the state transition model are both linear, as in that case the EKF is identical to the regular one). In addition, if the initial estimate of the state is wrong, or if the process is modeled incorrectly, the filter may quickly diverge, owing to its linearization.

2.2.2.3 Particle filter

Filters based on Kalman filter principle such as EKF and unscented Kalman filter (Wan and Van Der Merwe (2000)) can get better results in some nonlinear systems, but they cannot get rid of Gaussian approximations.

Particle filters (PF), or sequential Monte Carlo methods (Del Moral (1997); Liu and Chen (1998)), are a set of Monte Carlo algorithms used to solve filtering problems arising in Bayesian statistical inference. Particle filtering uses a set of particles (also called samples) to represent the posterior distribution of a stochastic process given the noisy and/or partial observations. The state-space model can be nonlinear and the initial state and noise distributions can take any form required. Its core idea is to express its distribution through random state particles drawn from the posterior probability. As the number of samples increases, the posterior density function provides a closer approximation to the true representation (optimal Bayesian solution).

The Monte Carlo sampling method means to find a posterior probability distribution $p(x|o)$. In the most case, we are interested in the expectation related to this posterior, for example, $\mathbf{E}_{x|o}[f(x)]$:

$$\mathbf{E}_{x|o}[f(x)] = \int f(x)p(x|o)dx \approx \frac{1}{N}\sum_{i=1}^{N} f(x^i). \tag{2.34}$$

Here N samples are sampled from $p(x|o)$.

When $p(x|o)$ is complicated or has high dimension, it is difficult to sample directly from distribution $p(x|o)$, so we need to introduce importance sampling. In importance sampling, we involve a simple and easily-sampled distribution $q(x|o)$, called proposal distribution. Then we have

$$\mathbf{E}_{x|o}[f(x)] = \int f(x)p(x|o)dx = \int f(x)\frac{p(x|o)}{q(x|o)}q(x|o)dx \approx \frac{1}{N}\sum_{i=1}^{N} f(x^i)\frac{p(x^i|o)}{q(x^i|o)}. \tag{2.35}$$

Here N samples are sampled from $q(x|o)$, and $W^i = \frac{p(x^i|o)}{q(x^i|o)}$ is called the weight of the ith sample.

Now we can introduce the importance sampling to a filter problem, i.e., $p(x_k|o_k)$. At time k, the weight of ith sample can be written as $W_k^i = \frac{p(x_k^i|o_k)}{q(x_k^i|o_k)}$.

In the iterative calculation of the filtering problem, we need to find N weights at each moment, which is computationally demanding. Therefore, we hope to find a recursive formula so that the weight of current moment W_k can be derived by the weight of previous moment W_{k-1}. From this idea, a sequential (i.e., recursive) version of importance sampling is introduced, that is, sequence importance sampling (SIS).

We first take a look at the molecule of weight

$$\begin{aligned} p(x_k|o_k) &= \frac{p(x_k, o_k)}{p(o_k)} \\ &= \frac{1}{p(o_k)} p(o_k|x_k, o_{k-1})p(x_k, o_{k-1}) \\ &= \frac{1}{p(o_k)} p(o_k|x_k)p(x_k|x_{k-1})p(x_{k-1}, o_{k-1}) \\ &= \frac{p(o_{k-1})}{p(o_k)} p(o_k|x_k)p(x_k|x_{k-1})p(x_{k-1}|o_{k-1}) \end{aligned} \tag{2.36}$$

Here $p(o_{k-1})$ and $p(o_k)$ are known constant.

Algorithm 2.2: Particle filter

```
Parameters:
  $N$ number of particles;
Initialization:
  current time instant $k$ ;
  transition distribution $p(x_{k+1}|x_k)$ ;
  observation distribution $p(o_k|x_k)$ ;
  previous weights $W_{k-1}$ ;
for $j = 1 : N$ do
    Sample from proposal distribution (2.39) $x_k^j \sim q(x_k|x_{k-1}, o_k)$;
    Compute weight by (2.40) $W_k^i \propto p(o_k|x_k)W_{k-1}^i$;
end
Normalize weights $W_k^j$ by $W_k^j = \frac{W_k^j}{\sum_{i=1}^N W_k^i}$ ;
Calculate new estimation by (2.35);
```

By assuming the proposal distribution q be the following form:

$$q(x_k|o_k) = q(x_k|x_{k-1}, o_k)q(x_{k-1}, o_{k-1}), \tag{2.37}$$

we can obtain the update equation of W_k^i:

$$W_k^i \propto \frac{p(x_k|o_k)}{q(x_k|o_k)} \propto \frac{p(o_k|x_k)p(x_k|x_{k-1})p(x_{k-1}|o_{k-1})}{q(x_k|x_{k-1}, o_k)q(x_{k-1}, o_{k-1})} \propto \frac{p(o_k|x_k)p(x_k|x_{k-1})}{q(x_k|x_{k-1}, o_k)} W_{k-1}^i. \tag{2.38}$$

Further, we can let

$$q(x_k|x_{k-1}, o_k) = p(x_k|x_{k-1}) \tag{2.39}$$

since the distribution q is arbitrary. Then the weight update equation will become

$$W_k^i \propto p(o_k|x_k)W_{k-1}^i. \tag{2.40}$$

The basic particle filter algorithm is summarized in Algorithm 2.2

2.2.3 Chance constraints

Consider constraints on state and input vectors described in very general terms by the inequalities

$$g(x_k, u_k) \leq 0 \tag{2.41}$$

where the function $g : \mathbb{R}^{n_x \times n_u} \to \mathbb{R}^{n_c}$. Depending on the noise characteristics, these constraints can be formulated as *hard constraints*, i.e., the constraint must be satisfied deterministically, or *stochastic constraints* to denote that a partial violation is allowed to consider that their deterministic fulfillment can be too tight or even impossible due to the presence of the stochastic noise (Farina, Giulioni, and Scattolini (2016)). One choice of the stochastic constraints is considering the expectation of the constraint (Primbs and Sung (2009)), i.e.,

$$\mathbf{E}[g(x_k, u_k)] \leq 0 \tag{2.42}$$

The use of expectation constraints amounts to ensure that the constraints are satisfied on average for the control problem. In this way, however, the number of occurred violations is not controlled explicitly.

In most of the SMPC literature (Cannon et al. (2010, 2009); Korda et al. (2014); Schildbach et al. (2014)), the stochastic constraint is formulated by the chance (or probabilistic) constraint:

$$\mathrm{P}[g(x_k, u_k) \leq 0] \geq 1 - p_{vio} \tag{2.43}$$

where p_{vio} is a design parameter to be tuned to obtain a trade-off between performance and constraint violation. In this thesis, we focus on the chance constraints (2.43). For a more detailed explanation of the stochastic constraints and a clear analysis of their effects on the control problem, please see Cinquemani, Agarwal, Chatterjee, and Lygeros (2011).

In general, $g(x_k, u_k)$ is a vector, when the purpose of the constraint is to express the probability that the state and control are inside a certain set, the chance constraint is called *joint chance constraint.* Otherwise, if the constraint is defined element-wise, the constraint is called *individual chance constraint.*

Under an individual chance constraint, each line of the original constraint can be transformed individually. The chance constraint (2.43) would be expanded out to be

$$\mathrm{P}[g_i(x_k, u_k) \leq 0] \geq 1 - p_{vio},\ i = 1, ..., n_c \tag{2.44}$$

With this transformation, each line is now a single chance constraint. They can be transformed into linear deterministic inequalities and analytical solutions can be obtained easily.

Under a joint chance constraint, the original constraint as a whole is reformulated as one chance constraint.

$$\mathrm{P}[g_i(x_k, u_k) \leq 0,\ i = 1, ..., n_c] \geq 1 - p_{vio}, \tag{2.45}$$

Individual chance constraints are easy to solve, but they can only guarantee that each line satisfies the constraint to a certain confidence level. Joint chance constraint ensures that the constraint as a whole is satisfied to a certain confidence level. However, it is incredibly difficult to solve.

In simple cases, where decision and random variables can be decoupled, the constraint can be transformed into deterministic constraints using probability density functions, and deterministic optimization techniques can be used to solve the problem. In more complicated cases where decision and random variables can interact in a way such that it is impossible to decouple them, the problem is currently impossible to solve. Next, we will introduce how to reformulate the chance constraint in a simple case.

Consider the linear chance constraints where decision and random variables can be decoupled, and the random variable ω is Gaussian, i.e., $\omega \sim \mathcal{N}(\mu, \Sigma_\omega)$:

$$\mathrm{P}[H^{\mathrm{T}}\omega + \chi \leq h] \geq \mathbf{1} - p_{vio}, \tag{2.46}$$

where χ is the decision variable. It can be analytically reformulated as a deterministic constraint:

$$\mathrm{P}[\Lambda^{-1}(H^{\mathrm{T}}\omega + \chi - H^{\mathrm{T}}\mu - \chi) \leq \Lambda^{-1}(h - H^{\mathrm{T}}\mu - \chi)] \geq \mathbf{1} - p_{vio}, \tag{2.47}$$

where $H^{\mathrm{T}}\Sigma_\omega H^{\mathrm{T}} = \Lambda\Lambda^{\mathrm{T}}$, and Λ can be obtained by matrix decomposition methods such as the Cholesky decomposition or LDL decomposition. By defining the cumulative distribution function (CDF), this equation can be rewritten as

$$\Phi(\Lambda^{-1}(h - H^{\mathrm{T}}\mu - \chi)) \geq \mathbf{1} - p_{vio}, \tag{2.48}$$

where Φ denotes the standard Gaussian CDF. Considering the inverse of the standard Gaussian CDF Φ^{-1}, we obtain the following result:

$$H^{\mathrm{T}}\mu + \chi + \Lambda\Phi^{-1}(\mathbf{1} - p_{vio}) \leq h, \tag{2.49}$$

and the Φ^{-1} can be computed offline only once with arbitrary precision.

We can summarize the analytic reformulation of the chance constraint as a deterministic constraint for μ:

$$H^{\mathrm{T}}\mu + \chi + t(\mathbf{1} - p_{vio}) \leq h, \tag{2.50}$$

where $t(\mathbf{1} - p_{vio}) = \Lambda\Phi^{-1}(\mathbf{1} - p_{vio})$ is the constraint-tightening level of a chance constraint, which implies that the chance constraint is a type of tightened constraint. Furthermore, even if the distribution of the random variable is not specified, if its mean μ and covariance matrix Σ_ω are known, it is still possible to find an approximation using the Chebyshev inequality (Marshall and Olkin (1960)).

Chapter 3

Efficient Control Parameterization Method for Linear SMPC

3.1 Introduction

One of the main characteristics of SMPC is the requirement of a closed-loop control policy. It is common knowledge that optimizing over open-loop control sequences directly leads to very conservative control performance and may be infeasible when a constraint exists (Mayne et al. (2000)). Thus, the optimization should be done using families of feedback control policies. The core difficulty with this type of feedback policy is that optimizing the feedback policy over arbitrary nonlinear functions is extremely difficult. Proposals that take this approach are typically intractable (e.g., based on dynamic programming (Diehl and Bjornberg (2004)) or based on the generation of disturbance sequences from a certain set, as in Scokaert and Mayne (1998)).

In the case of linear SMPC, pre-parameterize the control policy in terms of the affine functions of the sequence of states is a natural choice. However, the set of constraint-admissible policies of this form is non-convex (Lofberg (2003)). Many approaches to solving this problem have been proposed. A widely adopted approach is to fix a stabilizing feedback gain over the prediction horizon, which is called *tube-based* approaches (Cannon, Cheng, Kouvaritakis, and Raković (2012); Lee and Kouvaritakis (1999); Mayne, Seron, and Raković (2005)). Though tractable, this approach is problematic since the method of selecting the gain to minimize conservativeness is unclear.

Another solution is to parameterization of the control policy as an affine function of the disturbance, called the *affine disturbance feedback* control policy. This kind of parameterization is very old in stochastic programming (Garstka and Wets (1974)) and has been shown to be equivalent to the affine state feedback policy in Goulart et al. (2006). The advantage of affine disturbance feedback control parameterization is that the set of its decision variables is guaranteed to be convex. However, the main disadvantage of this parameterization is that the number of decision variables grows quadratically with the prediction horizon, so the real-time calculation is disastrous when the prediction horizon grows.

In this chapter, we intend to reduce the computation time while taking advantage of the affine disturbance feedback control policy. Many researchers have considered related works. In Muñoz-Carpintero, Kouvaritakis, and Cannon (2016), a striped prediction scheme was proposed with number of variables and constraints that only grow linearly with the prediction horizon. In Kouvaritakis, Cannon, and Muñoz-Carpintero (2013), the authors used a striped lower triangular control policy for disturbance compensation in stochastic MPC and further reduced the computation time by performing the computation of the disturbance compensation matrix offline. In Kouvaritakis et al. (2013), the control parameterization combined state feedback, feedforward, and disturbance feedback to guarantee feasibility and stability, making its structure rather complex.

Our control parameterization method proposed in this chapter combines the related simplification techniques of affine disturbance feedback to create a concise controller form; it is shown to be equivalent to state feedback control policies. This equivalence allows us, in principle, to directly apply the stability results discussed in research related to state feedback control policies (e.g., discussion on systems with bounded disturbances, see C. Wang, Ong, and Sim (2009), Cannon et al. (2010) or unbounded disturbances Farina, Giulioni, Magni, and Scattolini (2015), Cannon et al. (2009)). As shown in a later section, our control policy admits a feasible domain that is the same as a corresponding state feedback control policy but whose set of feasible decision variables is convex. Compared with the original affine disturbances control law, this approach significantly reduces the number of decision variables with stability results similar to those of state feedback parameterization.

The rest of this chapter is organized as follows. Section 3.2 discusses the problem settings used in this chapter. Section 3.3 discusses the existing control parameterization methods. Section 3.4 discusses our proposed simplified affine disturbance feedback policy and the properties of this method. Section 3.5 presents our simulation results. Section 3.6 summarizes this chapter.

3.2 Problem Statement

In this chapter, we consider a linear discrete-time system with an additive disturbance:

$$x_{k+1} = Ax_k + Bu_k + Ew_k, \tag{3.1}$$

where k is the discrete time, $x_k \in \mathbb{R}^{n_x}$ denotes the state, $u_k \in \mathbb{R}^{n_u}$ denotes the control input, and $w_k \in \mathbb{R}^{n_w}$ is a random disturbance. For convenience, the prediction of the system's behavior over a finite horizon N is described as

$$\mathbf{x} = \mathbf{A}x_0 + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w}, \tag{3.2}$$

where $\mathbf{x} = [x_0^{\mathrm{T}}, x_1^{\mathrm{T}}, ..., x_N^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{(N+1)n_x}$ denotes a sequence of system states, $\mathbf{u} = [u_0^{\mathrm{T}}, ..., u_{N-1}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{Nn_u}$ denotes a sequence of control inputs, and $\mathbf{w}_t = [w_0^{\mathrm{T}}, ..., w_{N-1}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{Nn_w}$ is a sequence of stochastic disturbances. Matrices $\mathbf{A} \in \mathbb{R}^{(N+1)n_x \times n_x}$, $\mathbf{B} \in \mathbb{R}^{(N+1)n_x \times Nn_u}$, and $\mathbf{E} \in \mathbb{R}^{(N+1)n_x \times Nn_w}$ are given as follows:

$$\mathbf{A} := \begin{bmatrix} I \\ A \\ \vdots \\ A^N \end{bmatrix}, \quad \mathbf{B} := \begin{bmatrix} O & O & \cdots & O \\ B & O & \cdots & O \\ AB & B & \cdots & O \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1}B & \cdots & AB & B \end{bmatrix},$$

$$\mathbf{E} := \begin{bmatrix} O & O & \cdots & O \\ E & O & \cdots & O \\ AE & E & \cdots & O \\ \vdots & \ddots & \ddots & \vdots \\ A^{N-1}E & \cdots & AE & E \end{bmatrix}.$$

We make the following assumptions in this chapter:

Assumption 3.1. *A measurement of all states is available at each sample instant.*

Assumption 3.2. *Matrix E is column full rank.*

Assumption 3.3. *The disturbances are assumed to be independent and identically normally distributed random variables (i.e., $\mathbf{w} \sim \mathcal{N}(0, I)$).*

Besides the dynamics, the chance constraints on state and control input are considered as follows

$$\mathrm{P}[\mathbf{G}\mathbf{x} \leq \mathbf{g}] \geq 1 - p_{vio,x}, (i = 1, \ldots, a), \tag{3.3a}$$
$$\mathrm{P}[\mathbf{S}\mathbf{u} \leq \mathbf{s}] \geq 1 - p_{vio,u}, (i = 1, \ldots, b), \tag{3.3b}$$

where $\mathbf{G} \in \mathbb{R}^{l\times(N+1)n_x}$, $\mathbf{g} \in \mathbb{R}^l$, $\mathbf{S} \in \mathbb{R}^{q\times Nn_u}$, and $\mathbf{s} \in \mathbb{R}^q$; $p_{vio,x}$ and $p_{vio,u}$ denote the probability level of constraint violation for state and input, respectively.

The SMPC problem with chance constraints is stated as follows.

Problem 3.1 (SMPC)**.**

$$\begin{aligned}
\min_{\mathbf{u}} \mathbf{E} & \left[\sum_{k=0}^{N-1}(x_k^{\mathrm{T}}Qx_k + u_k^{\mathrm{T}}Ru_k) + x_N^{\mathrm{T}}Q_Nx_N\right] \\
\text{s.t.}\quad & \mathbf{x} = \mathbf{A}x_0 + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w} \\
& \mathrm{P}[\mathbf{G}\mathbf{x} \le \mathbf{g}] \ge 1 - \alpha_x \\
& \mathrm{P}[\mathbf{S}\mathbf{u} \le \mathbf{s}] \ge 1 - \alpha_u \\
& \mathbf{w} \sim \mathcal{N}(0, I),
\end{aligned}$$

where $Q > 0$, $Q_N > 0$, and $R > 0$ are given symmetric matrices of appropriate dimensions.

3.3 Various Control Parameterization Methods

3.3.1 Affine state feedback parameterization

Tube-based SMPC uses a pre-fixed feedback gain to construct the control policy as

$$u_k = Kx_k + v_k, \tag{3.4}$$

where $v_k \in \mathbb{R}^{n_u}$ are optimization variables, and $K \in \mathbb{R}^{n_u\times n_x}$ is a pre-computed gain which stabilizes $A + BK$. In this way, we say that there is at least some kind of feedback in the system via closed-loop prediction $A + BK$, although not optimal.

Remark 3.1. *It is common to choose K as the solution of a linear quadratic control problem, with state and control weights Q, R, for the nominal model of system (3.1).*

To incorporate feedback predictions in our framework, write the feedback predictions in an augmented form.

$$\mathbf{u} = \mathbf{K}\mathbf{x} + \mathbf{v} \tag{3.5}$$

where $\mathbf{v}$ is a column vector, and $\mathbf{K}$ is a block diagonal matrix as follows

$$\begin{aligned}
\mathbf{v} :&= \left[v_0^{\mathrm{T}}, \cdots, v_{N-1}^{\mathrm{T}}\right]^{\mathrm{T}}, \\
\mathbf{K} :&= \begin{bmatrix} K & \cdots & \cdots & O & O \\ 0 & K & \cdots & O & O \\ \vdots & \ddots & \ddots & \vdots & O \\ 0 & \cdots & 0 & K & O \end{bmatrix}.
\end{aligned} \tag{3.6}$$

We can naturally extend the form of the $\mathbf{K}$ matrix to more general forms, for example, different feedback matrices along the diagonal, or feedback terms also in the lower triangular part of the matrix as follows

$$\mathbf{K} := \begin{bmatrix} K_1 & \cdots & \cdots & O & O \\ 0 & K_2 & \cdots & O & O \\ \vdots & \ddots & \ddots & \vdots & O \\ 0 & \cdots & 0 & K_N & O \end{bmatrix} \text{ or } \mathbf{K} : = \begin{bmatrix} K_{1,1} & \cdots & \cdots & O & O \\ K_{2,1} & K_{2,2} & \cdots & O & O \\ \vdots & \ddots & \ddots & \vdots & O \\ K_{N,1} & \cdots & K_{N,N-1} & K_{N,N} & O \end{bmatrix}, \tag{3.7}$$

where all elements in $\mathbf{K}$ are optimization variables. The only requirement is that the control policy is causal in the sense that u_j only depends on x_i, $i \leq j$.

3.3.2 Affine disturbance feedback parameterization

Let us first take a look at the predictions obtained by the affine state feedback policy (3.5):

$$\begin{aligned} \mathbf{x} &= \mathbf{A}x_0 + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w} \\ \mathbf{u} &= \mathbf{K}\mathbf{x} + \mathbf{v} \end{aligned} \tag{3.8}$$

Solving the above parametric equation for $\mathbf{x}$ and $\mathbf{u}$:

$$\mathbf{x} = (I - \mathbf{B}\mathbf{K})^{-1}(\mathbf{A}x_0 + \mathbf{B}\mathbf{v} + \mathbf{E}\mathbf{w}) \tag{3.9a}$$

$$\mathbf{u} = \mathbf{K}(I - \mathbf{B}\mathbf{K})^{-1}(\mathbf{A}x_0 + \mathbf{B}\mathbf{v} + \mathbf{E}\mathbf{w}) + \mathbf{v} \tag{3.9b}$$

The problem is that the mapping from $\mathbf{K}$ and $\mathbf{v}$ to $\mathbf{x}$ and $\mathbf{u}$ is nonlinear, hence optimization over both $\mathbf{K}$ and $\mathbf{v}$ is likely to be a non-convex problem, where the original problem 3.1 is just a simple QP with linear constraints.

To overcome the problem caused by affine state feedback parameterization, we intend to solve an equivalent convex optimization problem for the exact optimal solution. Let us take a closer look at equation (3.9b)

$$\mathbf{u} = (\mathbf{K}(I - \mathbf{B}\mathbf{K})^{-1}(\mathbf{A}x_0 + \mathbf{B}\mathbf{v}) + \mathbf{v}) + \mathbf{K}(I - \mathbf{B}\mathbf{K})^{-1}\mathbf{E}\mathbf{w}. \tag{3.10}$$

We can find that $\mathbf{u}$ is composed of a constant part, $\mathbf{K}(I - \mathbf{B}\mathbf{K})^{-1}(\mathbf{A}x_0 + \mathbf{B}\mathbf{v}) + \mathbf{v}$, and one linear term with respect to the disturbances, $\mathbf{K}(I - \mathbf{B}\mathbf{K})^{-1}\mathbf{E}\mathbf{w}$. Keep in mind that the control action $\mathbf{u}$ is causal, which means u_j is only affected by w_i, $i < j$.

Now we can give an alternative parameterization by the above discussions:

$$\mathbf{u} = \mathbf{M}'\mathbf{w} + \mathbf{h}' \tag{3.11}$$

where

$$\begin{aligned}\mathbf{h}' &:= \left[h_0^{\mathrm{T}}, \cdots, h_{N-1}^{\mathrm{T}}\right]^{\mathrm{T}}, \\ \mathbf{M}' &:= \begin{bmatrix} O & \cdots & \cdots & O \\ M_{1,0} & O & \cdots & O \\ \vdots & \ddots & \ddots & \vdots \\ M_{N-1,0} & \cdots & M_{N-1,N-2} & O \end{bmatrix}.\end{aligned} \tag{3.12}$$

and $M_{i,j} \in \mathbb{R}^{n_u \times n_w}$ and $h_i \in \mathbb{R}^{n_u}$ are decision variables. The control parameterization (3.11) is called *affine disturbance feedback* (ADF) since the control sequence is now parameterized directly in the disturbance (Lofberg (2003)).

3.4 Simplified Affine Disturbance Feedback

3.4.1 SADF parameterization

The affine disturbance feedback control law (3.11) can provide a convex optimization problem. However, the number of the decision variables in $\mathbf{M}'$ and $\mathbf{v}'$ is

$$n_u n_w (N-1) + n_u n_w (N-2) + \cdots + n_u n_w + n_u = \frac{N(N-1) n_u n_w}{2} + n_u, \tag{3.13}$$

which means it grows quadratically with the prediction horizon, making it difficult to use in real-time calculations. Regarding calculation time, we consider a kind of ADF that contains fewer decision variables, called *simplified affine disturbance feedback* (SADF), which takes the form of:

$$u_i = \sum_{k=0}^{i-1} M_{i-k} w_k + h_i, \tag{3.14}$$

where u_i is an affine function of the i disturbances , $M_{i-k} \in \mathbb{R}^{n_x \times n_u}$ is the matrix of coefficients associated with the past k-step disturbances, and w_k is a disturbance that was realized at the last step. For convenience, we also define matrix $\mathbf{M} \in \mathbb{R}^{N n_u \times N n_w}$ and vector $\mathbf{h} \in \mathbb{R}^{N n_u}$ in such a way that

$$\begin{aligned}\mathbf{h} &:= \left[h_0^{\mathrm{T}}, \cdots, h_{N-1}^{\mathrm{T}}\right]^{\mathrm{T}}, \\ \mathbf{M} &:= \begin{bmatrix} O & \cdots & \cdots & O \\ M_1 & O & \cdots & O \\ \vdots & \ddots & \ddots & \vdots \\ M_{N-1} & \cdots & M_1 & O \end{bmatrix}.\end{aligned} \tag{3.15}$$

This parameterization is a simplified version of the original affine disturbance feedback control policy (3.11). Note that $\mathbf{M}$ has a strictly lower triangular Toeplitz structure with a constant block in each descending diagonal from left to right and only contains just $(N-1)n_u n_w$ different decision variables, that grow linearly with the prediction horizon.

3.4.2 Problem reformulation

Next we consider how to reformulate the stochastic problem into a deterministic one based on the SADF (3.14). Linear individual chance constraints in the form (3.3a) have been shown to be equivalent to a second-order cone constraint in Schwarm and Nikolaou (1999). In our setup, $\mathbf{w} \sim \mathcal{N}(0, I)$, the individual chance constraint (3.3a) can be transformed into a deterministic constraint with the help of a standard Gaussian cumulative distribution function (CDF). By standardizing the distribution, we have

$$\mathrm{P}\left[\frac{\mathbf{GEw}}{\|\mathbf{GE}\|} \leq \frac{-\mathbf{G}(\mathbf{A}x_0 + \mathbf{Bu}) + \mathbf{g}}{\|\mathbf{GE}\|}\right] \geq 1 - p_{vio,x}, \tag{3.16}$$

where $\frac{\mathbf{GEw}}{\|\mathbf{GE}\|}$ is a standard Gaussian distribution vector.

Given the definition of CDF, this equation can be rewritten as

$$\Phi\left(\frac{-\mathbf{G}(\mathbf{A}x_0 + \mathbf{Bu}) + \mathbf{g}}{\|\mathbf{GE}\|}\right) \geq 1 - p_{vio,x}, \tag{3.17}$$

where Φ is the standard Gaussian CDF. Thus, the chance constraint (3.3a) can be transformed into a deterministic one by using the inverse of standard Gaussian CDF as

$$\mathbf{G}(\mathbf{A}x_0 + \mathbf{Bu}) + \Phi^{-1}(1 - p_{vio,x})\|\mathbf{GE}\| \leq \mathbf{g}, \tag{3.18}$$

and so can (3.3b).

By substituting the SADF parameterization (3.14) into (3.18), the chance constraints under the SADF control parameterization can be transformed into deterministic ones as

$$\mathbf{G}(\mathbf{A}x_0 + \mathbf{Bh}) + \Phi^{-1}(1 - p_{vio,x})\|\mathbf{G}(\mathbf{BM} + \mathbf{E})\| \leq \mathbf{g}. \tag{3.19}$$

Similarly, chance constraints on control input can also be transformed into deterministic ones as

$$\mathbf{Sh} + \Phi^{-1}(1 - p_{vio,u})\|\mathbf{SM}\| \leq \mathbf{s}. \tag{3.20}$$

These constraints are second-order cone constraints that concern the decision variable $(\mathbf{M}_t, \mathbf{h}_t)$, and deterministic equivalent chance constraints on the state (3.19) and control input (3.20) can be rewritten as $(\mathbf{x}, \mathbf{u}) \in Z$ for convenience.

Thus, a reformulation of Problem 3.1 using SADF (3.14) is stated as follows:

Problem 3.2 (SMPC with SADF)**.**

$$
\begin{aligned}
\min_{\mathbf{M},\mathbf{h}} \mathbf{E} & \left[\sum_{k=0}^{N-1} (x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k) + x_N^{\mathrm{T}} Q_N x_N \right] \\
\text{s.t.} \quad & \mathbf{x} = \mathbf{A} x_0 + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w} \\
& \mathbf{u} = \mathbf{M}\mathbf{w} + \mathbf{h} \\
& (\mathbf{x}, \mathbf{u}) \in Z \\
& \mathbf{w} \sim \mathcal{N}(0, I).
\end{aligned}
$$

The expectation $\mathbf{E}$ in the objective function can be obtained by substituting the expectation and variance information of disturbance w_k. Thus, Problem 3.2 can be treated as a deterministic MPC with nonlinear constraints, and it can be solved in principle by general constrained optimization methods, e.g., interior-point method introduced in Section 2.1.2. Existing solvers such as CasADi (Andersson, Gillis, Horn, Rawlings, and Diehl (2019)) in MATLAB, C/GMRES (Ohtsuka (2004)) in MAPLE can also be used to solve this problem.

3.4.3 Properties of SADF

Although Problem 3.2 inherits the characteristics of the convex optimization problem of affine disturbance feedback parameterization, there is still a question of whether SADF is the reasonable feedback control policy we want. In this section, we will discuss the relationship between SADF and the state feedback control policy further.

The set of admissible pair $(\mathbf{M}, \mathbf{h})$ for SADF is defined as

$$
\Xi^{\text{sadf}}(x) := \left\{ (\mathbf{M}, \mathbf{h}) \;\middle|\; \begin{aligned} & (\mathbf{M}, \mathbf{h}) \text{ satisfies (3.15)}, \\ & \mathbf{x} = \mathbf{A}x + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w}, \\ & \mathbf{u} = \mathbf{M}\mathbf{w} + \mathbf{h}, \\ & (\mathbf{x}, \mathbf{u}) \in Z, \mathbf{w} \sim \mathcal{N}(0, I), \end{aligned} \right\} \tag{3.21}
$$

and the set of initial state x for which an admissible control law of the form (3.14) exists as

$$
X^{\text{sadf}} := \{x \in \mathbb{R}^n | \Xi^{\text{sadf}} \neq \emptyset\}. \tag{3.22}
$$

Remark 3.2. *Note that the original affine disturbance feedback law (3.11) subsumes the proposed SADF (3.14). Therefore, (3.14) is a more conservative approximation to (3.11).*

The original affine disturbance feedback control law (3.11) was proven to be equivalent to an affine state feedback control law in Goulart et al. (2006). Thus, the important question is whether the proposed SADF (3.11) is also an equivalent and tractable formulation of a certain state feedback law. In other words, is this a reasonable approximation?

Consider the state feedback control law:

$$u_i = \sum_{k=0}^{i} K_{i-k} x_k + v_i. \tag{3.23}$$

where matrix $\mathbf{K}$ and column vector $\mathbf{v}$ are defined as follows:

$$\begin{aligned} \mathbf{v} : &= \left[v_0^{\mathrm{T}}, \cdots, v_{N-1}^{\mathrm{T}}\right]^{\mathrm{T}} \\ \mathbf{K} : &= \begin{bmatrix} K_0 & O & \cdots & O & O \\ K_1 & K_0 & \cdots & O & O \\ \vdots & \ddots & \ddots & \vdots & O \\ K_{N-1} & \cdots & K_1 & K_0 & O \end{bmatrix}. \end{aligned} \tag{3.24}$$

The number of free variables of this structure is between the left and right sides of structure (3.7), with knowledge of prior states.

In a manner similar to (3.21), we can also define the set of admissible $(\mathbf{K}, \mathbf{v})$ as

$$\Xi^{\mathrm{sf}}(x) := \left\{ (\mathbf{K}, \mathbf{v}) \left| \begin{aligned} &(\mathbf{K}, \mathbf{v}) \text{ satisfies (3.24)}, \\ &\mathbf{x} = \mathbf{A}x + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w}, \\ &\mathbf{u} = \mathbf{K}\mathbf{x} + \mathbf{v}, \\ &(\mathbf{x}, \mathbf{u}) \in Z, \mathbf{w} \sim \mathcal{N}(0, I) \end{aligned} \right. \right\} \tag{3.25}$$

and the same for the set of initial states x for which an admissible control law of the form (3.14) is

$$X^{\mathrm{sf}} := \{x \in \mathbb{R}^n | \Xi^{\mathrm{sf}} \neq \emptyset\}. \tag{3.26}$$

Next, we show that the state feedback parameterization (3.23) and the SADF parameterization (3.14) are equivalent.

Theorem 3.1. *The sets of admissible states for control policy, (3.14) and (3.23), are identical (i.e., $X^{\mathrm{adf}} = X^{\mathrm{sf}}$ holds). Moreover, given any $x \in X^{\mathrm{adf}}$, for any admissible $(\mathbf{M}, \mathbf{h})$, there exists a pair of $(\mathbf{K}, \mathbf{v})$ yields the same state and input sequence for the same disturbance sequence $\mathbf{w}$, and vice versa.*

Proof. $X^{\mathrm{sf}} \subseteq X^{\mathrm{adf}}$: Based on the definition of an admissible set for any given $x \in X^{\mathrm{sf}}$, there exists a pair of $(\mathbf{K}, \mathbf{v})$ satisfies the constraints in (3.25). Therefore, for a given disturbance sequence $\mathbf{w}$, we can eliminate $\mathbf{x}$ for $\mathbf{u}$ through simultaneous equations (3.2) and (3.23) as

$$\mathbf{u} = \mathbf{K}(I - \mathbf{BK})^{-1}(\mathbf{A}x_0 + \mathbf{Bv} + \mathbf{Ew}) + \mathbf{v}. \tag{3.27}$$

Notably, $(I - \mathbf{BK})$ is always nonsingular since $\mathbf{BK}$ is strictly lower block triangular. The control input can be divided into terms related to disturbance $\mathbf{w}$ and terms not related to $\mathbf{w}$ as

$$\mathbf{u} = \mathbf{K}(I - \mathbf{BK})^{-1}\mathbf{Ew} + (\mathbf{K}(I - \mathbf{BK})^{-1}(\mathbf{A}x_0 + \mathbf{Bv}) + \mathbf{v}). \tag{3.28}$$

Choose $(\mathbf{M}, \mathbf{h})$ as

$$\mathbf{M} = \mathbf{K}(I - \mathbf{BK})^{-1}\mathbf{E}, \tag{3.29a}$$

$$\mathbf{h} = \mathbf{K}(I - \mathbf{BK})^{-1}(\mathbf{A}x_0 + \mathbf{Bv}) + \mathbf{v}. \tag{3.29b}$$

By applying the results in Kucerovsky, Mousavand, and Sarraf (2016), the product of two lower triangular Toeplitz matrices, $\mathbf{B}$ and $\mathbf{K}$, is again a lower triangular Toeplitz matrix. Moreover, if a lower triangular Toeplitz matrix is invertible, then its inverse is also Toeplitz. One can easily verify that $\mathbf{M}$ has a strictly lower block triangular Toeplitz structure, meaning the pair of $(\mathbf{M}, \mathbf{h})$ satisfies (3.21) and can yield the same input sequence as the given pair of $(\mathbf{K}, \mathbf{v})$. Thus, $x \in X^{\mathrm{sf}}$ implies $x \in X^{\mathrm{adf}}$.

$X^{\mathrm{adf}} \subseteq X^{\mathrm{sf}}$: For any given $x \in X^{\mathrm{adf}}$, there exists a pair of $(\mathbf{M}, \mathbf{h})$ that satisfies the constraints in (3.21). For a given disturbance sequence $\mathbf{w}$, we can also eliminate $\mathbf{w}$ for $\mathbf{u}$ through simultaneous equations (3.2) and (3.14) as

$$\mathbf{u} = (I + \mathbf{ME}^{-}\mathbf{B})^{-1}(\mathbf{ME}^{-}(\mathbf{x} - \mathbf{A}x_0) + \mathbf{h}), \tag{3.30}$$

where $\mathbf{E}^{-} \in \mathbb{R}^{Nr \times Nn}$ denotes the left inverse of $\mathbf{E}$, so that $\mathbf{E}^{-}\mathbf{E} = I$, and it is given explicitly as

$$\mathbf{E}^{-} := \begin{bmatrix} O & E^{-} & O & \cdots & O \\ O & -E^{-}A & E^{-} & \cdots & O \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ O & O & \cdots & -E^{-}A & E^{-} \end{bmatrix},$$

where $E^{-} \in \mathbb{R}^{r \times n}$ denotes the left inverse of E. Note that E^{-} exists because E is a full column rank matrix by Assumption 2. Since $\mathbf{ME}^{-}\mathbf{B}$ is strictly lower block triangular, $I + \mathbf{ME}^{-}\mathbf{B}$ is always nonsingular. So, we can choose $(\mathbf{K}, \mathbf{v})$ as

$$\mathbf{K} = (I + \mathbf{ME}^{-}\mathbf{B})^{-1}\mathbf{ME}^{-}, \tag{3.31a}$$

$$\mathbf{v} = (I + \mathbf{ME}^{-}\mathbf{B})^{-1}(\mathbf{h} - \mathbf{ME}^{-}\mathbf{A}x_0). \tag{3.31b}$$

It is easy to confirm $\mathbf{ME}^{-}$ and $\mathbf{ME}^{-}\mathbf{B}$ as being strictly lower block triangular Toeplitz. Through the properties of Toeplitz matrix multiplication, we see that $\mathbf{K}$ has a strictly lower triangular Toeplitz structure, meaning the pair of $(\mathbf{K}, \mathbf{v})$ satisfies (3.25) and can yield the same input sequence as the given pair of $(\mathbf{M}, \mathbf{h})$. Thus, $x \in X^{\text{adf}}$ implies $x \in X^{\text{sf}}$. □

Theorem 3.1 tells us that the optimal solution $(\mathbf{M}^{*}, \mathbf{h}^{*})$ obtained by SADF is actually equivalent to a state feedback control policy $(\mathbf{K}^{*}, \mathbf{v}^{*})$. In MPC implementation, the first optimal control input is applied; i.e.,

$$\pi(x_0) = K_0^{*}(x_0)x_0 + v_0^{*}(x_0). \tag{3.32}$$

Thus, the closed-loop system is given by

$$x_{k+1} = Ax_k + B\pi(x_k) + Ew_k. \tag{3.33}$$

In addition, if we suppose the disturbances are bounded with an unknown bound, and if the MPC control is defined as in (3.32), we can guarantee the closed-loop stability and feasibility. Although the assumption of bounded disturbances contravenes the original assumption that the disturbance w_k has infinite support, it is meaningful in practice. The discussion of closed-loop stability and feasibility under bounded disturbance is beyond the scope of this thesis, some similar results can be found in the robust MPC case (Goulart et al. (2006)).

3.5 Case Study

3.5.1 Numerical example settings

We tested our method with the control problem of room temperature Gwerder and Tödtli (2005). The basic control target was keeping the room temperature higher than a certain level in the presence of external disturbances. The system contained three states: let x_1 be the room temperature, let x_2 be the temperature in the wall connected to another room, and let x_3 be the temperature in the wall connected to the outside. The system was subject to three disturbances in which w_1 denoted the outside temperature, w_2 denoted the solar radiation, and w_3 denoted the internal heat gains (e.g., people, electronic devices, etc.) and all the external disturbance subjects to $\mathcal{N}(0, I)$. The only control input u was the heating, which was constrained to $0 \leq u \leq 45$ [W/m^2]. The control objective was keeping the room temperature above 21°C with minimum energy; thus, the weighting matrix is $Q = 0$, and the state constraint is treated like a chance constraint $\mathbb{P}(x \geq 21) \geq 1 - \alpha$. The parameters of the system are taken from Oldewurtel et al. (2013) as follows:

$$A := \begin{bmatrix} 0.8511 & 0.0541 & 0.0707 \\ 0.1293 & 0.8635 & 0.0055 \\ 0.0989 & 0.0032 & 0.7541 \end{bmatrix},$$
$$E := 10^{-3} \cdot \begin{bmatrix} 22.2170 & 1.7912 & 42.2123 \\ 1.5376 & 0.6944 & 2.9214 \\ 103.1813 & 0.1032 & 196.0444 \end{bmatrix},$$
$$B := \begin{bmatrix} 0.0035 \\ 0.0003 \\ 0.0002 \end{bmatrix}, \quad R := 1.$$

The following three control policies are compared:

1. SADF control policy (3.14),
2. Original affine disturbance feedback (ADF) control policy (3.11), and
3. Open-loop prediction (OP) control policy, i.e., $\mathbf{M} = 0$ in (3.14).

We carried out the simulation on a laptop computer with a 2.60 GHz Intel Core i7-6700HQ and CasADi toolkit Andersson et al. (2019) in MATLAB 2020a. In all the simulations, we subjected the system to the same disturbance realizations and constraint violation degree $\alpha_{x,i} = 0.1, \alpha_{u,i} = 0.005$.

3.5.2 Simulation analysis

Figures 3.1 and 3.2 show a 150-minute simulation with the initial state $[22, 22, 15]^{\mathrm{T}}$ and the prediction horizon $N = 6$. The results show that an OP control policy led to conservative control behavior; the room temperature was almost always higher than the room temperature of the other two control policies. In contrast, ADF and SADF led to relatively less conservative control behavior.

To compare the three controllers more intuitively, we considered the objective function value for the same optimization problem (i.e., only a single optimization starting from the same state within a prediction horizon $N = 30$). Table 3.1 compares three states. The results show that the OP controller consumes the most energy for an identical optimization problem (i.e., most conservative), while the control SADF performance is slightly more conservative than ADF.

Table 3.2 shows the computation times of the three controllers. The table shows that the computation times of OP and SADF did not grow as quickly with respect to

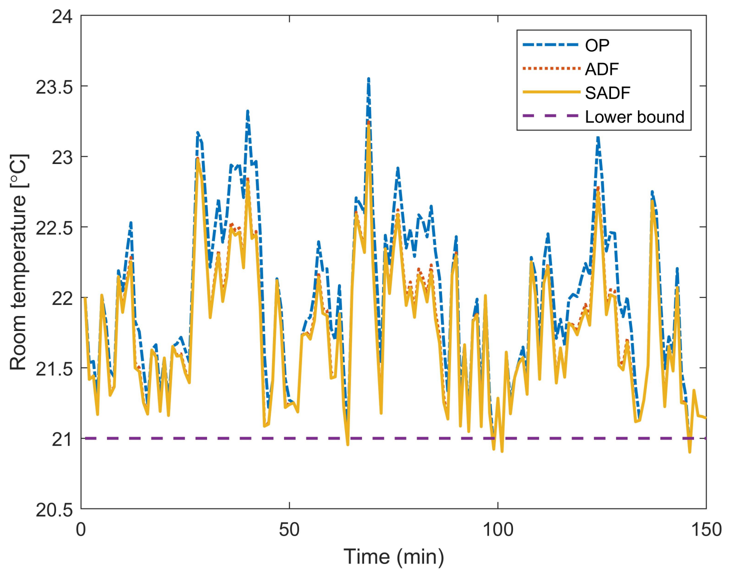

Figure 3.1: Room temperature profile [°C].

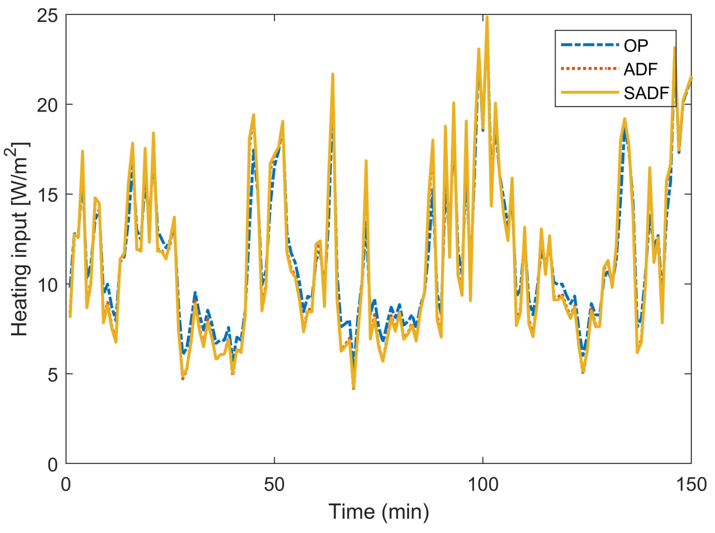

Figure 3.2: Heating [W/m^2].

the length of the prediction horizon, while the computation time of ADF grew very quickly. Our method provides an acceptable computation time in this example.

3.6 Summary

In this chapter, we proposed a simplified affine disturbance feedback control law for linear SMPC. The decision variable number decreased to $\mathcal{O}(N)$ compared with $\mathcal{O}(N^2)$

Table 3.1: Comparison of objective function values for different states under the same settings ($N = 30$).

	OP	ADF	SADF
state 1: $[28, 28, 21]^{\mathrm{T}}$	1778	1242	1256
state 2: $[22, 18, 15]^{\mathrm{T}}$	835	505	567
state 3: $[25, 25, 15]^{\mathrm{T}}$	2004	1430	1448

Table 3.2: Average computation time per update (ms).

	OP	ADF	SADF
$N = 5$	9	19	13
$N = 15$	16	697	150
$N = 30$	31	21320	1420

of the original affine disturbance feedback control law, resulting in a preferable trade-off between real-time calculation and control performance. This parameterization is shown to be equivalent to a state feedback control law, and the closed-loop stability of the SMPC problem can also be guaranteed under mild assumptions.

Chapter 4

Stochastic Dynamic Programming for Linear SMPC

4.1 Introduction

In principle, Bellman's principle of optimality (Bellman (1966)) can be used to obtain an ideal closed-loop performance; however, it is always intractable, especially in large-scale optimization (Mesbah (2018)). In a certainty-equivalent SMPC scheme, the expectation formulations of the cost function and chance constraints are transformed into deterministic formulations (Mayne et al. (2000)), which can achieve the so-called "open-loop" control performance. There are also some heuristic methods for reformulating the stochastic optimization problem for better control performance. One reformulation is the control parameterization method we discussed Chapter 3, such as pre-fixed state feedback (Bemporad (1998)), affine state feedback (Lofberg (2003)), affine disturbance feedback (Lofberg (2003)), and simplified affine disturbance feedback in Chapter 3. Another formulation directly considers the minimization of variance in the cost function rather than adding a feedback feature to the control inputs (D. Li et al. (2002)). Although these methods can artificially include some degree of closed-loop control effects, they still face some difficulties, such as the increase in decision variables and the difficulty of explicitly analyzing their degree of conservativeness.

Solving stochastic dynamic programming directly is still an attractive approach for stochastic optimal control. In control of a linear Gaussian system, LQG controller separates the estimation and control and designs the optimal control policy by solving infinite-horizon dynamic programming (Lindquist (1973)), but the separation theorem can only hold in linear Gaussian system without constraints. Kumar et al. (2018)

proposed a stochastic dual dynamic programming framework to deal with finite horizon problem, and explained how to construct terminal costs and performance bounds for SMPC by deriving and interpreting stochastic dual dynamic programming from an MPC perspective.

However, constraint handling in dynamic programming is still problematic. Bemporad, Morari, Dua, and Pistikopoulos (2002) proposed to solve a parametric solution for constrained dynamic programming which is called "explicit MPC", but it suffers from the curse of dimensionality, and can only be applied to small-scale problem. Darup and Mönnigmann (2012) and Kvasnica, Holaza, Takács, and Ingole (2015) tried to reduce the computational complexity by finding some approximate solution of explicit MPC. However, the computation burden of a parametric solution is still not acceptable in online calculation. Ferreau, Bock, and Diehl (2008) proposed an online active set strategy to overcome the limitations of explicit MPC, which assumed that the active set does not change much from one QP to the next in MPC problem, so it is also an approximation.

In this chapter, we intend to directly find an SDP solution for a linear SMPC problem with chance constraints. Our approach involves converting the stage-wise stochastic optimization problem into an equivalent deterministic problem. A recursive Riccati interior-point method (RRIPM) is proposed to solve the ensuing constrained optimization problems. The proposed method eliminates active sets in conventional explicit MPC and does not suffer from the curse of dimensionality because it finds the value function and feedback policy only for a given state using the interior-point method. Moreover, the proposed method is proven to converge globally to a stationary solution Q-superlinearly. The numerical experiment reveals that the proposed method achieves a less conservative performance with low computational complexity compared to existing methods.

This chapter is organized as follows. Section 4.2 discusses the problem settings in this chapter. Section 4.3 reformulates the problem by Bellman equation and discusses optimal conditions. Section 4.4 presents the main algorithm, recursive Riccati interior-point method. Section 4.5 considers the global convergence and local convergence rate of RRIPM algorithm. Section 4.6 gives a practical way to deal with infeasibility. Section 4.7 presents simulation results. Finally, Section 4.8 summarizes this chapter.

4.2 Problem Statement

In this chapter, we consider a linear discrete-time system with an additive disturbance:

$$x_{k+1} = Ax_k + Bu_k + Ew_k, \tag{4.1}$$

where k denotes the discrete time, $x_k \in \mathbb{R}^{n_x}$ denotes the state, $u_k \in \mathbb{R}^{n_u}$ denotes the control input, and $w_k \in \mathbb{R}^{n_w}$ denotes a random disturbance. The chance constraints on the state and polytopic constraints on the input are given as

$$\mathrm{P}[Fx_k \leq f] \geq \mathbf{1} - p_{vio},\ Gu_k \leq g, \tag{4.2}$$

where $F \in \mathbb{R}^{l_x \times n_x}$, $f \in \mathbb{R}^{l_x}$, $G \in \mathbb{R}^{l_u \times n_u}$, $g \in \mathbb{R}^{l_u}$, P denotes the probability, $p_{vio} \in \mathbb{R}^{l_x}$ denotes the constraint violation level vector.

Define an N-horizon quadratic cost function as

$$J := \mathbf{E}[\sum_{k=0}^{N-1} \frac{1}{2}(x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k) + \frac{1}{2} x_N^{\mathrm{T}} Q_N x_N] \tag{4.3}$$

where $Q \geq 0$ and $R > 0$ are symmetric matrices with appropriate dimensions, and $Q_N \geq 0$ is the terminal cost.

We make the following assumptions throughout this chapter:

Assumption 4.1. *Measurement of all states are available at each sample instant.*

Assumption 4.2. *The disturbances are subjected to an independent Gaussian distribution, i.e., $w_k \sim \mathcal{N}(0, \Sigma_w)$, and the covariance matrix Σ_w is positive definite.*

The SMPC solves the following finite-horizon SOCP with a known initial state $\bar{x}_0$ at each time instant:

Problem 4.1 (SOCP)**.**

$$\begin{aligned} \min_{(u_k)_{k=0}^{N-1}} \quad & J \\ \text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k + w_k \\ & \mathrm{P}[Fx_k \leq f] \geq \mathbf{1} - p_{vio} \\ & Gu_k \leq g \\ & x_0 = \bar{x}_0,\ w_k \sim \mathcal{N}(0, \Sigma_w). \end{aligned}$$

4.3 Bellman Equation and Optimality Conditions

4.3.1 Problem reformulation

In theory, Bellman's principle of optimality can be used to solve Problem 4.1. At each stage k, the optimal *cost-to-go* $V_k(x_k)$ (i.e., the value function) should satisfy the Bellman equation in the SDP scheme.

Problem 4.2 (Bellman equation)**.**

$$\begin{aligned} V_k(x_k) = \min_{u_k(x_k)} \ & \mathbf{E}[\frac{1}{2}x_k^{\mathrm{T}}Qx_k + \frac{1}{2}u_k^{\mathrm{T}}Ru_k + V_{k+1}(x_{k+1})]. \\ \text{s.t. } & x_{k+1} = Ax_k + Bu_k + w_k \\ & \mathrm{P}[Fx_{k+1} \le f] \ge \mathbf{1} - p_{vio} \\ & Gu_k \le g \end{aligned}$$

The solution to Problem 4.1 comprises a sequence of solutions $(V_k(x_k), u_k(x_k))$ for Problem 4.2 for $k = 0, 1, ..., N-1$, and the terminal value function is $V_N(x_N) = \frac{1}{2}x_N^{\mathrm{T}}Q_Nx_N$. Note that $V_k(x_k)$ is defined recursively backward from $V_N(x_N)$ and is not necessarily defined for every $x_k \in \mathbb{R}^n$ because of the constraints.

The available measurement information at each stage plays an important role in stochastic control and leads to different policy types. A discussion on various policies can be found in Bar-Shalom and Tse (1974). The *open-loop* policy is defined as the absence of immediate observation data at each stage, i.e., only the initial measurement can be used. The *feedback* policy indicates that the measurement information till current stage k is available for the computation of the contro1.

To obtain the closed-loop control performance, we use the concept of a feedback policy, i.e., we know the measurement of x_k at stage k, and the mean and covariance are $\bar{x}_k$ and 0, respectively. Because the full state is observable according to Assumption 4.1, the future information of x_{k+1} can be obtained as follows:

$$\bar{x}_{k+1} = \mathbf{E}[x_{k+1}] = A\bar{x}_k + Bu_k \tag{4.4}$$

$$\Sigma_{k+1} = A\Sigma_kA^{\mathrm{T}} + \Sigma_w = \Sigma_w \tag{4.5}$$

where $\bar{x}_{k+1}$ and Σ_{k+1} denote the estimated mean and covariance respectively.

Using these settings, we can remove the stochastic descriptions in Problem 4.2. The chance constraint can be converted to deterministic inequalities as in Section (2.2.3):

$$F\bar{x}_{k+1} + t(\mathbf{1} - p_{vio}) \le f, \tag{4.6}$$

where the constraint-tightening level

$$t(\mathbf{1}-\alpha) = \Phi^{-1}(\mathbf{1}-p_{vio})\Lambda_1, \tag{4.7}$$

$F\Sigma_w F^{\mathrm{T}} = \Lambda_1\Lambda_1^{\mathrm{T}}$. By substituting (4.4) into this deterministic constraint, it can be expressed by the current states and inputs as follows:

$$F(A\bar{x}_k + Bu_k) + t(\mathbf{1}-p_{vio}) \leq f. \tag{4.8}$$

Together with the input constraints, the admissible region Π_k can be rewritten as

$$\Pi_k := \{(\bar{x}_k, u_k) | C_x\bar{x}_k + C_u u_k + C_c \leq 0\}, \tag{4.9}$$

where

$$C_x = \begin{bmatrix} FA \\ 0 \end{bmatrix} \; C_u = \begin{bmatrix} FB \\ G \end{bmatrix}, \; C_c = \begin{bmatrix} t(\mathbf{1}-p_{vio}) - f \\ -g \end{bmatrix}$$

The remaining question concerns the expectation of the cost function. From the dynamic programming solution in the constrained explicit MPC case (Faísca, Kouramas, Saraiva, Rustem, and Pistikopoulos (2008)), we know that the exact form of the optimal value function in the linear case has the following quadratic form in an appropriate region for each stage:

$$V_{k+1}(x_{k+1}) = \frac{1}{2}x_{k+1}^{\mathrm{T}}P_{k+1}x_{k+1} + q_{k+1}^{\mathrm{T}}x_{k+1} + r_{k+1}. \tag{4.10}$$

where P_{k+1}, q_{k+1} and r_{k+1} are second-order, first-order and constant coefficient, respectively.

Remark 4.1. *Explicit MPC seeks to find an offline dynamic programming solution; therefore, the optimal value function and feedback policy will be a piecewise function that depends on the state x_k. By contrast, in our case, we want to find an online solution only for a given $\bar{x}_0$. Therefore, the value function and feedback policy are determined only in a certain sequence of regions for a given $\bar{x}_0$, rather than by using the piecewise form over the state space.*

Using the mean and covariance information of x_k, we can transform the expected value function as follows:

$$\begin{aligned}&\mathbf{E}[\frac{1}{2}x_k^{\mathrm{T}}Qx_k + \frac{1}{2}u_k^{\mathrm{T}}Ru_k + V_{k+1}(x_{k+1})]\\ =&\frac{1}{2}\bar{x}_k^{\mathrm{T}}Q\bar{x}_k + \frac{1}{2}\mathrm{tr}(\Sigma_k Q) + \frac{1}{2}u_k^{\mathrm{T}}Ru_k + V_{k+1}(\bar{x}_{k+1}) + \frac{1}{2}\mathrm{tr}(\Sigma_{k+1}P_{k+1}).\end{aligned} \tag{4.11}$$

Because the covariance of x_k and x_{k+1} are all constants, they can be omitted for simplicity. Now, we can provide an equivalent deterministic Bellman equation defined by mean value of state variables, and for a concise expression, we use x_k to directly represent the mean value $\bar{x}_k$ hereafter in this chapter.

Problem 4.3 (Deterministic Bellman equation)**.**

$$V_k(x_k) = \min_{u_k(x_k)} [\frac{1}{2}x_k^{\mathrm{T}}Qx_k + \frac{1}{2}u_k^{\mathrm{T}}Ru_k + V_{k+1}(x_{k+1})].$$
$$\text{s.t. } x_{k+1} = Ax_k + Bu_k$$
$$C_x x_k + C_u u_k + C_c \leq 0$$

4.3.2 Optimality conditions

The Lagrange multiplier method in equation (2.2) (Nocedal and Wright (1999)) can be used to rewrite the value function in Problem 4.3 as follows:

$$V_k(x_k) = \min_{u_k} \max_{s_k}[l(x_k, u_k, s_k) + V_{k+1}(x_{k+1})], \tag{4.12}$$

where

$$l(x_k, u_k, s_k) = \frac{1}{2}x_k^{\mathrm{T}}Qx_k + \frac{1}{2}u_k^{\mathrm{T}}Ru_k + s_k^{\mathrm{T}}(C_x x_k + C_u u_k + C_c)$$

and s_k denotes the Lagrange multiplier.

Let the stage-wise cost be $Q_k := l(x_k, u_k, s_k) + V_{k+1}(x_{k+1})$. Substituting (4.10) and the mean prediction equation (4.4) into (4.12), we can write Q_k as

$$Q_k(x_k, u_k, s_k) = \frac{1}{2}\begin{bmatrix} x_k \\ u_k \\ s_k \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Q_{xx} & Q_{xu} & Q_{xs} \\ Q_{ux} & Q_{uu} & Q_{us} \\ Q_{sx} & Q_{su} & Q_{ss} \end{bmatrix} \begin{bmatrix} x_k \\ u_k \\ s_k \end{bmatrix} + \begin{bmatrix} Q_x \\ Q_u \\ Q_s \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} x_k \\ u_k \\ s_k \end{bmatrix} + Q_c \tag{4.13}$$

where

$$\begin{aligned} Q_{xx} &= Q + A^{\mathrm{T}}P_{k+1}A,\ Q_{uu} = R + B^{\mathrm{T}}P_{k+1}B \\ Q_{ux} &= Q_{xu}^{\mathrm{T}} = B^{\mathrm{T}}P_{k+1}A,\ Q_{ss} = 0 \\ Q_{sx} &= Q_{xs}^{\mathrm{T}} = C_x,\ Q_{su} = Q_{us}^{\mathrm{T}} = C_u \\ Q_u &= B^{\mathrm{T}}q_{k+1},\ Q_x = A^{\mathrm{T}}q_{k+1},\ Q_s = C_c \\ Q_c &= \mathrm{tr}(\Sigma P_{k+1}) + r_{t+1} \end{aligned}$$

The KKT conditions for the single-stage optimization problem (4.12) called stage-wise KKT (SKKT) conditions are

$$Q_{uu}u_k + Q_{ux}x_k + Q_{us}s_k + Q_u = 0 \tag{4.14a}$$
$$S_k(Q_{sx}x_k + Q_{su}u_k + Q_s) = 0 \tag{4.14b}$$
$$Q_{sx}x_k + Q_{su}u_k + Q_s \leq 0,\ s_k \geq 0, \tag{4.14c}$$

where $S_k = \mathrm{diag}[s_k]$.

This SKKT condition can be considered a parametric equation with respect to x_k. Therefore, if we can solve these equations at each stage k, we can obtain a sequence of state-feedback solutions $u_k(x_k)$ and the corresponding optimal sequence (x_k^*, u_k^*, s_k^*). We present our main results regarding solutions to the SKKT conditions (4.14) in the next section.

4.4 Recursive Riccati Interior-Point Method

In this section, we introduce the recursive Riccati interior-point method (RRIPM) algorithm for solving constrained programming.

Note that the SKKT conditions (4.14) are nonlinear equations with nonsmooth complementary conditions, which cause numerical difficulties in the direct solution. We use the interior-point method to avoid the complicated searching of active sets and find solutions of (4.14) by applying Newton's method to the perturbed SKKT conditions El-Bakry et al. (1996) and iteratively reducing the smooth parameter. More specifically, our proposed method improves the solution of each Newton iteration by solving the perturbed SKKT conditions in the vicinity of the current trajectory for the search directions through a *backward pass* and by computing a new trajectory through a *forward pass*.

By further introducing the slack variable y_k to transform inequalities into equalities as (2.5), we can write the SKKT conditions (4.14) in the slack variable form:

$$Q_{uu}u_k + Q_{ux}x_k + Q_{us}s_k + Q_u = 0 \tag{4.15a}$$

$$S_k y_k = 0 \tag{4.15b}$$

$$Q_{sx}x_k + Q_{su}u_k + Q_s + y_k = 0 \tag{4.15c}$$

$$s_k \geq 0,\ y_k \geq 0. \tag{4.15d}$$

Let $\mathbf{x}^0 := (x_0^0, ..., x_N^0)$ and $\mathbf{u}^0 := (u_0^0, ..., u_{N-1}^0)$ denote the initial guesses of the state and control, respectively, which are feasible for the nominal system equation. $\mathbf{s}^0 := (s_0^0, ..., s_{N-1}^0)$ and $\mathbf{y}^0 := (y_0^0, ..., y_{N-1}^0)$ denote the initial guesses of the Lagrange multipliers and slack variables, respectively, which are all non-negative. At the ith iteration, the trajectories of the last iteration, $(\mathbf{x}^{i-1}, \mathbf{u}^{i-1}, \mathbf{s}^{i-1}, \mathbf{y}^{i-1})$, are known. The *backward pass* and *forward pass* complete a Newton iteration, and the entire algorithm terminates after the obtained trajectory satisfies the optimality condition, which is judged in the *outer loop*.

4.4.1 Backward pass

Initialize: let

$$P_N = Q_N,\ q_N = r_N = 0$$

At stage k, represent the optimization variables by their search direction:

$$\begin{aligned} x_k^i &= x_k^{i-1} + \delta x_k \\ u_k^i &= u_k^{i-1} + \delta u_k \\ s_k^i &= s_k^{i-1} + \delta s_k \\ y_k^i &= y_k^{i-1} + \delta y_k. \end{aligned} \tag{4.16}$$

The new trajectory satisfies the perturbed SKKT conditions:

$$Q_{uu}u_k^i + Q_{ux}x_k^i + Q_{us}s_k^i + Q_u = 0 \tag{4.17a}$$

$$S_k^i y_k^i = \mu_k^i \tag{4.17b}$$

$$Q_{sx}x_k^i + Q_{su}u_k^i + Q_s + y_k^i = 0 \tag{4.17c}$$

$$s_k^i \geq 0,\ y_k^i \geq 0, \tag{4.17d}$$

where $\mu_k^i = (s_k^{i-1})^{\mathrm{T}} y_k^{i-1}/(l_x + l_u)$ is the current duality measure (central parameter) and the update of μ_k^i is $\mu_k^i = \sigma\mu_k^{i-1}$, where $\sigma \in [0, 1]$ is the reduction factor that we wish to achieve the duality measure at this step. When $\sigma > 0$, we solve for a *perturbed KKT point* at each Newton iteration. It is well known that a *standard Newton direction* with $\sigma = 0$ often does not make much progress toward a solution; therefore, it is better to choose a positive σ to reduce the central parameter iteratively Nocedal and Wright (1999).

Representing all the variables by their search direction, the only nonlinear equation is $(s_k^{i-1} + \delta s_k)(y_k^{i-1} + \delta y_k) = \mu_k^i$. The interior-point algorithm finds solutions by applying Newton's method to the three equations in (4.17), we can obtain the following linear equations:

$$\begin{bmatrix} Q_{uu} & Q_{us} & 0 \\ 0 & Y_k & S_k \\ Q_{su} & 0 & I \end{bmatrix} \begin{bmatrix} \delta u_k \\ \delta s_k \\ \delta y_k \end{bmatrix} = -\begin{bmatrix} Q_{ux} \\ 0 \\ Q_{sx} \end{bmatrix} \delta x_k - \begin{bmatrix} \xi_o^{i-1} \\ \xi_d^{i-1} - \mu_k^i \\ \xi_f^{i-1} \end{bmatrix} \tag{4.18}$$

where

$$\begin{bmatrix} \xi_o^{i-1} \\ \xi_d^{i-1} \\ \xi_f^{i-1} \end{bmatrix} = \begin{bmatrix} Q_{uu}u_k^{i-1} + Q_{ux}x_k^{i-1} + Q_{us}s_k^{i-1} + Q_u \\ S_k^{i-1} y_k^{i-1} \\ Q_{sx}x_k^{i-1} + Q_{su}u_k^{i-1} + Q_s + y_k^{i-1} \end{bmatrix} \tag{4.19}$$

are the primal and dual infeasiblities of the last iteration.

The parametric system (4.18) can be solved directly to obtain the solution

$$\begin{aligned}\delta u_k &= \eta_{1k}\delta x_k + \theta_{1k}\\ \delta s_k &= \eta_{2k}\delta x_k + \theta_{2k}\\ \delta y_k &= \eta_{3k}\delta x_k + \theta_{3k}.\end{aligned} \tag{4.20}$$

Writing (u, s) as a function of x,

$$u_k^i = u_k^{i-1} + \eta_{1k}(x_k^i - x_k^{i-1}) + \theta_{1k} = K_u x_k^i + v_u \tag{4.21a}$$

$$s_k^i = s_k^{i-1} + \eta_{2k}(x_k^i - x_k^{i-1}) + \theta_{2k} = K_s x_k^i + v_s, \tag{4.21b}$$

By substituting it into (4.12), we obtain the expressions for the coefficients of V_{k-1}, which are given by the following Riccati-type recursion:

$$\begin{aligned}P_{k-1} =& Q_{xx} + K_u^{\mathrm{T}} Q_{uu} K_u + Q_{xu} K_u + K_u^{\mathrm{T}} Q_{ux} + 2K_s^{\mathrm{T}}(Q_{sx} + Q_{su}K_u)\\ q_{k-1} =& Q_x + Q_u K_u + v_u^{\mathrm{T}}(Q_{ux} + Q_{uu}K_u) + (Q_{su}v_u + Q_s)K_s^{\mathrm{T}} + v_s^{\mathrm{T}}(Q_{sx} + Q_{su}K_u).\end{aligned} \tag{4.22}$$

The update of the constant term r_{k-1} is omitted (for simplicity) because it does not affect the solution.

The backward pass above is iteratively performed from $k = N - 1$ to $k = 1$ to finish the backward pass.

4.4.2 Forward pass

Starting from $k = 0$, $x_0^i = \bar{x}_0$ is already known by measurement, which means that $\delta x_0 = 0$.

At stage k, the search directions $(\delta u_k, \delta s_k, \delta y_k)$ are calculated using (4.20). The interior-point method generates a new trajectory $(\mathbf{x}^i, \mathbf{u}^i, \mathbf{s}^i, \mathbf{y}^i)$ in a wide neighborhood, as proposed in Kojima, Megiddo, and Mizuno (1993).

Definition 4.1. *Let $\gamma \in (0, 1)$, $\epsilon_o > 0$ and $\epsilon_f > 0$. A wide neighborhood $\mathcal{N}(\gamma, \epsilon_o, \epsilon_f)$ is a set of variables $(x_k^i, u_k^i, s_k^i, y_k^i)$ satisfying the following conditions:*

$$\begin{aligned}& s_k^i > 0, y_k^i > 0,\\ & (s_k^i)_j (y_k^i)_j \geq \gamma (s_k^i)^{\mathrm{T}} y_k^i / (l_x + l_u), j = 1, ..., (l_x + l_u)\\ & (s_k^i)^{\mathrm{T}} y_k^i \geq \gamma \|\xi_o^i\| \vee \|\xi_o^i\| \leq \epsilon_o,\\ & (s^i)^{\mathrm{T}} y^i \geq \gamma \|\xi_f^i\| \vee \|\xi_f^i\| \leq \epsilon_f,\end{aligned} \tag{4.23}$$

where $\|\cdot\|$ denotes the infinity norm.

Define the new iterate (u_k^i, s_k^i, y_k^i) as

$$\begin{bmatrix} u_k^i \\ s_k^i \\ y_k^i \end{bmatrix} = \begin{bmatrix} u_k^{i-1} \\ s_k^{i-1} \\ y_k^{i-1} \end{bmatrix} + \alpha_k^i \begin{bmatrix} \delta u_k \\ \delta s_k \\ \delta y_k \end{bmatrix}. \tag{4.24}$$

where the step size α_k^i is determined by the following line search rules:

STEP 1. Let $\bar{\alpha} \in (0, 1]$ be the maximum value for all $\alpha \in (0, \bar{\alpha}]$. The following conditions are satisfied:

$$\begin{aligned} &(u_k^{i-1}, s_k^{i-1}, y_k^{i-1}) + \alpha(\delta u_k, \delta s_k, \delta y_k) \in \mathcal{N}(\gamma, \epsilon_o, \epsilon_f) \\ &(s_k^{i-1} + \alpha \delta s_k)^{\mathrm{T}} (y_k^{i-1} + \alpha \delta y_k) \leq (1 - \alpha(1 - \sigma))(s_k^{i-1})^{\mathrm{T}} y_k^{i-1}, \end{aligned} \tag{4.25}$$

STEP 2. Choose the α_k^i to determine (u_k^i, s_k^i, y_k^i) such that

$$\begin{aligned} &(u_k^i, s_k^i, y_k^i) \in \mathcal{N}(\gamma, \epsilon_o, \epsilon_f) \\ &(s_k^i)^{\mathrm{T}} y_k^i \leq (1 - \bar{\alpha}(1 - \sigma))(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} \end{aligned} \tag{4.26}$$

Generally, it is common to let $\alpha_y = \alpha_s = \bar{\alpha}$. We show the existence of $\bar{\alpha}$ by proving the existence of a positive number α^* such that $\bar{\alpha} \geq \alpha^*$ holds as long as the iteration continues, which also leads to a global convergence result for our algorithm.

After calculating (u_k^i, s_k^i, y_k^i), we can compute x_{k+1}^i using the nominal system equation

$$x_{k+1}^i = A x_k^i + B u_k^i \tag{4.27}$$

The above update is repeated recursively from $k = 0$ to $N - 1$ to obtain a new trajectory $(\mathbf{x}^i, \mathbf{u}^i, \mathbf{s}^i, \mathbf{y}^i)$and complete the forward pass.

4.4.3 Outer loop

The *backward pass* and *forward pass* together complete the Newton steps. The algorithm terminates when the optimality, duality, or constraint feasibility measurements meet the stop criteria:

$$\|\boldsymbol{\xi}_o^i\| \leq \epsilon_o, \ \|\boldsymbol{\xi}_d^i\| \leq \epsilon_d, \text{ and } \|\boldsymbol{\xi}_f^i\| \leq \epsilon_f \tag{4.28}$$

or the duality measurement satisfies

$$\|\boldsymbol{\xi}_d^i\|_1 > \iota \tag{4.29}$$

for any large ι.

Remark 4.2. *If (4.29) holds, we can derive information on infeasibility such that there is no feasible solution in a certain wide region of the primal-dual space Kojima, Mizuno, and Yoshise (1993).*

We summarize the entire process in Algorithm 4.1.

Algorithm 4.1: RRIPM

```
Parameters:
  $\sigma \in [0,1]$ reduction parameter;
  $\alpha_f = 0.995$ fraction-to-the-boundary parameter;
  $(\epsilon_o, \epsilon_d, \epsilon_f, \iota)$ terminal criteria;
Initialization:
  iteration counter $i = 0$ ;
  known initial state for all $i$, $x_0^i = x_0$ ;
  initial guess $(\mathbf{x}^0, \mathbf{u}^0, \mathbf{s}^0, \mathbf{y}^0)$ ;
  central parameter $\boldsymbol{\mu}^0$ ;
  optimality measurements $(\|\boldsymbol{\xi}_o^0\|, \|\boldsymbol{\xi}_d^0\|, \|\boldsymbol{\xi}_f^0\|)$ ;
while $\|\boldsymbol{\xi}_o^i\| > \epsilon_o \vee \iota > \|\boldsymbol{\xi}_o^i\| > \epsilon_d \vee \|\boldsymbol{\xi}_f^i\| > \epsilon_f$ do
    $P_N \leftarrow Q_N, q_N\ r_N \leftarrow 0.$ ;
    for $k \leftarrow N-1$ to $0$ do
        // Backward Pass
        Solve linear equation (4.18);
        $\delta u_k \leftarrow \eta_{1k}\delta x_k + \theta_{1k}$;
        $\delta s_k \leftarrow \eta_{2k}\delta x_k + \theta_{2k}$;
        $\delta y_k \leftarrow \eta_{3k}\delta x_k + \theta_{3k}$;
        $(P_k, q_k, r_k) \leftarrow (P_{k+1}, q_{k+1}, r_{k+1})$;
    end
    for $k \leftarrow 0$ to $N-1$ do
        // Forward Pass
        Find $\alpha_k^i$ by line search rules (4.25)-(4.26);
        Calculate new iterate $(x_k^i, s_k^i, y_k^i)$;
        $x_{k+1}^i \leftarrow Ax_k^i + Bu_k^i$;
    end
    Reduce the barrier parameter $\boldsymbol{\mu}^i = \sigma\boldsymbol{\mu}^i$;
    Update the iteration counter $i = i + 1$;
    Compute optimality measurements $(\xi_o^i, \xi_d^i, \xi_f^i)$.
end
```

4.5 Convergence Analysis of RRIPM

The global convergence property and the local convergence rate of the proposed algorithm are discussed in this subsection. Let $z = (x, u, s, y)$ and $F_k(z_k) = 0$ denote the SKKT conditions (4.15), and let z^* denote the stationary point of $F_k(z_k) = 0$. The following assumptions are made in this subsection.

Assumption 4.3. *The optimal solution $\mathbf{z}^*$ exists and is unique in the neighborhood $\mathcal{N}(\gamma, \epsilon_o, \epsilon_f)$ for a given initial state x_0, and it holds a strict complementarity.*

Assumption 4.4. *Q_{uu} is positive definite.*

Before discussing the convergence result, we provide the following lemma to guarantee the nonsingularity of the coefficient matrix on the left-hand side of (4.18).

Lemma 4.1. *Under Assumption 4.4, the SKKT matrix*

$$\begin{bmatrix} Q_{uu} & Q_{us} & 0 \\ 0 & Y_k & S_k \\ Q_{su} & 0 & I \end{bmatrix} \tag{4.30}$$

is nonsingular for all $k = 0, ..., N-1$.

Proof. If $\mathbf{z}^*$ is a solution point for which strict complementarity holds, then for every iteration i at stage k, under the line search rules (4.25)–(4.26), either s_k^i or y_k^i remains bounded away from zero as the iterates approach z_k^*, ensuring that the second block row $\begin{bmatrix} 0 & Y_k & S_k \end{bmatrix}$ has a full row rank. Because Q_{uu} is nonsingular, the first and third block rows also have full row ranks. Together with the linear independence of these three block rows, we can conclude that (4.30) is nonsingular. □

4.5.1 Global convergence

The following theorem provides a global convergence result for Algorithm 4.1.

Theorem 4.2. *For any $(\mathbf{x}^0, \mathbf{u}^0)$, there always exists $(\mathbf{s}^0, \mathbf{y}^0)$ such that $(x_k^0, u_k^0, s_k^0, y_k^0) \in \mathcal{N}(\gamma, \epsilon_o, \epsilon_f)$ holds for every k. Moreover, for any $(x_k^0, u_k^0, s_k^0, y_k^0) \in \mathcal{N}(\gamma, \epsilon_o, \epsilon_f)$, Algorithm 4.1 is terminated after a finite number of iterations.*

Proof. The first half is trivial; for any $(\mathbf{x}^0, \mathbf{u}^0)$, $(\mathbf{s}^0, \mathbf{y}^0)$ satisfies

$$(s_k^0)_j (y_k^0)_j \geq \max\{\frac{\gamma (s_k^0)^{\mathrm{T}} y_k^0}{l_x + l_u}, \frac{\gamma \|\xi_o^0\|}{l_x + l_u}, \frac{\gamma \|\xi_f^0\|}{l_x + l_u}\} \tag{4.31}$$

for all $k = 0, ..., N-1$, $j = 1, ..., (l_x + l_u)$.

The second half arises from a contradiction. Suppose Algorithm 4.1 does not terminate. At iteration i, for all k, we have

$$(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} \geq \epsilon^*, \text{ and } \|\xi_d^{i-1}\|_1 \leq \iota, \tag{4.32}$$

where $\epsilon^* = \min\{\gamma\epsilon_o, \epsilon_d, \gamma\epsilon_f\}$; otherwise, $\mathbf{z}^{i-1}$ satisfies either the stopping criteria (4.28) or (4.29). Therefore, the entire sequence $(x_k^{i-1}, u_k^{i-1}, s_k^{i-1}, y_k^{i-1})$ lies within the compact set.

$$\bar{\mathcal{N}} = \{(x, u, s, y) | s^{\mathrm{T}} y \geq \epsilon^*, \text{ and } \|\xi_d\|_1 \leq \iota\} \tag{4.33}$$

Together with the nonsingularity of the SKKT matrix from Lemma 4.1, the Newton direction $(\delta u_k, \delta s_k, \delta y_k)$ generated by (4.18) is bounded for all $(x_k^{i-1}, u_k^{i-1}, s_k^{i-1}, y_k^{i-1})$ over the compact set $\bar{N}$.

We define the functions g_o, g_d^j $(j = 1, ..., (l_x + l_u))$, g_f, and h from the conditions in (4.25) as follows:

$$\begin{aligned} g_o(\alpha) &= (s_k^i)^{\mathrm{T}} y_k^i - \gamma \|\xi_o^i\| && (4.34a)\\ g_d^j(\alpha) &= (s_k^i)^j (y_k^i)^j - \gamma (s_k^i)^{\mathrm{T}} y_k^i / (l_x + l_u) && (4.34b)\\ g_f(\alpha) &= (s_k^i)^{\mathrm{T}} y_k^i - \gamma \|\xi_f^i\| && (4.34c)\\ h(\alpha) &= (1-\alpha)(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} - (s_k^i)^{\mathrm{T}} y_k^i. && (4.34d) \end{aligned}$$

From (4.19) and (4.25), we have

$$\begin{aligned} \xi_o^i =& Q_{uu} u_k^i + Q_{ux} x_k^i + Q_{us} s_k^i + Q_u \\ =& Q_{uu}(u_k^{i-1} + \alpha \delta u_k) + Q_{ux}(x_k^{i-1} + \alpha \delta x_k) + Q_{us}(s_k^{i-1} + \alpha \delta s_k) + Q_u. \end{aligned} \tag{4.35}$$

By (4.18), we know that

$$Q_{uu}(u_k^{i-1} + \delta u_k) + Q_{ux}(x_k^{i-1} + \delta x_k) + Q_{us}(s_k^{i-1} + \delta s_k) + Q_u = 0. \tag{4.36}$$

Simultaneous with (4.35) and (4.36), we have

$$\xi_o^i = (1-\alpha)\xi_o^{i-1}. \tag{4.37}$$

Similarly, we have

$$\xi_f^i = (1-\alpha)\xi_f^{i-1}. \tag{4.38}$$

Hence, α are chosen to satisfy the following conditions

$$\begin{aligned} & g_d^j(\alpha) \geq 0 \\ & g_o(\alpha) \geq 0 \vee \|(1-\alpha)\xi_o^{i-1}\| \leq \epsilon_o \\ & g_f(\alpha) \geq 0 \vee \|(1-\alpha)\xi_f^{i-1}\| \leq \epsilon_f \\ & h(\alpha) \geq 0, \end{aligned} \tag{4.39}$$

such that the new iteration satisfies equation (4.25). Because the above four equations are similar, let us consider the second equation about $g_o(\alpha)$ as an example:

$$\begin{aligned} g_o(\alpha) =& (s_k^{i-1} + \alpha \delta s_k)^{\mathrm{T}} (y_k^{i-1} + \alpha \delta y_k) - \gamma \|(1-\alpha)\xi_o^{i-1}\| \\ =& (s_k^{i-1})^{\mathrm{T}} y_k^{i-1} + (\alpha \delta s_k^{\mathrm{T}} y_k^{i-1} + \alpha \delta y_k^{\mathrm{T}} s_k^{i-1}) + \alpha^2 \delta s_k^{\mathrm{T}} \delta y_k - \gamma \|(1-\alpha)\xi_o^{i-1}\| \\ =& (s_k^{i-1})^{\mathrm{T}} y_k^{i-1} - (\alpha (s_k^{i-1})^{\mathrm{T}} y_k^{i-1} - \alpha\sigma (s_k^{i-1})^{\mathrm{T}} y_k^{i-1}) + \alpha^2 \delta s_k^{\mathrm{T}} \delta y_k - \gamma \|(1-\alpha)\xi_o^{i-1}\| \\ =& (1-\alpha)((s_k^{i-1})^{\mathrm{T}} y_k^{i-1} - \gamma \|\xi_o^{i-1}\|) + \alpha\sigma (s_k^{i-1})^{\mathrm{T}} y_k^{i-1} + \alpha^2 \delta s_k^{\mathrm{T}} \delta y_k \end{aligned} \tag{4.40}$$

Because $(x_k^{i-1}, u_k^{i-1}, s_k^{i-1}, y_k^{i-1}) \in \mathcal{N}(\gamma, \epsilon_o, \epsilon_f)$, we have

$$(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} \geq \gamma \|\xi_o^{i-1}\| \vee \|\xi_o^{i-1}\| \leq \epsilon_o. \tag{4.41}$$

For $\|\xi_o^{i-1}\| \leq \epsilon_o$, for any $\alpha \in (0,1]$, we have

$$\|\xi_o^i\| = \|(1-\alpha)\xi_o^{i-1}\| < \epsilon_o. \tag{4.42}$$

When $(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} \geq \gamma \|\xi_o^{i-1}\|$, the first term on the right-hand side of (4.40) is always positive for any $\alpha \in (0,1]$. From the boundness of $(\delta u_k, \delta s_k, \delta y_k)$, we find a positive constant ϖ such that

$$|\delta s_k^{\mathrm{T}} \delta y_k| \leq \varpi, \tag{4.43}$$

Therefore, we have the inequality

$$g_o(\alpha) \geq \alpha\sigma\epsilon^* - \alpha^2\varpi, \tag{4.44}$$

because $(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} \geq \epsilon^*$.

Similarly, we obtain the conditions for $g_d^j(\alpha)$, $g_f(\alpha)$, and $h(\alpha)$:

$$\begin{aligned} g_d^j(\alpha) &\geq \alpha\sigma(1-\gamma)\epsilon^*/(l_x + l_u) - \alpha^2\varpi \\ g_f(\alpha) &\geq \alpha\sigma\epsilon^* - \alpha^2\varpi \\ h(\alpha) &\geq \alpha(\bar{\sigma} - \sigma)\epsilon^* - \alpha^2\varpi. \end{aligned} \tag{4.45}$$

We can find a positive α^* by letting

$$\alpha^* = \min\{1, \frac{\sigma\epsilon^*}{\varpi}, \frac{\sigma(1-\gamma)\epsilon^*}{(l_x + l_u)\varpi}, \frac{(\bar{\sigma} - \sigma)\epsilon^*}{\varpi}\} \tag{4.46}$$

such that for any $\alpha \in (0, \alpha^*]$, (4.39) holds true.

From the definition of $\bar{\alpha}$, $\bar{\alpha} \geq \alpha^*$. We can find the step size α_k^i to determine a new trajectory that satisfies (4.26), i.e.,

$$\begin{aligned} (s_k^i)^{\mathrm{T}} y_k^i &\leq (1 - \bar{\alpha}(1-\sigma))(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} \\ &\leq (1 - \alpha^*(1-\sigma))(s_k^{i-1})^{\mathrm{T}} y_k^{i-1} \\ &\leq (1 - \alpha^*(1-\sigma))^i (s_k^0)^{\mathrm{T}} y_k^0 \end{aligned} \tag{4.47}$$

Evidently, the last equation converges to zero as iteration i tends to ∞, which contradicts (4.32). □

Theorem 4.2 reveals the finite termination of Algorithm 4.1. Together with Assumption 4.3, it can be concluded that our algorithm converges to a sequence that satisfies the SKKT conditions; in other words, the algorithm globally converges to $\mathbf{z}^*$.

4.5.2 Local convergence rate

Next, we consider the local convergence rate in Algorithm 4.1. Let $\boldsymbol{\alpha}^i := (\alpha_0^i, ..., \alpha_{N-1}^i)$.

Theorem 4.3. *If $\boldsymbol{\alpha}^i \to \mathbf{1}$ and $\boldsymbol{\mu}^i \to \mathbf{0}$, the sequence $\{\mathbf{z}^i\}$ generated by Algorithm 4.1 converges to the optimal solution $\mathbf{z}^*$ Q-superlinearly.*

Proof. Let $\zeta_k^i = (u_k^i, s_k^i, y_k^i)$. We define the coefficient matrix in $F_k(z_k) = 0$ as

$$\nabla_x F_k = \begin{bmatrix} Q_{ux} \\ 0 \\ Q_{sx} \end{bmatrix} \quad \nabla_\zeta F_k = \begin{bmatrix} Q_{uu} & Q_{us} & 0 \\ 0 & Y_k & S_k \\ Q_{su} & 0 & I \end{bmatrix}.$$

At each stage, consider the Taylor approximation

$$0 = F_k(z_k^*) = F_k(z_k^i) + \nabla_x F_k(z_k^i)(x_k^* - x_k^i) + \nabla_\zeta F_k(z_k^i)(\zeta_k^* - \zeta_k^i) + o(z_k^* - z_k^i)^2. \quad (4.48)$$

From Lemma 4.1, it is known that $\nabla_\zeta F_k(z_k^i)$ is nonsingular. By multiplying $\Psi = \nabla_\zeta F_k^{-1}(z_k^i)$ to both sides of (4.48), we have

$$\Psi F_k(z_k^i) + \Psi \nabla_x F_k(z_k^i)(x_k^* - x_k^i) + \zeta_k^* - \zeta_k^i + o(z_k^* - z_k^i)^2 = 0. \quad (4.49)$$

From the forward update (4.24), we know the update of ζ.

$$\zeta_k^{i+1} = \zeta_k^i + \alpha_k^i \delta\zeta_k, \quad (4.50)$$

where $\delta\zeta_k$ is the solution of the linear equation (4.18):

$$\delta\zeta_k = -\Psi \nabla_x F_k(z_k^i)(x_k^{i+1} - x_k^i) - \Psi F_k(z_k^i) + \Psi \mu_k^i \hat{p}, \quad (4.51)$$

where $\hat{p}$ denotes a fixed vector. Substituting (4.50) into (4.51)yields

$$\Psi F_k(z_k^i) = -\frac{\zeta_k^{i+1} - \zeta_k^i}{\alpha_k^i} - \Psi \nabla_x F_k(z_k^i)(x_k^{i+1} - x_k^i) + \Psi \mu_k^i \hat{p}. \quad (4.52)$$

Then, simultaneous (4.49) and (4.52) to eliminate the term $\Psi F(z_k^i)$:

$$\zeta_k^{i+1} - \zeta_k^* = (1 - \alpha_k^i)(\zeta_k^i - \zeta_k^*) + \alpha_k^i \Psi \nabla_x F_k(z_k^i)(x_k^{i+1} - x_k^*) + \alpha_k^i \Psi \mu_k^i \hat{p} + o(z_k^* - z_k^i)^2. \quad (4.53)$$

From the system equation (4.1), because $x_0^{i+1} = x_0^* = x_0$, we have

$$\begin{aligned} x_1^{i+1} - x_1^* &= A(x_0^{i+1} - x_0^*) + B(u_0^{i+1} - u_0^*) \\ &= B(u_0^{i+1} - u_0^*). \end{aligned} \quad (4.54)$$

By continuing by induction until stage k and representing the right-hand side by ζ, we obtain

$$x_k^{i+1} - x_k^* = \sum_{j=0}^{k-1} M_j(\zeta_j^{i+1} - \zeta_j^*), \tag{4.55}$$

where M_j denotes an appropriate constant matrix. Substituting (4.55) into (4.53), when $k = 0$, we have

$$\zeta_0^{i+1} - \zeta_0^* = (1 - \alpha_0^i)(\zeta_0^i - \zeta_0^*) + \alpha_0^i \Psi \mu_0^i \hat{p} + o(z_0^* - z_0^i)^2. \tag{4.56}$$

Similarly, at $k = 1$,

$$\begin{aligned} \zeta_1^{i+1} - \zeta_1^* =&(1 - \alpha_1^i)(\zeta_1^i - \zeta_1^*) + \alpha_1^i \Psi \mu_1^i \hat{p} + o(z_1^* - z_1^i)^2 \\ &+ \alpha_1^i \Psi \nabla_x F_k(z_k^i) M_0(\zeta_0^{i+1} - \zeta_0^*) \\ =&(1 - \alpha_1^i)(\zeta_1^i - \zeta_1^*) + \alpha_1^i \Psi \mu_1^i \hat{p} + o(z_1^* - z_1^i)^2 \\ &+ \alpha_1^i \Psi \nabla_x F_k(z_k^i) M_0((1 - \alpha_0^i)(\zeta_0^i - \zeta_0^*) \\ &+ \alpha_0^i \Psi \mu_0^i \hat{p} + o(z_0^* - z_0^i)^2). \end{aligned} \tag{4.57}$$

Let $\boldsymbol{\zeta} = [\zeta_0, \zeta_1, ..., \zeta_{N-1}]$. By continuing this induction until $k = N - 1$, we obtain

$$\boldsymbol{\zeta}^{i+1} - \boldsymbol{\zeta}^* = \Upsilon_\zeta(\boldsymbol{\zeta}^i - \boldsymbol{\zeta}^*) + \Upsilon_\mu \boldsymbol{\mu}^i + o(\mathbf{z}^* - \mathbf{z}^i)^2, \tag{4.58}$$

where Υ_ζ and Υ_μ are the appropriate coefficient matrices, and Υ_ζ is composed of the product of $(1 - \alpha_k^i)$ and other bounded matrices. By combining (4.55) and (4.58), we have

$$\mathbf{z}^{i+1} - \mathbf{z}^* = \Upsilon'_\zeta(\mathbf{z}^i - \mathbf{z}^*) + \Upsilon'_\mu \boldsymbol{\mu}^i + o(\mathbf{z}^* - \mathbf{z}^i)^2, \tag{4.59}$$

where $\Upsilon'_\zeta \to 0$ holds true when $\boldsymbol{\alpha}^i \to \mathbf{1}$. Therefore, we can conclude that if $\boldsymbol{\alpha}^i \to \mathbf{1}$ and $\boldsymbol{\mu}^i \to \mathbf{0}$, then sequence $\{\mathbf{z}^i\}$ converges to $\mathbf{z}^*$ Q-superlinearly. $\square$

4.6 Practical Implement Issues for a Feasible Algorithm

In the context of analytic approximated chance constraints, the problem of guaranteeing recursive feasibility of the MPC optimization problem is very important in practical implementation. This issue relies on the fact that the disturbance is unbounded, the initial state x_0 may take (even with small probability) unboundedly large (or small) values and therefore there may not exist a feasible control action that can force the predicted trajectory into admissible region Π_k (4.9). In this section,

Algorithm 4.2: Feasible RRIPM

```
Parameters:
  $\rho$ penalty coefficient;
Input: Initial state $(\mathbf{x}^0, \mathbf{u}^0, \mathbf{s}^0, \mathbf{y}^0)$ ;
Output: Optimal solution $\mathbf{u}^*$ ;
if RRIPM (4.1) converges then
    $\mathbf{u}^* \leftarrow RRIPM(\mathbf{x}^0, \mathbf{u}^0, \mathbf{s}^0, \mathbf{y}^0)$ ;
else
    Run soft constraint RRIPM (Algorithm A.1);
    $\mathbf{u}^* \leftarrow soft\ constraint\ RRIPM(\mathbf{x}^0, \mathbf{u}^0, \mathbf{s}^0, \mathbf{y}^0)$ ;
end
```

we reformulate the problem constraints in a soft fashion. Therefore, in the case of infeasibility, an alternative softened optimization problem is solved.

By introducing the soft variable (Kerrigan and Maciejowski (2000)), we can rewrite the deterministic Bellman equation (4.3) as follows

Problem 4.4 (Deterministic Bellman equation with soft variable)**.**

$$\begin{aligned} V_k(x_k) = \min_{u_k(x_k)} \ & [\frac{1}{2}x_k^{\mathrm{T}} Q x_k + \frac{1}{2}u_k^{\mathrm{T}} R u_k + \rho^{\mathrm{T}} e_k + V_{k+1}(x_{k+1})] \\ \text{s.t. } & x_{k+1} = Ax_k + Bu_k \\ & C_x x_k + C_u u_k + C_c \leq e_k \\ & e_k \geq 0, \end{aligned}$$

where $e_k \in \mathbb{R}^{l_x + l_u}$ is the soft variable and ρ is the penalty coefficient.

Based on the KKT conditions with soft variable in (2.5), we can also give the "softened" SKKT conditions of based on Problem (4.4):

$$Q_{uu}u_k + Q_{ux}x_k + Q_{us}s_k + Q_u = 0 \tag{4.60a}$$

$$S_k y_k = 0 \tag{4.60b}$$

$$Q_{sx}x_k + Q_{su}u_k + Q_s + y_k - e_k = 0 \tag{4.60c}$$

$$E_k(\rho - s_k) = 0 \tag{4.60d}$$

$$s_k \geq 0, \ y_k \geq 0, \ e_k \geq 0, \tag{4.60e}$$

where $E_k = \operatorname{diag}[e_k]$. Next, the soft constraint RRIPM based on this "softened" SKKT condition is also straightforward. The slightly different point is that there are two non-smooth equations (4.60b) and (4.60d) in this equation, and it needs to be smoothed with two central parameters, the details of soft constraint RRIPM can be found in Appendix (A).

We summarize the feasible RRIPM algorithm in Algorithm 4.2.

4.7 Case Study

4.7.1 Numerical example settings

We tested our method with a linear system: the control problem of room temperature Gwerder and Tödtli (2005). The basic control target was to maintain the room temperature above a certain level in the presence of external disturbances. The system contained three states: let x_1 be the room temperature, let x_2 be the temperature of the wall connected to another room, and let x_3 be the temperature of the wall connected to the outside. The system was subjected to three disturbances in which w_1 denotes the outside temperature, w_2 denotes the solar radiation, and w_3 denotes the internal heat gains (e.g., people and electronic devices), and the entire disturbances w subjected to $\mathcal{N}(0, I)$. The only control input, u, was the heating.

The control objective was to maintain the room temperature above 21°C with minimum energy; thus, the weighting matrix was $Q = 0$ and $R = 1$. The state constraint was treated as a chance constraint:

$$\Pr(x_1 \geq 21) \geq 1 - p_{vio}.$$

The control input also had limitations imposed by the actuator, $0 \leq u \leq 45$ [W/m^2]. The parameters of the system were obtained from Oldewurtel et al. (2013) as follows:

$$A := \begin{bmatrix} 0.8511 & 0.0541 & 0.0707 \\ 0.1293 & 0.8635 & 0.0055 \\ 0.0989 & 0.0032 & 0.7541 \end{bmatrix}, B := \begin{bmatrix} 0.0035 \\ 0.0003 \\ 0.0002 \end{bmatrix}.$$

Because explicit MPC is intractable in the present problem settings when the prediction horizon is larger than two, we compared the following three control strategies in terms of control performance, constraint violation rate, and computation time:

1. Open-loop policy (Mayne et al. (2000)): Only the information of the initial measurement x_0 can be used.

2. Pre-parameterization (PP) (Lofberg (2003)): Parameterizes the control policy as $u_k = K_k x_k + v_k$ first and searches directly over the decision variables (K_k, v_k), which is a heuristic formulation of the *feedback* policy.

3. Proposed RRIPM.

We carried out the simulation on a laptop computer with a 2.60 GHz Intel Core i7-6700HQ in MATLAB 2020a. The first and second problems were solved using the interior-point method with direct shooting (El-Bakry et al. (1996)) and coded in the MATLAB environment.

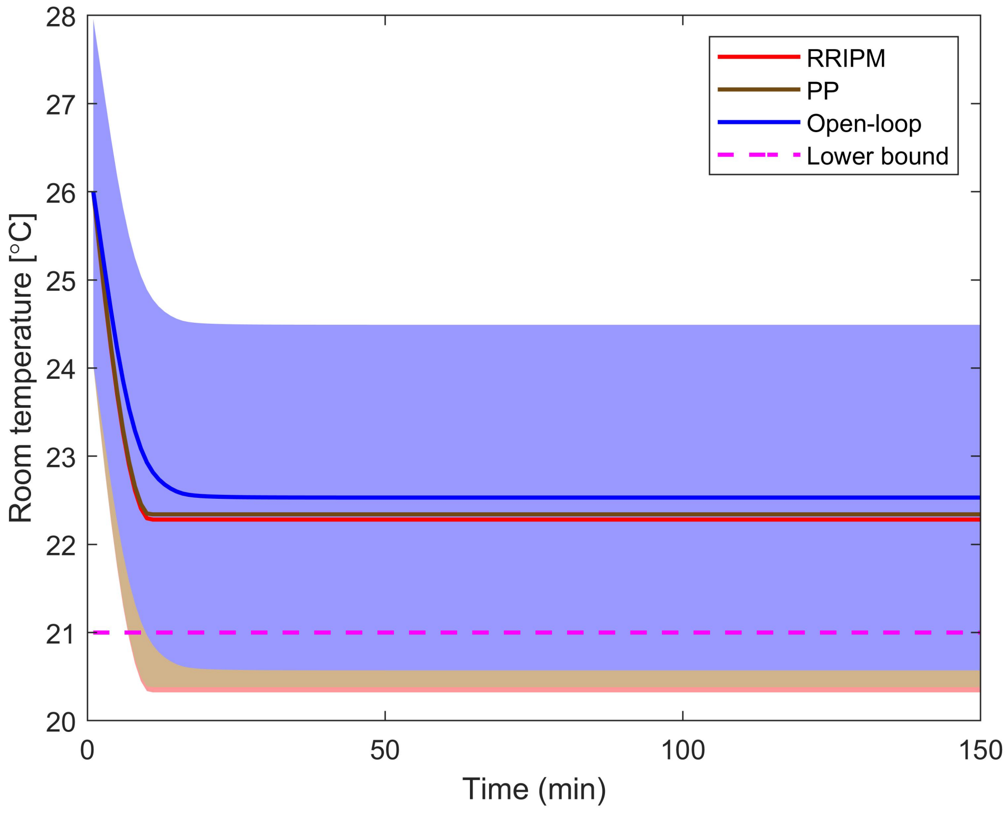

Figure 4.1: Room temperature profile and corresponding 95% confidence region.

4.7.2 Simulation analysis

Figure 4.1 shows the time evolution of the mean of x_1 and corresponding 95% confidence region for all three control methods with initial state $[26, 26, 21]^{\mathrm{T}}$, prediction horizon $N = 6$ and allowed constraint violation rate $p_{vio} = 10\%$. The results reveal that the most conservative control performance can be achieved with an open-loop policy; the room temperature was almost always higher than the room temperature of the other two control policies while our proposed method resulted in the least conservative control performance. The means and confidence regions of PP and RRIPM

Table 4.1: Comparison of the objective function values for different states under the same settings ($N = 6$).

	Open-loop	PP	RRIPM
state 1: $[28, 28, 21]^{\mathrm{T}}$	313	187	62
state 2: $[22, 18, 15]^{\mathrm{T}}$	2855	2216	1859
state 3: $[25, 25, 15]^{\mathrm{T}}$	1205	828	611

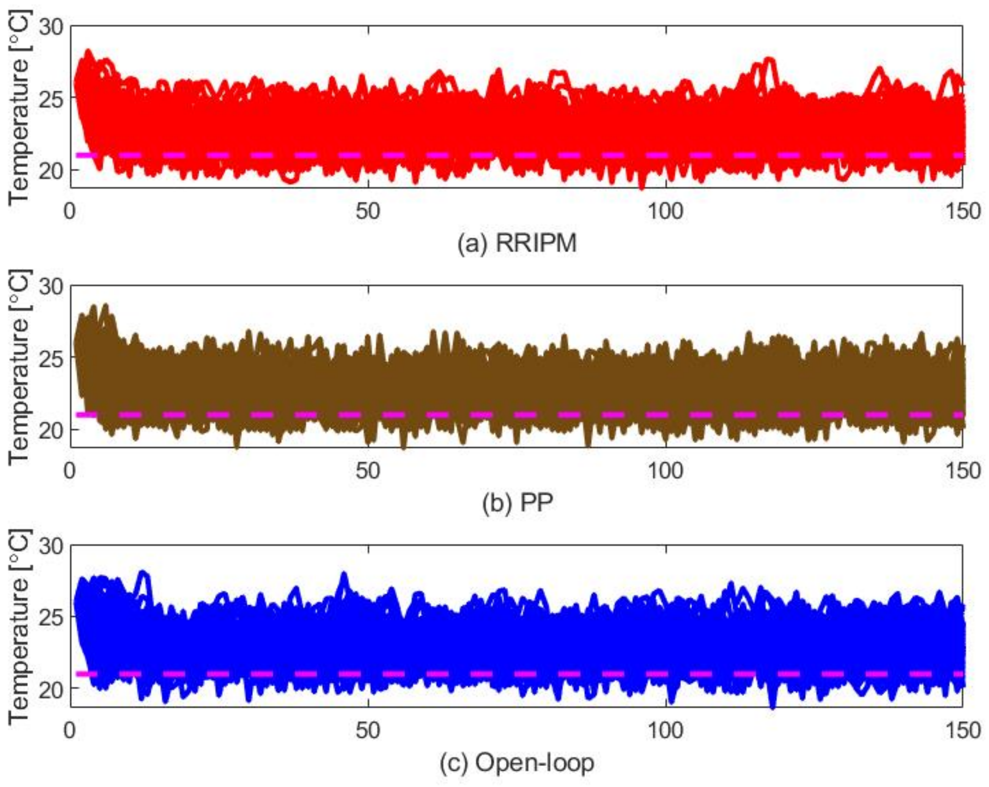

Figure 4.2: 100 Monte Carlo simulations. ($\alpha_x = 10\%$)

almost coincided; however, PP was slightly more conservative than the proposed RRIPM, which may be due to the fact that pre-parameterization is a heuristic formulation of a feedback policy. A more intuitive comparison of control performance can be seen in Table 4.1. We considered optimal solutions for several initial states. The results reveal that the open-loop policy consumes the most energy for an identical optimization problem (i.e., most conservative), while the proposed RRIPM always achieves the smallest objective function value.

The constraint violation rates were compared based on 100 Monte Carlo simulations. Figure 4.2 shows the time evolution of x_1 for $p_{vio} = 10\%$. As can be seen from

Table 4.2: Comparison of the constraint violation rate based on 100 Monte Carlo simulations. (%)

	Open-loop	PP	RRIPM
$p_{vio} = 1$	0.61	0.93	0.94
$p_{vio} = 5$	3.82	4.20	4.72
$p_{vio} = 10$	5.67	9.52	10.00

Table 4.3: Average computation time per update (ms).

	Open-loop	PP	RRIPM
$N = 5$	0.56	9.19	1.41
$N = 50$	2.36	85.10	2.77
$N = 100$	12.37	175.35	3.49

the figure, all three control methods exhibit certain degrees of constraint violation. Table 4.2 provides a more intuitive comparison, showing the constraint violation rate for $p_{vio} = 1\%$, $p_{vio} = 5\%$, and $p_{vio} = 10\%$. The results reveal that, for different values of α_x, our proposed RRIPM always remained the closest to the set value, the PP and RRIPM were close but slightly smaller, and the open-loop policy was significantly lower than the set value.

The computation times are listed in Table 4.3. The data in the table reveals that, when the prediction horizon is short, the computation times of the open-loop policy and RRIPM are similar and smaller than those of the PP because they have fewer decision variables. However, when the prediction horizon increases, RRIPM has obvious advantages because the computational complexity of Riccati recursion increases linearly with the prediction horizon.

Figure 4.3 shows the shape of the obtained first step of the optimal feedback control law $u_0(x)$. In principle, the dynamic programming solution of a linear system with linear constraints should be a piecewise linear function w.r.t. state variables. We obtained $u_0(x)$ by pointwise computation of RRIPM within a certain area of x, i.e., $x_1 \in [18, 30], x_2 \in [18, 30]$ and fixed $x_3 = 10$. The figure reveals that the optimal solution u_0 exhibits the shape of a piecewise linear function in the (x_1, x_2) space. In other words, our algorithm generates same control inputs as a piecewise affine feedback law of explicit MPC while getting rid of active sets and does not suffer from the curse of dimensionality. Moreover, our proposed method also achieves low computational complexity compared to existing methods.

4.8 Summary

In this chapter, a full-state observable linear SMPC with chance constraints is considered. We reformulate the SOCP with chance constraints as a deterministic Bellman equation under the concept of a feedback policy. A RRIPM is proposed to solve the stage-wise KKT conditions. Our proposed method avoids the complicated discussion

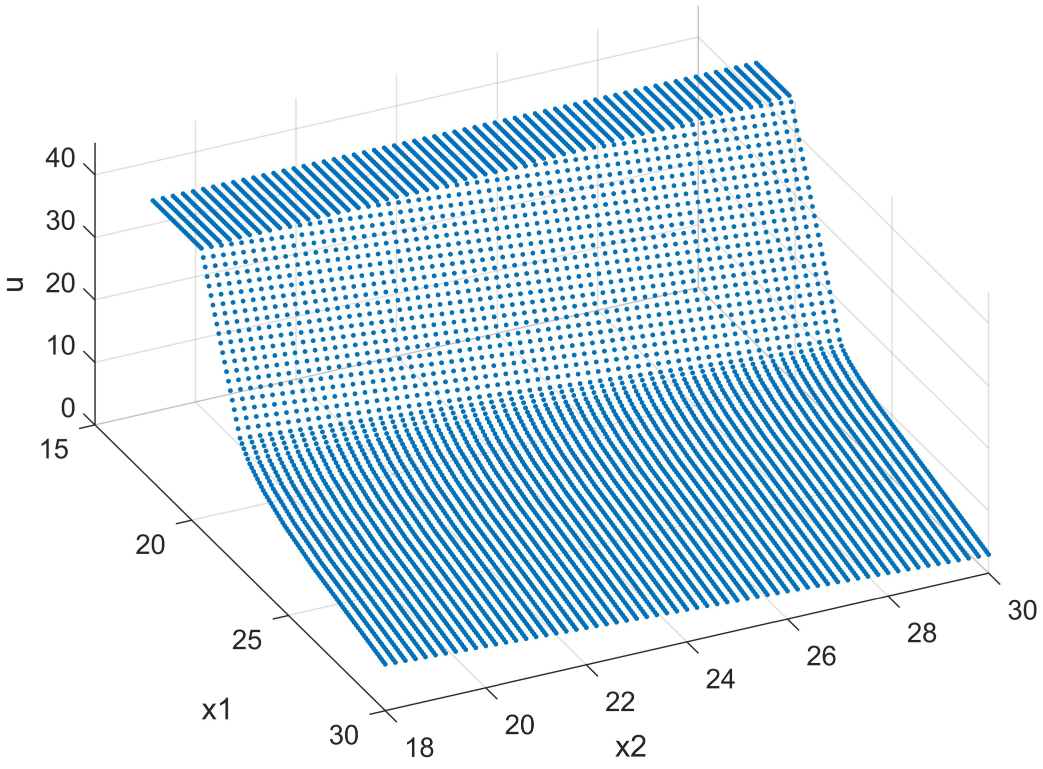

Figure 4.3: Optimal solution $u_0(x)$ obtained by pointwise computation of RRIPM.

of the active set and determines that the optimal control policy depends on the current state. We proved the global convergence and local Q-superlinear convergence rate of our algorithm for the optimal solution. The simulation results demonstrate that our proposed algorithm can achieve ideal less conservative control performance, that is, to make the constraint violation rate as close as possible to the set value. Moreover, our proposed method achieves low computational complexity compared to existing methods.

Chapter 5

Approximate Dynamic Programming for Output-feedback Nonlinear SMPC

5.1 Introduction

SMPC of nonlinear system has received less attention compared with the linear case. The key challenge of nonlinear SMPC is the efficient uncertainty propagation method through nonlinear dynamics. In Weissel et al. (2009), the Gaussian-mixture approximation (Maz'ya and Schmidt (1996)) was used to describe the transition probability distributions of states. Generalized polynomial chaos expansion (gPC) (Xiu and Karniadakis (2002)) is also widely used in nonlinear SMPC. In Mayne (2014), a sample-based method was adopted to adapt the coefficients of gPC based on the history data. Streif et al. (2014) used Galerkin projection (Ghanem and Spanos (2003)) for adapting the coefficients of gPC, and efficiently constructing the probability distributions of state through Monte Carlo methods to evaluate the chance constraints. The disadvantage of the above methods is that they require a lot of calculation or sampling time to simulate the propagation of uncertainty through nonlinear system.

On the other hand, if the output-feedback is considered in the SMPC, we need to incorporate state estimation into the controller like LQG does (Lindquist (1973)). However, the separation principle is not optimal for nonlinear stochastic system. In Yan and Bitmead (2005), the output-feedback SMPC for linear system was considered, which uses a Kalman filter to estimate the state. In Sehr and Bitmead (2017), a particle filter was used to estimate the system. Mesbah (2018) gave the general framework of dealing with output-feedback cases, where a Bayesian filter is used to estimate the information state. However, Bayesian filter is generally intractable.

In Chapter 4, we introduced the numerical algorithm for solving constrained dynamic programming. Approximate dynamic programming (ADP) is a natural extension of dynamic programming in nonlinear cases. ADP-based methods, such as differential dynamic programming (DDP) (Mayne (1966)) and iterative LQR (iLQR) (W. Li and Todorov (2004)) have received considerable attention in the trajectory optimization and machine learning fields (Lewis and Liu (2013)). To avoid solving the exact solution of the original Bellman equation, ADP decouples the nested optimization problem and decomposes a large problem across a control sequence into a recursive series of small problems and then recursively solves each stage of the problem. This method has the advantage of computational efficiency because the size of each subproblem is time-independent, and the computational complexity grows only linearly with the prediction horizon, which is very attractive for real-time computing.

However, constraint handling in ADP methods remains problematic even in deterministic cases. Algorithms in Maz'ya and Schmidt (1996) and W. Li and Todorov (2004) only consider unconstrained problems. In Rodriguez and Sideris (2010), an active-set-based method was proposed to solve a constrained LQR problem. In Xie, Liu, and Hauser (2017), an active-set method was applied to DDP to address general nonlinear constraints. The disadvantages of the active-set method include inefficiencies in handling highly nonlinear optimization problems and computational degradation as the problem size increases. Moreover, existing methods cannot deal with chance constraints directly. Chapter 4 gives RRIPM algorithm for constrained dynamic programming. However, it is only applicable to linear, full-state feedback systems.

In this chapter, we developed a stochastic output-feedback MPC scheme for chance-constrained nonlinear systems. A novel way of dealing with chance constraints using an appropriate information state propagation method and a constrained ADP algorithm is proposed for efficiently solving stochastic optimal control problems. We extend the optimal value function concept to the output-feedback case, such that the controller inherently has closed-loop performance. The proposed algorithm was proven to exhibit a Q-superlinear local convergence rate. The numerical example reveals that the proposed method achieves good control performance and a reasonable level of constraint violation and is computationally efficient owing to the dynamic programming structure.

This chapter is organized as follows. Section 5.2 discusses the problem settings in this chapter. Section 5.3 discusses the features output-feedback SMPC and its difficulties. Section 5.4 presents the constrained SADP algorithm. Section 4.5 considers the local convergence rate of the proposed constrained SADP algorithm. Section 5.6 presents simulation results. Finally, Section 5.7 summarizes this chapter.

5.2 Problem Statement

In this chapter, we consider a nonlinear discrete-time uncertain system:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w_k) \\ y_k &= h(x_k, v_k), \end{aligned} \tag{5.1}$$

where k denotes the time index; $x_k \in \mathbb{R}^{n_x}$ denotes the system states; $u_k \in \mathbb{R}^{n_u}$ denotes the control inputs; $y_k \in \mathbb{N}^{n_y}$ denotes the outputs; $w_k \in \mathbb{R}^{n_w}$ denotes stochastic process noise; $v_k \in \mathbb{N}^{n_v}$ denotes measurement noise; and $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ and $h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_y}$ denote the system state and output equations, respectively.

Let $\mathcal{I}_k$ denote the matrix of available system information at time k

$$\begin{aligned} \mathcal{I}_k &:= [y_k, ..., y_0, u_{k-1}, ..., u_0] \\ \mathcal{I}_0 &:= [y_0]. \end{aligned} \tag{5.2}$$

We make the following assumptions in this chapter:

Assumption 5.1. *$f(\cdot)$ and $h(\cdot)$ are differentiable almost everywhere with full-rank Jacobians.*

Assumption 5.2. *The noise sequences $\{w_k\}$ and $\{v_k\}$ are subjected to independent zero-mean Gaussian distributions, that is, $w_k \sim \mathcal{N}(0, \Sigma_w)$ and $v_k \sim \mathcal{N}(0, \Sigma_v)$.*

Assumption 5.3. *Random variables x_0, $\{w_k\}$, and $\{v_l\}$ are mutually independent for all $k, l \geq 0$.*

Define an N-horizon cost function:

$$J := \mathbf{E}[\sum_{k=0}^{N-1} l(x_k, u_k) + l_N(x_N)], \tag{5.3}$$

where $\mathbf{E}$ denotes the expected value, $l : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ and $l_N : \mathbb{R}^{n_x} \to \mathbb{R}$ are the stage-wise and terminal costs, respectively, and are twice continuously differentiable.

In addition to minimizing cost (5.3), we impose chance constraints on the state of the form

$$\mathrm{P}[g(x_k) \leq 0] \geq 1 - p_{vio}, \text{ or } \mathrm{P}[x_k \in \mathbb{X}] \geq 1 - p_{vio} \tag{5.4}$$

where P denotes the probability, $g : \mathbb{R}^{n_x} \to \mathbb{R}^{n_c}$, and $p_{vio} \in (0, 0.5]$ denotes the maximum allowed probability of state constraint violation. The hard constraints on the control inputs are as follows:

$$u_k \in \mathbb{U}. \tag{5.5}$$

Given system equations (5.1) and a prediction horizon of length N, we can define the finite-horizon stochastic optimal control problem with initial state $x_0 \sim \mathcal{N}(\bar{x}_0, \Sigma_0)$ as follows:

Problem 5.1 (SOCP)**.**

$$\begin{aligned} \min_{(u_k)_{k=0}^{N-1}} \quad & \mathbf{E}[\sum_{k=0}^{N-1} l(x_k, u_k) + l_N(x_N)] \\ \text{s.t.} \quad & x_{k+1} = f(x_k, u_k, w_k) \\ & \mathrm{P}[x_k \in \mathbb{X}] \geq 1 - p_{vio} \\ & u_k \in \mathbb{U} \end{aligned}$$

5.3 Output-feedback SMPC

5.3.1 Bellman equation for general SOCP

In the output-feedback case, the Bellman equation of SOCP (5.1) should be defined as a function of information state $\xi_{k|k}$ (see Section 2.2.2) as follows

$$\begin{aligned} V_k(\xi_{k|k}) := \min_{u_k} \; & \mathbf{E}_{x_k}[l(x_k, u_k) + V_{k+1}(\xi_{k+1|k+1})] \\ \text{s.t.} \; & \xi_{k+1|k} = \mathcal{G}(\xi_{k|k}, u_k) \\ & \xi_{k+1|k+1} = \mathcal{H}(\xi_{k+1|k}, I_{k+1}) \\ & \mathrm{P}[x_{k+1} \in \mathbb{X} | I_k] \geq \mathbf{1} - p_{vio} \\ & u_k \in \mathbb{U} \end{aligned} \tag{5.6}$$

where $V_N(\xi_{N|N}) := \mathbf{E}_{x_N}[l_N(x_N)]$. Note that, at stage k, the chance constraint is related to the conditional probability distribution $\xi_{k+1|k} = p[x_{k+1}|\mathcal{I}_k]$ with a given $\mathcal{I}_k$. Because the control inputs are causal, the information patterns of these observations can only be defined in terms of their probability distributions. By solving these optimization problems backward from $k = N - 1$ to zero, we incorporate the causal anticipation of future system observations $\mathcal{I}_k$ up to the end-stage N, which implies a closed-loop policy (Bar-Shalom and Tse (1974)).

The solution to the optimization problem on the right-hand side of (5.6) is the core difficulty in SOCP. This difficulty is due to the requirement of an explicit solution for nonlinear optimization. The value function of general nonlinear problems does not have an analytical form such as the LQG. This difficulty can be addressed using ADP, which solves for a numerical solution.

The second difficulty is that even if the numerical solution is considered, the "curse of dimensionality" cannot be avoided, that is, the exponential growth of the computational and storage requirements as the information sequences $\mathcal{I}_k$ grows. To handle this difficulty, an estimator must be introduced to summarize all information up to time k. Note that although the noise terms w_k and v_k in (5.1) are Gaussian, the distributions of x_k and y_k can be non-Gaussian because Gaussian noise passes through the nonlinear functions f and h. To derive a practically efficient algorithm, we assume that the distribution of the state conditioned on the measurement and inputs can be approximated by a Gaussian distribution in this chapter. Therefore, the update of the hyperstate can be represented by the update of its sufficient statistics, that is, mean and variance.

Most nonlinear filters can be used to update the hyperstate, but filters based on Monte Carlo methods, such as particle filters, may cause excessive computational burden in real-time calculations. Considering the real-time calculation factors, the extended Kalman filter (EKF) may be a better choice. Although the EKF has a certain level of estimation error for nonlinear systems, we can reuse the Jacobian matrix obtained when solving the optimization problem in the EKF. Therefore, the EKF has significant computational cost advantages. Considering the estimation error and real-time requirements, we chose to use EKF to update the hyperstate to obtain an efficient numerical algorithm.

5.3.2 Gaussian belief dynamic model

The EKF provides the predicted state and covariance estimates (Ribeiro (2004)):

$$\begin{aligned} \hat{x}_{k+1|k} &= f(\hat{x}_{k|k}, u_k, 0) \\ \Sigma_{k+1|k} &= f_{x,k}\Sigma_{k|k}f_{x,k}^{\mathrm{T}} + f_{w,k}\Sigma_w f_{w,k}^{\mathrm{T}}, \end{aligned} \tag{5.7}$$

where $f_{x,k}$ and $f_{w,k}$ denote the derivatives of f with respect to x and w at time k, respectively. $\hat{x}_{k+1|k}$ denotes the estimate of x at time $k+1$ given observations up to and including at time k, which corresponds to the hyperstate update (2.26) in the Gaussian belief.

The measurement update can be expressed as follows:

$$\begin{aligned} L_{k+1} &= \Sigma_{k+1|k} h_{x,k}^{\mathrm{T}} (h_{x,k} \Sigma_{k+1|k} h_{x,k}^{\mathrm{T}} + h_{v,k} \Sigma_v h_{x,k}^{\mathrm{T}})^{-1} \\ \hat{x}_{k+1|k+1} &= \hat{x}_{k+1|k} + L_{k+1}(y_{k+1} - h(\hat{x}_{k+1|k}, 0)) \\ \Sigma_{k+1|k+1} &= (I - L_{k+1} h_{x,k}) \Sigma_{k+1|k}, \end{aligned} \tag{5.8}$$

which corresponds to the hyperstate update equation (2.27).

Note that measurement y_{k+1} is unavailable when solving the SOCP because the controller is causal. We must approximate it using its probabilistic knowledge, for example, a certainty equivalent:

$$y_{k+1} = h(x_{k+1}, v_{k+1}) \approx h(\hat{x}_{k+1|k}, 0). \tag{5.9}$$

By substituting (5.9) into (5.8) and combining (5.7) and (5.8), we can obtain the recursive equation of hyperstate update:

$$\begin{aligned} \bar{x}_{k+1} &= f(\bar{x}_k, u_k, 0) \\ \Sigma'_{k+1} &= f_{x,k} \Sigma_k f_{x,k}^{\mathrm{T}} + f_{w,k} \Sigma_w f_{w,k}^{\mathrm{T}} \\ \Sigma_{k+1} &= (I - L_{k+1} h_{x,k}) \Sigma'_{k+1}, \end{aligned} \tag{5.10}$$

where $\bar{x}_k$, $\bar{x}_{k+1}$, Σ_k, Σ'_k, and Σ_{k+1} denote $\hat{x}_{k|k}$, $\hat{x}_{k+1|k+1}$, $\Sigma_{k|k}$, $\Sigma_{k+1|k}$, and $\Sigma_{k+1|k+1}$, respectively.

The Bellman equation (5.6) can be rewritten as

$$\begin{aligned} V_k(x_k) := \min_{u_k} \ & \mathbf{E}_{x_k}[l(x_k, u_k) + V_{k+1}(x_{k+1})] \\ \text{s.t. } & \bar{x}_{k+1} = f(\bar{x}_k, u_k, 0) \\ & \Sigma'_{k+1} = f_{x,k} \Sigma_k f_{x,k}^{\mathrm{T}} + f_{w,k} \Sigma_w f_{w,k}^{\mathrm{T}} \\ & \Sigma_{k+1} = (I - L_{k+1} h_{x,k}) \Sigma'_{k+1} \\ & \mathrm{P}[x'_{k+1} \in \mathbb{X}] \geq \mathbf{1} - p_{vio}, \ u_k \in \mathbb{U} \end{aligned} \tag{5.11}$$

where $x_k \sim \mathcal{N}(\bar{x}_k, \Sigma_k)$, $x'_{k+1} \sim \mathcal{N}(\bar{x}_{k+1}, \Sigma'_{k+1})$ and $x_{k+1} \sim \mathcal{N}(\bar{x}_{k+1}, \Sigma_{k+1})$.

5.3.3 Output-feedback SMPC algorithm

This subsection presents the entire algorithm for output-feedback SMPC.

Let i be the current time instant of control systems, and u_i^* be the optimal control input, obtained by solving the Bellman equation (5.11) recursively at time instant i. The receding-horizon implementation of this optimal policy produces measurement information at each time instant; therefore, the EKF (5.7)–(5.8) is applied to obtain

Algorithm 5.1: Output-feedback SMPC

```
Initialization:
    system equations (2.15) ;
    prediction horizon $N$ ;
    time instant counter $i = 0$ ;
    initial state distribution $x_{0|0} \sim \mathcal{N}(\hat{x}_{0|0}, \Sigma_{0|0})$ ;
    probability distribution of $w$ and $v$ ;
    $w \sim \mathcal{N}(0, \Sigma_w)$, $v \sim \mathcal{N}(0, \Sigma_v)$ ;
Repeat
    For given $x_i \sim \mathcal{N}(\hat{x}_{i|i}, \Sigma_{i|i})$ ;
    $\bar{x}_i \leftarrow \hat{x}_{i|i}$ ;
    $\Sigma_i \leftarrow \Sigma_{i|i}$ ;
    Solve the Bellman equation (5.11) to get $u_i^*$ for $x_0 \sim \mathcal{N}(\bar{x}_i, \Sigma_i)$ ;
    Apply first control input to get $y_{i+1}$ ;
    Run EKF (5.7)–(5.8) ;
    $(\hat{x}_{i+1|i+1}, \Sigma_{i+1|i+1}) \leftarrow \text{EKF}(\hat{x}_{i|i}, \Sigma_{i|i}, y_{i+1})$ ;
    Update the time subscript $i = i + 1$ ;
Until finished
```

the state estimate by the current observation y_i. The receding-horizon implementation is summarized in Algorithm 5.1.

The difficulty of output-feedback SMPC in Algorithm 5.1 is finding the solution of SOCP (5.1) by solving the chance-constrained stochastic Bellman equation (5.11) online. In the next section, we use the Gaussian belief dynamic model to convert stochastic optimization to deterministic optimization, and an efficient algorithm is presented for online calculations based on ADP.

5.4 Stochastic Approximate Dynamic Programming Algorithm

We have introduced the RRIPM in Chapter 4 which solves the linear constrained dynamic programming via the interior-point method. Now we intend to introduce the ADP algorithm in nonlinear case.

ADP is an iterative method that decomposes a large OCP into a recursive series of small problems, and each solves an approximated Bellman equation at a single time instant, which is implemented backward in time. Specifically, we let $(\tilde{x}_k, \tilde{u}_k)$ be a given trajectory that satisfies the nominal state equation, the value function is approximated by a quadratic fit around this given trajectory. By performing a *backward pass* and a *forward pass*, the trajectory iteratively moves toward the minima of the quadratic approximations and is progressively improved toward a local optimum. During the backward pass, the algorithm approximates the value function as a quadratic function along the current trajectory. In the forward pass, forward propagation is performed to produce a new trajectory based on the value function computed in the backward pass. This process is repeated until the desired convergence level is achieved.

5.4.1 Deterministic reformulation via the Gaussian belief dynamic model

Let $(\delta x_k, \delta u_k)$ be the deviation from a given trajectory. Thus, we have $(x_k, u_k) = (\tilde{x}_k + \delta x_k, \tilde{u}_k + \delta u_k)$. First, we focus on the chance constraints. In general, nonlinear chance constraints are intractable, because the propagation of stochastic variables along nonlinear functions are difficult to compute.

Note that the stochastic variable in chance constraints is $x'_{k+1} \sim \mathcal{N}(\bar{x}_{k+1}, \Sigma'_{k+1})$. Linearizing nonlinear constraint function g near $\tilde{x}_{k+1}$:

$$g(x'_{k+1}) \approx g(\tilde{x}_{k+1}) + g_{x,k+1}\delta x'_{k+1} \tag{5.12}$$

where $\delta x'_{k+1} \sim \mathcal{N}(\delta\bar{x}_{k+1}, \Sigma'_{k+1})$ and $\delta\bar{x}_{k+1} = \bar{x}_{k+1} - \tilde{x}_{k+1}$. The analytical reformulation of linear chance constraint (5.12) can be obtained using the method introduced in Subsection 2.2.3. By standardizing the Gaussian distribution, we obtain

$$P[\Lambda^{-1}(g_{x,k+1}\delta x'_{k+1} - g_{x,k+1}\delta\bar{x}_{k+1}) \leq \Lambda^{-1}(g(\tilde{x}_k + 1) - g_{x,k+1}\delta\bar{x}_{k+1}))] \geq 1 - p_{vio}, \tag{5.13}$$

where $g_{x,k+1}\Sigma'_{k+1}g^{\mathrm{T}}_{x,k+1} = \Lambda\Lambda^{\mathrm{T}}$, and Λ can be obtained by matrix decomposition methods such as the Choleskey or LDL decomposition (Meyer (2000)). From the definition of the cumulative distribution function (CDF), the equation can be rewritten as

$$\Phi(\Lambda^{-1}(g(\tilde{x}_k + 1) - g_{x,k+1}\delta\bar{x}_{k+1})) \geq 1 - p_{vio}, \tag{5.14}$$

where Φ denotes the standard Gaussian CDF. The inverse function of standard Gaussian CDF Φ^{-1} can be computed offline with arbitrary precision, whereafter the chance constraints can be reformulated as:

$$g(\tilde{x}_{k+1}) + g_{x,k+1}\delta\bar{x}_{k+1} + t(\mathbf{1} - p_{vio}) \leq 0 \tag{5.15}$$

where $t(\mathbf{1} - p_{vio}) = \Lambda\Phi^{-1}(\mathbf{1} - p_{vio})$.

By further linearizing f near $(\tilde{x}_k, \tilde{u}_k)$, the propagation of $\delta\bar{x}_k$ can be expressed as follows:

$$\delta\bar{x}_{k+1} = f_{x,k}\delta\bar{x}_k + f_{u,k}\delta u_k. \tag{5.16}$$

Substituting (5.16) into (5.15), we obtain:

$$g_1 \delta \bar{x}_k + g_2 \delta u_k + g_3 \leq 0 \tag{5.17}$$

where

$$g_1 = g_{x,k+1} f_{x,k}, \quad g_2 = g_{x,k+1} f_{u,k}, \quad g_3 = g(\tilde{x}_{k+1}) + t(\mathbf{1} - p_{vio}).$$

The hard constraints on the control input (5.5) can also be handled using a similar linearization procedure. By combining the linear constraint on state and input, we can obtain the deterministic constraint form

$$g'(\bar{x}_k, u_k, \Sigma'_{k+1}) := C_{x,k} \delta \bar{x}_k + C_{u,k} \delta u_k + C_{c,k} \leq 0 \tag{5.18}$$

where $C_{x,k}$, $C_{u,k}$, and $C_{c,k}$ are the merged coefficient matrices of $\delta \bar{x}_k$, δu_k, and the constant term, respectively.

We can now define the action-value function under a dynamic programming framework. By adding a term involving Lagrange multipliers to the stage cost, we can rewrite the cost function of the minimization problem in the value function (5.11), as follows:

$$\min_{u_k} \max_{s_k} \mathbf{E}[l(x_k, u_k) + s_k^{\mathrm{T}} g' + V_{k+1}(x_{k+1})] \tag{5.19}$$

where s_k is the Lagrange multiplier.

We let $Q : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ be the action-value function of (5.19). More conveniently, we define Q in terms of the deviation from the given state and action.

$$\begin{aligned} Q_k =& \mathbf{E}[l(\tilde{x}_k + \delta x_k, \tilde{u}_k + \delta u_k)] + \\ & V_{k+1}(\tilde{x}_{k+1} + \delta x_{k+1}) + (\tilde{s}_k + \delta s_k) g'(\tilde{x}_k + \delta x_k, \tilde{u}_k + \delta u_k)], \end{aligned} \tag{5.20}$$

where $\tilde{s}_k$ is a given sequence of Lagrange multipliers, and δs_k denotes the deviation from $\tilde{s}_k$.

To approximate the value function as a quadratic function, we consider the second-order Taylor expansion of Q along the given trajectory:

$$\begin{aligned} Q_k \approx& \mathbf{E}[\tilde{Q}_k + Q_{x,k}^{\mathrm{T}} \delta x_k + Q_{u,k}^{\mathrm{T}} \delta u_k + Q_{s,k}^{\mathrm{T}} \delta s_k + \frac{1}{2}(\delta x_k^{\mathrm{T}} Q_{xx,k} \delta x_k + \delta u_k^{\mathrm{T}} Q_{uu,k} \delta u_k) + \\ & \delta u_k^{\mathrm{T}} Q_{ux,k} \delta x_k + \delta s_k^{\mathrm{T}} Q_{sx,k} \delta x_k + \delta s_k^{\mathrm{T}} Q_{su,k} \delta u_k] \end{aligned} \tag{5.21}$$

where

$$\begin{aligned}
\tilde{Q}_k &= Q(\tilde{x}_k, \tilde{u}_k) + V_{k+1}(\tilde{x}_{k+1}) \\
Q_{x,k} &= l_{x,k} + f_{x,k}^{\mathrm{T}} V_{x,k+1} \\
Q_{u,k} &= l_{u,k} + f_{u,k}^{\mathrm{T}} V_{x,k+1} \\
Q_{xx,k} &= l_{xx,k} + f_{x,k}^{\mathrm{T}} V_{xx,k+1} f_{x,k} + V_{x,k+1} f_{xx,k} \\
Q_{uu,k} &= l_{uu,k} + f_{u,k}^{\mathrm{T}} V_{xx,k+1} f_{u,k} + V_{x,k+1} f_{uu,k} \\
Q_{ux,k} &= l_{ux,k} + f_{u,k}^{\mathrm{T}} V_{xx,k+1} f_{x,k} + V_{x,k+1} f_{ux,k} \\
Q_{s,k} &= C_{c,k},\ Q_{sx,k} = C_{x,k},\ Q_{su,k} = C_{u,k}.
\end{aligned}$$

The expected value of Q can be evaluated as follows:

$$\begin{aligned}
Q_k =& \tilde{Q}_k + Q_{x,k}^{\mathrm{T}} \delta\bar{x}_k + Q_{u,k}^{\mathrm{T}} \delta u_k + Q_{s,k}^{\mathrm{T}} \delta s_k + \frac{1}{2}(\delta\bar{x}_k^{\mathrm{T}} Q_{xx,k} \delta\bar{x}_k + \delta u_k^{\mathrm{T}} Q_{uu,k} \delta u_k) \\
&+ \delta u_k^{\mathrm{T}} Q_{ux,k} \delta\bar{x}_k + \delta s_k^{\mathrm{T}} Q_{sx,k} \delta\bar{x}_k + \delta s_k^{\mathrm{T}} Q_{su,k} \delta u_k + \Gamma(\Sigma_k, \Sigma_{k+1}', \Sigma_{k+1})
\end{aligned} \tag{5.22}$$

where $\Gamma(\cdot)$ denotes a combination of terms related to the covariance matrix. As mentioned in the previous section, the covariance update depends only on the linearized points, and the term with the covariance matrix does not affect the solution of the optimization problem. Therefore, we can omit these terms from the Q function. The value function can be expressed as a quadratic approximation of Q:

$$V_k(x_k) := \min_{\delta u_k} \max_{\delta s_k} \ Q(\delta\bar{x}_k, \delta u_k, \delta s_k) \tag{5.23}$$

5.4.2 Constrained SADP algorithm

Thus far, we have introduced a method to reformulate the stochastic ADP as a deterministic ADP in a Gaussian belief fashion and the corresponding stage-wise optimization problem (5.23). Hereafter, we will present the proposed constrained stochastic ADP (SADP) algorithm, which contains a backward pass for updating the value function and policy, and a forward pass for calculating a new trajectory as well as posterior estimation. Details of dealing with constrained dynamic programming via an *interior-point*-based method can be found in Chapter 3; here, we briefly introduce the entire algorithm.

The optimality conditions of (5.23) can be summarized as the following stage-wise KKT (SKKT) conditions:

$$Q_{uu,k}\delta u_k + Q_{ux,k}\delta\bar{x}_k + Q_{us,k}\delta s_k + Q_{u,k} = 0 \tag{5.24a}$$

$$(\tilde{s}_k + \delta s_k)(\tilde{y}_k + \delta y_k) = 0 \tag{5.24b}$$

$$Q_{sx,k}\delta\bar{x}_k + Q_{su,k}\delta u_k + Q_{s,k} + \tilde{y}_k + \delta y_k = 0 \tag{5.24c}$$

$$\tilde{s}_k + \delta s_k \geq 0,\ \tilde{y}_k + \delta y_k \geq 0, \tag{5.24d}$$

Algorithm 5.2: Constrained SADP

```
Parameters:
  $\sigma \in (0,1)$ reduction parameter, ;
  $\alpha_f = 0.995$ fraction-to-boundary parameter;
  $(\epsilon_o, \epsilon_d, \epsilon_f) = 10^{-6}$ terminal tolerance;
Initialization:
  known initial state, $(\bar{x}_0, \Sigma_0)$ ;
  initial guess $(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}, \tilde{\mathbf{s}}, \tilde{\mathbf{y}}, \Sigma)$ ;
  central parameter $\boldsymbol{\mu}$ ;
  optimality measurements $(\|\boldsymbol{\rho}_o\|, \|\boldsymbol{\rho}_d\|, \|\boldsymbol{\rho}_f\|)$ ;
while $\|\boldsymbol{\rho}_o\| > \epsilon_o \vee \|\boldsymbol{\rho}_o\| > \epsilon_d \vee \|\boldsymbol{\rho}_f\| > \epsilon_f$ do
    $V_{xx,N} \leftarrow l_{xx,N}$, $V_{x,N} \leftarrow l_{x,N}$ ;
    for $k \leftarrow N-1$ to $0$ do
        // Backward Pass
        Solve the linear equation (5.27);
        $\delta u_k \leftarrow \eta_{1k}\delta x_k + \theta_{1k}$;
        $\delta s_k \leftarrow \eta_{2k}\delta x_k + \theta_{2k}$;
        $\delta y_k \leftarrow \eta_{3k}\delta x_k + \theta_{3k}$;
        - Update the coefficients using (5.30) ;
        $(V_{xx,k}, V_{x,k}) \leftarrow (V_{xx,k+1}, V_{x,k+1})$;
    end
    for $k \leftarrow 0$ to $N-1$ do
        // Forward Pass
        Find $\alpha^i$ using line search rules (5.32);
        Calculate the new trajectory using (5.31);
        $(u_k, s_k, y_k) \leftarrow (\tilde{u}_k, \tilde{s}_k, \tilde{y}_k)$. - Calculate the mean and covariance using
          (5.33) ;
        $(\bar{x}_{k+1}, \Sigma'_{k+1}, \Sigma_{k+1}) \leftarrow (\bar{x}_k, \Sigma_k)$.
    end
    Update new trajectory ;
    $(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}, \tilde{\mathbf{s}}, \tilde{\mathbf{y}}, \Sigma) \leftarrow (\bar{\mathbf{x}}, \mathbf{u}, \mathbf{s}, \mathbf{y}, \Sigma)$;
    Reduce the barrier parameter $\boldsymbol{\mu} = \sigma\boldsymbol{\mu}$;
    Compute the optimality measurements $(\zeta_o, \zeta_d, \zeta_f)$.
end
```

where y_k is a slack variable that transforms inequalities into equalities to simplify the calculation.

Let $(\tilde{x}_k, \tilde{u}_k, \Sigma_k)$ be the initial guess, which is feasible for the posterior estimation update (5.10), $(\tilde{s}_k, \tilde{y}_k)$ be a non-negative initial estimate of the Lagrange multiplayer and slack variables. We summarize the algorithm in Algorithm 5.2, which contains a backward pass, forward pass, and an outer loop below.

5.4.2.1 Backward pass

Let the approximated optimal value function be the following quadratic form:

$$V_k(x_k) := \frac{1}{2}\delta\bar{x}_k^{\mathrm{T}} V_{xx,k}\delta\bar{x}_k + V_{x,k}^{\mathrm{T}}\delta\bar{x}_k + V_{c,k} \tag{5.25}$$

where $V_{xx,N} = l_{xx,N}$, $V_{x,N} = l_{x,N}$, and $V_{c,k}$ are constant terms irrelevant to the algorithm, while the remaining coefficients $V_{xx,k}$ and $V_{x,k}$ are updated recursively in the backward pass.

The difficulty of solving SKKT conditions (5.24) is caused by the nonsmooth equation (5.24b) and inequalities (5.24d), where the difficulty of inequalities will be handled in the forward pass, and the nonsmooth equation can be handled by adding a smooth parameter, that is,

$$(\tilde{s}_k + \delta s_k)(\tilde{y}_k + \delta y_k) = \mu_k \tag{5.26}$$

where $\mu_k = \tilde{s}_k^{\mathrm{T}}\tilde{y}_k/n_c$ is the current smoothing parameter. By applying Newton's method to the smoothed SKKT conditions, and treating $\delta\bar{x}_k$ as a parameter, we obtain the following equations:

$$\begin{bmatrix} Q_{uu,k} & Q_{us,k} & 0 \\ 0 & \tilde{Y}_k & \tilde{S}_k \\ Q_{su,k} & 0 & I_{n_c} \end{bmatrix} \begin{bmatrix} \delta u_k \\ \delta s_k \\ \delta y_k \end{bmatrix} = - \begin{bmatrix} Q_{ux,k} \\ 0 \\ Q_{sx,k} \end{bmatrix} \delta\bar{x}_k - \begin{bmatrix} \rho_o \\ \rho_d - \mu_k \\ \rho_f \end{bmatrix} \tag{5.27}$$

where the optimality, duality, and constraint feasibility measurements, ρ_o, ρ_d and ρ_f are defined as follows:

$$\begin{bmatrix} \rho_o \\ \rho_d \\ \rho_f \end{bmatrix} = \begin{bmatrix} Q_{u,k} \\ \tilde{S}_k\tilde{Y}_k \\ Q_{s,k} + \tilde{y}_k \end{bmatrix}, \tag{5.28}$$

$\tilde{S}_k := \mathrm{diag}[\tilde{s}_k]$, as does $\tilde{Y}_k$.

The parametric solution of (5.27) as a function of $\bar{x}_k$ is:

$$\begin{aligned} \delta u_k &= \eta_{1k}\delta\bar{x}_k + \theta_{1k} \\ \delta s_k &= \eta_{2k}\delta\bar{x}_k + \theta_{2k} \\ \delta y_k &= \eta_{3k}\delta\bar{x}_k + \theta_{3k}. \end{aligned} \tag{5.29}$$

Note that at stage $N-1$, solution (5.29) can be obtained because $V_{xx,N}$ and $V_{x,N}$ are given. Then, for $k = N-1, ..., 0$, by substituting (5.29) into (5.23) and comparing the coefficients of the second-order and first-order terms, we obtain the following Riccati equations:

$$\begin{aligned} V_{xx,k} =& Q_{xx,k} + \eta_{1k}^{\mathrm{T}} Q_{uu,k}\eta_{1k} + 2\eta_{1k}^{\mathrm{T}} Q_{ux,k} + 2\eta_{2k}^{\mathrm{T}}(Q_{sx,k} + Q_{su,k}\eta_{1k}) \\ V_{x,k} =& Q_{x,k} + Q_{u,k}\eta_{1k} + \theta_{1k}^{\mathrm{T}}(Q_{ux,k} + Q_{uu,k}\eta_{1k}) + \\ & (Q_{su,k}\theta_{1k} + Q_{s,k})\eta_{2k}^{\mathrm{T}} + \theta_{2k}^{\mathrm{T}}(Q_{sx,k} + Q_{su,k}\eta_{1k}). \end{aligned} \tag{5.30}$$

5.4.2.2 Forward pass

In the forward pass, we updated the Gaussian belief dynamics. At $k = 0$, the information of the initial state is $(\bar{x}_0, \Sigma_0)$; thus, we have $\delta x_0 = 0$. For $k = 1, ..., N$, we can calculate the deviation pairs $(\delta u_k, \delta s_k, \delta y_k)$ by (5.29). The interior-point-based method generates a new trajectory that satisfies inequalities (5.24d) in SKKT; therefore, we employ a line search procedure by including a step size α_k as follows:

$$\begin{bmatrix} u_k \\ s_k \\ y_k \end{bmatrix} = \begin{bmatrix} \tilde{u}_k \\ \tilde{s}_k \\ \tilde{y}_k \end{bmatrix} + \alpha_k \begin{bmatrix} \delta u_k \\ \delta s_k \\ \delta y_k \end{bmatrix} \tag{5.31}$$

where

$$\begin{aligned} \alpha_k &= \operatorname{diag}\begin{bmatrix} \alpha_f \alpha_y & \alpha_f \alpha_s & \alpha_f \alpha_y \end{bmatrix} \\ \alpha_s &= \max\{\alpha \in (0, 1]) : \tilde{s} + \alpha \delta s_k \geq 0\} \\ \alpha_y &= \max\{\alpha \in (0, 1]) : \tilde{y} + \alpha \delta y_k \geq 0\} \end{aligned} \tag{5.32}$$

and α_f is the fraction of the boundary parameter that prevents the Lagrange multiplayer and slack variables from approaching zero too quickly.

Next, we can calculate the mean and covariance using

$$\begin{aligned} \bar{x}_{k+1} &= f(\bar{x}_k, u_k, 0) \\ \Sigma'_{k+1} &= f_{x,k} \Sigma_k f_{x,k}^{\mathrm{T}} + f_{w,k} \Sigma_w f_{w,k}^{\mathrm{T}} \\ \Sigma_{k+1} &= (I - L_{k+1} h_{x,k}) \Sigma'_{k+1}. \end{aligned} \tag{5.33}$$

The new $\delta \bar{x}_{k+1}$ can be obtained from $\delta \bar{x}_{k+1}$ by $\delta \bar{x}_{k+1} = \bar{x}_{k+1} - \tilde{x}_{k+1}$.

Perform the above update recursively from $k = 0$ to $N - 1$ and let

$$(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}, \tilde{\mathbf{s}}, \tilde{\mathbf{y}}, \Sigma) \leftarrow (\bar{\mathbf{x}}, \mathbf{u}, \mathbf{s}, \mathbf{y}, \Sigma), \tag{5.34}$$

we obtained a new trajectory.

5.4.2.3 Outer loop

The backward and forward passes together complete the Newton steps. The algorithm terminates when the optimality, duality, or constraint feasibility measurements satisfy the following stop criteria:

$$\|\boldsymbol{\rho}_o\| \leq \epsilon_o, \ \|\boldsymbol{\rho}_d\| \leq \epsilon_d, \ \text{and} \ \|\boldsymbol{\rho}_f\| \leq \epsilon_f \tag{5.35}$$

where $\boldsymbol{\rho} = [\rho_0, ..., \rho_{N-1}]$. The selection of $(\epsilon_o, \epsilon_d, \epsilon_f)$ depends on the control accuracy requirements.

Remark 5.1. *The feasibility of Constrained SADP algorithm also remains problematic. A practical way is to solve an alternative softened problem as we introduced in Section 4.6 and Appendix A.*

5.5 Convergence Analysis of Constrained SADP Algorithm

The local convergence property of Algorithm 5.2 is discussed in this section. Let $z = (x, u, s, y)$ and $F_k(z_k) = 0$ denote the SKKT conditions of the original optimization problem (5.19). We make the following assumptions.

Assumption 5.4. *There exists an optimal solution z_k^* ($k = 0, ..., N$) that satisfies $F_k(z_k^*) = 0$ and strict complementarity holds.*

Assumption 5.5. *$l(\cdot)$, $l_N(\cdot)$, $f(\cdot)$ and $h(\cdot)$ are locally twice Lipschitz continuous at z^**

Assumption 5.6. *$Q_{uu,k}$ is a positive definite for $k = 0, ..., N$.*

Assumptions 5.4–5.6 represent existence, smoothness, and second-order sufficiency, respectively, which are standard for Newton's method in most of the literature.

The following lemma is necessary before proving the local convergence rate:

Lemma 5.1. *Under Assumptions 5.4–5.6, the SKKT matrix*

$$\begin{bmatrix} Q_{uu,k} & Q_{us,k} & 0 \\ 0 & \tilde{Y}_k & \tilde{S}_k \\ Q_{su,k} & 0 & I_{n_c} \end{bmatrix} \tag{5.36}$$

is nonsingular and locally Lipschitz continuous at z^*.

Proof. Local Lipschitz continuity can be easily obtained from Assumption 5.5 because $l(\cdot)$, $l_N(\cdot)$, $f(\cdot)$ and $h(\cdot)$ are all locally twice Lipschitz continuous at z^*.

If z_k^* is a solution for which strict complementarity holds, then for each Newton iteration at time k, under the line search rule (5.32) in the forward pass, either s_k or y_k remains bounded away from zero as the iteration approaches z_k^*, ensuring that both $\tilde{Y}_k$ and $\tilde{S}_k$ are positive definite. In addition, there are positive definite matrices $Q_{uu,k}$, indefinite matrix $Q_{us,k}$ and its transpose $Q_{su,k}$ in (5.36), so we can rewrite the SKKT matrix (5.36) as follows:

$$\left[\begin{array}{cc|c} A & B & 0 \\ 0 & C & D \\ \hline B^{\mathrm{T}} & 0 & I \end{array}\right] \tag{5.37}$$

where matrices A, C, and D are positive definite and C and D are diagonal. By calculating the Schur complement of the block matrix:

$$M = \begin{bmatrix} A & B \\ 0 & C \end{bmatrix} - \begin{bmatrix} 0 \\ D \end{bmatrix} \begin{bmatrix} B^{\mathrm{T}} & 0 \end{bmatrix} = \begin{bmatrix} A & B \\ -DB^{\mathrm{T}} & C \end{bmatrix}. \tag{5.38}$$

The entire matrix is invertible if M is invertible. By recalculating the Schur complement of the matrix M, we know that M is invertible if $A + BC^{-1}DB^{\mathrm{T}}$ is invertible. From the positive definite and diagonal property of C and D, we know that $B^{\mathrm{T}}C^{-1}DB$ is positive semidefinite. Together with the positive definite property of A, we can conclude that $A + BC^{-1}DB^{\mathrm{T}}$ is positive definite and therefore invertible. $\square$

Let us now discuss the local convergence property of Algorithm 5.2. Let z_k^j be the old trajectories $\tilde{z}_k$ at iteration j and let z_k^{j+1} be the new trajectory generated by (5.31).

Theorem 5.1. *Let $\mathbf{z} = (\mathbf{x}, \mathbf{u}, \mathbf{s}, \mathbf{y})$. For all $k = 0, ..., N-1$, if $\alpha_k^j \to 1$ $(j \to \infty)$, $\|\alpha_k^j - 1\| \leq \varepsilon$, and $\|\mu_k^0\| \leq \varepsilon \|\mathbf{z}^0 - \mathbf{z}^*\| \leq \varepsilon^2$ for a sufficiently small $\varepsilon > 0$, then the iteration sequences generated by Algorithm 5.2 converge to $\mathbf{z}^*$ Q-superlinearly.*

Proof. Let $\zeta = (u, s, y)$. We define the coefficient matrix in (5.27) as

$$\nabla_x F_k = \begin{bmatrix} Q_{ux,k} \\ 0 \\ Q_{sx,k} \end{bmatrix} \quad \nabla_\zeta F_k = \begin{bmatrix} Q_{uu,k} & Q_{us,k} & 0 \\ 0 & \tilde{Y}_k & \tilde{S}_k \\ Q_{su,k} & 0 & I_{n_c} \end{bmatrix}.$$

According to Taylor's theorem, there exist functions $h_1(z_k^j)$ such that

$$0 = F_k(z_k^*) = F_k(z_k^j) + \nabla_x F_k(z_k^j)(x_k^* - x_k^j) + \nabla_\zeta F_k(z_k^j)(\zeta_k^* - \zeta_k^j) + h_1(z_k^j), \tag{5.39}$$

and norms of the residual functions are bounded.

$$\|h_1(z_k^j)\| \leq M_1 \|\mathbf{z}^* - \mathbf{z}^j\|^2, \tag{5.40}$$

where M_1 denotes a constant.

From Lemma 5.1, we know that $\nabla_\zeta F_k(z_k^j)$ is nonsingular. Multiplying $\Psi = \nabla_\zeta F_k^{-1}(z_k^j)$ by both sides of (5.39), we obtain

$$\Psi F_k(z_k^j) + \Psi \nabla_x F_k(z_k^j)(x_k^* - x_k^j) + \zeta_k^* - \zeta_k^j + \Psi h_1(z_k^j) = 0. \tag{5.41}$$

From the forward update (5.31), we obtain the update of ζ.

$$\zeta_k^{j+1} = \zeta_k^j + \alpha_k^j \delta\zeta_k, \tag{5.42}$$

where $\delta\zeta_k$ is the solution of the linear equation (5.27):

$$\delta\zeta_k = -\Psi \nabla_x F_k(z_k^j)(x_k^{j+1} - x_k^j) - \Psi F_k(z_k^j) + \Psi \mu_k', \tag{5.43}$$

where μ'_k denotes $[0\ \mu_k^j\ 0]^\mathrm{T}$ and $\mu_k^j = \sigma^j \mu_k^0$, which is iteratively reduced by the reduction parameter $\sigma \in (0,1)$. Substituting (5.42) into (5.43) yields:

$$\Psi F_k(z_k^j) = -\frac{\zeta_k^{j+1} - \zeta_k^j}{\alpha_k^j} - \Psi \nabla_x F_k(z_k^j)(x_k^{j+1} - x_k^j) + \Psi \mu'_k. \tag{5.44}$$

We can then combine (5.41) and (5.44) to eliminate the term $\Psi F(z_k^j)$:

$$\zeta_k^{j+1} - \zeta_k^* = (1-\alpha_k^j)(\zeta_k^j - \zeta_k^*) + \alpha_k^j \Psi \nabla_x F_k(z_k^j)(x_k^{j+1} - x_k^*) + \alpha_k^j \Psi \mu'_k + \Psi h_1(z_k^j). \tag{5.45}$$

At stage $k=0$, since $x_0^{j+1} = x_0^* = \bar{x}_0$, we have

$$\zeta_0^{j+1} - \zeta_0^* = (1-\alpha_0^j)(\zeta_0^j - \zeta_0^*) + \alpha_0^j \Psi \mu'_0 + \Psi h_1(z_0^j). \tag{5.46}$$

At stage $k=1$, we apply the Taylor theorem to the nominal system equation (5.1)

$$\begin{aligned} x_1^{j+1} - x_1^* =& f(x_0^{j+1}, u_0^{j+1}, 0) - f(x_0^*, u_0^*, 0) \\ \le& f_{x,0}(x_0^{j+1} - x_0^*) + f_{u,0}(u_0^{j+1} - u_0^*) + A_1 \|(x_0^{j+1} - x_0^*), (u_0^{j+1} - u_0^*)\|^2 \\ \le& A_2(\zeta_0^{j+1} - \zeta_0^*) \\ \le& A_3 \|\mathbf{z}^* - \mathbf{z}^j\|^2 \end{aligned} \tag{5.47}$$

for some constant matrices A_1, A_2 and A_3. By continuing this induction, there exists a constant matrix M_2, such that

$$x_k^{j+1} - x_k^* \le M_2 \|\mathbf{z}^* - \mathbf{z}^j\|^2 \tag{5.48}$$

Furthermore, considering (5.45) for $k = 0, ..., N-1$. At $k=0$, we have (5.46), and when $k=1$:

$$\begin{aligned} \zeta_1^{j+1} - \zeta_1^* =& (1-\alpha_1^j)(\zeta_1^j - \zeta_1^*) + \alpha_1^j \Psi \nabla_x F_1(z_1^j)(x_1^{j+1} - x_1^*) + \alpha_1^j \Psi \mu'_1 + \Psi h_1(z_1^j) \\ \le& (1-\alpha_1^j)(\zeta_1^j - \zeta_1^*) + \alpha_1^j \Psi \nabla_x F_1(z_1^j) A_3 \|\mathbf{z}^* - \mathbf{z}^j\|^2 + \alpha_1^j \Psi \mu'_1 + \Psi M_1 \|\mathbf{z}^* - \mathbf{z}^j\|^2 \end{aligned} \tag{5.49}$$

Continuing this induction until $k = N-1$, together with inequalities (5.40) and (5.48), we have

$$\|\mathbf{z}^{j+1} - \mathbf{z}^*\| \le \Upsilon_\zeta \|\mathbf{z}^j - \mathbf{z}^*\| + \Upsilon_\mu \|\boldsymbol{\mu}^0\| + M' \|\mathbf{z}^* - \mathbf{z}^j\|^2 \tag{5.50}$$

where Υ_ζ is composed of the product of $(1-\alpha_k^j)$ and α_k^j, Υ_μ is composed of σ to the power of j and other bonded matrices, and M' is a constant matrix. When $j=0$, from the assumption $\|\alpha_k^j - 1\| \le \varepsilon$, and $\|\mu_k^0\| \le \varepsilon \|\mathbf{z}^0 - \mathbf{z}^*\| \le \varepsilon^2$ hold for a sufficiently small $\varepsilon > 0$, so there must exist a γ^0 that satisfies $0 < \gamma^0 < 1$, such that

$$\begin{aligned} \|\mathbf{z}^1 - \mathbf{z}^*\| \le& (\Upsilon_\zeta + \varepsilon \Upsilon_\mu + M' \|\mathbf{z}^0 - \mathbf{z}^*\|) \|\mathbf{z}^0 - \mathbf{z}^*\| \\ <& \gamma^0 \|\mathbf{z}^0 - \mathbf{z}^*\| \end{aligned} \tag{5.51}$$

By continuing this induction from $j = 0$ to ∞, we know $\mathbf{z}^j \to \mathbf{z}^*$.

Finally, if $\alpha_k^j \to 1$ holds when $j \to \infty$, which implies $\Upsilon_\zeta \to 0$, together with the fact that $\Upsilon_\mu \to 0$ since $0 < \sigma < 1$, we have

$$\lim_{j \to \infty} \frac{\|\mathbf{z}^{j+1} - \mathbf{z}^*\|}{\|\mathbf{z}^j - \mathbf{z}^*\|} = 0 \tag{5.52}$$

$\square$

5.6 Case Study

5.6.1 Numerical example settings

We tested our algorithm via a nonlinear, continuous-time stirred tank reactor (CSTR) problem using the dilution rate as the manipulated variable. The reactor has a constant volume, and its dynamics are described as follows:

$$\dot{x}_1 = -k_1 x_1 - k_3 x_1^2 + (x_f - x_1)u \tag{5.53a}$$

$$\dot{x}_2 = -k_1 x_1 - k_2 x_2 + x_2 u, \tag{5.53b}$$

that models the Van de Vusse series of reactions (Scokaert and Rawlings (1998)):

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C, \quad 2A \xrightarrow{k_3} D$$

where x_1 and x_2 represent the concentrations of A and B, and u represents the dilution rate Van de Vusse (1964). We assume that $k_1 = 50$, $k_2 = 100$, $k_3 = 10$, and $x_f = 10$. This state is partially observed using the following equation:

$$y = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{5.54}$$

We discretize the continuous-time system (5.53) with a sampling time of 0.002 s and assume a stochastic disturbance w_k and measurement noise v_k with the following covariance matrix:

$$\Sigma_w = \begin{bmatrix} 0.005^2 & 0 \\ 0 & 0.005^2 \end{bmatrix}, \quad \Sigma_v = 0.003^2.$$

The control cost function is of the quadratic form $x^{\mathrm{T}} Q x + u^{\mathrm{T}} R u$ with

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1.$$

The real initial state is $\begin{bmatrix} 0.5 & 0.18 \end{bmatrix}^{\mathrm{T}}$ and the estimated initial distribution of x is $x_0 \sim \mathcal{N}(\begin{bmatrix} 0.45 & 0.2 \end{bmatrix}^{\mathrm{T}}, \mathrm{diag}\begin{bmatrix} 0.1 & 0.1 \end{bmatrix})$. x_2 is required to satisfy the chance constraint,

$$\mathrm{P}[x_2 \geq 0.12] \geq 1 - p_{vio}. \tag{5.55}$$

5.6.2 Simulation analysis

For comparison, we also present the simulation results for three control policies.

1. Open-loop policy (see Section 4.7);
2. Proposed Constrained SADP;
3. Pre-parameterized policy: Parameterizes the control policy as $u_k = K_k x_k + v_k$ first and searches directly over the decision variables (K_k, v_k) (see Section 4.7).

We performed the simulation on a laptop computer with a 2.60 GHz Intel Core i7-6700HQ in MATLAB 2020a. The proposed algorithm was coded in the MATLAB environment, and the open-loop controller and the PP controller were implemented in the CasADi toolkit with the interior-point-method-based solver IPOPT Andersson et al. (2019).

Figure 5.1 shows the time evolution of x_2 using the proposed constrained SADP, and it shows that the real system value can be tracked well by EKF under noisy initial states and partial observations, which means that our proposed output-feedback SMPC scheme can perform the control task well.

Table 5.1: Comparison between three policies ($N = 20, p_{vio} = 10\%$)

Control policy	Cost	Constraint violation(%)
Open-loop	1077	5.26
Constrained SADP	902	9.99
PP	944	7.23

Figure 5.2 shows the time histories for all three controllers with a constraint violation allowance $p_{vio} = 10\%$ based on 100 Monte Carlo simulations. As shown in the figure, all three control methods exhibit certain degrees of constraint violation. Table 5.1 lists the average cost and constraint violation rate in the Monte Carlo simulation. The results revealed that the open-loop policy achieved the most conservative control performance, with the highest cost and lowest degree of constraint violation. The control performance of the proposed constrained SADP and PP policies is better than that of the open-loop policy; however, it is slightly more conservative than the proposed method, which may be because pre-parameterization is a heuristic formulation of a feedback policy, and the resulting optimization problem is also difficult to solve. Our proposed method achieved the most outstanding control effect: the closest to the set constraint violation value and the lowest cost.

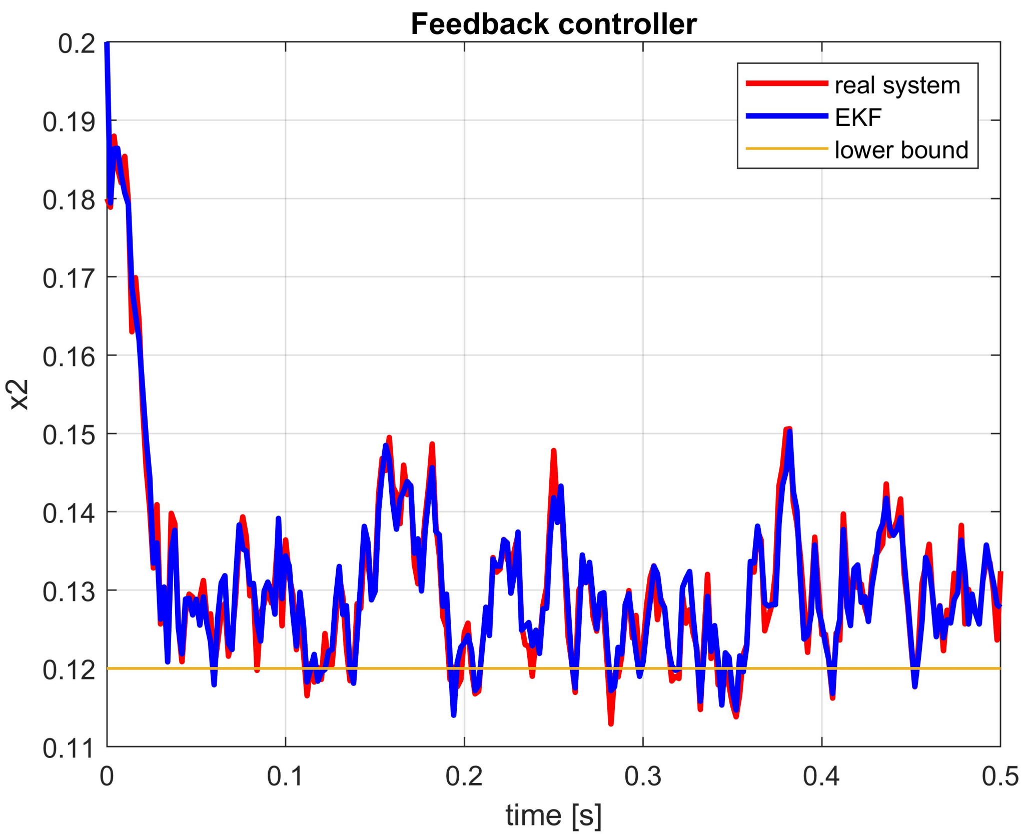

Figure 5.1: Time evolution profile of real system and EKF

The constraint violation rate were further compared in Table 5.2, showing the constraint violation rate for $p_{vio} = 1\%$, $p_{vio} = 5\%$ and $p_{vio} = 20\%$ based on 100 Monte Carlo simulations. The results show that the constrained SADP always remains the closest to the set value, whereas the open-loop policy is always much lower than the set value, revealing that the proposed method can achieve the least conservative control effect while not violating the chance constraint.

The computation times for the three controllers are compared in Table 5.3. The data in the table reveal that when the prediction horizon is short, our proposed algorithm has a computational efficiency similar to that of the IPOPT solver, while the computation time of PP is longer because it introduces extra optimization parameters (K_k, v_k). However, when the prediction horizon is long, our algorithm has an excellent advantage because the computational complexity of ADP increases linearly with the prediction horizon. In contrast, the computational complexity of a general interior-point-method-based solver is cubic for the prediction horizon. Moreover, the average number of iterations to reach the terminal tolerance is 10, and the average number of

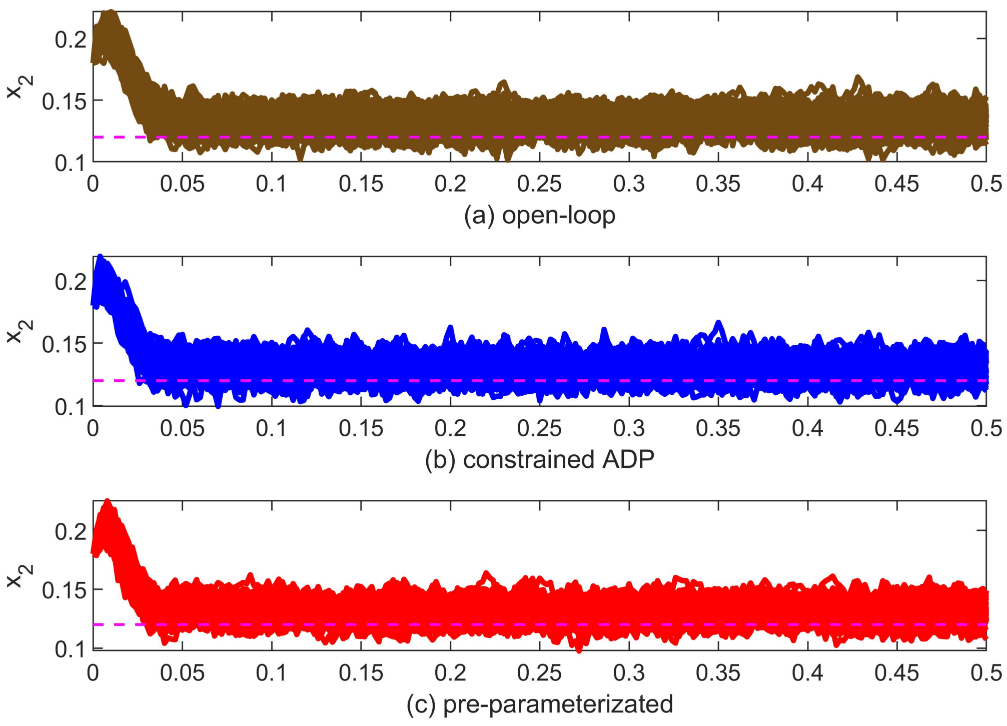

Figure 5.2: Time evolution of real system state x_2 for all three policies in 100 Monte Carlo simulations. The constraint is represented by the dashed magenta line

iterations for obtaining the open-loop controller by IPOPT is 8, which implies that the proposed algorithm is as efficient as solving a nonlinear programming problem by IPOPT.

5.7 Summary

In this chapter, we designed an output-feedback stochastic MPC controller for chance-constrained nonlinear systems. The stochastic optimal control problem was solved in a stochastic dynamic programming fashion, and the output-feedback control was performed with the extended Kalman filter. The information state in SDP is summarized by a Gaussian belief dynamics to avoid "curse of dimensionality," and the stochastic optimization problem is reformulated as a deterministic one by this Gaussian belief dynamics. We proposed a constrained SADP algorithm to solve the SDP, which has a Q-superlinear local convergence rate. Simulation results showed that our proposed algorithm can achieve good control performance and computational efficiency.

Table 5.2: Comparison of the constraint violation rate based on 100 Monte Carlo simulations (%)

Control policy	$p_{vio} = 1$	$p_{vio} = 5$	$p_{vio} = 20$
Open-loop	0.41	2.04	4.46
Constrained SADP	0.98	4.88	19.90
PP	0.69	4.11	17.20

Table 5.3: Average computation time per update (ms)

Control policy	$N = 5$	$N = 20$	$N = 100$
Open-loop	0.77	5.04	31.93
Constrained SADP	1.33	1.71	3.09
PP	3.42	19.55	110.81

Chapter 6

Bayesian Reinforcement Learning for Unknown Model

6.1 Introduction

In the past few decades, reinforcement learning (RL, or adaptive optimal control, see Sutton, Barto, and Williams (1992)) has emerged as an elegant and popular technique to handle optimal decision-making problems when the model is unknown (Kaelbling, Littman, and Moore (1996); Wiering and Van Otterlo (2012)). RL is a class of learning problems in which an agent interacts with an environment, with the goal of finding a policy, to optimize some measure of the long-term performance (Sutton and Barto (2018)). A major challenge in RL is the balance between the need to explore the space of all possible policies and the desire to focus data collection towards trajectories that yield better outcomes. This is known as the *exploitation-exploration* trade-off, which is also called the *dual control effect* in the context of control theory (Feldbaum (1961)).

A principled solution to this problem is offered by Bayesian reinforcement learning (BRL), which uses Bayesian inference to incorporate information into the learning process (Duff (2002)). In BRL, a probability distribution (belief) over the dynamics and cost of the environment can be used not just to simulate and plan trajectories, but also to reason about changes to the belief from future observations, and their influence on future decisions.

Model-based BRL is an extension of model-based RL, that explicitly maintains a posterior over the model parameters and uses this posterior to select actions. Initial work on model-based BRL appeared in the control literature, under the topic of dual control (Feldbaum (1961); Filatov and Unbehauen (2000)). Both model-based BRL and dual control are intended to explicitly represent uncertainty over the model parameter, then the optimal policy can be selected. Duff (2002) proposed a general framework for model-based BRL, called Bayes-Adaptive MDP, which is an extension of the conventional MDP model involving the idea of posterior information state.

The main difficulty in model-based BRL is the learning efficiency, which relies on how "good" the control policy is in the learning process. In principle, a "good" policy should satisfy the Bayes optimality under the current learned belief (Ghavamzadeh and Engel (2006)), but the difficulty of solving stochastic dynamic programming prevents obtaining such a policy. Poupart, Vlassis, Hoey, and Regan (2006) proposed an offline algorithm called BEETLE to solve Bayes-Adaptive MDP. They used a multi-variate polynomial functions to represent the value function, thus the stochastic dynamic programming can be solved analytically. The main disadvantage of this problem is the number of terms in the polynomials increases exponentially with the planning horizon, which is a big challenge even in the offline case. In Strens (2000) a Bayesian dynamic programming method was proposed to find the optimal policy. In this method, a model is sampled from the posterior belief, and this model is used in dynamic programming to select actions. Convergence to the optimal policy is achievable by sampling the model from the full posterior over uncertainty. However, the convergence could be very slow, since it needs to explore the full belief space. Instead of directly solving stochastic dynamic programming, Dearden, Friedman, and Andre (1999) proposed a value of information heuristic method, which involved a value of information term in the reward, to estimate the expected improvement in policy following an exploration action, but this heuristic approximation may provide only a very limited view of potential information gain of certain actions.

The model-based BRL has a strong relation to SMPC, or we can say model-based BRL is a more general form of stochastic optimal control with additional model learning parts. When the system model is obtained, they all solve similar stochastic dynamic programming problems. This means that the algorithm studied in SMPC can be well applied to solve model-based BRL. In the previous chapters, we have introduced the SMPC algorithms for linear systems, and nonlinear systems with Gaussian assumption. In this chapter, we will focus on a more general problem, i.e., nonlinear non-Gaussian systems with unknown models. A novel sample-based BRL framework is proposed for solving the planning problem of current learned belief. We used the particle filter to estimate the belief update, and a sample-based method is adopted to solve the stochastic dynamic programming with chance constraints. The numerical example reveals that the proposed method can achieve good learning efficiency and control performance in both linear Gaussian and nonlinear non-Gaussian systems.

The rest of this chapter is organized as follows. Section 6.2 discusses the problem settings used in this chapter. Section 6.3 introduces the model-based BRL and some existing methods. Section 6.4 discusses the Gaussian process dynamic model for model learning. Section 6.5 presents the main algorithm, i.e., sample-based BRL. Section 6.6 presents our simulation results. Section 6.7 summarizes this chapter.

6.2 Problem Statement

6.2.1 Partially observable Markov decision process

In this section, we consider a partially observable Markov decision process (POMDP) to be defined by the tuple (X, A, O, T, Ω, R) (Kaelbling, Littman, and Cassandra (1998)):

- $X \subset \mathbb{R}^{n_x}$: A set of states.
- $A \subset \mathbb{R}^{n_u}$: A set of control actions.
- $O \subset \mathbb{R}^{n_y}$: A set of observations.
- $T(\cdot|x, a)$: Probability distribution over next state, conditioned on action a being taken at state x.
- $\Omega(\cdot|x)$: Probability distribution over possible observations, conditioned on action a being taken to reach state x where the observation is perceived.
- $R(\cdot|x, a)$: Random variable r representing the reward obtained when action a is taken in state x.

Since the state is not directly observed, at stage k the agent must rely on the recent information history of actions and observations $(o_k, ..., o_0, u_{k-1}, ..., u_0)$ to infer a distribution over states, which is called *belief* (or *information state* see Section 2.2.2).

Since the state is Markovian (also known as Markov chain), which means current belief over the states requires knowledge of the previous belief state, the action taken, and the current observation. In Section 2.2.2, we have introduced that the belief update of a dynamic system can be obtained by recursive Bayesian estimation (2.25). For convenience, we denote the belief transition equation as $b_{k+1} = \tau(b_k, a, o)$.

A Markovian belief state allows a POMDP to be formulated as a Markov decision process where every belief is a state. The resulting belief MDP will thus be defined on a continuous state space. Formally, the belief MDP is defined as a tuple (B, A, O, τ, r):

- B: A set of belief states over the POMDP states.
- A: A same set of action as for the original POMDP.
- O: A same set of observations as for the original POMDP.
- $\tau(b, a, o)$: Belief state transition function.
- $r(b, a)$: Reward function on belief states.

Among these, the belief MDP reward function r is the expected reward from the POMDP reward function over the belief state distribution:

$$r(b,a) = \mathbf{E}[\int_X b(x)R(x,a)dx] \tag{6.1}$$

6.2.2 Control target

The goal of the control, when modeled as a belief MDP (or original POMDP), is to choose a policy π which maximize the expected sum of rewards over the finite horizon N. Beside the reward, the states are also constrained by the chance constraints

$$P[c(x_k) \leq 0] \geq 1 - p_{vio}, \tag{6.2}$$

where $c : \mathbb{R}^{n_x} \to [0, \infty)$ denotes the stage constraints function, and $p_{vio} \in (0, 0.5]$ denotes the maximum allowed probability of state constraint violation.

We can also give the Bellman equation at time k defined by belief state as (similar to Bellman equation in Section 5.3.1):

$$\begin{aligned} V_k^*(b_k) = \min_{a_k} \mathbf{E}[r(b_k, a_k) + \int_O f(o_{k+1}|b_k, a_k)V_{k+1}^*(b_{k+1})] \\ \text{s.t. } b_{k+1} = \tau(b_k, a_k, o_{k+1}) \\ P[c(x_k) \leq 0] \geq 1 - p_{vio} \end{aligned} \tag{6.3}$$

where

$$f(o_{k+1}|b_k, a_k) = \int_S \int_S \Omega(o_{k+1}|x_{k+1})T(x_{k+1}|x_k, a_k)b_k dx_k dx_{k+1}.$$

For our problem of optimal control, we assume the dynamics can be expressed in the following form:

$$\begin{aligned} x_{k+1} &= T(x_k, a_k) + w_k \\ o_k &= \Omega(x_k) + v_k \end{aligned} \tag{6.4}$$

where w_k and v_k are zero-mean Gaussian noise, i.e., $w_k \sim \mathcal{N}(0, \Sigma_w)$ and $v_k \sim \mathcal{N}(0, \Sigma_v)$. The functions $T(\cdot)$ and Ω are unknown deterministic function that returns the next state and observations respectively. The reward function $R(x_k, a_k)$ is assumed known in this section, but the results in this section can be easily extended to unknown case via using additional learning model to reward function.

6.3 Preliminaries of Model-based Bayesian Reinforcement Learning

The basic idea of model-based BRL is to explicitly learn a posterior distribution (belief) over the model parameters θ, i.e. $b(\theta)$ and use this posterior to select actions. Actions can be selected for both exploration and exploitation.

In the BRL framework, exploration and exploitation are naturally balanced in a coherent mathematical framework. Policy is selected over the full belief, including the stochastic part. More specifically, any policy that minimizes the stochastic Bellman equation (6.3) is called a *Bayes-optimal* policy (Ghavamzadeh and Engel (2006)). In general, the cost of a Bayes-optimal policy is higher than the optimal policy solved by Bellman equation defined by exact states, because it may require additional actions to do "exploration". We first introduce some existing methods in model-based BRL.

Bayesian dynamic programming (Bayesian DP) method, closely related to Thompson sampling (Thompson (1935)), was proposed by Strens (2000). For a given prior belief P_{prior}, we first run a random policy to collect data set $\hat{D}$, which includes past actions and observations, then update prior belief to get posterior belief P_{post}. At each episode, we sample a model from the posterior belief P_{post}, solve this model using dynamic programming, and use the solved model to select actions. Models are re-sampled and updated at the end of an episode. The Bayesian DP method is summarized in Algorithm 6.1. The approach is simple to implement and does not rely on any heuristics. Convergence to the optimal policy is achievable by sampling the model from the full posterior over uncertainty. From Figure 6.1, we can see that for a system with little uncertainty or near deterministic, the convergence will be fast since we do not need large samples to cover the full posterior over uncertainty. However, the convergence could be very slow with systems with large uncertainties.

The Bayesian DP method does not explicitly consider the posterior uncertainty for a Bayes-optimal policy in each trial. Abbasi-Yadkori and Szepesvári (2015) proposed a "lazy" version of Bayesian DP, they updated the belief when the solved policy with a sample makes improvement on the Bayes optimality, but they still did not provide some good approach to find a better policy by considering the uncertainties. In this chapter, our main goal is to derive an algorithm that can find a better policy based on the current belief to improve learning efficiency.

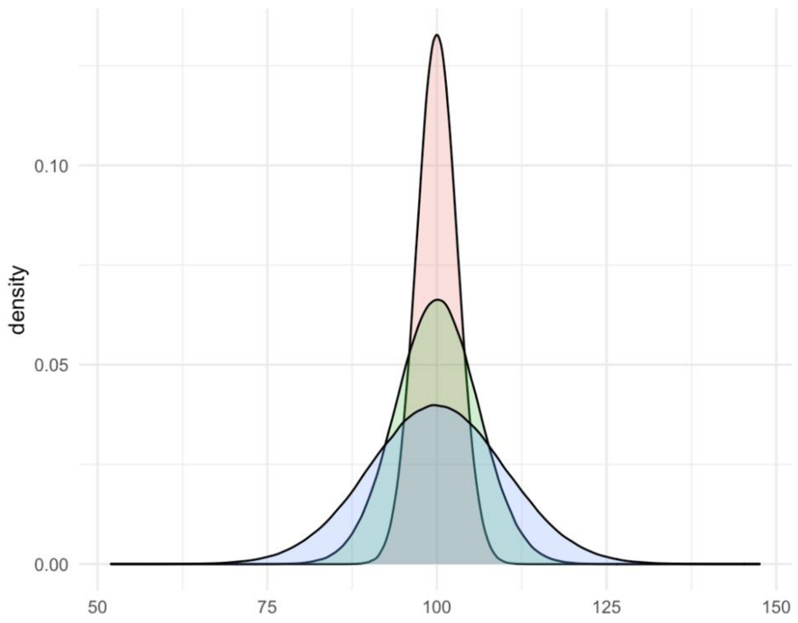

Figure 6.1: Probability distributions with different levels of uncertainties

Algorithm 6.1: Bayesian dynamic programming

```
Initialization:
   prior belief $P_{\text{prior}}$ ;
   episode number $i = 0$ ;
Run random policy to collect data set $\hat{D}$ $P_{\text{post}} \leftarrow P_{\text{prior}}$ ;
for $i = 1, 2, ...$ do
   Sample $\hat{\theta}$ from $P_{\text{post}}$;
   Solve dynamic programming to obtain optimal policy of this sample $\pi^*$ ;
   Collect data set $\hat{D}$ and update $P_{\text{post}}$ ;
end
```

6.4 Gaussian Process Dynamic Model Learning

6.4.1 Gaussian process dynamic model

In this section, We will introduce a Bayesian approach to modeling dynamics. Gaussian process dynamic model (GPDM) is fully defined by a set of low dimensional representations of the training data, with both dynamics and observation mappings learned from Gaussian process (GP) regression (J. M. Wang, Fleet, and Hertzmann (2005)). The dynamic mapping $X \times A \to X$ and observation mapping $X \to O$ are defined as linear combinations of nonlinear basis functions

$$x_{k+1} = \sum_i \mathbf{b}_i \phi(x_k, a_k) + n_x \tag{6.5a}$$

$$o_k = \sum_i \mathbf{c}_i \psi(x_k) + n_o. \tag{6.5b}$$

Here, $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, ...]$ and $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, ...]$ are weights for basis function ϕ_i and ψ_j, n_x and n_o are zero-mean Gaussian noise. The GPDM is obtained by marginalizing out the parameters of the two mappings, and optimizing the latent coordinates of training data.

With an isotropic Gaussian prior on the columns of $\mathbf{C}$. Marginalizing over observation function (6.5b) be done in closed form (MacKay, Mac Kay, et al. (2003); Neal (2012)) to yield a multivariate Gaussian likelihood

$$p(\mathbf{O}|\mathbf{X}, \bar{\beta}) = \frac{|\mathbf{W}|}{\sqrt{(2\pi)^{Nn_y}|\mathbf{K}_O|^{n_y}}} \times \exp(-\frac{1}{2}\mathrm{tr}(\mathbf{K}_O^{-1}\mathbf{O}\mathbf{W}^2\mathbf{O}^{\mathrm{T}})), \tag{6.6}$$

where $\mathbf{O} = [o_1, ..., o_N]$ is a sequence of N observations, $\mathbf{K}_o$ is a kernel matrix, and $\bar{\beta} = [\beta_1, \beta_2, ..., \mathbf{W}]$ are the kernel parameters. The matrix $\mathbf{W}$ is a scaling matrix $\mathbf{W} := \mathrm{diag}[W_1, ..., W_{n_o}]$ to account for different variance in observed data dimensions. The elements of kernel matrix are defined by a kernel function $(\mathbf{K}_O)_{i,j} = k_O(x_i, x_j)$. For the unknown mapping, $X \rightarrow O$, the radial basis function (RBF) kernel is always used

$$k_O(x, x') = \beta_1 \exp(-\frac{\beta_2}{2}\|x - x'\|^2) + \beta_3^{-1}\delta_{x,x'} \tag{6.7}$$

where hyperparameter α_1 denotes the output variance, β_2 denotes the smoothness of the function, β_3^{-1} denotes the variance of the additional noise n_o, and δ is a Dirac delta function.

The dynamic mapping (6.5a) is conceptually similar but need to consider the Markov property. As above, we form the joint probability density over the states and the dynamic weights $\mathbf{B}$ in (6.5a). We then marginalize over the weights $\mathbf{B}$ to get the probability density over states:

$$p(\mathbf{X}|\mathbf{A}, \bar{\alpha}) = \frac{p(x_0)}{\sqrt{(2\pi)^{N(n_x+n_u)}|\mathbf{K}_Z|^{n_x+n_u}}} \times \exp(-\frac{1}{2}\mathrm{tr}(\mathbf{K}_Z^{-1}\mathbf{X}\mathbf{X}^{\mathrm{T}})), \tag{6.8}$$

where $\bar{\alpha} = [\alpha_1, \alpha_2, ...]$ is a vector of kernel hyperparameters, $p(x_0)$ is the initial state distribution, $\mathbf{X} = [x_1, ..., x_N]$. The kernel matrix $\mathbf{K}_Z$ is constructed by state and action data, where $\mathbf{Z} = [z_1, ..., z_N]$ and $z_i = [x_i, a_i]$. The "linear + RBF" kernel is used here (J. M. Wang et al. (2005)):

$$k_Z(z, z') = \alpha_1 \exp(-\frac{\alpha_2}{2}\|z - z'\|^2) + \alpha_4 z^{\mathrm{T}} z' + \alpha_3^{-1}\delta_{z,z'} \tag{6.9}$$

where the additional hyperparameter α_4 compared with (6.7) denotes the output scale of the linear term.

We assume the hyperparameters subject to uniform distribution, i.e., $p(\bar{\alpha}) \propto \prod_i \alpha_i^{-1}$ and $p(\bar{\beta}) \propto \prod_i \beta_i^{-1}$. Together, we can get a general model for action-observation/state sequence

$$p(\mathbf{O}, \mathbf{X}, \bar{\alpha}, \bar{\beta}|\mathbf{A}) = p(\mathbf{O}|\mathbf{X}, \bar{\beta})p(\mathbf{X}|\mathbf{A}, \bar{\alpha})p(\bar{\alpha})p(\bar{\beta}). \tag{6.10}$$

6.4.2 Model learning

Learning the GPDM from known information $\mathbf{O}$ and $\mathbf{A}$ entails using numerical optimization to estimate some or all of the unknowns in the model $(\mathbf{X}, \bar{\alpha}, \bar{\beta}, \mathbf{W})$. The training data is obtained from random trials drawn from models beforehand and the online learning process.

Maximum a posteriori (MAP) algorithm, which is proposed in is proposed in J. M. Wang, Fleet, and Hertzmann (2007), is an efficient algorithm to estimate all unknowns. The goal of model learning can be summarized by minimizing the joint negative log-posterior of the unknowns $-\ln p(\mathbf{X}, \bar{\alpha}, \bar{\beta}, \mathbf{W}|\mathbf{O}, \mathbf{A})$, which is given by

$$\mathcal{L} = \mathcal{L}_O + \mathcal{L}_Z + \sum_i \ln \alpha_i + \sum_i \ln \beta_i \tag{6.11}$$

where

$$\mathcal{L}_O = \frac{n_y}{2} \ln |\mathbf{K}_O| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_O^{-1}\mathbf{O}\mathbf{W}^2\mathbf{O}^{\mathrm{T}}) - N \ln |\mathbf{W}|$$
$$\mathcal{L}_Z = \frac{n_x + n_u}{2} \ln |\mathbf{K}_Z| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_Z^{-1}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}) + \frac{1}{2} z_0^{\mathrm{T}} z_0$$

The detailed MAP algorithm can be found in J. M. Wang et al. (2007). The resulting estimation of unknowns $(\mathbf{X}, \bar{\alpha}, \bar{\beta}, \mathbf{W})$ is then used to construct the transition and observation function. The probability distribution over the next state can be represented by $p(x_{k+1}|x_k, a_k) \sim \mathcal{N}(\mu_k^z(z_k), \Sigma_k^z(z_k))$, where

$$\begin{aligned} \mu^z(z) &= \mathbf{Z}^{\mathrm{T}}\mathbf{K}_Z\mathbf{k}_Z(z) \\ \Sigma^z(z) &= k_Z(z, z) - \mathbf{k}_Z(z)^{\mathrm{T}}\mathbf{K}_Z^{-1}\mathbf{k}_Z(z)^{\mathrm{T}}. \end{aligned} \tag{6.12}$$

Here $\mathbf{k}_Z(z)$ is a vector denotes the covariance between working point and training set, i.e., containing $k_Z(z, z_i)$ in the ith entry, where z_i is the ith training vector.

The probability distribution over observation is $p(o_k|x_k) \sim \mathcal{N}(\mu_k^o(x_k), \Sigma_k^o(x_k))$, where

$$\begin{aligned} \mu^o(x) &= \mathbf{O}^{\mathrm{T}}\mathbf{K}_O\mathbf{k}_O(x) \\ \Sigma^o(x) &= k_O(x, x) - \mathbf{k}_O(x)^{\mathrm{T}}\mathbf{K}_O^{-1}\mathbf{k}_O(x)^{\mathrm{T}}, \end{aligned} \tag{6.13}$$

and $\mathbf{k}_O(x)$ is also a vector denotes the covariance between working point and training set. Then, the probability distribution $p(x_{k+1}|x_k, a_k)$ and $p(o_k|x_k)$ with moments (6.12) and (6.13) can be used to solve the planning problem in a model-based learning scheme.

6.5 Sample-based Bayesian Reinforcement Learning

In the context of a model-based Reinforcement learning method. The planning problem is solved after a model is learned. In this section, we will discuss the planning algorithm based on the previously learned GPDM.

An important insight in planning of stochastic system is the posterior belief (information state) update should be used while solving Bellman equation, as we have done in Chapter 5. This can provide a dual control effect, or in machine learning field, we call this a compromise between exploration and exploitation (Gupta, Smith, and Shalley (2006)). In Chapter 5, an extended Kalman filter is used to approximate the belief update, but with a Gaussian assumption. In this section, we use the particle filter to estimate the belief and solve the Bellman equation via sample-based method without any additional assumption.

6.5.1 Belief update via particle filter

Particle filter (PF) uses a set of particles to represent the posterior distribution of a stochastic process given the noisy and/or partial observations. The advantages of particle filter are the state-space model can be nonlinear and the initial state and noise distributions can take any form.

In Section 2.2.2.3, we have introduced the basic idea of PF and sequential importance sampling (SIS). At stage k, we have the belief $b_k(x_k)$. By denoting the posterior estimate of state as $\hat{x}_{k|k}$, and prior estimate of state as $\hat{x}_{k+1|k} \sim p(x_{k+1}|x_k, a_k)$, the update of this estimated state is

$$\hat{x}_{k+1|k+1} \approx f(\hat{x}_{k+1|k}, o_{k+1}) = \sum_{j=1}^{N} W_{k+1}^{j} \hat{x}_{k+1|k}^{j}, \quad j = 1, ..., M \tag{6.14}$$

where j denotes the jth particle randomly generate from a simple proposal distribution $p(x_{k+1}|x_k, a_k)$, M is the number of particles, and W_{k+1}^{j} is the weight of the

Algorithm 6.2: Belief update via PF

```
Parameters:
  $M$ number of particles;
Initialization:
  horizon counter $k$ ;
  initial posterior distribution $p(\hat{x}_{k|k})$ ;
  transition distribution $p(x_{k+1}|x_k, a_k)$ ;
  observation distribution $p(o_k|x_k)$ ;
  initial weights $W_k$ ;
Input: observations $o_{k+1}$ ;
Output: deterministic belief update function $f(\cdot)$ ;
for $j = 1 : M$ do
    Compute prior estimate of state $\hat{x}^j_{k+1|k} \sim p(x_{k+1}|x_k, a_k)$;
    Compute weight for each particle by (6.15);
end
Normalize weights $W^j_{k+1}$ by (6.16) ;
Obtain belief update function $f(\cdot)$ from equation (6.14) ;
```

jth particle, which can be updated recursively by the following equation (see Section 2.2.2.3)

$$W^j_{k+1} \propto p(o_{k+1}|x_{k+1})W^j_k. \tag{6.15}$$

After getting all the associated weights, a normalization is applied to make sure the summation of conditional probability equals to one:

$$W^j_{k+1} = \frac{W^j_{k+1}}{\sum_{i=1}^M W^i_{k+1}} \tag{6.16}$$

From the above discussion, we know that once a new observation is known, we can obtain a deterministic update equation of $\hat{x}_{k|k}$, which can be interpreted as an estimation of belief update equation $b_{k+1} = \tau(b_k, a_k, o_{k+1})$. The whole PF algorithm is summarized in Algorithm 6.2.

6.5.2 Sample-based constrained dynamic programming

The belief update equation (6.14) obtained in the previous section can be used to construct the Bellman equation (6.3). However, it is still a stochastic dynamic programming problem that remains difficult to solve. In Chapter 5, we have introduced an approximate dynamic programming algorithm based on the Gaussian assumption. In this chapter, the belief obtained by PF is no longer Gaussian, so we propose a sample-based constrained dynamic programming in this general case.

The basic idea of the sample-based method is to compute an optimal policy that is feasible under K of "samples" of the uncertainty. Note that the uncertainties in Bellman equation is arisen from the observations, i.e.,

$$\hat{x}_{k+1|k+1} = f(\hat{x}_{k|k}, a_k, o_{k+1}) \tag{6.17}$$

where $p(o_k|x_k) \sim \mathcal{N}(\mu_k^o(x_k), \Sigma_k^o(x_k))$ from (6.13).

Drawing K samples from the observation distribution for the whole prediction steps $k = 0, ..., N-1$, they are combined into full-horizon samples, also called scenarios. Thus, for each sample j, we can have a deterministic belief update equation

$$\bar{x}_{k+1} = f_j(\bar{x}_k, a_k), \quad j = 1, ..., K \tag{6.18}$$

where $\bar{x}$ is a deterministic state, and the Bellman equation (6.3) can also be reformulated into a deterministic one with these samples

$$\begin{aligned} V_k^*(\bar{x}_k) = \min_a \; & r(\bar{x}_k, a_k) + V_{k+1}^*(\bar{x}_{k+1}) \\ \text{s.t. } & \bar{x}_{k+1} = f_j(\bar{x}_k, a_k), \quad j = 1, ..., K \\ & c(\bar{x}_{k+1}) \leq 0 \end{aligned} \tag{6.19}$$

Note that for each sample, stochastic nature is removed, so we just need to directly remove the stochastic description of chance constraint. We can find the approximated local policy for each deterministic constrained Bellman equation with given initial state x_0 by constrained SADP algorithm 5.2 introduced in Section 5.4.2 (this can be done in parallel).

The remaining problem is that this method renders the stochastic optimization problem into multiple deterministic systems by substituting particular scenarios. This significantly simplifies the problem. However, these samples lead to a randomization of the control policy. More specifically, the randomized policy can not meet the requirement of a desired constraint satisfactory level in chance constraints.

This randomization effect can be mitigated by a posterior scenario removal (Campi and Garatti (2008, 2011)). A posterior scenario removal means to remove the state constraints of L scenarios after the outcomes of all samples have been observed. For a specific constraint violation level p_{vio}, L has to be varied in proper combination with K.

Definition 6.1 (Admissible sample-removal pair). *An admissible sample-removal pair is considered to be a combination (K, L) that does not exceed the desired constraint violation level p_{vio}.*

The details of how to determine the admissible sample-removal pair can be found in Campi and Garatti (2011), but a fundamental result is the larger K you choose, the more accurate result you can get. Here, we directly use the existing rule of choosing admissible sample-removal pair (K, L) in Campi and Garatti (2011), i.e., the sample number K and removal number L have a fixed mapping $L = \Omega(K)$ for a specific problem setting, Hence, if either K or L is fixed, an admissible sample-removal pair (K, L) can be determined. Moreover, if L is fixed there always exists a K large enough to generate an admissible pair (K, L).

After K scenarios have been sampled the selection of the L removed scenarios is performed by choosing a practical algorithm from below (Schildbach et al. (2012)):

- Optimal removal: The deterministic Bellman equation (6.19) is solved for all possible combinations of choosing L out of K scenarios. Then the combination that yields the lowest cost function value of all the solutions is selected. This requires the solution to K choose L instances of the optimization problem, a complexity that is usually prohibitive for larger values of L.

- Removal via multipliers: For each sample, the deterministic Bellman equation (6.19) for all remaining sampled constraints is solved (initially, that is all sampled constraints). Then the constraint in scenario associated with the highest Lagrange multiplier is removed, and the procedure is repeated till L scenarios are removed.

Remark 6.1. *If the violated constraints are selected via optimal removal, then the solution to the sample-based dynamic programming problem with violated constraints is able to arbitrarily approximate that of the original stochastic problem, as the number L increases (Calafiore (2010)).*

Even though the optimal removal can approximate the problem precisely, it needs quite large computation cost. Thus, in a practice implementation, the removal via multipliers algorithm is always used.

After obtaining the solution from all K samples, we can calculate the policy by combing all sampled solutions. More specifically, optimal policy is chosen by maximum likelihood estimation (MLE) of parameters. In our algorithm, we obtain K local linear feedback policies from all scenarios as the form of

$$u_k^i = G_k^i x_k^i + v_k^i, \ i = 1, ..., K, \tag{6.20}$$

where G_k^i and v_k^i are parameters of policy in ith scenario. Then the MLE of parameters in this linear policy can be easily obtained by take the average values of all scenario solutions.

6.5.3 Bayesian reinforcement learning algorithm

Through model learning, the belief update, and the DP-based planning algorithm, we can give the entire sample-based BRL framework.

At initialization, we first apply a random policy on the agent and collect data, which is used to initialize the GPDM. In each trial, according to the current learned GPDM, K samples are first generated according to the observation distribution $p(o_k|x_k)$, and K different deterministic brief update equations are generated through Algorithm 6.2. Then, the optimal control law is obtained by solving the sampling-based dynamic programming, which is the "optimal" control policy under the stochastic model currently learned. Finally, apply this control policy to the agent and collect a new sequence of data until a satisfactory control effect is obtained. The whole algorithm is summarized in Algorithm 6.3

6.6 Case Study

6.6.1 Numerical example settings

We tested our algorithm in two cases. The first case is a discrete-time linear Gaussian room temperature system used in Section 4.7 with additional observation function:

$$\begin{aligned} x_{k+1} &= Ax_k + Ba_k + w_k \\ o_k &= Cx_k + v_k \end{aligned} \tag{6.21}$$

where the parameters of the system are taken from Oldewurtel et al. (2013) as follows:

$$A := \begin{bmatrix} 0.8511 & 0.0541 & 0.0707 \\ 0.1293 & 0.8635 & 0.0055 \\ 0.0989 & 0.0032 & 0.7541 \end{bmatrix},$$
$$B := \begin{bmatrix} 0.0035 \\ 0.0003 \\ 0.0002 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The noises w_k and w_k are all zero-mean Gaussian white noises with distribution $w_k \sim \mathcal{N}(0, 0.1^2 I)$ and $v_k \sim \mathcal{N}(0, 0.1^2 I)$.

Algorithm 6.3: Sample-based BRL

```
Parameters:
  $K$ number of samples;
  $L$ number of constraint removal;
Initialization:
  trial counter $i$ ;
  a prior of GPDM $p(x'|x,a)$ and $p(o|x)$ ;
  initial policy $\pi_{\theta_a}$;
Apply random policy on agent, collect date $\tilde{\mathbf{D}} = (\mathbf{O}, \mathbf{A})$, update GPDM via
  MAP algorithm ;
for $i = 1, 2, ...$ do
    Generate $K$ samples from observation distribution ;
    Derive $K$ deterministic belief equation by Algorithm 6.2 ;
    while $L > 0$ do
        Solve $K$ deterministic Bellman equations (6.19) ;
        Remove the constraint of the particle associated with the highest
          Lagrange multiplier ;
        $L \leftarrow L - 1$ ;
        $K \leftarrow K - 1$ ;
    end
    Compute the optimal policy $\pi^*$ via MLE ;
    Implement optimal policy $\pi^*$ to agent ;
    Collect new date set $\tilde{\mathbf{D}}$, update GPDM;
end
```

The objective function of this system was defined as $r(x_k, a_k) = a_k^{\mathrm{T}} a_k$ with planning horizon $N = 150$. State constraint was treated as a chance constraint:

$$P[x_1 \geq 21] \geq 1 - p_{vio}.$$

Control input also had limitations imposed by the actuator, $0 \leq u \leq 45$.

The second case is a nonlinear, non-Gaussian continuous-time CSTR system used in Section 5.6. The reactor has a constant volume, and its dynamics are described as follows:

$$\begin{aligned} \dot{x}_1 &= -k_1 x_1 - k_3 x_1^2 + (x_f - x_1)a \\ \dot{x}_2 &= -k_1 x_1 - k_2 x_2 + x_2 a, \end{aligned} \tag{6.23}$$

where the value of these parameters are $k_1 = 50$, $k_2 = 100$, $k_3 = 10$, and $x_f = 10$. This state is partially observed using the following equation:

$$o = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{6.24}$$

The continuous-time system (6.23) was discretized with a sampling time of 0.002s. Stochastic noises w_k and v_k were assumed to be added on the dynamic (6.23) and observation equation (6.24) respectively, and they are subject to zero-mean uniform distribution $w_k \sim \mathcal{U}(0, 0.2^2 I)$ and $v_k \sim \mathcal{U}(0, 0.2^2 I)$.

The objective function of this system is of the quadratic form $x_k^{\mathrm{T}} Q x_k + a_k^{\mathrm{T}} R a_k$ with

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1,$$

and the planning horizon is $N = 150$. x_2 is required to satisfy the chance constraint,

$$P[x_2 \geq 0.12] \geq 1 - p_{vio}.$$

6.6.2 Simulation analysis

For comparison, we also present the simulation results for three methods.

1. Proposed sample-based BRL;
2. Constrained SADP (Algorithm 5.2), which directly solve information-state-based DP with Gaussian assumption;
3. Bayesian DP (Algorithm 6.1).

Note that the constrained SADP algorithm cannot deal with unknown model, so it was also implemented in the Bayesian reinforcement learning framework with GPDM model learning.

We performed the simulation on a laptop computer with a 2.60 GHz Intel Core i7-6700HQ in MATLAB 2020a. All of these methods were coded in the MATLAB environment, and the performance of these methods are evaluated by 100 times Monte Carlo simulations.

6.6.3 Linear Gaussian case

We first tested how the number of samples K affect the performance. Table 6.1 compares the control performance of proposed sample-based BRL with different (K, L) pairs $(N = 150, p_{vio} = 10\%)$. All (K, L) pairs are admissible sample-removal pairs, and Figure 6.2 shows the time evolution of x_1 with $(K, L) = (5706, 100)$ after the model learning process converges. The results show that when K (or L) increases, we can obtain less conservative control performance, i.e., lower cost and closer to the

Table 6.1: Control performance of proposed sample-based BRL with different (K, L) pairs $(N = 150, p_{vio} = 10\%)$

(K, L)	Cost($\times 10^4$)	Constraint violation(%)
(495,20)	5.44	7.46
(1123,50)	5.40	8.03
(5706,100)	5.22	9.45

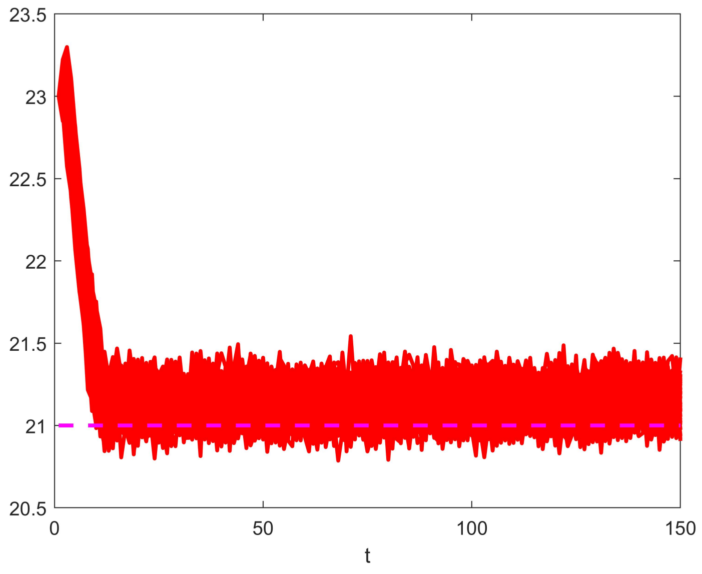

Figure 6.2: Control performance of proposed sample-based BRL with sample-removal pair $(K, L) = (5706, 100)$

Table 6.2: Comparison of sample-based BRL, constrained SADP and Bayesian DP $(N = 150,\ p_{vio} = 10\%)$

	Cost($\times 10^4$)	Constraint violation(%)	Trials
Sample-based BRL $(K = 5706)$	5.22	8.45	7
Constrained SADP	5.21	9.86	6
Bayesian DP	6.49	–	16

set value of constraint violation level p_{vio}. Meanwhile, the results also reveal that our proposed sample-based BRL works well when we use relatively large samples.

Table 6.2 compares the average control performance of proposed sample-based BRL, constrained SADP, and Bayesian DP methods, and the time evolution of x_1

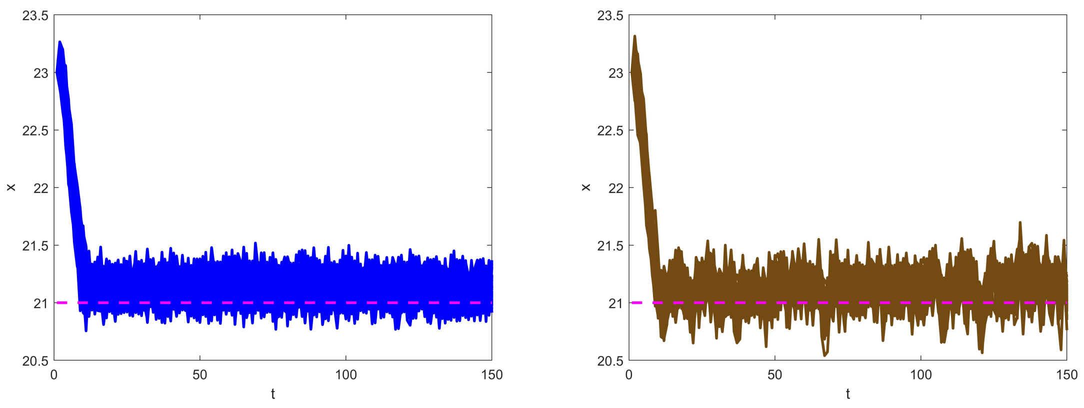

Figure 6.3: Time evolution of x_1 using constrained SADF (left) and Bayesian dynamic programming (right)

using constrained SADP and Bayesian DP are given in Figure 6.3. The result shows that the Bayesian DP method obviously needs more trials. Moreover, the total cost of Bayesian DP method is significantly higher than the other two methods, and the Bayesian DP cannot even handle chance constraint since just one sample ($K = 1$) is used. The control performance of sample-based BRL proposed in this chapter is slightly worse than the constrained SADP method, i.e., higher cost and more conservative constraint violation level. This is because the tested system is linear Gaussian, which can satisfy the assumptions in constrained SADP, while the sample-based BRL is a more general method for arbitrary systems.

6.6.4 Nonlinear non-Gaussian case

In the nonlinear non-Gaussian case, we first tested the model learning effect via GPDM. Figure 6.4 shows the time evolution of x_1 under a given policy after the learning process converge to the set accuracy. The result shows that the model learned by GPDM can track the nonlinear non-Gaussian system very well, which provides a powerful guarantee for the performance of model-based Bayesian reinforcement learning.

Table 6.3 compares the control performance of the three methods in the nonlinear non-Gaussian system. The result shows that Bayesian DP failed to converge in this case. Constrained SADP with Gaussian assumption can not satisfy the constraint violation requirement this time, which means its control performance is unacceptable in this case. The sample-based BRL obtained a reasonable control performance with samples $K = 2210$, and also provided the possibility to get a better control effect by

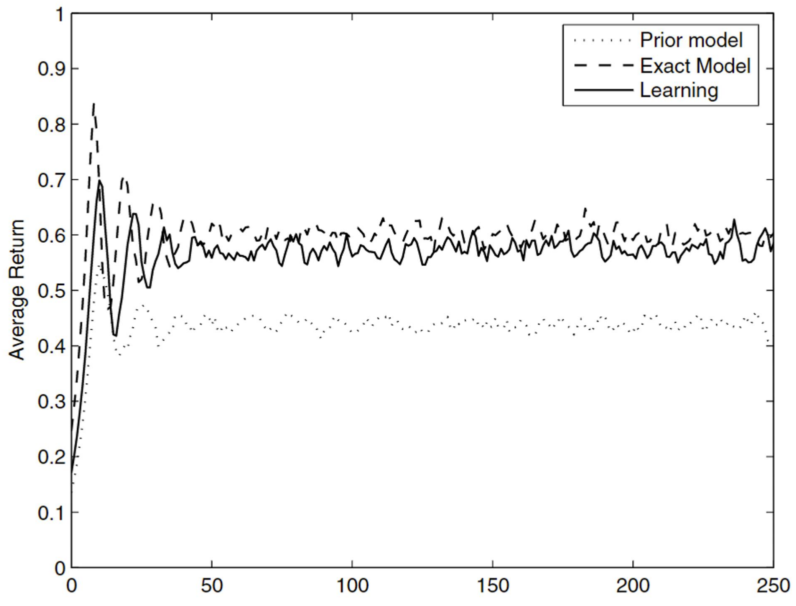

Figure 6.4: Response of x_2 under a given policy by learned GPDM

Table 6.3: Comparison of sample-based BRL, constrained SADP and Thompson sampling in nonlinear non-Gaussian system ($N = 150,\ p_{vio} = 10\%$)

	Cost	Constraint violation(%)	Trials
Sample-based BRL ($K = 2210$)	902	7.95	20
Constrained SADP	766	23.82	16
Bayesian DP	–	–	–

increasing the number of K, which can be treated as a design parameter that can be adjusted according to requirements.

6.7 Summary

In this chapter, we designed a sample-based Bayesian reinforcement learning method for general systems with unknown model. The Gaussian process dynamic model was used to model the system in a model-based learning fashion. The inner planning problem was constructed by the stochastic Bellman equation with respect to the posterior belief. The posterior belief update equation was calculated by a particle filter algorithm and formulated as a conditional distribution of observations. We proposed a sample-based algorithm for solving stochastic dynamic programming. The whole sample-based Bayesian reinforcement learning algorithm can obtain a good control performance. Simulation results showed that our proposed algorithm can achieve good learning efficiency and control performance in both linear and non-linear systems.

Chapter 7

Conclusions

7.1 Summary of Contributions

In this thesis, we have proposed several algorithms of SMPC with chance constraints for different system settings. SMPC explicitly considers the stochastic nature of uncertainties and finds an optimal policy to obtain a closed-loop control performance. However, policy optimization is still challenging due to stochastic settings and chance constraints, and the efficient propagation method for stochastic uncertainties through the system dynamics is a key challenge in SMPC of nonlinear systems. This thesis has consistently tackled these challenges to establish efficient SMPC algorithms for different systems.

In Chapter 3, we proposed an efficient parameterization method called simplified affine disturbance feedback parameterization, it is proved to be equivalent to a state feedback policy, and it obtained a trade-off between computation cost and control performance. In Chapter 4 we proposed a recursive Riccati interior-point method for directly solving stochastic dynamic programming in linear SMPC case, we proved the global convergence and local Q-superlinear convergence rate of this algorithm, and simulation results showed that this algorithm can achieve ideal control performance and low computational complexity. In Chapter 5, we proposed an approximate dynamic programming algorithm for output-feedback nonlinear SMPC, it is proved to have a Q-superlinear local convergence rate, and this method can achieve both closed-loop control performance and computational efficiency. In Chapter 6, we proposed a sample-based Bayesian reinforcement learning method for systems with unknown model, this method provided a general framework for handling POMDP and pretty good learning efficiency.

7.2 Discussion and Future Work

Finally, we discuss the limitations of our methods and future work for these problems.

The first problem is the treatment of chance constraints. In Chapters 3, 4 and 5, we derived the algorithms based on an analytic approximation of chance constraints, these approximations are often characterized by a quite high level of conservativeness. In addition, these algorithms were derived in linear (or linearization) chance constraint cases, while their extension to the exact nonlinear chance constraints is still largely unexplored. In Chapter 6, the sample-based method was used to deal with chance constraints affected by general-type noise. However, the feasibility and stability properties issues are still open in the sample-based method.

Estimation error in nonlinear case is another problem. In linear case like LQG and problem in Chapter 4, the Kalman filter is the optimal recursive Bayesian estimator. However, in nonlinear system, we used extended Kalman filter in Chapter 4 and particle filter in Chapter 5, these filters will bring a certain level of estimation errors. How to evaluate the effects of estimation errors in control performance need to be considered in future research.

MPC is currently a widely used algorithm in industry, but its disadvantage is that it needs a more accurate model, and BRL considering the unknown model can solve this problem well, so the algorithm combining BRL and SMPC will receive wider attention. In Chapter 6, we used a sample-based method to deal with very general systems, but it brings large computation burdens if we want a higher control accuracy. Therefore, how to reduce the computation time for real-time implementation could be nice future work.

Appendix A

Soft Constraint RRIPM

A.1 Optimality conditions

In Section 4, we have introduced the deterministic Bellman equation with soft variable e_k (4.4):

$$\begin{aligned} V_k(x_k) = \min_{u_k(x_k)} & [\frac{1}{2}x_k^{\mathrm{T}}Qx_k + \frac{1}{2}u_k^{\mathrm{T}}Ru_k + \rho^{\mathrm{T}}e_k + V_{k+1}(x_{k+1})] \\ \text{s.t. } & x_{k+1} = Ax_k + Bu_k \\ & C_x x_k + C_u u_k + C_c \leq e_k \\ & e_k \geq 0, \end{aligned} \tag{A.1}$$

Similar to (4.12), the Lagrangian of the above constrained optimization problem is

$$V_k(x_k) = \min_{u_k} \max_{s_k} [l(x_k, u_k, e_k, s_k) + V_{k+1}(x_{k+1})], \tag{A.2}$$

where

$$l(x_k, u_k, s_k) = \frac{1}{2}x_k^{\mathrm{T}}Qx_k + \frac{1}{2}u_k^{\mathrm{T}}Ru_k + \rho^{\mathrm{T}}e_k + s_k^{\mathrm{T}}(C_x x_k + C_u u_k + C_c - e_k)$$

and s_k denotes the Lagrange multiplier.

We can also give the "softened" SKKT conditions of based on Lagrangian (A.2) with slack variable y_k:

$$Q_{uu}u_k + Q_{ux}x_k + Q_{us}s_k + Q_u = 0 \tag{A.3a}$$

$$S_k y_k = 0 \tag{A.3b}$$

$$Q_{sx}x_k + Q_{su}u_k + Q_s + y_k - e_k = 0 \tag{A.3c}$$

$$E_k(\rho - s_k) = 0 \tag{A.3d}$$

$$s_k \geq 0,\ y_k \geq 0,\ e_k \geq 0, \tag{A.3e}$$

where $E_k = \mathrm{diag}[e_k]$, and the parameters $(Q_{uu}, Q_{ux}, Q_{us}, Q_{sx}, Q_{su}, Q_s)$ is the same as in equation (4.14).

A.2 Soft Constraint RRIPM Algorithm

At the ith iteration, the trajectories of the last iteration, $(\mathbf{x}^{i-1}, \mathbf{u}^{i-1}, \mathbf{s}^{i-1}, \mathbf{y}^{i-1}, \mathbf{e}^{i}-1)$, then the backward pass, forward pass and outer loop can be summarized as follows.

A.2.1 Backward pass

Initialize: let

$$P_N = Q_N,\ q_N = r_N = 0$$

At stage k, represent the optimization variables by their search direction:

$$\begin{aligned} x_k^i &= x_k^{i-1} + \delta x_k \\ u_k^i &= u_k^{i-1} + \delta u_k \\ s_k^i &= s_k^{i-1} + \delta s_k \\ y_k^i &= y_k^{i-1} + \delta y_k \\ e_k^i &= e_k^{i-1} + \delta e_k. \end{aligned} \tag{A.4}$$

The new trajectory satisfies the perturbed SKKT conditions:

$$Q_{uu}u_k^i + Q_{ux}x_k^i + Q_{us}s_k^i + Q_u = 0 \tag{A.5a}$$

$$S_k^i y_k^i = \mu_{1k}^i \tag{A.5b}$$

$$Q_{sx}x_k^i + Q_{su}u_k^i + Q_s + y_k^i - e_k^i = 0 \tag{A.5c}$$

$$E_k^i(\rho - s_k^i) = \mu_{2k}^i \tag{A.5d}$$

$$s_k^i \geq 0,\ y_k^i \geq 0,\ e_k^i \geq 0, \tag{A.5e}$$

where μ_{1k}^i and μ_{2k}^i are used to smooth the complementarity conditions (A.5b) and (A.5d).

Applying Newton's method to (A.5), we can obtain the following linear equations:

$$\begin{bmatrix} Q_{uu} & Q_{us} & 0 & 0 \\ 0 & Y_k & S_k & 0 \\ Q_{su} & 0 & I & -I \\ 0 & -E_k & 0 & \rho - S_k \end{bmatrix} \begin{bmatrix} \delta u_k \\ \delta s_k \\ \delta y_k \\ \delta e_k \end{bmatrix} = -\begin{bmatrix} Q_{ux} \\ 0 \\ Q_{sx} \\ 0 \end{bmatrix} \delta x_k - \begin{bmatrix} \xi_o^{i-1} \\ \xi_d^{i-1} - \mu_{1k}^i \\ \xi_f^{i-1} \\ \xi_e^{i-1} - \mu_{2k}^i \end{bmatrix} \tag{A.6}$$

where

$$\begin{bmatrix} \xi_o^{i-1} \\ \xi_d^{i-1} \\ \xi_f^{i-1} \\ \xi_e^{i-1} \end{bmatrix} = \begin{bmatrix} Q_{uu}u_k^{i-1} + Q_{ux}x_k^{i-1} + Q_{us}s_k^{i-1} + Q_u \\ S_k^{i-1}y_k^{i-1} \\ Q_{sx}x_k^{i-1} + Q_{su}u_k^{i-1} + Q_s + y_k^{i-1} \\ E_k^i(\rho - s_k^i) \end{bmatrix} \tag{A.7}$$

are the primal and dual infeasiblities of the last iteration.

The parametric system (4.18) can be solved directly to obtain the solution

$$\begin{aligned}\delta u_k &= \eta_{1k}\delta x_k + \theta_{1k}\\ \delta s_k &= \eta_{2k}\delta x_k + \theta_{2k}\\ \delta y_k &= \eta_{3k}\delta x_k + \theta_{3k}\\ \delta e_k &= \eta_{4k}\delta x_k + \theta_{4k}.\end{aligned} \tag{A.8}$$

Writing (u, s) as a function of x,

$$u_k^i = u_k^{i-1} + \eta_{1k}(x_k^i - x_k^{i-1}) + \theta_{1k} = K_u x_k^i + v_u \tag{A.9a}$$
$$s_k^i = s_k^{i-1} + \eta_{2k}(x_k^i - x_k^{i-1}) + \theta_{2k} = K_s x_k^i + v_s \tag{A.9b}$$
$$y_k^i = y_k^{i-1} + \eta_{3k}(x_k^i - x_k^{i-1}) + \theta_{3k} = K_y x_k^i + y_s \tag{A.9c}$$
$$e_k^i = e_k^{i-1} + \eta_{4k}(x_k^i - x_k^{i-1}) + \theta_{4k} = K_e x_k^i + v_e. \tag{A.9d}$$

By substituting it into (4.12), we obtain the expressions for the coefficients of V_{k-1}, which are given by the following Riccati recursion:

$$\begin{aligned}P_{k-1} =& Q_{xx} + K_u^{\mathrm{T}} Q_{uu} K_u + Q_{xu} K_u + K_u^{\mathrm{T}} Q_{ux} + 2K_s^{\mathrm{T}}(Q_{sx} + Q_{su}K_u) - K_s^{\mathrm{T}} K_e\\ q_{k-1} =& Q_x + Q_u K_u + v_u^{\mathrm{T}}(Q_{ux} + Q_{uu}K_u) + (Q_{su}v_u + Q_s)K_s^{\mathrm{T}} +\\ & v_s^{\mathrm{T}}(Q_{sx} + Q_{su}K_u) + K_e^{\mathrm{T}}\rho - K_s^{\mathrm{T}} v_e - K_e^{\mathrm{T}} v_s.\end{aligned} \tag{A.10}$$

The backward pass above is iteratively performed from $k = N - 1$ to $k = 1$ to finish the backward pass.

A.2.2 Forward pass

Starting from $k = 0$, $x_0^i = \bar{x}_0$ is already known by measurement, which means that $\delta x_0 = 0$.

At stage k, the search directions $(\delta u_k, \delta s_k, \delta y_k, \delta e_k)$ are calculated using (A.8). Then the new iteration $(u_k^i, s_k^i, y_k^i, e_k^i)$

$$\begin{bmatrix} u_k^i \\ s_k^i \\ y_k^i \\ e_k^i \end{bmatrix} = \begin{bmatrix} u_k^{i-1} \\ s_k^{i-1} \\ y_k^{i-1} \\ e_k^{i-1} \end{bmatrix} + \alpha_k^i \begin{bmatrix} \delta u_k \\ \delta s_k \\ \delta y_k \\ \delta e_k \end{bmatrix}. \tag{A.11}$$

where the step size is calculated by

$$\begin{aligned}\alpha_k &= \operatorname{diag}\begin{bmatrix}\alpha_f\alpha_y & \alpha_f\alpha_s & \alpha_f\alpha_y & \alpha_f\alpha_e\end{bmatrix}\\ \alpha_s &= \max\{\alpha \in (0,1]) : s_k^{i-1} + \alpha\delta s_k \geq 0\}\\ \alpha_y &= \max\{\alpha \in (0,1]) : y_k^{i-1} + \alpha\delta y_k \geq 0\}\\ \alpha_e &= \max\{\alpha \in (0,1]) : e_k^{i-1} + \alpha\delta e_k \geq 0\}\end{aligned} \tag{A.12}$$

Algorithm A.1: Soft constraint RRIPM

```
Parameters:
    $\sigma \in [0, 1]$ reduction parameter;
    $\alpha_f = 0.995$ fraction-to-the-boundary parameter;
    $(\epsilon_o, \epsilon_d, \epsilon_f, \epsilon_e)$ terminal criteria;
Initialization:
    iteration counter $i = 0$ ;
    known initial state for all $i$, $x_0^i = x_0$ ;
    initial guess $(\mathbf{x}^0, \mathbf{u}^0, \mathbf{s}^0, \mathbf{y}^0, \mathbf{e}^0)$ ;
    central parameter $\boldsymbol{\mu_1}^0$,$\boldsymbol{\mu_2}^0$ ;
    optimality measurements $(\|\boldsymbol{\xi}_o^0\|, \|\boldsymbol{\xi}_d^0\|, \|\boldsymbol{\xi}_f^0\|, \|\boldsymbol{\xi}_e^0\|)$ ;
while $\|\boldsymbol{\xi}_o^i\| > \epsilon_o \vee \|\boldsymbol{\xi}_o^i\| > \epsilon_d \vee \|\boldsymbol{\xi}_f^i\| > \epsilon_f \vee \boldsymbol{\xi}_e^i\| > \epsilon_e$ do
    $P_N \leftarrow Q_N, q_N \; r_N \leftarrow 0.$ ;
    for $k \leftarrow N-1$ to $0$ do
        // Backward Pass
        Solve linear equation (A.6);
        Calculate search direction by (A.8) ;
        $(P_k, q_k, r_k) \leftarrow (P_{k+1}, q_{k+1}, r_{k+1})$ by (A.10);
    end
    for $k \leftarrow 0$ to $N-1$ do
        // Forward Pass
        Find $\alpha_k^i$ by line search rules (A.12);
        Calculate new iterate $(x_k^i, s_k^i, y_k^i, e_k^i)$;
        $x_{k+1}^i \leftarrow Ax_k^i + Bu_k^i$;
    end
    Reduce the barrier parameter $\boldsymbol{\mu}_1^i = \sigma\boldsymbol{\mu}_1^i$, $\boldsymbol{\mu}_2^i = \sigma\boldsymbol{\mu}_2^i$;
    Update the iteration counter $i = i + 1$;
    Compute optimality measurements $(\xi_o^i, \xi_d^i, \xi_f^i, \xi_e^i)$.
end
```

and α_f is the fraction of the boundary parameter.

After calculating (u_k^i, s_k^i, y_k^i), we can compute x_{k+1}^i using the nominal system equation

$$x_{k+1}^i = Ax_k^i + Bu_k^i \tag{A.13}$$

The above update is repeated recursively from $k = 0$ to $N-1$ to obtain a new trajectory $(\mathbf{x}^i, \mathbf{u}^i, \mathbf{s}^i, \mathbf{y}^i, \mathbf{e}^i)$and complete the forward pass.

A.2.3 Outer loop

The algorithm terminates when the optimality, duality, or constraint feasibility measurements meet the stop criteria:

$$\|\boldsymbol{\xi}_o^i\| \leq \epsilon_o,\ \|\boldsymbol{\xi}_d^i\| \leq \epsilon_d,\ \|\boldsymbol{\xi}_f^i\| \leq \epsilon_f, \text{and} \|\boldsymbol{\xi}_e^i\| \leq \epsilon_e \tag{A.14}$$

The whole soft constraint RRIPM is summarized in Algorithm A.1.

References

Abbasi-Yadkori, Y., & Szepesvári, C. (2015). Bayesian optimal control of smoothly parameterized systems. In *Uncertainty in Artificial Intelligence: Proceedings of the 31st Conference, UAI 2015* (pp. 1–11).

Andersson, J. A., Gillis, J., Horn, G., Rawlings, J. B., & Diehl, M. (2019). CasADi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, *11*(1), 1–36.

Bar-Shalom, Y., & Tse, E. (1974). Dual effect, certainty equivalence, and separation in stochastic control. *IEEE Transactions on Automatic Control*, *19*(5), 494–500.

Bayard, D. S., & Eslami, M. (1985). Implicit dual control for general stochastic systems. *Optimal Control Applications and Methods*, *6*(3), 265–279.

Bayard, D. S., & Schumitzky, A. (2010). Implicit dual control based on particle filtering and forward dynamic programming. *International Journal of Adaptive Control and Signal Processing*, *24*(3), 155–177.

Bellman, R. (1966). Dynamic programming. *Science*, *153*(3731), 34–37.

Bemporad, A. (1998). Reducing conservativeness in predictive control of constrained systems with disturbances. In *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No. 98CH36171)* (Vol. 2, pp. 1384–1389).

Bemporad, A., & Morari, M. (1999). Robust model predictive control: A survey. In *Robustness in identification and control* (pp. 207–226). Springer.

Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, *38*(1), 3–20.

Ben-Tal, A., Goryashko, A., Guslitzer, E., & Nemirovski, A. (2004). Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, *99*(2), 351–376.

Bertsekas, D. (2012). *Dynamic programming and optimal control: Volume I.* Athena Scientific.

Blanchini, F. (1990). Control synthesis for discrete time systems with control and state bounds in the presence of disturbances. *Journal of Optimization Theory and Applications*, *65*(1), 29–40.

Boyd, S., Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization.* Cambridge university press.

Buehler, E. A., Paulson, J. A., & Mesbah, A. (2016). Lyapunov-based stochastic nonlinear model predictive control: Shaping the state probability distribution functions. In *2016 American Control Conference (ACC)* (pp. 5389–5394).

Calafiore, G. C. (2010). Random convex programs. *SIAM Journal on Optimization*, *20*(6), 3427–3464.

Campi, M. C., & Garatti, S. (2008). The exact feasibility of randomized solutions of uncertain convex programs. *SIAM Journal on Optimization*, *19*(3), 1211–1230.

Campi, M. C., & Garatti, S. (2011). A sampling-and-discarding approach to chance-constrained optimization: feasibility and optimality. *Journal of Optimization Theory and Applications*, *148*(2), 257–280.

Cannon, M., Cheng, Q., Kouvaritakis, B., & Raković, S. V. (2012). Stochastic tube MPC with state estimation. *Automatica*, *48*(3), 536–541.

Cannon, M., Kouvaritakis, B., Raković, S. V., & Cheng, Q. (2010). Stochastic tubes in model predictive control with probabilistic constraints. *IEEE Transactions on Automatic Control*, *56*(1), 194–200.

Cannon, M., Kouvaritakis, B., & Wu, X. (2009). Probabilistic constrained MPC for multiplicative and additive stochastic uncertainty. *IEEE Transactions on Automatic Control*, *54*(7), 1626–1632.

Charnes, A., & Cooper, W. W. (1959). Chance-constrained programming. *Management Science*, *6*(1), 73–79.

Chen, J., Shimizu, Y., Sun, L., Tomizuka, M., & Zhan, W. (2021). Constrained iterative LQG for real-time chance-constrained Gaussian belief space planning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 5801–5808).

Chen, Z., et al. (2003). Bayesian filtering: From Kalman filters to particle filters, and beyond. *Statistics*, *182*(1), 1–69.

Cinquemani, E., Agarwal, M., Chatterjee, D., & Lygeros, J. (2011). Convexity and convex approximations of discrete-time stochastic control problems with constraints. *Automatica*, *47*(9), 2082–2087.

Darup, M. S., & Mönnigmann, M. (2012). Low complexity suboptimal explicit NMPC. *IFAC Proceedings Volumes*, *45*(17), 406–411.

Dearden, R., Friedman, N., & Andre, D. (1999). Model based Bayesian exploration. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence* (pp. 150–159).

Del Moral, P. (1997). Nonlinear filtering: Interacting particle resolution. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, *325*(6), 653–658.

Diehl, M., & Bjornberg, J. (2004). Robust dynamic programming for min-max model predictive control of constrained uncertain systems. *IEEE Transactions on Automatic Control*, *49*(12), 2253–2257.

Dombrovskii, V., & Obyedko, T. (2015). Model predictive control for constrained systems with serially correlated stochastic parameters and portfolio optimization. *Automatica*, *54*, 325–331.

Duff, M. O. (2002). *Optimal learning: Computational procedures for bayes-adaptive markov decision processes.* University of Massachusetts Amherst.

El-Bakry, A., Tapia, R. A., Tsuchiya, T., & Zhang, Y. (1996). On the formulation and theory of the newton interior-point method for nonlinear programming. *Journal of Optimization theory and Applications*, *89*(3), 507–541.

Faísca, N. P., Kouramas, K. I., Saraiva, P. M., Rustem, B., & Pistikopoulos, E. N.

(2008). A multi-parametric programming approach for constrained dynamic programming problems. *Optimization Letters*, *2*(2), 267–280.

Farina, M., Giulioni, L., Magni, L., & Scattolini, R. (2013). A probabilistic approach to model predictive control. In *52nd IEEE Conference on Decision and Control* (pp. 7734–7739).

Farina, M., Giulioni, L., Magni, L., & Scattolini, R. (2015). An approach to output-feedback MPC of stochastic linear discrete-time systems. *Automatica*, *55*, 140–149.

Farina, M., Giulioni, L., & Scattolini, R. (2016). Stochastic linear model predictive control with chance constraints–a review. *Journal of Process Control*, *44*, 53–67.

Farrokhsiar, M., & Najjaran, H. (2012). An unscented model predictive control approach to the formation control of nonholonomic mobile robots. In *2012 IEEE International Conference on Robotics and Automation* (pp. 1576–1582).

Feldbaum, A. (1961). Theory of dual control. *Avtomat. i Telemekh*, *21*.

Ferreau, H. J., Bock, H. G., & Diehl, M. (2008). An online active set strategy to overcome the limitations of explicit mpc. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, *18*(8), 816–830.

Fiacco, A. V., & McCormick, G. P. (1990). *Nonlinear programming: sequential unconstrained minimization techniques.* SIAM.

Filatov, N. M., & Unbehauen, H. (2000). Survey of adaptive dual control methods. *IEE Proceedings-Control Theory and Applications*, *147*(1), 118–128.

Garatti, S., & Campi, M. C. (2013). Modulating robustness in control design: Principles and algorithms. *IEEE Control Systems Magazine*, *33*(2), 36–51.

Garstka, S. J., & Wets, R. J.-B. (1974). On decision rules in stochastic programming. *Mathematical Programming*, *7*(1), 117–143.

Geletu, A., Klöppel, M., Zhang, H., & Li, P. (2013). Advances and applications of chance-constrained approaches to systems optimisation under uncertainty. *International Journal of Systems Science*, *44*(7), 1209–1232.

Ghanem, R. G., & Spanos, P. D. (2003). *Stochastic finite elements: a spectral approach.* Courier Corporation.

Ghavamzadeh, M., & Engel, Y. (2006). Bayesian policy gradient algorithms. *Advances in Neural Information Processing Systems*, *19*.

Goulart, P. J., Kerrigan, E. C., & Maciejowski, J. M. (2006). Optimization over state feedback policies for robust control with constraints. *Automatica*, *42*(4), 523–533.

Gupta, A. K., Smith, K. G., & Shalley, C. E. (2006). The interplay between exploration and exploitation. *Academy of Management Journal*, *49*(4), 693–706.

Gwerder, M., & Tödtli, J. (2005). Predictive control for integrated room automation. In *8th REHVA World Congress Clima.*

Herzog, F., Dondi, G., & Geering, H. P. (2007). Stochastic model predictive control and portfolio optimization. *International Journal of Theoretical and Applied Finance*, *10*(02), 203–233.

Hooshmand, A., Poursaeidi, M. H., Mohammadpour, J., Malki, H. A., & Grigoriads, K. (2012). Stochastic model predictive control method for microgrid management. In *2012 IEEE PES Innovative Smart Grid Technologies (ISGT)* (pp. 1–7).

Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, *101*(1-2), 99–134.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237–285.

Kakade, S. M. (2001). A natural policy gradient. *Advances in Neural Information Processing Systems*, *14*.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, *82*(1), 35-45.

Kerrigan, E. C., & Maciejowski, J. M. (2000). Soft constraints and exact penalty functions in model predictive control. In *Control 2000 Conference, Cambridge* (pp. 2319–2327).

Kojima, M., Megiddo, N., & Mizuno, S. (1993). A primal—dual infeasible-interior-point algorithm for linear programming. *Mathematical Programming*, *61*(1), 263–280.

Kojima, M., Mizuno, S., & Yoshise, A. (1993). A little theorem of the big M in interior point algorithms. *Mathematical Programming*, *59*(1), 361–375.

Korda, M., Gondhalekar, R., Oldewurtel, F., & Jones, C. N. (2014). Stochastic MPC framework for controlling the average constraint violation. *IEEE Transactions on Automatic Control*, *59*(7), 1706–1721.

Kouvaritakis, B., Cannon, M., & Muñoz-Carpintero, D. (2013). Efficient prediction strategies for disturbance compensation in stochastic MPC. *International Journal of Systems Science*, *44*(7), 1344–1353.

Kouvaritakis, B., Cannon, M., Raković, S. V., & Cheng, Q. (2010). Explicit use of probabilistic distributions in linear predictive control. *Automatica*, *46*(10), 1719–1724.

Kucerovsky, D., Mousavand, K., & Sarraf, A. (2016). On some properties of Toeplitz matrices. *Cogent Mathematics*, *3*(1), 1154705.

Kumar, R., Wenzel, M. J., Ellis, M. J., ElBsat, M. N., Drees, K. H., & Zavala, V. M. (2018). A stochastic dual dynamic programming framework for multiscale MPC. *IFAC-PapersOnLine*, *51*(20), 493–498.

Kvasnica, M., Holaza, J., Takács, B., & Ingole, D. (2015). Design and verification of low-complexity explicit MPC controllers in MPT3. In *2015 European Control Conference (ECC)* (pp. 2595–2600).

Lee, Y., & Kouvaritakis, B. (1999). Constrained receding horizon predictive control for systems with disturbances. *International Journal of Control*, *72*(11), 1027–1032.

Lewis, F. L., & Liu, D. (2013). *Reinforcement learning and approximate dynamic programming for feedback control.* John Wiley & Sons.

Li, D., Qian, F., & Fu, P. (2002). Variance minimization approach for a class of dual control problems. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)* (Vol. 5, pp. 3759–3764).

Li, W., & Todorov, E. (2004). Iterative linear quadratic regulator design for nonlinear biological movement systems. In *ICINCO (1)* (pp. 222–229).

Lindquist, A. (1973). On feedback control of linear stochastic systems. *SIAM Journal on Control*, *11*(2), 323–343.

Liu, J. S., & Chen, R. (1998). Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, *93*(443), 1032–1044.

Lofberg, J. (2003). Approximations of closed-loop minimax MPC. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)* (Vol. 2, pp. 1438–1442).

MacKay, D. J., Mac Kay, D. J., et al. (2003). *Information theory, inference and learning algorithms.* Cambridge university press.

Marshall, A. W., & Olkin, I. (1960). Multivariate Chebyshev inequalities. *The Annals of Mathematical Statistics*, 1001–1014.

Mayne, D. Q. (1966). A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, *3*(1), 85–95.

Mayne, D. Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, *50*(12), 2967-2986.

Mayne, D. Q., Rawlings, J. B., Rao, C. V., & Scokaert, P. O. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, *36*(6), 789–814.

Mayne, D. Q., Seron, M. M., & Raković, S. (2005). Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica*, *41*(2), 219–224.

Maz'ya, V., & Schmidt, G. (1996). On approximate approximations using gaussian kernels. *IMA Journal of Numerical Analysis*, *16*(1), 13–29.

Mesbah, A. (2016). Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, *36*(6), 30–44.

Mesbah, A. (2018). Stochastic model predictive control with active uncertainty learning: A survey on dual control. *Annual Reviews in Control*, *45*, 107–117.

Mesbah, A., Streif, S., Findeisen, R., & Braatz, R. D. (2014). Stochastic nonlinear model predictive control with probabilistic constraints. In *2014 American control conference* (pp. 2413–2419).

Meyer, C. D. (2000). *Matrix analysis and applied linear algebra* (Vol. 71). Siam.

Muñoz-Carpintero, D., Kouvaritakis, B., & Cannon, M. (2016). Striped parameterized tube model predictive control. *Automatica*, *67*, 303–309.

Neal, R. M. (2012). *Bayesian learning for neural networks* (Vol. 118). Springer Science & Business Media.

Nocedal, J., & Wright, S. J. (1999). *Numerical optimization.* Springer.

Ohtsuka, T. (2004). A continuation/GMRES method for fast computation of nonlinear receding horizon control. *Automatica*, *40*(4), 563–574.

Oldewurtel, F., Jones, C. N., & Morari, M. (2008). A tractable approximation of chance constrained stochastic MPC based on affine disturbance feedback. In *2008 47th IEEE Conference on Decision and Control* (pp. 4731–4736).

Oldewurtel, F., Jones, C. N., Parisio, A., & Morari, M. (2013). Stochastic model predictive control for building climate control. *IEEE Transactions on Control Systems Technology*, *22*(3), 1198–1205.

Oldewurtel, F., Parisio, A., Jones, C. N., Morari, M., Gyalistras, D., Gwerder, M., . . . Wirth, K. (2010). Energy efficient building climate control using stochastic model predictive control and weather predictions. In *Proceedings of the 2010 American Control Conference* (pp. 5100–5105).

Patchell, J., & Jacobs, O. (1971). Separability, neutrality and certainty equivalence. *International Journal of Control*, *13*(2), 337–342.

Patrinos, P., Trimboli, S., & Bemporad, A. (2011). Stochastic MPC for real-time market-based optimal power dispatch. In *2011 50th IEEE Conference on Decision and Control and European Control Conference* (pp. 7111–7116).

Poupart, P., Vlassis, N., Hoey, J., & Regan, K. (2006). An analytic solution to discrete Bayesian reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning* (pp. 697–704).

Powell, W. B. (2007). *Approximate dynamic programming: Solving the curses of dimensionality* (Vol. 703). John Wiley & Sons.

Prandini, M., Garatti, S., & Lygeros, J. (2012). A randomized approach to stochastic model predictive control. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)* (pp. 7315–7320).

Primbs, J. A., & Sung, C. H. (2009). Stochastic receding horizon control of constrained linear systems with state and control multiplicative noise. *IEEE transactions on Automatic Control*, *54*(2), 221–230.

Ribeiro, M. I. (2004). Kalman and extended Kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics*, *43*, 46.

Rodriguez, L. A., & Sideris, A. (2010). An active set method for constrained linear quadratic optimal control. In *Proceedings of the 2010 American Control Conference* (pp. 5197–5202).

Schildbach, G., Calafiore, G. C., Fagiano, L., & Morari, M. (2012). Randomized model predictive control for stochastic linear systems. In *2012 American Control Conference (ACC)* (pp. 417–422).

Schildbach, G., Fagiano, L., Frei, C., & Morari, M. (2014). The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations. *Automatica*, *50*(12), 3009–3018.

Schwarm, A. T., & Nikolaou, M. (1999). Chance-constrained model predictive control. *AIChE Journal*, *45*(8), 1743–1752.

Scokaert, P. O., & Mayne, D. Q. (1998). Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic Control*, *43*(8), 1136–1142.

Scokaert, P. O., & Rawlings, J. B. (1998). Constrained linear quadratic regulation. *IEEE Transactions on Automatic Control*, *43*(8), 1163–1169.

Sehr, M. A., & Bitmead, R. R. (2017). Particle model predictive control: Tractable stochastic nonlinear output-feedback MPC. *IFAC-PapersOnLine*, *50*(1), 15361–15366.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *International Conference on Machine Learning* (pp. 387–395).

Streif, S., Karl, M., & Mesbah, A. (2014). Stochastic nonlinear model predictive control with efficient sample approximation of chance constraints. *arXiv preprint arXiv:1410.4535*.

Strens, M. J. (2000). A Bayesian framework for reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning* (pp. 943–950).

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction.* MIT press.

Sutton, R. S., Barto, A. G., & Williams, R. J. (1992). Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems Magazine*, *12*(2), 19–22.

Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, *12*.

Thompson, W. R. (1935). On the theory of apportionment. *American Journal of Mathematics*, *57*(2), 450–456.

Todorov, E., & Li, W. (2005). A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the 2005, American Control Conference* (pp. 300–306).

Van de Vusse, J. (1964). Plug-flow type reactor versus tank reactor. *Chemical Engineering Science*, *19*(12), 994–996.

Van Hessem, D., & Bosgra, O. (2006). Stochastic closed-loop model predictive control of continuous nonlinear chemical processes. *Journal of Process Control*, *16*(3), 225–241.

Visintini, A. L., Glover, W., Lygeros, J., & Maciejowski, J. (2006). Monte Carlo optimization for conflict resolution in air traffic control. *IEEE Transactions on Intelligent Transportation Systems*, *7*(4), 470–482.

Wan, E. A., & Van Der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)* (pp. 153–158).

Wang, C., Ong, C.-J., & Sim, M. (2009). Convergence properties of constrained linear system under MPC control law using affine disturbance feedback. *Automatica*, *45*(7), 1715–1720.

Wang, J. M., Fleet, D. J., & Hertzmann, A. (2005). Gaussian process dynamical models. *Advances in Neural Information Processing Systems*, *18*.

Wang, J. M., Fleet, D. J., & Hertzmann, A. (2007). Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(2), 283–298.

Weissel, F., Huber, M. F., & Hanebeck, U. D. (2009). Stochastic nonlinear model predictive control based on Gaussian mixture approximations. In *Informatics in Control, Automation and Robotics* (pp. 239–252). Springer.

Wiering, M. A., & Van Otterlo, M. (2012). Reinforcement learning. *Adaptation, Learning, and Optimization*, *12*(3), 729.

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

Xiu, D., & Karniadakis, G. E. (2002). The Wiener–Askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, *24*(2), 619–644.

Yan, J., & Bitmead, R. R. (2005). Incorporating state estimation into model predictive control and its application to network traffic control. *Automatica*, *41*(4), 595–604.

Zhou, Z., & Cogill, R. (2013). Reliable approximations of probability-constrained stochastic linear-quadratic control. *Automatica*, *49*(8), 2435–2439.

List of Publications

Journal articles

1. Zhang, J., & Ohtsuka, T. (2020). Stochastic model predictive control using simplified affine disturbance feedback for chance-constrained systems. *IEEE Control Systems Letters*, 5(5), 1633-1638. **Chapter 3**

2. Zhang, J., & Ohtsuka, T. A recursive Riccati interior-point method for chance-constrained stochastic model predictive control. *SICE Journal of Control, Measurement, and System Integration* (under review). **Chapter 4**

3. Zhang, J., & Ohtsuka, T. Output-feedback stochastic model predictive control of chance-constrained nonlinear system. *IET Control Theory & Applications* (under review). **Chapter 5**

Conference proceedings

1. Zhang, J., & Ohtsuka, T. (2021, May). Stochastic model predictive control using simplified affine disturbance feedback for chance-constrained systems. In *2021 American Control Conference (ACC)* (pp. 1256–1261). IEEE. **Chapter 3** (Conference version of journal article 1)