

Stochastic LQ Control with Probabilistic Constraints

---

A Dissertation

Presented to  
the faculty of the School of Engineering and Applied Science  
University of Virginia

---

in partial fulfillment  
of the requirements for the degree

Doctor of Philosophy

by

Zhou Zhou

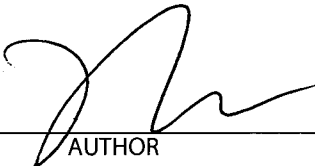
December

2012

---

APPROVAL SHEET

The dissertation  
is submitted in partial fulfillment of the requirements  
for the degree of  
Doctor of Philosophy

  
\_\_\_\_\_  
AUTHOR

The dissertation has been read and approved by the examining committee:

Prof. Randy Cogill

\_\_\_\_\_  
Advisor

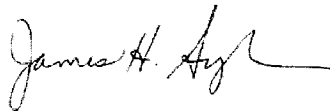
Prof. Steve Patek

\_\_\_\_\_  
Prof. Alfredo Garcia

\_\_\_\_\_  
Prof. Peter Beling

\_\_\_\_\_  
Prof. Zongli Lin

Accepted for the School of Engineering and Applied Science:



Dean, School of Engineering and Applied Science

December  
2012

## Abstract

In this thesis we focus on stochastic LQ control problems with probabilistic constraints. We will briefly review the history of LQ control and the related classical results for constrained and unconstrained cases. Then we formulate the problem studied, where the system has linear dynamics and a quadratic cost, and states are required to satisfy a probabilistic constraint. We sample the most recent techniques for such probability constrained control problems and propose our own approach. We consider two types of probability constraints: all-stage and per-stage. In the first case, there is a joint probabilistic constraint on all the system states over the whole predicting horizon while in the second one the states are restricted by individual probability constraints at each stage. The contribution of this thesis has two parts: first we develop a recursive state feedback control algorithm for a special class of state constrained stochastic LQR, and a disturbance feedback controller for the general case using quadratic programming for the all-stage problems. Second, we design a recursive algorithm for the per-stage problem based on sub-gradient method. We also implement a practical Model Predictive Control algorithm for such problems. The control algorithm is tested on a simple temperature control problem for analysis.



## Acknowledgements

I dedicate this dissertation to my family, in particular to my mother. Without your unconditional love this thesis would not have been possible.

First I want to thank my advisor Professor Cogill. He provides constant guidance and support during my study. I enjoy the freedom he gives me to explore the topic that I like most. And I rely on his knowledge and judgement whenever I encounter difficulties in my research. I want to thank my committee members: Professor Patek, Professor Beling, Professor Garcia and Professor Lin. The courses I took from them and their feedback has been crucial to the preparation of this thesis. I also want to extend my gratitude to all the people here in UVA who took part in educating me. The knowledge and experience I have learned from them will prove to be invaluable in the future. I want to thank Jennifer, Terri, Ig and Debra. There are tons of things in the department that I would not know what to do with without their help.

I sincerely thank all my friends here: My best friends Zhengjun Ma, Xiaojing Ma, Qiaohua Tan and Siyu Lin. Thank you for being there for me in the tough days. My brilliant, wonderful officemate Qifeng Qiao and my roommate Qiang Qian who make my daily life enjoyable. Xin Yao, Qian Zhang, Qian Zhou, Xiao Zhou, Yuting Wang, Cheng Peng, Chuanchuan Shen, Chenyang Li, Emma, George and many others, you are the most colorful chapter of my memories in Charlottesville.

The last but not the least I am grateful to the staff members at the International Student Office and the language development center. As an international student I feel at home with their help.

Four years can change much about a person, I sincerely thank you all for being part of this crucial change of my life.

# Contents

List of Figures	v
List of Tables	vii
<b>1 Introduction</b>	<b>1</b>
1.1 Overview . . . . .	1
1.1.1 Problem formulation and terminology . . . . .	1
<b>2 Literature review</b>	<b>5</b>
2.1 Constrained deterministic LQ control . . . . .	6
2.2 Unconstrained Linear Quadratic Control . . . . .	12
2.2.1 Dynamic Programming . . . . .	12
2.2.2 Closed form solution of unconstrained LQ Control . . . . .	15
2.3 Probabilistic Constraints . . . . .	18
2.3.1 Exact Evaluation of Probabilistic Constraints . . . . .	18
2.3.2 Approximating Probabilistic Constraints . . . . .	19
2.4 Constrained Stochastic Linear Quadratic Control . . . . .	22
2.4.1 Generalized Linear Quadratic Control . . . . .	22
2.4.2 Semi-definite Programming Formulation . . . . .	27
2.4.3 A Tractable approximation of probability constrained stochastic LQR . . . . .	30
<b>3 Feedback controller design for Probability Constrained SLQR</b>	<b>35</b>
3.1 A Chebyshev bound approach to convert probabilistic constraints	35
3.2 SLQR with all-stage probabilistic constraint . . . . .	38
3.2.1 State Separable Approximation . . . . .	39

## CONTENTS

---

3.2.2	State Non-Separable Approximation . . . . .	43
3.2.3	Numerical Example . . . . .	48
3.3	SLQR with per-stage probabilistic constraints . . . . .	52
3.3.1	A sub-gradient method approach . . . . .	52
3.3.2	Model Predictive Control implementation . . . . .	60
3.3.3	Numerical Example . . . . .	62
3.4	Conclusion and Extensions . . . . .	66
	<b>References</b>	<b>69</b>



# List of Figures

3.1	2D trajectories . . . . .	49
3.2	state trajectories . . . . .	50
3.3	Important statistics . . . . .	63
3.4	Control Output . . . . .	63
3.5	Disturbances . . . . .	64
3.6	Important statistics . . . . .	64

## LIST OF FIGURES

---

# List of Tables

3.1	Recursive algorithm for State Separable Approximation . . . . .	43
3.2	Convex programming algorithm for State Non-separable Approximation . . . . .	47
3.3	Running results . . . . .	51
3.4	MPC implementation . . . . .	61

## LIST OF TABLES

---

# 1

## Introduction

This thesis is concerned with theoretical development of stochastic linear quadratic stochastic control problems with probabilistic constraints and its extensions. In this chapter, we will introduce the background of the thesis and walk through the existing results in the area.

### 1.1 Overview

This section is an overview of the thesis, aiming to give the reader a sense of what the problem is and what is the value of the research. We will formulate our problem mathematically and summarize the difficulties and the major contributions of this thesis. The contents are covered with more care and technical detail in later chapters.

#### 1.1.1 Problem formulation and terminology

Here we formulate the control problem we consider through out the thesis. In control theory, a system is described by a set of states  $x \in R^n$  and a system evolution equation that captures the dynamics. In this thesis, without particularly pointing out, we are studying linear systems with quadratic costs

$$x_{k+1} = Ax_k + Bu_k + w_k$$

# 1. INTRODUCTION

---

for  $k \in \{0, \dots, N-1\}$ . In period  $k$ ,  $x_k$  is the system state,  $u_k$  is the control input, and  $w_k$  is the disturbance input. The disturbance inputs are independent, have zero mean, and have covariance matrix  $W$ . Our goal is to minimize the classical LQR objective in expectation,

$$\mathbf{E} \left[ \sum_{k=0}^{N-1} x_{k+1}^T Q x_{k+1} + u_k^T R u_k \right],$$

where the matrix  $Q$  is positive semidefinite and the matrix  $R$  is positive definite. In this thesis we consider both state feedback and disturbance feedback control laws. The state feedback control law considered in the thesis is of the form

$$u_k = \bar{u}_k + K_k(x_k - \bar{x}_k).$$

Here,  $\bar{x}_k$  denotes the expected value of the state and  $\bar{u}_k$  denotes the expected value of the control input. In a following section, we consider a disturbance feedback control law of the form

$$u_k = \bar{u}_k + \sum_{j=0}^{k-1} K_{(k,j)} w_j.$$

So far it is very similar to the standard unconstrained stochastic linear quadratic control problem, which can be solved recursively by Algebraic Riccati equation. So what is the big difference here? The complicating factor in our problem is a linear or quadratic constraint on the system states that must hold with some specified probability. Specifically, for a given  $\alpha \in [0, 1)$ , we must select a control law that ensures

$$\mathbf{P}(g_k(x_k, u_k) \leq 0) \geq \alpha_k$$

In this thesis, we assume that  $g_k$  is a linear or quadratic function. This is the part that complicates the entire problem. The probabilistic constraint above means that the constraint:

$$g_k(x_k, u_k) \leq 0$$

must be satisfied with at least probability  $\alpha_k$ . A few simple questions can be helpful to grasp a glimpse of the difficulties lie ahead.

- How to test joint probabilistic constraints?

- What is the structure of the controller if we want to introduce feedback?
- What is the complexity of this problem?

All these issues must be tackled in one shot when we are developing an algorithm for it. And what makes things worse is that they are highly correlated to each other too. For example, the complexity of the approach to test the probabilistic constraint is critical to the total solving time of the problem as the testing is called repeatedly. Also, a different feedback structure may bring fundamental change to the complexity of the problem. In fact, there are research papers covering each of the questions above. Stochastic control with probabilistic constraints lies right at the intersection of those subareas and has attracted much attention in the recent few years. However, it is still a very open problem today because of its many difficulties. This thesis aims to derive new algorithms to solve stochastic linear quadratic control problem with probabilistic constraints. The contributions of this thesis are, in response to the questions we raised before:

- Proposed to use a multi-dimensional Chebyshev bound to replace the probabilistic constraint
- Designed feedback controllers for the problem
- Developed efficient algorithms to solve the probability constrained stochastic LQ control problem
- Compared our algorithms to other existing results in the literature

## 1. INTRODUCTION

---



## 2

# Literature review

Constrained stochastic linear quadratic control is a relatively new research area. Although unconstrained SLQR was solved a few decades ago, constrained problems are still not well understood. In this chapter we will sample the most recent development in this area, especially new algorithms for probabilistic constrained SLQR. We will first start with constrained deterministic LQ control. Then we introduce the classical recursive algorithm for unconstrained SLQR and the representative approaches to solve constrained SLQR in the literature. As we mentioned before, this area is still relatively new so there exist different but similar formulations for the problem. This is basically because there are many possible ways to replace constraints on a SLQR problem and the different feedback control structure. We will try to extract the most fundamental aspects of the different formulations and make clear the differences of the approaches as well as their respective strengths and weaknesses. Despite the different setups, one will find the formulations soon to be investigated have the following things in common. The system evolution is linear with additive noise and for simplicity we assume the noise has zero mean. Besides introducing control algorithms we will also briefly talk about the techniques used to approximate probabilistic constraints as they play a key role in probability constrained SLQR problems.

## 2. LITERATURE REVIEW

---

### 2.1 Constrained deterministic LQ control

In practice, so many control problems are formulated deterministically. And it is hard to image that a system runs without some restrictions. That is the motivation to study constrained deterministic LQ control. It is one of the most important areas in control theory and is widely used in many other fields. To date, the most elegant approach to solve a constrained LQR is proposed by Bemporad and Morari [1]. Let us first look at the problem formulation

$$\begin{aligned}
 \min \quad & \sum_{k=0}^{N-1} \{x_{k+1}^T Q x_{k+1} + u_k^T R u_k\} \\
 \text{s.t.} \quad & \underline{x}_{k+1} \leq x_{k+1} \leq \bar{x}_{k+1} \\
 & \underline{u}_k \leq u_k \leq \bar{u}_k \\
 & x_{k+1} = A x_k + B u_k, \\
 & k = 0, \dots, N-1
 \end{aligned}$$

The above finite state constrained LQR is solved repeatedly in a model predictive control(MPC) algorithm [Chmielewski96]. And it is proved that for deterministic LQR, open loop control is equivalent to closed loop control. The authors in [1] thus proposed to formulate the above problem as a multi-parametric programming problem. They proved that the optimal control is a piece-wise linear function of system states and correspondingly the optimal cost function is piece-wise quadratic in states.

We will now show how to formulate the constrained deterministic LQ control as a multi-parametric program. Let us rewrite the above formulation in a more compact way. First we need some definitions

$$\mathcal{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}, \mathcal{U} = \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix}, \bar{\mathcal{X}} = \begin{bmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_N \end{bmatrix}, \underline{\mathcal{X}} = \begin{bmatrix} \underline{x}_1 \\ \vdots \\ \underline{x}_N \end{bmatrix}, \bar{\mathcal{U}} = \begin{bmatrix} \bar{u}_0 \\ \vdots \\ \bar{u}_{N-1} \end{bmatrix}, \underline{\mathcal{U}} = \begin{bmatrix} \underline{u}_0 \\ \vdots \\ \underline{u}_{N-1} \end{bmatrix},$$

Using the new notations, the system dynamics become

$$\mathcal{X} = \mathbb{F}x_0 + \mathbb{H}\mathcal{U}$$

Where the block matrices  $\mathbb{F}$  and  $\mathbb{H}$  are given by:

$$\mathbb{F} = \begin{bmatrix} A \\ \vdots \\ A^N \end{bmatrix} \quad \mathbb{H} = \begin{bmatrix} B & & & \\ AB & B & & \\ \vdots & & \ddots & \\ A^{N-1}B & A^{N-2}B & \dots & B \end{bmatrix}$$

$$\mathbb{Q} = \begin{bmatrix} Q & & \\ & \ddots & \\ & & Q \end{bmatrix}, \quad \mathbb{R} = \begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix}$$

We can then equivalently reformulate the problem as

$$\min x_0^T \mathbb{F}^T \mathbb{Q} \mathbb{H} \mathbf{u} + \frac{1}{2} \mathbf{u}^T (\mathbb{R} + \mathbb{H}^T \mathbb{Q} \mathbb{H}) \mathbf{u} \quad (2.1.1)$$

$$s.t. \quad \underline{\mathbf{u}} \leq \mathbf{u} \leq \overline{\mathbf{u}} \quad (2.1.2)$$

$$\underline{\mathcal{X}} - \mathbb{F}x_0 \leq \mathbb{H}\mathbf{u} \leq \mathbb{H}\mathbf{u} + \mathbb{F}x_0 \quad (2.1.3)$$

Further more, (2.1.2) and (2.1.3) can be combined as

$$\hat{\mathbb{H}}\mathbf{u} \leq \hat{\mathbb{E}} + \mathbb{E}x_0 \quad (2.1.4)$$

where

$$\hat{\mathbb{H}} = \begin{bmatrix} \mathbb{H} \\ I \\ -\mathbb{H} \\ -I \end{bmatrix}, \quad \hat{\mathbb{E}} = \begin{bmatrix} \underline{\mathcal{X}} \\ \overline{\mathbf{u}} \\ -\underline{\mathcal{X}} \\ -\underline{\mathbf{u}} \end{bmatrix}, \quad \mathbb{E} = \begin{bmatrix} -\mathbb{F} \\ 0 \\ \mathbb{F} \\ 0 \end{bmatrix}$$

Note that now the only variable is  $\mathbf{u}$  and it is clear that (2.1.1) together with (2.1.4) is a quadratic programming (QP) with linear inequality constraints, parameterized in initial states  $x_0$ .

Obviously, for any given  $x_0$  the QP can be solved by a conventional solver. However, in a MPC setting the QP is solved at the beginning of each stage and only the first control is actually applied. When the program size grows big, solving the QP real-time becomes difficult and that was one of the biggest issues for a MPC algorithm until the idea of multi-parametric programming was introduced into the context. Parametric programming is not a new topic, there already were many results a few decades ago. However, most of them are for scalar parameters. Multi-parametric cases were first studied by Pistikopoulos, Aecedevo and Duo [2]

## 2. LITERATURE REVIEW

---

[3], [4], based on the post-optimality sensitivity analysis results of Fiacco [5] and Gal [6]. We will only summarize the results of multi-parametric programming that used to solve (2.1.1) with (2.1.4), which we called **MPQP**( $x_0$ ) from now on. There are more in the literature such as [7] and [8] about multi-parametric programming if the reader is interested.

Now we show why the optimal solution of **MPQP**( $x_0$ ) is linear in  $x_0$ . The KKT conditions are

$$\mathcal{L}(\lambda, \mathcal{U}) = x_0^T \mathbb{F}^T \mathbb{Q} \mathbb{H} + \mathcal{U}^T (\mathbb{R} + \mathbb{H}^T \mathbb{Q} \mathbb{H}) + \sum_{i=1}^{2Nn} \lambda_i \hat{\mathbb{H}}_i = 0 \quad (2.1.5)$$

$$\lambda_i (\hat{\mathbb{H}}_i \mathcal{U} - (\hat{\mathbb{E}}_i + \mathbb{E}_i x_0)) = 0 \quad (2.1.6)$$

$$\hat{\mathbb{H}}_i \mathcal{U} - (\hat{\mathbb{E}}_i + \mathbb{E}_i x_0) \leq 0 \quad (2.1.7)$$

$$\lambda_i \geq 0 \quad (2.1.8)$$

where  $\mathbb{H}_i, \hat{\mathbb{H}}_i, \mathbb{E}_i, \hat{\mathbb{E}}_i$  present the  $i$ th row of  $\mathbb{H}, \hat{\mathbb{H}}, \mathbb{E}, \hat{\mathbb{E}}$  respectively and  $i = 1, \dots, 2Nn$ . Given an optimal solution pair  $(\mathcal{U}^*, \lambda^*)$  associated with a parameter  $x_0$ , we can obtain the active constraint set  $\mathcal{A}$  and the inactive set  $\tilde{\mathcal{A}}$ , which are defined as

$$\begin{cases} \mathcal{A} = \{i | \hat{\mathbb{H}}_i \mathcal{U}^* - (\hat{\mathbb{E}}_i + \mathbb{E}_i x_0) = 0\} \\ \tilde{\mathcal{A}} = \{i | \hat{\mathbb{H}}_i \mathcal{U}^* - (\hat{\mathbb{E}}_i + \mathbb{E}_i x_0) < 0\} \end{cases}$$

We then use  $\lambda_{\mathcal{A}}$  and  $\lambda_{\tilde{\mathcal{A}}}$  to denote the Lagrangian multipliers associated with  $\mathcal{A}$  and  $\tilde{\mathcal{A}}$  respectively (we use the same notation for other vectors/matrices). Now we summarize the main results. First we need an assumption. We say an active set satisfies **linearly independence constraint qualification**(LICQ) if the set of active constraint gradients are linearly independent. In our case, this assumption means that  $\hat{\mathbb{H}}_{\mathcal{A}}$  has full row rank. Using (2.1.5) and (2.1.6) we get

$$\mathcal{U}^* = -(\mathbb{R} + \mathbb{H}^T \mathbb{Q} \mathbb{H})^{-1} (\mathbb{H}^T \mathbb{Q} \mathbb{F} x_0 + \hat{\mathbb{H}}^T \lambda^*) \quad (2.1.9)$$

Replace  $\mathcal{U}$  in (2.1.6) with (2.1.9) and under the assumption of LICQ, we obtain

$$\lambda_{\mathcal{A}}^* = -\mathbb{M} \left( \hat{\mathbb{H}}_{\mathcal{A}} (\mathbb{R} + \mathbb{H}^T \mathbb{Q} \mathbb{H})^{-1} \mathbb{H}^T \mathbb{Q} \mathbb{F} + \mathbb{E}_{\mathcal{A}} \right) x_0 + \mathbb{M} \hat{\mathbb{E}}_{\mathcal{A}} \quad (2.1.10)$$

where  $\mathbb{M} = \left( \hat{\mathbb{H}}_{\mathcal{A}} (\mathbb{R} + \mathbb{H}^T \mathbb{Q} \mathbb{H})^{-1} \hat{\mathbb{H}}_{\mathcal{A}}^T \right)^{-1}$ . We can see from (2.1.10) that  $\lambda^*$  is an affine function of parameter  $x_0$  and with (2.1.9) we can conclude that so is  $\mathcal{U}^*$ . Note that

we assume at the beginning of the above analysis that we already have an optimal solution pair  $(U^*, \lambda^*)$  and the corresponding active constraint set  $\mathcal{A}$ . So the above conclusion is valid locally within the region in parameter space associated with  $\mathcal{A}$ , which is called critical region. The critical region is defined by (2.1.7) and (2.1.8). As one can see, the whole parameter space is divided into polyhedrons, each of which is associated with an affine structure of the corresponding optimal solution pair of the MPQP. We formally summarize the above result with a theorem

**Theorem 2.1.** *Consider the optimization problem (2.1.1) and (2.1.4). Suppose the feasible region defined by (2.1.4) is a polyhedron, then the optimal solution pair  $(U^*, \lambda^*)$  is piece-wise affine in the parameter  $x_0$ . Further more, if LICQ holds everywhere in the polyhedron,  $\lambda^*(x_0)$  is continuous.*

**Proof.** We have shown the first part in our previous analysis. Please refer to [1] for the second half. ■

So if we know a feasible parameter  $x_0$  and the active set  $\mathcal{A}$ , we can characterize the optimal solution of the MPQP in the corresponding critical region. And if we can explore all the critical regions in parameter space we can then obtain explicit solutions for the control problem in terms of the parameter  $x_0$  using (2.1.9) and (2.1.10). That is the basic idea of solving the constrained deterministic LQR as a MPQP. There are a few things missing before a method to explore the parameter space is developed. The first one is how to obtain a starting point of  $x_0$ . Assume that the parameter space  $\Theta$  we want to explore is defined as

$$\Theta = \{\theta | T\theta \leq Z\}$$

We can obtain an initial feasible  $x_0$  in  $\Theta$  by solving the following LP

$$\max_{x, u, \epsilon} \epsilon \tag{2.1.11}$$

$$s.t. \quad T_i x + \epsilon \|T_i\| \leq Z_i \tag{2.1.12}$$

$$\hat{\mathbb{H}}_i u - (\hat{\mathbb{E}}_i + \mathbb{E}_i x) \leq 0 \tag{2.1.13}$$

The above LP aims to find a point  $x \in \Theta$  and at the same time ensures that (2.1.7) defines a non-empty set. If the optimal  $\epsilon$  is positive, we can use the obtained  $x$  as the starting point to explore  $\Theta$ . Otherwise, the MPQP (2.1.1)

## 2. LITERATURE REVIEW

---

and (2.1.4) is infeasible with any  $x \in \Theta$ . Now if given a parameter subspace to explore we know how to find a starting point  $x_0$  in it. However, we don't know what are the critical regions. Remember we can only have the explicit solutions after we determine the critical regions. The following theorem can serve as a tool to answer the question [1].

**Theorem 2.2.** *Let  $\Theta$  be a polyhedron and  $R_0$  a non-empty subset of  $\Theta$  defined by*

$$R_0 = \{x \in \Theta | Ax \leq b\}$$

Let  $R_i$  be

$$R_i = \{x \in \Theta | A_i x > b_i, A_j x \leq b_j, \forall j < i\}, \quad i = 1, \dots, m\}$$

where  $m$  is the number of rows of  $A$ . If we define  $CR = \bigcup_{i=1}^m R_i$ , then  $\{R_0, \dots, R_m\}$  is a partition of  $\Theta$ .

**Proof.** It is easy to see that  $R_0, \dots, R_m$  are disjoint with each other. We need to show that  $\bigcup_{i=0}^m R_i = \Theta$ . It suffices to show that for any  $x \in \Theta$ , there exists  $i \in 0, \dots, m$  such that  $x \in R_i$ . Suppose  $x \in R_0$ , it is done. Now assume that  $x \notin R_0$ . Then there exists  $i$  such that  $A_i x > b_i$ . Without loss of generality we assume that  $i = \min_{i \leq m} i : A_i x > b_i$ . Obviously, by definition  $x \in R_i$ . ■

The above theorem gives us a way to partition a polyhedron into smaller polyhedrons. In particular, it can be used to enumerate all critical regions. Now we have enough tools to solve the constrained deterministic linear quadratic control problem (2.1.1) and (2.1.4) and obtain the piece-wise affine optimal solution. For more details and improvement of the procedure please refer to [1] and [9]. It seems that solving the deterministic LQC problem as a MPQP makes things complicated. However, it should be pointed out that most of the computation can be done off-line. First one needs to figure out a polyhedral region  $\Theta$  of parameter space to be explored. Then use the LP (2.1.11)-(2.1.13) to determine the starting point in  $\Theta$ . With this starting point, using (2.1.9) and (2.1.10) we can obtain the explicit optimal control  $\mathcal{U}$  and the corresponding Lagrangian multipliers  $\lambda$ . Note that the explicit solution is valid only within the critical region defined by (2.1.7) and (2.1.8). For the rest of  $\Theta$ , we can use the procedure described in theorem

## 2.1 Constrained deterministic LQ control

---

(2.1). The procedure is repeated until  $\Theta$  is fully explored. All of these can be done off-line. For a on-line MPC algorithm, all it has to do is to check its initial state  $x_0$  and look up the critical region where  $x_0$  lies in and use the pre-recorded explicit solution to obtain the optimal control. This approach is very suitable for on-line implementation.

## 2.2 Unconstrained Linear Quadratic Control

In this section we consider a linear system with additive disturbance

$$x_{k+1} = A_k x_k + B_k u_k + w_k, \quad k = 0, \dots, N - 1$$

Given initial condition  $x_0$ , our goal is to minimize the following quadratic cost function

$$E\left[\sum_{k=0}^{N-1} x_{k+1}^T Q x_{k+1} + u_k^T R u_k\right]$$

In the above expressions,  $x_k$  is a  $n$  dimensional vector representing system states and  $u_k$  is a  $m$  dimensional vector representing controls. We assume here  $A_k, B_k$  are constant matrices, but we point out that there exists extensions of this formulation where they are random [10].  $w_k$  is a random vector with zero mean and covariance matrix  $\Sigma_{w_k}$ . Note that although all these are in a stochastic setting, the corresponding deterministic problem has similar solution as it can be viewed as a special case of the stochastic case where  $w_k$  is constant zero. It is well-known that the unconstrained LQ control problem has a closed form solution obtained by solving it as a dynamic program. Before we show the explicit solution we briefly introduce dynamic programming.

### 2.2.1 Dynamic Programming

We use the same notation here for system states, controls and disturbances. In a dynamic program, the system evolution is not necessarily linear. It can be any form, as we denote by  $f_k$  and we write down the general case as

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, 1, \dots, N - 1$$

Suppose there is a random cost  $c_k(x_k, u_k, w_k)$  at stage  $0, \dots, N - 1$  when choosing action  $u_k$  at state  $x_k$  and a terminal cost  $c_N(x_N)$  at the last stage, our goal is to minimize the total expected cost over the problem horizon  $N$ .

$$E\left[c_N(x_N) + \sum_{k=0}^{N-1} c(x_k, u_k, w_k)\right]$$



Dynamic programming can be used to model many applications. Here we will give a water supplying example to illustrate the basic idea of such a problem.

### Example: Water supplying

Suppose we have a water processing plant. We are in the water supplying market as a provider. At the beginning of each month our internal analysts will provide a forecast (distribution) of our clients' demand of clean water. At the end of the month if our storage of clean water can not meet our client's demand we have to buy the difference from other water suppliers at market rate. If there is too much clean water left we have to either waste it or pay for extra storage. Let us denote

- $x_k$ : amount of clean water in our storage at the beginning of the  $k$ th month
- $u_k$ : amount of water we plan to process at the beginning of the  $k$ th month
- $w_k$ : random amount of demand of clean water of the  $k$ th month

Our objective is to minimize our operational cost. We assume the demands are independent of each other. The system evolution is:

$$x_{k+1} = x_k + u_k + w_k$$

At any month other than the final month  $N$ , the cost contains two parts: the cost of processing water and the penalty of insufficient supply or excessive processing. Let  $c(u_k)$  be the cost of processing and  $p(x_k)$  the cost of insufficient supplying or excessive processing respectively. The cost at month  $k$  is  $c(u_k) + p(x_k)$  and the cost of final month is  $p(x_N)$ . So our objective function is

$$E\left\{p(x_N) + \sum_{k=0}^{N-1} (p(x_k) + c(u_k))\right\}$$

Intuitively we are trying to find a solution so that we can satisfy our clients demands with our best effort while keeping the penalties low. It is possible that we decide to buy from the market instead of process water ourselves at some stages. Running a DP algorithm will tell us exactly what we should do when we are at a situation at stage  $k$ . Now let us consider the general mathematical

## 2. LITERATURE REVIEW

---

formulation for a dynamic programming problem. Suppose the system dynamics are

$$x_{k+1} = f_k(x_k, u_k, w_k), \quad k = 0, \dots, N-1$$

The cost at each stage is denoted by  $c_k(x_k, u_k, w_k)$  and  $c_N(x_N)$ . Our objective is to find the optimal control  $u_k$  at each stage associated with  $x_k$  to minimize the expected total cost

$$E\{c_N(x_N) + \sum_{k=0}^{N-1} c_k(x_k, u_k, w_k)\}$$

The basic idea to solve such a problem was first discovered by Richard Bellman: **An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.** This is captured mathematically in the famous Bellman's equation, if we define  $V_k(x_k)$  as the cost-to-go at stage  $k$  from stage  $x_k$ .

$$V_k(x_k) = \min_{u_k} E\{c_k(x_k, u_k, w_k) + V_{k+1}(f(x_k, u_k, w_k))\} \quad (2.2.1)$$

If we apply (2.2.1) backward, starting with  $V_N(x_N) = c_N(x_N)$ , it can be proved that the the solution is optimal and the optimal expected total cost is given by  $V_0(x_0)$ . This is proved by induction. Let  $V_k^*(x_k)$  be the optimal cost-to-go starting from state  $x_k$ . By definition,  $V_N^*(x_N) = c_N(x_N)$ . Assuming that for some  $k$  we have  $V_k^*(x_k) = V_k(x_k)$ , then

$$\begin{aligned} V_k^*(x_k) &= \min_{u_k, \dots, u_{N-1}} E\{c_k(x_k, u_k, w_k) + c_N(x_N) + \sum_{i=k+1}^{N-1} c_i(x_i, u_i, w_i)\} \\ &= \min_{u_k} E\{c_k(x_k, u_k, w_k) + \min_{u_{k+1}, \dots, u_{N-1}} E\{c_N(x_N) + \sum_{i=k+1}^{N-1} c_i(x_i, u_i, w_i)\}\} \\ &= \min_{u_k} E\{c_k(x_k, u_k, w_k) + V_{k+1}^*(f(x_k, u_k, w_k))\} \\ &= \min_{u_k} E\{c_k(x_k, u_k, w_k) + V_{k+1}(f(x_k, u_k, w_k))\} \\ &= V_k(x_k) \end{aligned}$$

Note that using the DP algorithm yields closed-loop solution, which incorporate feedback of previous information. Correspondingly, an open-loop solution does

not consider previous system trajectories and controls. Open loop solutions are often computed ahead of time. Computing closed-loop solutions ahead of time can be tricky (we will learn how to do that in later chapters), as the calculation of control at a certain stage  $k$  requires the information of stage  $k - 1$  and even earlier stages. The DP algorithm is evidence of this: the optimal actions are computed sequentially as the system marches forward. Dynamic programming has many applications. In general, any decision problem that can be formulated as a multi-stage optimization problem with finite state space and control space can be solved optimally using the DP algorithm (approximate DP can solve infinite state space or action space problems, but often approximately). However, the problem of dynamic programming is that the computational requirement grows rapidly with the size of the state space and action space as well as with the horizon. This issue is often known as **curse of dimensionality**. There are different techniques can be tried to mitigate the computational burden, which are beyond the scope of this thesis. Solving a big dynamic program is not always frustrating. Sometimes when special structures appear we can actually obtain elegant closed-form optimal solutions. For example, when the cost function is quadratic and the system evolution equation is linear. We will see how special is this structure in the following subsection.

### 2.2.2 Closed form solution of unconstrained LQ Control

In this subsection we show that when the cost function is quadratic and the system evolution equation is linear, using the DP algorithm we can derive a closed form solution. Let us recall the problem formulation at the beginning of this chapter

$$\begin{aligned}
 \min E \sum_{k=0}^{N-1} \{x_{k+1}^T Q x_{k+1} + u_k^T R u_k\} \\
 s.t. \quad \underline{x}_{k+1} \leq x_{k+1} \leq \bar{x}_{k+1} \\
 \underline{u}_k \leq u_k \leq \bar{u}_k \\
 x_{k+1} = A x_k + B u_k + w_k, \\
 k = 0, \dots, N - 1
 \end{aligned}$$

## 2. LITERATURE REVIEW

---

Now we show the optimal control  $u_k$  of unconstrained LQ Control is an affine function of  $x_k$ . We begin with the base case  $N - 1$ . According to DP algorithm

$$\begin{aligned} u_{N-1} &= \operatorname{argmin} E\{u_{N-1}^T R_{N-1} u_{N-1} + V_N(x_N)\} \\ &= \operatorname{argmin} E\{u_{N-1}^T R_{N-1} u_{N-1} + x_N^T Q_N x_N\} \\ &= \operatorname{argmin} E\{u_{N-1}^T R_{N-1} u_{N-1} \\ &\quad + (A_{N-1} x_{N-1} + B_{N-1} u_{N-1} + w_{N-1})^T Q_N (A_{N-1} x_{N-1} + B_{N-1} u_{N-1} + w_{N-1})\} \end{aligned}$$

We can see that expected cost is a quadratic function. The optimal control  $u_{N-1}$  is the one that gives zero gradient. So if we set the derivative of the right hand side of the above equation to zero we can obtain the optimal control at stage  $N - 1$

$$u_{N-1} = L_{N-1} x_{N-1}, \quad L_{N-1} = -(R_{N-1} + B_{N-1}^T Q_N B_{N-1})^{-1} B_{N-1}^T Q_N A_{N-1}$$

Note that we assume here that both  $R_k$  and  $Q_k$  are positive semidefinite and at least one of them is strictly positive definite so that the inverse exists. Now we look at  $V_{N-1}(x_{N-1})$

$$\begin{aligned} V_{N-1}(x_{N-1}) &= u_{N-1}^T R_{N-1} u_{N-1} + V_N(x_N) \\ &= (L_{N-1} x_{N-1})^T (R_{N-1} + B_{N-1}^T Q_N B_{N-1}) (L_{N-1} x_{N-1}) \\ &\quad + (A_{N-1} x_{N-1})^T Q_N (A_{N-1} x_{N-1}) + \operatorname{Tr}(W_{N-1} Q_N) \end{aligned}$$

We want to express  $V_{N-1}$  in terms of  $x_{N-1}$ . Replace  $u_{N-1}$  in the expression of  $V_{N-1}(x_{N-1})$  and define  $Y_N = Q_N$ , we get

$$\begin{aligned} V_{N-1}(x_{N-1}) &= x_{N-1}^T Y_{N-1} x_{N-1} + \operatorname{Tr}(W_{N-1} Y_N) \\ Y_{N-1} &= A_{N-1}^T (Y_N - Y_N B_{N-1} (B_{N-1}^T Y_N B_{N-1} + R_{N-1})^{-1} B_{N-1}^T Y_N) A_{N-1} + Q_N \end{aligned}$$

Now we need to verify the general case to see if the optimal solution indeed satisfies a recursive structure. Assume that for some  $k$  we have

$$\begin{cases} V_{k+1}(x_{k+1}) &= x_{k+1}^T Y_{k+1} x_{k+1} + \sum_{i=k+1}^{N-1} \operatorname{Tr}(W_i Y_{i+1}) \\ Y_{k+1} &= A_{k+1}^T (Y_{k+2} - Y_{k+2} B_{k+1} (B_{k+1}^T Y_{k+2} B_{k+1} + R_{k+1})^{-1} B_{k+1}^T Y_{k+2}) A_{k+1} \\ &\quad + Q_{k+1} \end{cases}$$

## 2.2 Unconstrained Linear Quadratic Control

---

With  $V_{k+1}$  and  $Y_{k+1}$  we can formulate the subproblem at stage  $k$  by applying Bellman's equation. The next step is to find the optimal control  $u_k$ .

$$\begin{aligned}
 u_k &= \operatorname{argmin} E\{u_k^T R_k u_k + V_{k+1}(x_{k+1})\} \\
 &= \operatorname{argmin} E\{u_k^T R_k u_k + x_{k+1}^T Y_{k+1} x_{k+1} + \operatorname{Tr}(W_{k+2} Y_{k+2})\} \\
 &= \operatorname{argmin} E\{u_k^T R_k u_k \\
 &\quad + (A_k x_k + B_k u_k + w_k)^T Y_{k+1} (A_k x_k + B_k u_k + w_k)\}
 \end{aligned}$$

It is not hard to verify that

$$u_k = L_k x_k \tag{2.2.2}$$

$$L_k = -(B_k^T Y_{k+1} B_k + R_k)^{-1} B_k^T Y_{k+1} A_k \tag{2.2.3}$$

and further we have that

$$V_k(x_k) = x_k^T Y_k x_k + \operatorname{Tr}(W_{k+1} Y_{k+1}) \tag{2.2.4}$$

$$Y_k = A_k^T (Y_k - Y_{k+1} B_k (B_k^T Y_{k+1} B_k + R_k)^{-1} B_k^T Y_{k+1}) A_k + Q_k \tag{2.2.5}$$

Compare the above expressions with the base case we can conclude that (2.2.2)-(2.2.5) give the closed-form optimal solution to the control problem. Note that although the result is for stochastic settings the deterministic version is identical except that the covariance matrices  $W_k$  will not present. (3.1.5) is called discrete time Riccati equation and when  $A_k, B_k, Q_k, R_k$  are constant and satisfy some mild assumptions (the system is controllable and observable), there exists a stationary solution  $Y$  to the following equation

$$Y = A^T (Y - Y B (B^T Y B + R)^{-1} B^T Y) A + Q$$

It suggests that for a time-invariant system the controls tend to converge over time. LQR is a special case where there exists a closed form solution of the DP algorithm and are able to construct a closed loop controller ahead of time. Things are not always nice, as we will see in later chapter. In fact, incorporate feedback in a controller without making the problem intractable is one of the big challenges we face.

### 2.3 Probabilistic Constraints

Probabilistic constraints or chance constraints were first introduced by Charnes, Cooper, and Symonds [11], Miller and Wagner [12]. Generally a probabilistic constraint of linear inequalities can be expressed as

$$\mathbf{P}(Ax \leq b) \geq \alpha \tag{2.3.1}$$

where  $b$  can be a scalar or a vector and correspondingly  $A$  can have one row or multiple rows.  $\alpha$  is the probability requirement. The uncertainty comes from either  $A$  or  $b$  (in some cases, both). The interpretation of the above expression is that the linear inequality  $Ax \leq b$  is satisfied at least with probability  $\alpha$ . So far there is no solver that can directly handle probabilistic constraints. To solve a problem with probabilistic constraints, one must translate it in a way existing solvers can read. The basic idea of dealing with the probabilistic constraints is to replace them with equivalent or conservative deterministic constraints. We will see how to do it in the following sections.

#### 2.3.1 Exact Evaluation of Probabilistic Constraints

There are different techniques to convert probabilistic constraints, corresponding to where the uncertainty is. We will use  $F_b(\cdot)$  to represent the CDF of  $b$ . In (2.3.1) if  $b$  is the uncertainty vector, we can simply substitute (2.3.1) with

$$F_b(Ax) \geq 1 - \alpha$$

The idea seems straightforward. However, the conversion is meaningful only if it is convex and smooth, which is exactly why this is a difficult topic. There are results established by Prekopa, Dupacova and Ruszczyński. To use the above exact conversion, one first needs to verify the concavity of  $F_b$  and its differentiability. It is guaranteed that  $F_b$  is continuously differentiable if the probability density function (PDF) of  $b$  as well as all its one-dimensional marginal PDFs are continuous. Using this approach, we need to know the explicit form of PDF and CDF of  $b$ . This is itself an active area of research and we refer the interested readers to [13], [14] and references therein. The problem of this approach is that

explicit PDFs and CDFs are not always available. Even we know them, not all of them are nice enough to apply the exact evaluation.

### 2.3.2 Approximating Probabilistic Constraints

As we saw that the conditions to exactly convert the probabilistic constraint (2.3.1) are difficult to meet in many cases, people often resort to approximations. There are two main-stream methods to approximate probabilistic constraints, namely sampling and probabilistic approximation. In the sampling approach, the uncertainty is sampled sufficiently and the probabilistic constraint is replaced by multiple deterministic ones where the uncertainty is substituted with sampled realizations.

$$Ax \leq b^i$$

where  $b^i$  are the realizations and  $i = 1, \dots, N_s$ .  $N_s$  is the number of realizations. The catch here is the number of realizations one needs to approximate a probabilistic constraint. Let  $\delta$  be an arbitrary number between 0 and 1. It is proved that if  $N_s$  satisfies

$$\text{ceil}[2n(1 - \alpha)^{-1} \log(12/(1 - \alpha)) + 2(1 - \alpha)^{-1} \log(2/\delta + 2n)]$$

then the probabilistic constraint will be satisfied with confidence level  $1 - \delta$ . This is proved by using independent Bernoulli trials. It is quite general and it is easy to implement. However, if  $\alpha$  is large and  $\delta$  is small,  $N_s$  can be huge, which can heavily drag down the solving process. To learn the details about how to set up the sampling procedure and how to compute the number of realizations needed, one can refer to [15] and [16], [17], and [18]. Sampling is an easy-to-apply method, however, it is not efficient to be used in large problems. In this thesis, we adopt the probabilistic approximation method, which is much faster with some sacrifice to accuracy.

In fact, (2.3.1) can have different variations depends on which side of the inequality contains the uncertainty. In the optimization publications one often finds cases where  $A$  is a random row vector whose components are independent and  $b$  a constant. We will briefly discuss how to treat them. Let  $\bar{a}$  be the mean

## 2. LITERATURE REVIEW

---

vector of  $A$  and  $\tilde{a}$  the vector of variance (of each component). The probabilistic constraint can be replaced by

$$\bar{a} + \Omega \left( \sum_{i=1}^n \tilde{a}_i^2 x_i^2 \right)^{\frac{1}{2}} \leq b$$

where  $\Omega$  is a carefully chosen constant. This approach is called Bernstein approximation and was first used in [19]. In [20], the authors proposed a way to test the tightness of the approximation and generalized the result. Similar approaches are found in [21].

Now we look at the case in which the uncertainty is on the right hand side of the inequality. One important thing to notice here is that if  $b$  is a scalar or a vector has a significant impact on the complexity of converting the probabilistic constraint. For example, in the case where the probability requirement  $\alpha$  is 0.5, the probabilistic constraint can be simply replaced by  $Ax \leq 0$ . Let us use  $F(\cdot)$  as the cumulative distribution function of the standard normal random variable. We all know that  $F(0) = 0.5$ . However, joint probabilistic constraints (when  $b$  is a vector) are totally different. For example, suppose we are to replace the joint probabilistic constraint

$$\mathbf{P} \left( \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} x \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \right) \geq \alpha$$

where  $b_1, b_2$  are i.i.d. standard normal variables, we know that there exist at least two realizations  $[0; 3.9]$  and  $[3.9; 0]$  (note that  $F(3.9) = 1$ ) that can be used to replace the probabilistic constraint.

$$\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} x \leq \begin{bmatrix} 3.9 \\ 0 \end{bmatrix}, \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} x \leq \begin{bmatrix} 0 \\ 3.9 \end{bmatrix}$$

So we have two different ways to replace the probabilistic constraints but they yield different feasible regions and hence may lead to quite different optimal objective values. The the problem is: do we know which one is the one that gives the best solution? In general when  $b$  is a continuous multi-dimensional random vector there are infinitely many of such valid realizations and this is a problem because it is not clear which realization we should choose before we have knowledge of the optimal objective of the optimization. One way to evaluate a joint probabilistic constraint is to approximate it with scalar probabilistic constraints.



This is approach that was mentioned in [20] and used in [22]. The basic idea is to use Boole's inequality. Suppose  $A$  is a matrix and  $b$  a vector and  $A_j$  represents the  $j$ th row of  $A$  and  $b_j$  the  $j$ th component respectively, we have

$$\mathbf{P}(Ax \leq b) = \mathbf{P}\left(\bigcap_j \{A_j x \leq b_j\}\right) = \mathbf{P}\left(\bigcup_j \{A_j x \geq b_j\}\right)$$

By Boole's inequality

$$\mathbf{P}\left(\bigcup_j \{A_j x \geq b_j\}\right) \leq \sum_j \mathbf{P}(A_j x \geq b_j)$$

Let us look at an example of applying this approach. Suppose we are trying to approximate

$$\mathbf{P}\left(Ax \leq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}\right) \geq 0.8$$

It is equivalent to

$$\mathbf{P}(A_1 x \geq b_1 \text{ or } A_2 x \geq b_2) \leq 0.2$$

Which can be safely replaced by

$$\mathbf{P}(A_1 x \leq b_1) \geq 1 - \alpha_1, \quad \mathbf{P}(A_2 x \leq b_2) \geq 1 - \alpha_2$$

As long as  $\alpha_1 + \alpha_2 \leq 0.2$ . This approach is easy to implement, however, it introduced another problem: how to assign the individual probabilities  $\mathbf{P}(A_j x \geq b_j)$ ? In [22] they were left as decision variables. It is reported that Boole's inequality is sharp when the probability requirement  $\alpha$  is close to 1 and the correlations between the random components ( $A_j$  for left-hand-side uncertainty or  $b_j$  for right-hand-side uncertainty) are weak.

## 2.4 Constrained Stochastic Linear Quadratic Control

Constrained stochastic control is an active research area recently and has drawn much attention within the optimization and control community [23],[24], [25], [26], [27], [28]. In chapter 2 we introduce unconstrained stochastic LQ control and how it can be solved recursively by dynamic programming. In this chapter we will show a couple of stochastic LQ control problems with constraints, which are much more difficult to solve in general. Stochastic LQ control is a relatively new area and there are quite a couple of different formulations of the problem because of the introduction of randomness. Any problem formulation is a combination of a set of chosen objective function, constraints and system dynamics. Unlike the deterministic versions, in stochastic LQ control, there is more than one way to enforce a constraint on states or controls such as constraints on expected value or covariance. The noise term in the system dynamics can be additive, multiplicative or even is embedded in the coefficient matrices. We point out here that in this thesis we consider the additive noise model of system dynamics if there is no special statement. Since the goal is to sample the most representative results in this area and develop the core content of this thesis, we will be explicit about the problem formulations and explore the differences.

### 2.4.1 Generalized Linear Quadratic Control

It is well-known that unconstrained LQ control has nice recursive solution. In this section we will show how to solve a stochastic LQ control problem with power constraints recursively [29]. The problem we are trying to solve is

$$\begin{aligned} \min \quad & \sum_{k=0}^{N-1} \{x_{k+1}^T Q x_{k+1} + u_k^T R u_k\} \\ \text{s.t.} \quad & x_{k+1} = A x_k + B u_k + w_k, \\ & E \begin{bmatrix} x_{k+1} \\ u_{k+1} \end{bmatrix}^T Q_{k+1} \begin{bmatrix} x_{k+1} \\ u_{k+1} \end{bmatrix} \leq \gamma_{k+1} \\ & k = 0, \dots, N-1 \end{aligned}$$

## 2.4 Constrained Stochastic Linear Quadratic Control

---

The quadratic constraint is also called power constraint, which can be used in stability analysis. Note that since we only place constraint on the system states, only the upper left part of  $Q_N$  is non-zero. Recall that we assume that  $w_k$  has zero mean and the initial state  $x_0$  is known. If we define

$$V(k) = E \begin{bmatrix} x_k \\ u_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T, F = \begin{bmatrix} I & 0 \end{bmatrix}$$

The system dynamics can be written as

$$\begin{aligned} FV(k+1)F^T &= E \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} x_{k+1} \\ u_{k+1} \end{bmatrix} \begin{bmatrix} x_{k+1} \\ u_{k+1} \end{bmatrix}^T \begin{bmatrix} I & 0 \end{bmatrix}^T \\ &= E x_{k+1} x_{k+1}^T \\ &= E (Ax_k + Bu_k + w_k)(Ax_k + Bu_k + w_k)^T \\ &= E \left\{ \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} A & B \end{bmatrix}^T + w_k w_k^T \right\} \\ &= \begin{bmatrix} A & B \end{bmatrix} V(k) \begin{bmatrix} A & B \end{bmatrix}^T + W_k \end{aligned}$$

and the initial condition is  $FV(0)F^T = x_0 x_0^T$ . Let  $V_{xx}(k)$  be the upper left part of  $V(k)$  (which is  $E x_k x_k^T$ ) and  $\tilde{Q} = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix}$ , the optimization problem can be written as **(P 3.1)**

$$\begin{aligned} \min_{V(k) \succeq 0} & Tr(QV_{xx}(N)) + \sum_{k=0}^{N-1} Tr(\tilde{Q}V(k)) \\ \text{s.t.} & FV(0)F^T \succeq x_0 x_0^T \\ & FV(k+1)F^T \succeq \begin{bmatrix} A & B \end{bmatrix} V(k) \begin{bmatrix} A & B \end{bmatrix}^T + W_k \\ & Tr(Q_{k+1}V(k+1)) \leq \gamma_{k+1} \\ & k = 0, \dots, N-1 \end{aligned}$$

Note that **(P 3.1)** is convex in  $V(k)$ . We would like to see that if there is a better way to solve the problem other than throwing it to a semi-definite programming

## 2. LITERATURE REVIEW

---

solver. We can dualize the constraints into the objective. The dual function is

$$\begin{aligned}
L(S, \lambda) = \min_{V(k) \succeq 0} & Tr(QV_{xx}(N)) + \sum_{k=0}^{N-1} \{Tr(\tilde{Q}V(k)) \\
& - Tr\left(S(k+1)(FV(k+1)F^T - [A \ B]V(k)[A \ B]^T - W_k)\right)\} \\
& - Tr\left(S(0)(FV(0)F^T - x_0x_0^T)\right) + \\
& \sum_{k=0}^{N-1} \lambda_{k+1} (Tr(Q_{k+1}V(k+1) - \gamma_{k+1})
\end{aligned}$$

where  $S(k) \in \mathbb{S}^n$  and  $\lambda \succeq 0$ . After recollecting the terms we have

$$\begin{aligned}
L(S, \lambda) = \min_{V(k) \succeq 0} & Tr\left((\tilde{Q} - F^T S(0)F + [A \ B]^T S(1)[A \ B])V(0)\right) + Tr(S(0)x_0x_0^T) \\
& + \sum_{k=1}^{N-1} Tr\left(Tr(\tilde{Q} - F^T S(k)F + [A \ B]^T S(k+1)[A \ B] + \lambda_k Q_k)V(i)\right) \\
& + Tr\left((F^T(Q - S(N))F + \lambda_N Q_N)V(N)\right) \\
& + \sum_{k=0}^{N-1} (Tr(S(k+1)W_i) - \lambda_{k+1}\gamma_{k+1})
\end{aligned}$$

Now if we define

$$J(0) = \tilde{Q} - F^T S(0)F + [A \ B]^T S(1)[A \ B] \quad (2.4.1)$$

$$J(k) = \tilde{Q} - F^T S(k)F + [A \ B]^T S(k+1)[A \ B] + \lambda_k Q_k, \quad k = 1, \dots, N-1 \quad (2.4.2)$$

$$J(N) = F^T(\tilde{Q} - S(N))F + \lambda_N Q_N \quad (2.4.3)$$

We know that  $J(k)$  must be positive semi-definite in a feasible solution, where  $k = 0, \dots, N$ , otherwise  $\max L(S, \lambda)$  will be unbounded. Since **(P 3.1)** is convex we can conclude that the optimal dual solution gives the same objective value as the optimal primal solution does. We can actually get the dual solution by solving a sequence of semi-definite programs and construct the dual variables so that  $J(k) = 0$ . Let us start with stage N and examine the following subproblem

$$\begin{aligned}
& \max_{S(N), \lambda_N} Tr(J(N)V(N)) + Tr(S(N)W_{N-1}) - \lambda_N \gamma_N \\
& \text{s.t. } F^T(Q - S(N))F + \lambda_N Q_N \succeq 0
\end{aligned}$$

## 2.4 Constrained Stochastic Linear Quadratic Control

---

Since  $V(N), W_{N-1}$  are positive semi-definite, we can conclude that the optimal  $S(N)$  given any fixed  $\lambda_N$  is

$$S(N) = Q + \lambda_N Q_N^{xx}, J(N) = 0$$

where  $Q_N = \begin{bmatrix} Q_N^{xx} & 0 \\ 0 & 0 \end{bmatrix}$ . So the optimal  $S(N)$  can be obtained by solving the following semi-definite program

$$\max_{S(N), \lambda_N} Tr(S(N)W_{N-1}) - \lambda_N \gamma_N \quad (2.4.4)$$

$$s.t. \quad S(N) \preceq Q + \lambda_N Q_N^{xx} \quad (2.4.5)$$

Once we have  $S(k+1)$ , we can solve for  $S(k)$ , for any  $k$  between 0 and  $N$ . Let us look at the following optimization

$$\begin{aligned} & \max_{S(k), \lambda_k} Tr(J(k)V(k)) + Tr(S(k)W_{k-1}) - \lambda_k \gamma_k \\ & s.t. \quad \tilde{Q} + \lambda_k Q_k + \begin{bmatrix} A^T S(k+1)A - S(k) & A^T S(k+1)B \\ B^T S(k+1)A & B^T S(k+1)B \end{bmatrix} \succeq 0 \end{aligned}$$

If we define

$$P(k) = \tilde{Q} + \lambda_k Q_k \quad (2.4.6)$$

$$L(k) = (P_{xu}(k) + A^T S(k+1)B) (P_{uu}(k) + B^T S(k+1)B)^{-1} \quad (2.4.7)$$

and assume that  $(B^T S(k+1)A + P_{ux}(k))$  is non-singular, we can apply the Schur complement condition to the constraint, which leads to

$$(P_{xx}(k) + A^T S(k+1)A - S(k)) - L(k) (P_{uu}(k) + B^T S(k+1)B) L(k)^T \succeq 0 \quad (2.4.8)$$

Now we can obtain the optimal  $S(k)$  given  $S(k+1)$  by solving this problem

$$\max_{S(k), \lambda_k} Tr(S(k)W_{k-1}) - \lambda_k \gamma_k \quad (2.4.9)$$

$$s.t. \quad S(k) \preceq (P_{xx}(k) + A^T S(k+1)A) - L(k) (P_{uu}(k) + B^T S(k+1)B) L(k)^T \quad (2.4.10)$$

We can drop the term  $Tr(J(k)V(k))$  because we know the optimal solution of the above problem will satisfy the equality of (2.4.8), and thus minimize

## 2. LITERATURE REVIEW

---

$Tr(J(k)V(k))$  for any fixed  $V(k)$ . This procedure can be repeated until we get all the optimal dual variables. The next step is to recover the optimal primal solution. Now we look back at the dual function with all the optimal dual variables known.

$$L(S, \lambda) = \min_{V(k) \succeq 0} \sum_{k=0}^N Tr(J(k)V(k)) + Tr(S(0)x_0x_0^T) \\ + \sum_{k=0}^{N-1} \{Tr(S(k+1)W(k)) - \lambda_{k+1}\gamma_{k+1}\}$$

Recall that with the optimal dual variables we see that  $J(k)$  has the following shape

$$J(k) = \begin{bmatrix} L(k)Y(k)L(k)^T & L(k)Y(k) \\ Y(k)L(k)^T & Y(k) \end{bmatrix}$$

where  $Y(k) = P_{uu}(k) + B^T S(k+1)B$ . Now we construct the optimal primal solution that satisfies  $Tr(J(k)V(k)) = 0$ . For any given  $V_{xx}(k)$ , we define

$$V(k) = \begin{bmatrix} V_{xx}(k) & -L(k)V_{xx}(k) \\ -V_{xx}(k)L^T(k) & L(k)V_{xx}(k)L(k)^T \end{bmatrix}$$

Since the initial condition  $V_{xx}(0)$  is given, we can obtain the optimal primal solution by

$$\begin{cases} V_{xx}(0) & = x_0x_0^T \\ V(k) & = \begin{bmatrix} V_{xx}(k) & -V_{xx}(k)L(k)^T \\ -L(k)V_{xx}(k) & L(k)^T V_{xx}(k)L(k) \end{bmatrix} \\ V_{xx}(k+1) & = \begin{bmatrix} A & B \end{bmatrix} V_{xx}(k) \begin{bmatrix} A & B \end{bmatrix}^T + W_k \end{cases}$$

where  $L(k)$  is defined in (2.4.7) and  $k = 0, \dots, N-1$ . The optimal cost is

$$Tr(S(0)x_0x_0^T) + \sum_{k=0}^{N-1} \{Tr(S(k+1)W(k)) - \lambda_{k+1}\gamma_{k+1}\}$$

And the optimal control law is given by

$$u_k = V_{xu}(k)V_{xx}^{-1}(k)x_k = -L(k)x_k, \quad k = 0, \dots, N-1.$$

This method is elegant as it only needs to solve up to  $N$  subproblems, whose size is in proportion to the dimension of system states. However, the weakness

of this approach is that the nice property that the dual function of the objective can be broken into subproblems and solved recursively does not preserve when the formulation changes. In other words, although the method is powerful it is not easy to be applied to other similar problems. For more details, please refer to [29].

### 2.4.2 Semi-definite Programming Formulation

In this section we introduce a general way to formulate stochastic LQ control as a semi-definite program and how different constraints can be added [30],[31],[32]. We will begin with the reformulation of the unconstrained stochastic LQR. Recall the problem formulation is

$$\begin{aligned} \min \quad & \sum_{k=0}^{N-1} \{x_{k+1}^T Q x_{k+1} + u_k^T R u_k\} \\ \text{s.t.} \quad & x_{k+1} = A x_k + B u_k + w_k \\ & k = 0, \dots, N-1 \end{aligned}$$

First we need to introduce some new notations

$$\bar{x}_k = E x_k; \quad \Sigma(k) = E[(x_k - \bar{x}_k)(x_k - \bar{x}_k)^T]$$

In this subsection we also assume the control has the following form

$$u_k = \bar{u}_k + K(k)(x_k - \bar{x}_k)$$

where in fact  $\bar{u}_k$  is the mean of  $u_k$  and  $K(k)$  is a gain matrix., both are to be determined. With the new notations the system dynamics become, assuming the disturbance  $w_k$  has zero mean.

$$\bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k \tag{2.4.11}$$

$$\Sigma(k+1) = (A + BK(k)) \Sigma(k) (A + BK(k))^T + W_k \tag{2.4.12}$$

The initial condition is

$$\bar{x}_0 = x_0; \quad \Sigma(0) = 0 \tag{2.4.13}$$

## 2. LITERATURE REVIEW

---

Note that (2.4.11) can not be processed by a semi-definite programming solver as it is not a linear matrix inequality (LMI). To address this issue we define  $\Psi(k) = K(k)\Sigma(k)$  and using Schur complement lemma we rewrite (2.4.11) as

$$\begin{bmatrix} \Sigma(k+1) & (A\Sigma(k) + B\Psi(k)) \\ (A\Sigma(k) + B\Psi(k))^T & \Sigma(k) \end{bmatrix} \succeq 0 \quad (2.4.14)$$

Note the in (2.4.12)  $\Sigma(k+1)$  becomes an upper bound instead of the exact expression of  $\Sigma(k)$  in (2.4.11), but we the inequality will be tight in the optimal solution. What we need to do next is to use the new notations to represent the terms in the objective. We have

$$\begin{aligned} & E \begin{bmatrix} x_k \\ u_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \\ &= E \begin{bmatrix} \bar{x}_k + (x_k - \bar{x}_k) \\ \bar{u}_k + K(k)(x_k - \bar{x}_k) \end{bmatrix} \begin{bmatrix} \bar{x}_k + (x_k - \bar{x}_k) \\ \bar{u}_k + K(k)(x_k - \bar{x}_k) \end{bmatrix}^T \\ &= \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix}^T + \begin{bmatrix} \Sigma(k) \\ \Psi(k) \end{bmatrix} \Sigma(k)^+ \begin{bmatrix} \Sigma(k) \\ \Psi(k) \end{bmatrix}^T \end{aligned}$$

where  $\Sigma(k)^+$  is the pseudo inverse of  $\Sigma(k)$ . If we define  $V(k)$  as the upper bound of the above expression. We have

$$V(k) \succeq \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix}^T + \begin{bmatrix} \Sigma(k) \\ \Psi(k) \end{bmatrix} \Sigma(k)^+ \begin{bmatrix} \Sigma(k) \\ \Psi(k) \end{bmatrix}^T \quad (2.4.15)$$

which again can be turned into a LMI using Schur complement lemma.

$$\begin{bmatrix} V(k) & \begin{bmatrix} \Sigma(k) \\ \Psi(k) \end{bmatrix} & \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} \\ \begin{bmatrix} \Sigma(k)^T & \Psi(k)^T \end{bmatrix} & \Sigma(k) & 0 \\ \begin{bmatrix} \bar{x}_k^T & \bar{u}_k^T \end{bmatrix} & 0 & 1 \end{bmatrix} \succeq 0 \quad (2.4.16)$$

The above LMI is for  $k = 1, \dots, N-1$ . As we assumed that the initial state  $x_0$  is known and we can also conclude that  $u_0$  has no random component. Thus we have

$$\begin{bmatrix} V(0) & \begin{bmatrix} x_0 \\ \bar{u}_0 \end{bmatrix} \\ \begin{bmatrix} x_0^T & \bar{u}_0^T \end{bmatrix} & 1 \end{bmatrix} \succeq 0 \quad (2.4.17)$$

And for  $k = N$  we have

$$\begin{bmatrix} V_{xx}(N) - \Sigma(N) & \bar{x}_N \\ \bar{x}_N & 1 \end{bmatrix} \succeq 0 \quad (2.4.18)$$



Using these new notation, the objective function becomes

$$\sum_{i=0}^{N-1} Tr \left( V(k) \tilde{Q} \right) + Tr(V_{xx}(N)Q) \quad (2.4.19)$$

Recall that  $\tilde{Q} = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix}$ . So the unconstrained SLQR problem is

$$\begin{aligned} & \min_{V(k), \Sigma(k), \Psi(k), \bar{x}_k, \bar{u}_k} \quad (2.4.18) \\ & s.t. \quad (2.4.10), (2.4.12) - (2.4.13), (2.4.15) - (2.4.17) \end{aligned}$$

Now we consider how to add constraints. Suppose the constraint we want to add has the following form

$$E \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T C(k) \begin{bmatrix} x_k \\ u_k \end{bmatrix} + d_k^T E \begin{bmatrix} x_k \\ u_k \end{bmatrix} \leq \gamma_k \quad (2.4.20)$$

It can be reformulated as

$$Tr(V(k)C(k)) + d_k^T \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} \leq \gamma_k \quad (2.4.21)$$

In [32], the authors proposed a way to convert a probabilistic constraint to a constraint has the form of (2.4.21). Suppose we are considering the following scalar case probabilistic constraint

$$\mathbf{P}(a^T x_k + c^T u_k \leq b) \geq \alpha$$

Using the idea of normal approximation, we can view  $a^T x_k + c^T u_k$  as a normal random variable  $z$ . And we have

$$\mathbf{P}\left(\frac{z - \bar{z}}{\sigma_z} \leq \frac{b - \bar{z}}{\sigma_z}\right) \geq \alpha$$

which is equivalent to

$$\frac{b - \bar{z}}{\sigma_z} \geq \mathcal{N}^{-1}(\alpha)$$

That is

$$(\mathcal{N}^{-1}(\alpha))^2 \sigma_z^2 \leq (b - \bar{z})^2$$

## 2. LITERATURE REVIEW

---

where  $\bar{z}$  is the mean of  $z$  and  $\sigma_z$  the standard deviation of  $z$ . We can express the above inequality in terms of moments of  $x$  and  $u$  as

$$(\mathcal{N}^{-1}(\alpha))^2 E \left( \begin{bmatrix} x_k - \bar{x}_k \\ u_k - \bar{u}_k \end{bmatrix}^T \begin{bmatrix} a \\ c \end{bmatrix} \begin{bmatrix} a \\ c \end{bmatrix}^T \begin{bmatrix} x_k - \bar{x}_k \\ u_k - \bar{u}_k \end{bmatrix} \right) \leq \left( b - \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} \right)^2$$

which can be replaced with

$$(\mathcal{N}^{-1}(\alpha))^2 \text{Tr}(V(k)M) \leq \left( b - \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} \right)^2 \quad (2.4.22)$$

where  $M$  is defined as

$$M = \begin{bmatrix} a \\ c \end{bmatrix} \begin{bmatrix} a \\ c \end{bmatrix}^T$$

Since  $V(k)$  and  $\bar{z}$  are both variables, (2.4.22) is not a convex constraint. We can approximate it with

$$(\mathcal{N}^{-1}(\alpha))^2 \text{Tr}(V(k)M) \leq b^2 - 2b\bar{z} + 2\bar{z}_0(\bar{z} - \bar{z}_0) \quad (2.4.23)$$

where  $\bar{z}_0 = a^T \bar{x}_k + c^T \bar{u}_k$  (which in a MPC setting can be computed using the information obtained from the previous LQR solution). Constraints like (2.4.21) and (2.4.23) can be added to the semi-definite programming formulation of the unconstrained LQR, and the resulting problem can be solved by a conventional SDP solver. This formulation of problem has the power of flexibility, as we showed that different constraints can be attached easily. However, we can see this formulation introduces extra variables so is more suitable for small or medium size problems.

### 2.4.3 A Tractable approximation of probability constrained stochastic LQR

In this subsection we introduce a method to approximate a stochastic LQR with probability constraints. This approach is different from the previous ones in that it approximates the probability constraints using linear inequalities and the value function is also a certainty equivalent version [33]. We summarize a slightly

different version here (P 2.4.3.1)

$$\begin{aligned}
 & \min \sum_{k=0}^{N-1} \{ \bar{x}_{k+1}^T Q \bar{x}_{k+1} + \bar{u}_k^T R \bar{u}_k \} \\
 & \text{s.t. } x_{k+1} = Ax_k + Bu_k + w_k, \\
 & \quad \mathbf{P}(T_k x_k \leq b_k) \geq \alpha \\
 & \quad w_k \sim \mathcal{N}(0, I) \\
 & \quad k = 0, \dots, N-1
 \end{aligned}$$

Note that in (P 2.4.3.1), the objective is a convex function. What complicates things is the probabilistic constraint  $\mathbf{P}(T_k x_k \leq b_k) \geq \alpha$ . Generally an exact, convex conversion is difficult to find for such a constraint. So people seek ways to approximate it. We will show how to do this using the method in [34],[33]. Let us introduce a more concise presentation. With the notations defined in section 2.1, the system dynamics can be rewritten as

$$\mathcal{X} = \mathbb{F}x_0 + \mathbb{H}\mathcal{U} + \mathbb{G}\mathcal{W}$$

where  $\mathbb{G}$  and  $\mathcal{W}$  are defined as

$$\mathbb{G} = \begin{bmatrix} I & & & & \\ A & I & & & \\ \vdots & & \ddots & & \\ A^{N-1} & A^{N-2} & \dots & I & \end{bmatrix}, \quad \mathcal{W} = \begin{bmatrix} w_0 \\ \vdots \\ w_{N-1} \end{bmatrix}$$

Our control  $\mathcal{U}$  has an affine disturbance feedback structure

$$\mathcal{U} = \bar{\mathcal{U}} + \mathbb{L}\mathcal{W}$$

where  $\mathbb{L}$  is the gain matrix given by

$$\mathbb{L} = \begin{bmatrix} 0 & & & & \\ L_{(1,0)} & 0 & & & \\ L_{(2,0)} & L_{(2,1)} & & & \\ \vdots & \ddots & \ddots & & \\ L_{(N-1,0)} & \dots & L_{(N-1,N-2)} & 0 & \end{bmatrix}$$

## 2. LITERATURE REVIEW

---

Note that  $\mathbb{L}$  has a lower triangular structure meaning that a control output at a certain stage can only use the revealed disturbances at previous stages. Now  $T_k x_k \leq b_k, k = 1, \dots, N$  can be written as

$$T\mathbb{F}x_0 + T\mathbb{H}\bar{\mathbf{u}} + T\mathbb{H}\mathbb{L}\mathcal{W} + T\mathbb{G}\mathcal{W} \leq b$$

For simplicity we define

$$g(\mathbb{L}, \bar{\mathbf{u}}, \mathcal{W}) = T\mathbb{F}x_0 + T\mathbb{H}\bar{\mathbf{u}} + T\mathbb{H}\mathbb{L}\mathcal{W} + T\mathbb{G}\mathcal{W} - b$$

If we define  $g_i$  represents the  $k$ th row of  $g$  then for any stage  $k$  between 1 and  $N$  by Boole's inequality we get

$$\begin{aligned} \sum_{k=in+1}^{(i+1)n} \alpha_k &= \sum_{k=in+1}^{(i+1)n} \mathbf{P}(g_k(\mathbb{L}, \bar{\mathbf{u}}, \mathcal{W}) \leq 0) \\ &\geq \mathbf{P}\left(\bigcup_{k=in+1}^{(i+1)n} g_k(\mathbb{L}, \bar{\mathbf{u}}, \mathcal{W}) \leq 0\right) \\ &= \mathbf{P}(T_i x_i \leq b_i) = \alpha \end{aligned}$$

In many real-world applications, the noise  $\mathcal{W}$  is often bounded. So it is reasonable for one to assume that there exists such  $\Omega$  that  $\|\mathcal{W}\| \leq \Omega$ . Under such assumption, it can be seen that  $g_i \leq 0$  can be replaced by

$$\max_{\|\mathcal{W}\| \leq \Omega} g_i(\mathbb{L}, \bar{\mathbf{u}}, \mathcal{W}) \leq 0$$

The remaining question is how to choose this  $\Omega$ . The following theorem 34 gives us a clue. It is also used to approximate the probabilistic constraints.

**Theorem 1.** *Let random variable  $w \sim \mathcal{N}(0, I)$  and be in  $\mathbf{R}^m$ . And let  $x$  be a  $n$  dimensional vector and*

$$\begin{cases} \Delta D = a^0 + \Delta A w \\ D^0 = b^0 + \Delta b^T w \end{cases}$$

where  $a^0 \in \mathbf{R}^n$ ,  $\Delta A \in \mathbf{R}^{n \times m}$ ,  $b^0 \in \mathbf{R}$  and  $\Delta b \in \mathbf{R}^m$ .

$$\mathbf{P}(D^0 + \Delta D^T x \geq 0) \leq \Omega \sqrt{e} \exp\left(-\frac{\Omega^2}{2}\right)$$

## 2.4 Constrained Stochastic Linear Quadratic Control

---

We can see that  $g_i$  can be rewritten as the form  $D^0 + \Delta D^T x$ . As a summary, the probabilistic constraint can be conservatively approximated by

$$\max_{\|v\| \leq \Omega} g(\mathbb{L}, \bar{\mathbf{u}}, v) \leq 0$$

where  $\Omega$  is determined by  $\Omega \sqrt{e} \exp(-\frac{\Omega^2}{2}) \geq \alpha$ . But obviously the above expression cannot be used directly to substitute a constraint in an optimization problem because itself is not a constraint. This can be addressed by using duality theory. Let us look at the above inequality row by row

$$\begin{aligned} \max (TFx_0 + TH\bar{\mathbf{u}} - b)_i + (THL + TG)_k v &\leq 0 \\ \text{s.t.} \quad -\Omega \mathbf{1} &\leq v \leq \Omega \mathbf{1} \end{aligned}$$

where the  $(\cdot)_i$  operator on a matrix means the  $i$ th row. Now consider  $v$  as the variable, using duality we can see it is equivalent to

$$\begin{aligned} \min (TFx_0 + TH\bar{\mathbf{u}} - b)_i + \Omega(\mathbf{1}^T \mu_i + \mathbf{1}^T \lambda_i) &\leq 0 \\ \text{s.t.} \quad (THL + TG)_i &= \mu_i - \lambda_i \\ \mu_i, \lambda_i &\geq 0 \end{aligned}$$

where  $i = 1, \dots, nN$ . The above expression can be used to replace the probabilistic constraint. We can now formulate the convex programming problem **P 2.4.3.2** after converting the probabilistic constraint

$$\begin{aligned} \min (\mathbb{F}x_0)^T \mathbb{Q}(\mathbb{F}x_0) + 2(\mathbb{F}x_0)^T \mathbb{Q}\mathbb{H}\bar{\mathbf{u}} + \bar{\mathbf{u}}^T (\mathbb{R} + \mathbb{H}^T \mathbb{Q}\mathbb{H})\bar{\mathbf{u}} \\ \text{s.t.} \quad (TFx_0 + TH\bar{\mathbf{u}} - b)_i + \Omega(\mathbf{1}^T \mu_i + \mathbf{1}^T \lambda_i) &\leq 0 \\ (THL + TG)_i &= \mu_i - \lambda_i \\ \mu_i, \lambda_i &\geq 0 \end{aligned}$$

where  $i = 1, \dots, nN$  and  $\mathbb{Q}, \mathbb{R}$  are  $nN \times nN$  are block diagonal matrices with  $Q$  and  $R$  at the diagonal respectively. The above is a quadratic program and can be solved by conventional solvers.

## 2. LITERATURE REVIEW

---

# 3

## Feedback controller design for Probability Constrained SLQR

In the previous chapter we introduced the probability constrained SLQR and sampled related techniques in the literature including the generalized SLQR method, the semi-definite programming approach and the tractable approximation approach. Those are the state-of-the-art results and their strengths and weaknesses were discussed. In this chapter, we will present the original work of the thesis, from a perspective different from the proposed ones. In particular we will design a disturbance feedback controller and two state feedback controllers for two differently formulated probability constrained SLQR problems. The goal of our work is to develop tractable, reliable and fast algorithms.

### 3.1 A Chebyshev bound approach to convert probabilistic constraints

In this section we present an approximation for the probabilistic constraint

$$\mathbf{P} \left( \sum_{k=0}^N T_k x_k \leq b \right) \geq \alpha.$$

Where  $T_k$  is a constant matrix and  $b$  is a random vector. As we introduced in the previous chapter, such a constraint can be replaced by using the cumulative

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

density function of  $b$  or the sampling technique. However, we do not assume the distribution of  $b$  so we cannot apply the first approach. And we tend to not use the sampling approach either because the large number of deterministic constraints it would generate. We will approximate it using the multi-dimensional Chebyshev inequality. As we will show, the Chebyshev-based constraint can be handled naturally within the framework of stochastic LQR. The Chebyshev constraint is an inner approximation, in the sense that control laws that satisfy this constraint are guaranteed to satisfy the original probabilistic constraint.

The goal for this section is to derive a method to approximate multidimensional chance constraints, which serves as the foundation of our control algorithms in the later sections. The following theorem provides a multi-dimensional Chebyshev inequality [35]:

**Theorem 3.1.** *Let  $z$  be a random vector in  $\mathbb{R}^d$  and  $\mathcal{S}$  a subset of  $\mathbb{R}^d$  defined by a collection of linear inequalities. If  $P \in \mathbb{S}^d$ ,  $q \in \mathbb{R}^d$  and  $r \in \mathbb{R}$  are chosen so that*

$$\{y \in \mathbb{R}^d \mid y^T P y + 2q^T y + r \leq 1\}$$

*is an inscribed ellipsoid of set  $\mathcal{S}$ , then we have*

$$1 - \mathbf{E}[z^T P z + 2q^T z + r] \leq \mathbf{P}(z \in \mathcal{S})$$

**Proof.** Let  $f(z) = z^T P z + 2q^T z + r$ . Then  $f(z) \geq 0$  for  $z \in \mathcal{S}$  and  $f(z) \geq 1$  for any  $z \in \mathcal{S}^c$ , where  $\mathcal{S}^c$  is the complement of  $\mathcal{S}$ . Let  $I_{\mathcal{S}^c}(\cdot)$  be the indicator function on  $\mathcal{S}^c$ , then

$$f(z) \geq I_{\mathcal{S}^c}(z)$$

Therefore,

$$\begin{aligned} \mathbf{E}[f(z)] &\geq \mathbf{E}[I_{\mathcal{S}^c}(z)] \\ &= \mathbf{P}(z \in \mathcal{S}^c), \end{aligned}$$

which is the same as

$$1 - \mathbf{E}[z^T P z + 2q^T z + r] \leq \mathbf{P}(z \in \mathcal{S}).$$

■



### 3.1 A Chebyshev bound approach to convert probabilistic constraints

---

The underlying probability distribution of  $z$  does not affect the nature of the bound. In other words, the bound is valid for any distribution, or an ambiguous distribution (see [36] for example), as long as its first and second moments coincide with the given ones.

The above theorem gives a lower bound on the probability that  $z$  falls into  $\mathcal{S}$ , or an upper bound on the probability that  $z$  falls outside  $\mathcal{S}$ . However, this theorem does not mention the way for selecting the inscribed ellipsoid. In [37] the authors proposed a way using this theorem to approximate probabilistic constraints. There are many possible ways to choose the ellipsoid and as one can expect, the quality of the bound largely relies on the choice of the ellipsoid. In our algorithms, we will use the maximum volume inscribed ellipsoid (see [35]). This ellipsoid can be easily computed using conventional semidefinite programming (SDP) solvers.

For the general case of approximating a linear constraint, suppose we have a probabilistic constraint of the form

$$\mathbf{P}(Tz \leq b) \geq \alpha.$$

Let  $T(i)$  be the  $i$ -th row of  $T$ ,  $b_i$  the  $i$ -th component of  $b$ , and  $\mathcal{S}$  the polyhedron defined by the linear inequalities. The maximum volume inscribed ellipsoid of  $\mathcal{S}$  is given by

$$\{\Phi u + c \mid \|u\|_2 \leq 1\}$$

where  $\Phi \in \mathbb{S}^d$ , and  $c \in \mathbb{R}^d$  are obtained from the following log-det program:

$$\begin{aligned} \max \quad & \log \det \Phi \\ \text{subject to:} \quad & \|\Phi T(i)^T\|_2 + T(i)c \leq b_i \text{ for } i = 1, \dots, m \end{aligned} \tag{3.1.1}$$

Using the affine mapping  $z = \Phi u + c$ , we can obtain the formulation of the maximum volume inscribed ellipsoid  $(P, q, r)$  in terms of  $z$ ,

$$\{z \mid z^T P z + 2q^T z + r \leq 1\},$$

where the transformation is given by

$$P = (\Phi \Phi^T)^{-1}, q = -Pc \text{ and } r = c^T P c \tag{3.1.2}$$

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

One potential problem arises when  $Tz \leq b$  is unbounded, so the maximum volume inscribed ellipsoid is not applicable. This issue can be addressed by adding box constraints,

$$-M \leq z \leq M,$$

where  $M$  is sufficiently large. There are different ways to use the theorem to convert the probabilistic constraint, depending on what the random variable  $z$  represents. It can be the system state or a combination of state and control. These lead to different design of controllers, as we will discuss in the following sections.

## 3.2 SLQR with all-stage probabilistic constraint

In this section we consider both state feedback and disturbance feedback controllers for the LQ control problem

$$\begin{aligned} \min \quad & E \sum_{k=0}^{N-1} \{x_{k+1}^T Q x_{k+1} + u_k^T R u_k\} \\ \text{subject to:} \quad & x_{k+1} = A x_k + B u_k + w_k \end{aligned}$$

With the presence of a all-stage probabilistic constraint

$$\mathbf{P} \left( T \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \leq b \right) \geq \alpha$$

We call the above constrained problem (**P 3.2.1**) from now on. Note that we call it an all-stage probabilistic constraint because it requires the linear inequality constraint on the system state of the whole predicting horizon to be met with at least probability  $\alpha$ . The probabilistic constraint is the major source of complexity of the problem. After we convert it to a deterministic constraint we still need to figure out how to design the controller. In this section, we will first design a state feedback controller of the form

$$u_k = \bar{u}_k + K_k(x_k - \bar{x}_k).$$

We will develop an efficient algorithm to solve for such state feedback controller when the probabilistic constraint is state separable. For cases where the probabilistic constraint is state non-separable we will derive an algorithm to construct

an affine disturbance feedback controller of the form

$$u_k = \bar{u}_k + \sum_{j=0}^{k-1} K_{(k,j)} w_j.$$

where  $\bar{u}_k$  and  $K_k$  or  $K_{(k,j)}$  are to be determined.

### 3.2.1 State Separable Approximation

In this subsection we study a special case in which the state probability constraint is approximated by a state separable Chebyshev bound, and show how that property can be used in an algorithm for computing a control law. First we will define the state separable Chebyshev bound of a state probability constraint.

**Definition 3.2.** *If an inscribed ellipsoid  $(P, q, r)$  of  $\sum_{k=0}^N T_k x_k \leq b$  has  $P$  such that the Chebyshev bound can be written as*

$$\sum_{k=0}^N \mathbf{E} [x_k^T P_k x_k + 2q_k^T x_k] + r - (1 - \alpha) \leq 0,$$

*then the ellipsoid gives what is called a **State Separable Chebyshev Bound**.*

If we replace the probabilistic constraint in **(P 3.2.1)** with a state separable Chebyshev bound, the resulting problem is called a **State Separable Approximation** of **(P 3.2.1)**. It turns out that a state separable approximation can be solved very effectively by recursive algorithms, as we will show soon. A nice property of the maximum volume ellipsoid algorithm is that it always gives a state separable Chebyshev bound when the probabilistic constraint satisfies a certain structure. In the next lemma, we will show that a state separable Chebyshev bound results from the special case where the linear constraint is equivalent to  $N + 1$  individual constraints of the form  $T_k x_k \leq b_k$ .

**Lemma 3.3.** *For the state probability constraint*

$$\mathbf{P} (T_0 x_0 \leq b_0, \dots, T_N x_N \leq b_N) \geq \alpha$$

*applying Theorem 3.1 using the maximum volume ellipsoid algorithm always gives a state separable Chebyshev bound.*

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

**Proof.** Let us look at the constraint  $\|\Phi T(i)^T\|_2 + T(i)c \leq b_i$  in (3.1.1). Again here  $T(i)$  represents the  $i$ -th row of  $T$ . Without loss of generality we assume that only the  $i$ -th block of  $n$  entries of  $T(i)$  is nonzero. For example, for  $T(0)$ , only the first  $n$  entries are nonzero. The constraint above is equivalent to:

$$T(i)\Phi^T\Phi T(i)^T \leq b_i - T(i)c$$

As we can see, the constraints are only placed on the diagonal blocks of  $\Phi^T\Phi$ . The off-diagonal blocks have no contribution to either the constraints or the volume of the ellipsoid, so they are set to zeros. Since  $P = (\Phi\Phi^T)^{-1}$ , it is also block diagonal. ■

Given the definition of the state separable Chebyshev bound, we are ready to show the convenience granted by this property. Now the original problem can be conservatively approximated by the following problem, which we call **(P 3.2.2)**:

$$\begin{aligned} \min \quad & \sum_{k=0}^{N-1} \left( \bar{x}_{k+1}^T Q \bar{x}_{k+1} + \bar{u}_k^T R \bar{u}_k + \mathbf{Tr}(Q \Sigma_{k+1}) + \mathbf{Tr}(R K_k \Sigma_k K_k^T) \right) \\ \text{s.t.} \quad & \sum_{k=0}^N \left( \bar{x}_k^T P_k \bar{x}_k + \mathbf{Tr}(P_k \Sigma_k) + 2q_k^T \bar{x}_k \right) \leq 1 - \alpha - r \\ & \bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k \\ & \Sigma_{k+1} = (A + B K_k) \Sigma_k (A + B K_k)^T + W \\ & k = 0, \dots, N-1 \end{aligned}$$

In fact, **(P 3.2.2)** is closely related to the classical unconstrained LQR and can be solved very efficiently. Before we establish the main algorithm of this subsection, we will establish some building blocks. Our algorithm for the state separable approximation will bring the Chebyshev constraint into the objective using Lagrange duality. Since there is a single Chebyshev constraint, this algorithm will alternately search over a single Lagrange multiplier and a controller for an unconstrained LQR problem. The search over a single Lagrange multiplier will make use of a simple but useful property, which we review next in a general setting.

Consider the following optimization problem:

$$\begin{aligned} \min \quad & f(\theta) \\ \text{s.t.} \quad & g(\theta) \leq 0 \end{aligned}$$

The following lemma characterizes the relationship between Lagrange multiplier values, objective values, and feasibility of the constraint.

**Lemma 3.4.** *Let  $\theta_i$  be a global minimizer of*

$$f(\theta) + \lambda_i g(\theta)$$

*for fixed  $\lambda_i \geq 0$ . If  $\lambda_1 < \lambda_2$ , then  $f(\theta_1) \leq f(\theta_2)$  and  $g(\theta_1) \geq g(\theta_2)$ .*

**Proof.** By definition of  $\theta_1$  and  $\theta_2$ ,

$$f(\theta_2) + \lambda_2 g(\theta_2) \leq f(\theta_1) + \lambda_2 g(\theta_1) \tag{3.2.1}$$

$$f(\theta_1) + \lambda_1 g(\theta_1) \leq f(\theta_2) + \lambda_1 g(\theta_2) \tag{3.2.2}$$

The inequality (3.2.1) can be written as

$$f(\theta_2) + \lambda_2 g(\theta_2) \leq f(\theta_1) + \lambda_1 g(\theta_1) + (\lambda_2 - \lambda_1)g(\theta_1)$$

Combining the right-hand side with the inequality (3.2.2) gives

$$f(\theta_2) + \lambda_2 g(\theta_2) \leq f(\theta_2) + \lambda_1 g(\theta_2) + (\lambda_2 - \lambda_1)g(\theta_1),$$

which is equivalent to

$$(\lambda_2 - \lambda_1)g(\theta_2) \leq (\lambda_2 - \lambda_1)g(\theta_1).$$

Since  $\lambda_2 - \lambda_1 > 0$ , this implies  $g(\theta_2) \leq g(\theta_1)$ .

Combining  $g(\theta_2) \leq g(\theta_1)$  with the inequality (3.2.2) yields

$$f(\theta_1) + \lambda_1 g(\theta_1) \leq f(\theta_2) + \lambda_1 g(\theta_1),$$

which is equivalent to  $f(\theta_1) \leq f(\theta_2)$ . ■

This property gives us a way to find the optimal Lagrange multiplier for the constraint in (P 3.2.2). Inspired by the above lemma, we examine the Lagrangian relaxation of (P 3.2.2). For any fixed  $\lambda$  we call the following problem **LRPA**( $\lambda$ ):

$$\begin{aligned} \min \quad & \sum_{k=1}^N \left( \bar{x}_k^T (Q + \lambda P_k) \bar{x}_k + 2\lambda q_k^T \bar{x}_k + \mathbf{Tr}((Q + \lambda P_k) \Sigma_k) \right) \\ & + \sum_{k=0}^{N-1} \left( \bar{u}_k^T R \bar{u}_k + \mathbf{Tr}(R K_k \Sigma_k K_k^T) \right) \\ \text{s.t.} \quad & \bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k \quad k \in \{0, \dots, N-1\} \\ & \Sigma_{k+1} = (A + B K_k) \Sigma_k (A + B K_k)^T + W \quad k \in \{0, \dots, N-1\} \end{aligned}$$

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

The above problem is equivalent to the classical unconstrained stochastic LQR problem, which can be solved using dynamic programming. The backward recursion for this dynamic program is given by

$$\begin{aligned} V_N(x_N) &= x_N^T (Q + \lambda P_N) x_N + 2\lambda q_N^T x_N \\ V_k(x_k) &= \min_{u_k} \{x_k^T (Q + \lambda P_k) x_k + u_k^T R u_k + 2\lambda q_k^T x_k + \mathbf{E} [V_{k+1}(Ax_k + Bu_k + w_k)]\} \end{aligned}$$

where  $k = 0, \dots, N - 1$ . It is well known that the dynamic programming value function in the recursion above has the form

$$V_k(x_k) = x_k^T Y_k x_k + s_k^T x_k + t_k$$

for all  $k$ , where the matrix  $Y_k$  is positive semidefinite. For a value function of this form, the optimal control input at period  $k$  is

$$u_k = -(R + B^T Y_{k+1} B)^{-1} B^T \left( \frac{1}{2} s_{k+1} + Y_{k+1} A x_k \right)$$

We can express this as a control law of the form  $u_k = \bar{u}_k + K_k(x_k - \bar{x}_k)$ , where

$$K_k = -(R + B^T Y_{k+1} B)^{-1} B^T Y_{k+1} A$$

and  $\bar{u}_k$  is obtained from the recursion

$$\begin{aligned} \bar{u}_k &= -(R + B^T Y_{k+1} B)^{-1} B^T \left( \frac{1}{2} s_{k+1} + Y_{k+1} A \bar{x}_k \right) \\ \bar{x}_{k+1} &= A \bar{x}_k + B \bar{u}_k \end{aligned}$$

The parameters of the quadratic value function are given by

$$\begin{aligned} Y_N &= Q + \lambda P_N \\ s_N &= 2\lambda q_N \\ t_N &= 0 \end{aligned}$$

and

$$\begin{aligned} Y_k &= Q + \lambda P_k + A^T Y_{k+1} A - A^T Y_{k+1} B (R + B^T Y_{k+1} B)^{-1} B^T Y_{k+1} A \\ s_k &= A^T s_{k+1} - A^T Y_{k+1} B (R + B^T Y_{k+1} B)^{-1} B^T s_{k+1} + 2\lambda q_k \\ t_k &= t_{k+1} + \mathbf{Tr}(Y_{k+1} W) - \frac{1}{4} s_{k+1}^T B (R + B^T Y_{k+1} B)^{-1} B^T s_{k+1} \end{aligned}$$

for  $k \in \{0, \dots, N - 1\}$ .

This recursion, together with Lemma 3.4, now provides a way to solve the Chebyshev bound constrained problem (P 3.2.2). Specifically, an optimal controller for (P 3.2.2) is provided by the solution to LRPA( $\lambda$ ) with the smallest  $\lambda$  such that the resulting controller is feasible for (P 3.2.2). This observation offers an efficient way to solve (P 3.2.2), however we still need a strategy to search for the optimal Lagrange multiplier  $\lambda^*$ . Here we simply use binary search as the basis of our implementation. For convenience, Table 3.1 summarizes the algorithm solving (P 3.2.2). Note that not all probabilistic constraints are State

**Table 3.1:** Recursive algorithm for State Separable Approximation

<b>Step 1:</b> Find an inscribed ellipsoid $(P, q, r)$ and corresponding Chebyshev bound.
<b>Step 2:</b> Choose an initial search region $[\lambda_L, \lambda_U]$ for $\lambda$ .
<b>Step 3:</b> Set $\lambda := \frac{1}{2}(\lambda_L + \lambda_U)$ and solve LRPA( $\lambda$ ).
<b>Step 4:</b> If the resulting solution is infeasible for (P 3.2.2), set $\lambda_L := \lambda$ . Otherwise, set $\lambda_U := \lambda$ .
<b>Step 5:</b> Stop if $\lambda_U - \lambda_L < \epsilon$ . Otherwise, return to Step 3.

Separable. When they are not we resort to the following approach.

### 3.2.2 State Non-Separable Approximation

In the previous subsection we derived an effective algorithm for the probabilistically constrained LQR problem in the case where a state separable Chebyshev bound can be utilized. In this subsection we will develop a convex program surrogate of (P 3.2.1) for the general case where there is no reasonable state separable approximation for the probabilistic constraint[(? )].

Here we recall some notation that is frequently used throughout this section. The quantities below are the vectorized states, controls, disturbances, and covariance matrix of the vectorized disturbances:

$$\mathcal{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \quad \mathcal{U} = \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix} \quad \mathcal{W} = \begin{bmatrix} w_0 \\ \vdots \\ w_{N-1} \end{bmatrix} \quad \mathbb{W} = \begin{bmatrix} W & & \\ & \ddots & \\ & & W \end{bmatrix}$$

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

In this subsection we adopt an affine disturbance feedback structure which is another major feedback strategy other than state feedback in stochastic control. It should be pointed out here that using an affine feedback of all past disturbances is proven in [38] to be equivalent to using an affine feedback of all past states in constrained stochastic LQR.

Let us briefly recall the notations we used in the previous chapter. In terms of the vectorized states and inputs, the system dynamics can be expressed as

$$\mathcal{X} = \mathbb{F}x_0 + \mathbb{H}\mathcal{U} + \mathbb{G}\mathcal{W},$$

where the block matrices  $\mathbb{F}$ ,  $\mathbb{H}$  and  $\mathbb{G}$  are given by:

$$\mathbb{F} = \begin{bmatrix} A \\ \vdots \\ A^N \end{bmatrix} \quad \mathbb{H} = \begin{bmatrix} B & & & & \\ AB & B & & & \\ \vdots & & \ddots & & \\ A^{N-1}B & A^{N-2}B & \dots & B & \end{bmatrix} \quad \mathbb{G} = \begin{bmatrix} I & & & & \\ A & I & & & \\ \vdots & & \ddots & & \\ A^{N-1} & A^{N-2} & \dots & I & \end{bmatrix}$$

Our next step is to reformulate the original problem (**P 3.2.1**) with this new notation. Let the diagonal  $N \times N$  block matrices  $\mathbb{Q}$  and  $\mathbb{R}$  be defined as follows:

$$\mathbb{Q} = \begin{bmatrix} Q & & & \\ & \ddots & & \\ & & Q & \end{bmatrix} \quad \mathbb{R} = \begin{bmatrix} R & & & \\ & \ddots & & \\ & & R & \end{bmatrix}$$

Using the notation above, we can rewrite (**P 3.2.1**) as a more compact form:

$$\begin{aligned} \min \quad & \mathbf{E}[\mathcal{X}^T \mathbb{Q} \mathcal{X} + \mathcal{U}^T \mathbb{R} \mathcal{U}] \\ \text{s.t.} \quad & \mathbf{P}(T\mathcal{X} \leq b) \geq \alpha \end{aligned} \tag{3.2.3}$$

Now we replace the probabilistic state constraint in (3.2.3) by the multi-dimensional Chebyshev inequality. Let

$$z = \begin{bmatrix} \mathcal{U} \\ \mathcal{W} \end{bmatrix}, \quad \hat{b} = b - T\mathbb{F}x_0$$

and

$$\hat{T} = T [\mathbb{H} \quad \mathbb{G}]$$

The state constraint simply becomes

$$\hat{T}z \leq \hat{b} \tag{3.2.4}$$





### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

where

$$\begin{aligned} \mathcal{X} &= \mathbb{F}x_0 + \mathbb{H}\bar{\mathbf{u}} + (\mathbb{G} + \mathbb{H}\mathbb{K})\mathcal{W} \\ \mathbf{u} &= \bar{\mathbf{u}} + \mathbb{K}\mathcal{W} \\ z &= \begin{bmatrix} \mathbf{u} \\ \mathcal{W} \end{bmatrix} \\ \mathbf{Tr}(P\mathbf{E}[zz^T]) + 2q^T\mathbf{E}[z] + r &\leq 1 - \alpha \end{aligned}$$

and the optimization variables are the vector  $\bar{\mathbf{u}}$  and the strictly block lower triangular matrix  $\mathbb{K}$ . Directly in terms of these optimization variables we can write this problem as (**P 3.2.2**):

$$\begin{aligned} \min \quad & \begin{bmatrix} x_0 \\ \bar{\mathbf{u}} \end{bmatrix}^T \begin{bmatrix} \mathbb{F}^T\mathbb{Q}\mathbb{F} & \mathbb{F}^T\mathbb{Q}\mathbb{H} \\ \mathbb{H}^T\mathbb{Q}\mathbb{F} & \mathbb{H}^T\mathbb{Q}\mathbb{H} + \mathbb{R} \end{bmatrix} \begin{bmatrix} x_0 \\ \bar{\mathbf{u}} \end{bmatrix} + \mathbf{Tr}(\mathbb{K}^T(\mathbb{H}^T\mathbb{Q}\mathbb{H} + \mathbb{R})\mathbb{K}\mathbb{W} + 2\mathbb{K}^T\mathbb{H}^T\mathbb{Q}\mathbb{G}\mathbb{W}) \\ \text{s.t.} \quad & \bar{\mathbf{u}}^T P_{11}\bar{\mathbf{u}} + 2q_1^T\bar{\mathbf{u}} + \mathbf{Tr}(\mathbb{K}^T P_{11}\mathbb{K}\mathbb{W} + 2\mathbb{K}^T P_{12}\mathbb{W}) \leq 1 - \alpha - r - \mathbf{Tr}(P_{22}\mathbb{W}) \end{aligned}$$

Here we have partitioned  $P$  and  $q$  as

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix} \quad q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$$

When the problem is solved, we obtain an optimal control policy expressed as a constant term plus a linear term associated with the disturbances. The problem (**P 3.2.2**) turns out to be a convex quadratic program with respect to  $(\bar{\mathbf{u}}, \mathbb{K})$ . This is shown in the following theorem.

**Theorem 3.5.** (**P 3.2.2**) *is a convex quadratic program.*

**Proof.** First we look at the objective. The objective is expressed as a sum of two quadratic terms, one exclusively in terms of the variable  $\bar{\mathbf{u}}$  and the other exclusively in terms of the variable  $\mathbb{K}$ . Since  $\mathbb{Q}$  and  $\mathbb{R}$  are both positive semidefinite, the term in  $\bar{\mathbf{u}}$  is convex. The term in  $\mathbb{K}$  is convex if  $\mathbf{Tr}(\mathbb{K}^T(\mathbb{H}^T\mathbb{Q}\mathbb{H} + \mathbb{R})\mathbb{K}\mathbb{W})$  is convex. Since any positive semidefinite matrix can be expressed as a sum of matrices of the type  $\beta\beta^T$ , we have

$$\begin{aligned} & \mathbf{Tr}(\mathbb{K}^T(\mathbb{H}^T\mathbb{Q}\mathbb{H} + \mathbb{R})\mathbb{K}\mathbb{W}) \\ &= \mathbf{Tr}(\mathbb{K}^T(\mathbb{H}^T\mathbb{Q}\mathbb{H} + \mathbb{R})\mathbb{K} \sum_{i=1}^t \beta_i\beta_i^T) \\ &= \sum_{i=1}^t (\mathbb{K}\beta_i)^T (\mathbb{H}^T\mathbb{Q}\mathbb{H} + \mathbb{R}) (\mathbb{K}\beta_i) \end{aligned}$$

### 3.2 SLQR with all-stage probabilistic constraint

---

which is a non-negative linear combination of convex functions of  $\mathbb{K}\beta_i$ . So this term is convex in  $\mathbb{K}$  and therefore the objective is convex.

Now we prove the constraint is also convex. As with the objective, the left hand side of the constraint is expressed as a sum of two quadratic terms, one exclusively in terms of the variable  $\bar{\mathbf{U}}$  and the other exclusively in terms of the variable  $\mathbb{K}$ . Since  $P_{11}$  is positive semidefinite, the term in  $\bar{\mathbf{U}}$  is convex. The term in  $\mathbb{K}$  is convex if  $\mathbf{Tr}(\mathbb{K}^T P_{11} \mathbb{K} \mathbb{W})$  is convex. Using the same analysis for the objective here, we conclude that  $\mathbf{Tr}(\mathbb{K}^T P_{11} \mathbb{K} \mathbb{W})$  is a non-negative linear combination of convex functions. Hence, the constraint is a convex quadratic constraint. ■

We showed the approach to replace the probabilistic constraint and designed a causal affine control law for the resulting problem. We further proved that the this control law can be computed by solving a convex program. For completeness, the algorithm that is used to solve **(P 3.2.1)** is summarized in Table 3.2.

**Table 3.2:** Convex programming algorithm for State Non-separable Approximation

<b>Step 1:</b> Compute $\mathbb{F}$ , $\mathbb{H}$ , $\mathbb{G}$ , $\mathbb{W}$ , $\mathbb{Q}$ and $\mathbb{R}$ .
<b>Step 2:</b> Construct $\hat{T}$ , $\hat{b}$ and the set $\mathcal{S}$ using (3.2.4).
<b>Step 3:</b> Find a qualified inscribed ellipsoid $(P, q, r)$ of $\mathcal{S}$ .
<b>Step 4:</b> Solve <b>(P2)</b> for $\bar{\mathbf{U}}$ and $\mathbb{K}$ .

The algorithm using convex programming is more computationally demanding than the recursive algorithm we proposed in the previous section. However, this algorithm offers a tractable approximation for a more general class of state constrained stochastic LQR. In the next subsection, we will test our algorithms on numerical examples and point out the connection between the two algorithms.

#### 3.2.3 Numerical Example

In this subsection we demonstrate our algorithms on two state-constrained LQR problems with 2 states and 2 control inputs. The examples are chosen in a way that the maximum volume ellipsoid gives state separable Chebyshev bounds, so that we can run all algorithms on the same examples. We compare our approaches with the certainty equivalent approach, which replaces random variables with their expected values. All the examples are implemented in Matlab, using Yalmip [39] and SDPT3 [40].

In our examples, the system dynamics and disturbance parameters are given by

$$A = \begin{bmatrix} 1.02 & -0.1 \\ 0.1 & 0.98 \end{bmatrix} \quad B = \begin{bmatrix} 0.5 & 0 \\ 0.05 & 0.5 \end{bmatrix}$$

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \quad R = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix} \quad Q_N = \begin{bmatrix} 50 & 0 \\ 0 & 50 \end{bmatrix}$$

The initial system state and the uniform box state constraint are given by

$$x_0 = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq x_k \leq \begin{bmatrix} 30 \\ 30 \end{bmatrix}$$

Note that this example contains a terminal cost matrix  $Q_N$ . The addition of terminal costs to our framework is trivial, and facilitates the use of our algorithms as part of a stochastic model predictive control procedure.

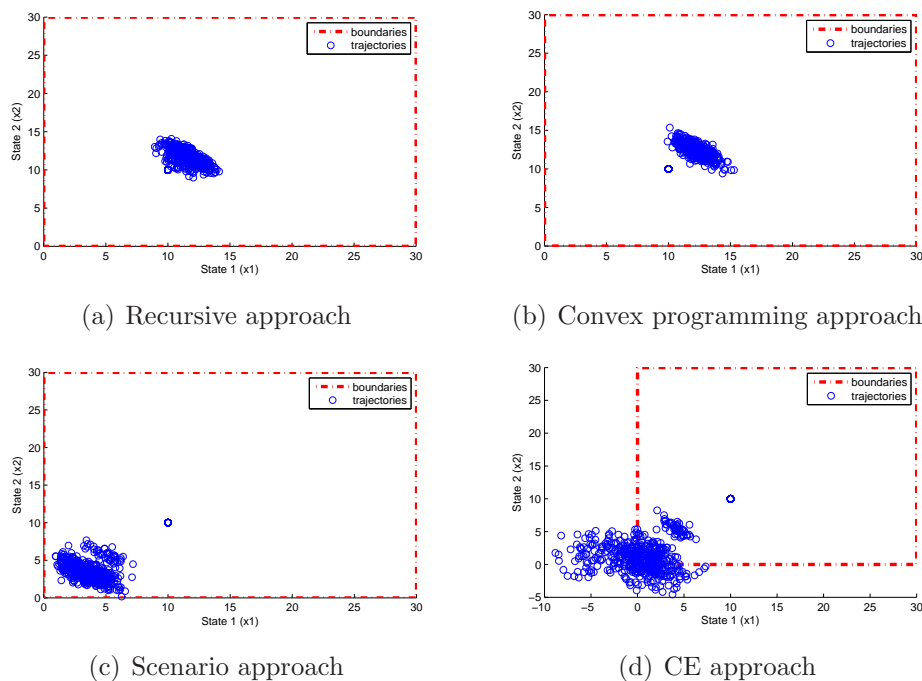
In our examples, the disturbance vectors are multi-dimensional normal and i.i.d.:

$$w_k \sim \mathcal{N} \left( \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.81 & -0.648 \\ -0.648 & 0.81 \end{bmatrix} \right)$$

The probability requirement  $\alpha$  is set to be 0.92. To make the comparison more interesting, all approaches here use feedback. The recursive approach uses the state feedback structure while the convex programming approach, scenario-based approach and CE approach adapts the disturbance feedback structure.

The results of the above example are shown in the following figures. Figure 3.1(a) contains the trajectories of the state-constrained stochastic LQR problem with a horizon containing 10 periods solved using the recursive algorithm. Figure 3.1(b) gives the trajectories generated using our convex programming algorithm,

### 3.2 SLQR with all-stage probabilistic constraint

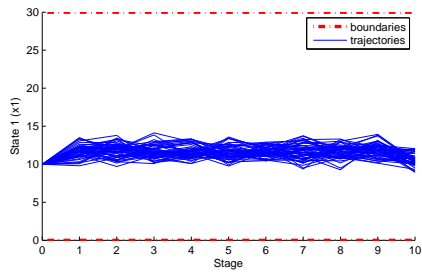


**Figure 3.1:** 2D trajectories

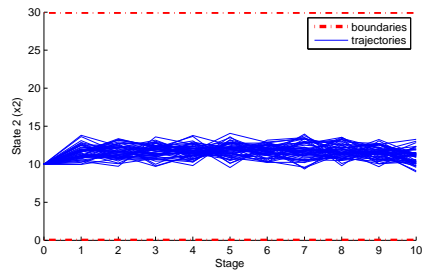
Figure 3.1(c) gives the result obtained from the scenario-based approach and Figure 3.1(d) shows the trajectories solved using the certainty equivalent approach. The simulation is repeated 50 times. As we can see, both of our approaches successfully keep the trajectories within the red dotted boundary box, which represents the state constraints. However compared with the scenario-based approach, our algorithms show some degree of conservatism. This is caused by the selection of ellipsoids in the Chebyshev bounds. The trajectories of the certainty equivalent control fall out of the boundaries quickly as the controller acts risky towards the origin, where the cost is minimized. Figure 2 gives the trajectories in detail by showing them state-by-state. Figure 3.2(a) and Figure 3.2(b) are the state trajectories for the recursive algorithm. Figure 3.2(c) and Figure 3.2(d) are those for the convex programming algorithm. Figure 3.2(e) and Figure 3.2(f) are for the scenario-based approach and Figure 3.2(g) and Figure 3.2(h) are for the certainty equivalent approach. From the above figures we see that in terms of solution quality, the scenario-based approach is the best. We then take solving time into consideration and look at exact numbers.

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONstrained SLQR

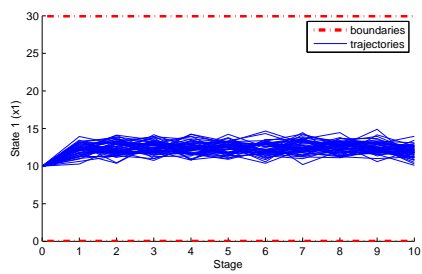
---



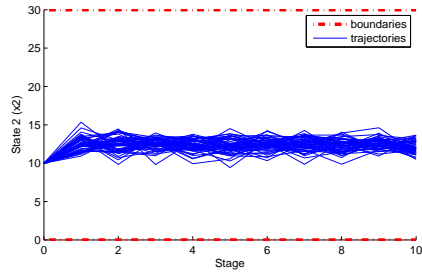
(a) Recursive approach: x1



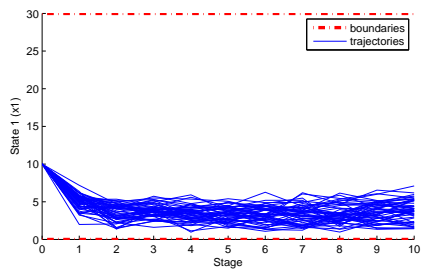
(b) Recursive approach: x2



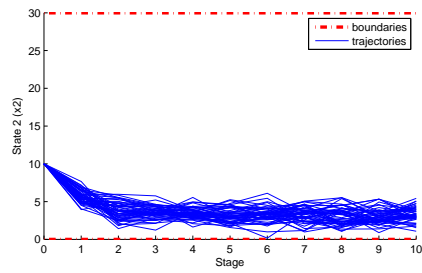
(c) convex programming approach: x1



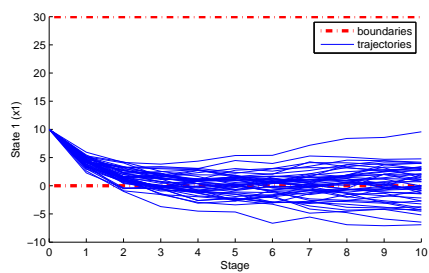
(d) convex programming approach: x2



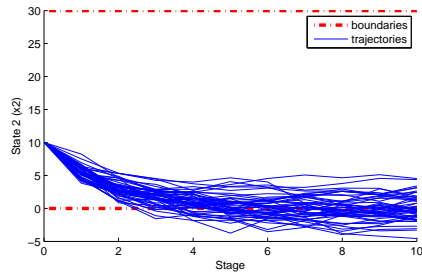
(e) scenario-based approach: x1



(f) scenario-based approach: x2



(g) CE approach: x1



(h) CE approach: x2

**Figure 3.2:** state trajectories

### 3.2 SLQR with all-stage probabilistic constraint

---

The running result comparison is shown in Table 3.3. The testing environment is Windows x64 on a PC with Intel Core i7 2630 2Ghz and 8 gig ram. We use maximum volume ellipsoids in the Chebyshev bound and the total solving time (including the time used to compute the ellipsoids) is shown below. The scenario-based approach takes a bit less than an hour to finish. While the recursive approach uses merely 1.8 seconds, and is almost as fast as solving the CE formulation. In the scenario-based approach, the probability that the test of the probabilistic constraint is insufficient is less than or equal to 1 percent which results in 21739 constraints. It is interesting to see in Table 3.3 that the optimal cost we get from the recursive approach is lower than the one provided by the convex programming approach. One reason is that when applying the Chebyshev bound to approximate the probabilistic constraint, the maximum volume ellipsoid algorithm is used for different polyhedrons as  $z$  is defined differently. Overall, our algorithms are more conservative than the scenario-based approach. This is alleviated by a MPC framework proposed in a working paper by the authors [41].

**Table 3.3:** Running results

Approach	Solving time	Optimal obj value
Recursive	1.82s	$3.01 \times 10^4$
Convex	35.33s	$4.25 \times 10^4$
Scenario	3278.4s	$4.49 \times 10^3$
CE	1.81s	$1.52 \times 10^3$

### 3.3 SLQR with per-stage probabilistic constraints

There are two main points in this section. The first one is that we propose a fast algorithm to solve a stochastic linear quadratic control problem with a probabilistic constraint at each stage. In the literature, most similar problems enforce probabilistic constraints on linear inequalities. In our formulation, the probabilistic constraints are placed on intersections of ellipsoids, which are represented by quadratic inequalities. We developed an algorithm based on a sub-gradient method to solve the SLQR. The second point is that we designed a MPC framework using the SLQR solver as a subroutine to solve long-horizon probability constrained control problems. This framework can significantly reduce the computational complexity of the problem while increasing the quality of the control. We test the MPC framework on a 120-hour-long temperature control problem and the results show that the framework is efficient and reliable against disturbances, and has the potential to be implemented online.

#### 3.3.1 A sub-gradient method approach

The problem formulation of this section is not fundamentally different from the one in the previous section. To make this section self-contained, we briefly review the problem setup and point out the difference here. We consider a probabilistically constrained version of the classical finite-horizon stochastic LQR problem. We aim to control a linear system with stochastic dynamics given by

$$x_{k+1} = Ax_k + Bu_k + w_k$$

for  $k \in \{0, \dots, N - 1\}$ . In period  $k$ ,  $x_k$  is the system state,  $u_k$  is the control input, and  $w_k$  is the disturbance input. The disturbance inputs are independent, have mean vector  $\bar{w}_k$ , and have covariance matrix  $W_k$ . Our goal is to minimize the classical LQR objective in expectation,

$$\mathbf{E} \left[ \sum_{k=0}^{N-1} x_{k+1}^T Q x_{k+1} + u_k^T R u_k \right],$$

where the matrix  $Q$  is positive semidefinite and the matrix  $R$  is positive definite. We assume the initial state vector  $x_0$  is known.  $N$  is the problem horizon. The



### 3.3 SLQR with per-stage probabilistic constraints

---

state feedback control we consider has the form

$$u_k = \bar{u}_k + K_k(x_k - \bar{x}_k).$$

Here,  $\bar{x}_k$  denotes the expected value of the state and  $\bar{u}_k$  denotes the expected value of the control input. The complicating factor in our problem is a linear constraint on the system states that must hold with some specified probability. Specifically, for a given  $\alpha \in [0, 1)$ , we must select a control law that ensures

$$\mathbf{P}(x_{k+1} \in \mathcal{S}_{k+1}) \geq \alpha$$

where  $\mathcal{S}_k$  is the intersections of some ellipsoids.

$$\mathcal{S}_{k+1} = \bigcap_{i=1}^{m_{k+1}} \{z | z^T T_{k+1,i} z + 2a_{k+1,i}^T z + b_{k+1,i} \leq 0\}$$

A similar problem was solved in the previous section [42]. The difference here is that the big probabilistic constraint is replaced by the per-stage probabilistic constraints and  $\mathcal{S}_k$  is defined by quadratic inequalities instead of linear inequalities. The new formulation is more flexible as now we can place different probability requirements on system state at each stage. This is useful when one wants to enforce a changing risk tolerance. For example, when using the model for portfolio management, one may care more about how much money he/she has at the end of the operation rather than the trajectory of getting there. In this case, one can place the probabilistic requirements of final stages to be more significant than those of the intermediate stages. The model can also be used in other applications such as supply chain management and temperature control. We will give a simple example of temperature control in the numerical experiment subsection.

To solve the per-stage constrained SLQR The first thing we need to do is to replace the probabilistic constraints. The idea is to substitute it with a deterministic approximation. We will adopt the approach proposed in the section 3.1, using a multi-dimensional Chebyshev inequality. An important question here is how to choose the inscribed ellipsoid in Theorem 3.1. We will get to that later and let us assume that the ellipsoids are given at this time.

Note that when finding the state feedback controller, the variables to be optimized are  $\bar{u}_k$  and  $K_k$ , representing the mean of the control and the gain of the

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

linear state feedback respectively. As we can see, the control contains feedback only if the system state is deviated from its mean. To see what it means, in an extreme case suppose the noise is so small that we can ignore it, then the controller reduces to an open-loop controller, which is optimal in that case. The feedback part  $K_k(x_k - \bar{x}_k)$  is used to compensate the offset of the state with respect to its projected mean. Now if we substitute the probabilistic constraints with the Chebyshev bounds and apply the control law, the original problem becomes

$$\begin{aligned}
& \text{minimize: } \sum_{k=0}^{N-1} \bar{x}_{k+1}^T Q \bar{x}_{k+1} + Tr(Q \Sigma_{k+1}) + Tr(RK_k \Sigma_k K_k^T) + \bar{u}_k^T R \bar{u}_k \\
& \text{subject to: } \bar{x}_{k+1}^T P_{k+1} \bar{x}_{k+1} + Tr(P_{k+1} \Sigma_{k+1}) + 2q_{k+1}^T \bar{x}_{k+1} + r_{k+1} - 1 + \alpha \leq 0 \\
& \quad \bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k + \bar{w}_k \\
& \quad \Sigma_{k+1} = (A + BK_k) \Sigma_k (A + BK_k)^T + W_k \\
& \quad \text{for } k = 0, \dots, N-1
\end{aligned}$$

We call the above problem **(P 3.3.1)**. The system dynamics are reflected by means and covariance matrices, which further reminds us our goal in this problem is to control not only the expected positions of the state but also the shape of its distribution. Note that the above formulation is a safe approximation of the original problem which means any feasible solution of **(P 3.3.1)** is guaranteed to be feasible in the original problem. There is something special about **(P 3.3.1)** if we examine it closely. Actually, **(P 3.3.1)** is very similar to the unconstrained SLQR, except that there is an additional constraint that is quadratic in the state. To see their connection clearly, let us dualize the Chebyshev bounds into the objective. For any fixed  $\lambda$  we call the following Lagrangian relaxation **LRPP**( $\lambda$ )

$$\begin{aligned}
& \text{minimize: } \sum_{k=0}^{N-1} \bar{x}_{k+1}^T (Q + \lambda_{k+1} P_{k+1}) \bar{x}_{k+1} + Tr((Q + \lambda_{k+1} P_{k+1}) \Sigma_{k+1}) \\
& \quad + Tr(RK_k \Sigma_k K_k^T) + \bar{u}_k^T R \bar{u}_k + 2\lambda_{k+1} q_{k+1}^T \bar{x}_{k+1} + \lambda_{k+1} (r_{k+1} - 1 + \alpha) \\
& \text{subject to: } \bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k + \bar{w}_k \\
& \quad \Sigma_{k+1} = (A + BK_k) \Sigma_k (A + BK_k)^T + W_k \\
& \quad \text{for } k = 0, \dots, N-1
\end{aligned}$$

On can see that now for any fixed Lagrangian multipliers **LRPP**( $\lambda$ ) is exactly an unconstrained SLQR, which can be solved quickly. Indeed,  $\max_{\lambda \geq 0} \mathbf{LRPP}(\lambda)$  is the problem we are going to solve, and therefore it is very important to investigate the relationship of its optimal solution and that of **(P 3.3.1)**. Are they equal to each other? Or  $\max_{\lambda \geq 0} \mathbf{LRPP}(\lambda)$  is just a lower bound of the optimal value of **(P 3.3.1)**? If **(P 3.3.1)** is a convex optimization problem, we are sure that

### 3.3 SLQR with per-stage probabilistic constraints

---

solving **(P 3.3.1)** and its dual  $\max_{\lambda \geq 0} \mathbf{LRPP}(\lambda)$  yield the same optimal solution. However, it is not obvious that **(P 3.3.1)** is convex or not. So it can not be directly told that whether the optimal objective value of  $\max_{\lambda \geq 0} \mathbf{LRPP}(\lambda)$  equals that of **(P 3.3.1)**. But after a reformulation and making a couple of connections we will be able to conclude that the two turn out to be the same. We summarize this in the following lemma.

**Lemma 3.6.** *The optimal objective value of **(P 3.3.1)** equals that of its dual  $\max_{\lambda \geq 0} \mathbf{LRPP}(\lambda)$*

**Proof.** We can rewrite **(P 3.3.1)** as **SDP1**

$$\text{minimize: } \sum_{k=0}^{N-1} \text{Tr} \left( V_k \tilde{Q} \right) + \text{Tr} (V_N^{xx} Q) \quad (3.3.1)$$

$$\text{subject to: } \bar{x}_{k+1} = A\bar{x}_k + B\bar{u}_k \quad (3.3.2)$$

$$\begin{bmatrix} \Omega_{k+1} - W_k & (A\Omega_k + B\Psi_k) \\ (A\Omega_k + B\Psi_k)^T & \Omega_k \end{bmatrix} \succeq 0 \quad (3.3.3)$$

$$V_{k+1} \succeq \begin{bmatrix} \bar{x}_{k+1} \\ \bar{u}_{k+1} \end{bmatrix} \begin{bmatrix} \bar{x}_{k+1} \\ \bar{u}_{k+1} \end{bmatrix}^T + \begin{bmatrix} \Omega_{k+1} \\ \Psi_{k+1} \end{bmatrix} \Omega_{k+1}^+ \begin{bmatrix} \Omega_{k+1} \\ \Psi_{k+1} \end{bmatrix}^T \quad (3.3.4)$$

$$\begin{bmatrix} V_0 & \begin{bmatrix} x_0 \\ \bar{u}_0 \end{bmatrix} \\ \begin{bmatrix} x_0^T & \bar{u}_0^T \end{bmatrix} & 1 \end{bmatrix} \succeq 0 \quad (3.3.5)$$

$$\begin{bmatrix} V_N^{xx} - \Omega_N & \bar{x}_N \\ \bar{x}_N & 1 \end{bmatrix} \succeq 0 \quad (3.3.6)$$

$$\bar{x}_{k+1}^T P_{k+1} \bar{x}_{k+1} + \text{Tr} (P_{k+1} V_{k+1}) + 2q_{k+1}^T \bar{x}_{k+1} + r_{k+1} - 1 + \alpha \leq 0 \quad (3.3.7)$$

$$k = 0, \dots, N-1 \quad (3.3.8)$$

where  $V_N^{xx}$  represents  $E[x_N x_N^T]$  and  $\Omega_k^+$  the pseudo inverse of  $\Omega_k$  and  $\tilde{Q}$  is the cost coefficient matrix defined as

$$\tilde{Q} = \begin{bmatrix} Q & \\ & R \end{bmatrix}$$

In the above formulation, the variables are  $\bar{x}_{k+1}$ ,  $\bar{u}_k$ ,  $\Omega_{k+1}$ ,  $V_{k+1}$  and  $\Psi_{k+1}$ . (3.3.2)-(3.3.6) capture the system dynamics and (3.3.7) is the Chebyshev bound. As we can see **SDP1** is semi-definite program and it is convex. Let us dualize (3.3.7) into the objective and call the resulting dual function **DSDP1**( $\lambda$ ). Assuming

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

feasibility, we can conclude that the optimal objective value of **SDP1** collides with that of  $\max_{\lambda} \mathbf{DSDP1}(\lambda)$  because of strong duality. We also know that for any fixed Lagrangian multipliers  $\lambda$ , **LRPP**( $\lambda$ ) gives the identical solution and objective value as **DSDP1**( $\lambda$ ). **DSDP1**( $\lambda$ ) is just a reformulated version of **LRPP**( $\lambda$ ). So we can conclude that maximizing **LRPP**( $\lambda$ ) is equivalent to  $\max_{\lambda} \mathbf{DSDP1}(\lambda)$ , which gives the same optimal objective value of solving **SDP1**. And since **SDP1** is just **(P 3.3.1)** reformulated, we now see that maximizing  $\max_{\lambda \geq 0} \mathbf{LRPP}(\lambda)$  gives the same optimal objective value as **(P 3.3.1)** does. ■

Lemma 3.6 tells us if we want to solve **(P 3.3.1)**, we only need to worry about the optimal Lagrangian multipliers  $\lambda^*$ . One way to approach it is to use a sub-gradient method. The sub-gradient method we use is similar to a conventional projected sub-gradient method, but we use the recursive LQR solver as a sub-routine instead of a nonlinear programming solver to solve **LRPP**( $\lambda$ ) for fixed  $\lambda$ . We need to prove that our sub-gradient method converges. Despite the difference of the internal solver, the convergence property of sub-gradient method remains the same in our analysis. Let  $e(\bar{u}, K)$  represent the objective function of **(P 3.3.1)** and  $g(\bar{u}, K)$  the inequality constraints of **(P 3.3.1)** (the Chebyshev bounds) at  $(\bar{u}, K)$ . If we dualize  $g$  into the objective function and call the objective of the resulting dual problem  $f(\lambda)$ , we have the following lemma [43]

**Lemma 3.7.** *Assuming  $\|g\|$  is bounded, if we update the sequence of Lagrangian multipliers as  $\lambda^{i+1} = \max[0, \lambda^i + a^i g^i]$ , and the positive step size sequence satisfies*

$$\sum_{i=1}^{\infty} (a^i)^2 \leq \infty, \sum_{i=1}^{\infty} a^i = \infty$$

then  $\lim_{i \rightarrow \infty} \max_i f(\lambda^i) = f(\lambda^*)$ , where  $\lambda^*$  is an optimizer of  $f(\lambda)$

**Proof.** First we point out some properties of  $f$  and  $g$ , which we are going to use. Notice that  $f(\lambda)$  is a minimization problem with  $\lambda$  fixed so it is always a concave function.  $g(\bar{u}, K)$  is a sub-gradient of  $f$  at  $\lambda$ . To see this note that for any  $\tilde{\lambda}$

$$\begin{aligned} f(\tilde{\lambda}) &\leq e(\bar{u}, K) + g(\bar{u}, K)^T \tilde{\lambda} \\ &= e(\bar{u}, K) + g(\bar{u}, K)^T \lambda + g(\bar{u}, K)^T (\tilde{\lambda} - \lambda) \\ &= f(\lambda) + g(\bar{u}, K)^T (\tilde{\lambda} - \lambda) \end{aligned}$$

### 3.3 SLQR with per-stage probabilistic constraints

---

Now we continue to prove the special property of the sequence  $\{\lambda^i\}$ . Let  $\lambda^*$  be an minimizer of  $f$ , we have

$$\begin{aligned} & \|\lambda^{i+1} - \lambda^*\| \\ &= \|\lambda^i + a^i g^i - \lambda^*\| \\ &= \|\lambda^i - \lambda^*\| + 2a^i (g^i)^T (\lambda^i - \lambda^*) + (a^i)^2 \|g^i\|^2 \\ &\leq \|\lambda^i - \lambda^*\| + 2a^i (f(\lambda^i) - f(\lambda^*)) + (a^i)^2 \|g^i\|^2 \end{aligned}$$

The last inequality is true because  $f(\lambda)$  is concave and  $g(\bar{u}^i, K^i)$  is a sub-gradient of it at  $\lambda^i$ , which means

$$f(\lambda^i) - f(\lambda^*) \geq (g^i)^T (\lambda^i - \lambda^*)$$

If we repeat the above inequality, we get

$$\|\lambda^{i+1} - \lambda^*\| \leq \|\lambda^1 - \lambda^*\| + 2 \sum_{j=1}^i a^j (f(\lambda^j) - f(\lambda^*)) + \sum_{j=1}^i (a^j)^2 \|g^j\|^2$$

which implies

$$2 \sum_{j=1}^i a^j (f(\lambda^*) - f(\lambda^j)) \leq \|\lambda^1 - \lambda^*\| + \sum_{j=1}^i (a^j)^2 \|g^j\|^2$$

Note that we also have

$$\sum_{j=1}^i a^j (f(\lambda^*) - f(\lambda^j)) \geq \sum_{j=1}^i a^j (f(\lambda^*) - \max_j f(\lambda^j))$$

Comparing the last two inequalities, we obtain

$$f(\lambda^*) - \max_j f(\lambda^j) \leq \frac{\|\lambda^1 - \lambda^*\| + \sum_{j=1}^i (a^j)^2 \|g^j\|^2}{2 \sum_{j=1}^i a^j}$$

Since the right hand side goes to zero if  $i \rightarrow \infty$  and

$$\sum_{j=1}^{\infty} (a^j)^2 \leq \infty, \quad \sum_{j=1}^{\infty} a^j = \infty$$

we can conclude that

$$\lim_{i \rightarrow \infty} \max_i f(\lambda^i) = f(\lambda^*)$$

■

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

Lemma 3.7 tells us how to get a set of optimal Lagrangian multipliers  $\lambda$ , which also means we can use the sub-gradient method to solve  $\mathbf{LRPP}(\lambda)$  given a pair  $(\bar{u}, K)$ . Combining Lemma 3.7 with Lemma 3.6 we find ourselves at a good position to summarize the optimality condition for solving  $(\mathbf{P} 3.3.1)$ .

**Theorem 3.8.** *For a given set of ellipsoids  $(P_k, q_k, r_k)$ , the optimality condition for  $\mathbf{LRPP}(\lambda)$  is*

$$\begin{cases} \lambda^T g(\bar{u}, K) = 0, \\ g(\bar{u}, K) \leq 0 \\ \lambda \geq 0 \end{cases}$$

*The optimal solution of  $(\mathbf{P} 3.3.1)$  is given by  $(\bar{u}, K)$ .*

**Proof.** When the condition is satisfied we know that  $(\bar{u}, K)$  is primal feasible and from Lemma 3.6 we know that complementary slackness implies zero duality gap between  $(\mathbf{P} 3.3.1)$  and its dual  $\max \mathbf{LRPP}(\lambda)$ . So from strong duality the optimality follows. ■

We have proved that solving  $\max \mathbf{LRPP}(\lambda)$  is equivalent to solving  $(\mathbf{P} 3.3.1)$  but we have not shown how to actually solve  $\mathbf{LRPP}(\lambda)$  for a fixed  $\lambda$ . As one can see that for any fixed  $\lambda$ ,  $\mathbf{LRPP}(\lambda)$  is just a unconstrained SLQR and can be solved recursively by Dynamic programming. The backward recursion is given by

$$\begin{aligned} V_N(x_N) &= x_N^T (Q + \lambda P_N) x_N + \lambda_N 2q_N^T x_N \\ V_k(x_k) &= \min_{u_k} x_k^T (Q + \lambda_k P_k) x_k + u_k^T R u_k + 2\lambda_k q_k^T x_k + E[V_{k+1}(Ax_k + Bu_k + w_k)] \end{aligned}$$

where  $k = 0, \dots, N - 1$ . It is well-know that the value function has a quadratic form

$$V_k(x_k) = x_k^T Y_k x_k + s_k^T x_k + t_k$$

The optimal control at stage  $k$  can be obtained by applying the recursion below:

$$u_k = - (R + B^T Y_{k+1} B)^{-1} B^T \left( \frac{1}{2} s_{k+1} + Y_{k+1} A x_k + Y_{k+1} \bar{w}_k \right)$$

Let us express the control in terms of the linear structure we specified before

$$u_k = \bar{u}_k + K_k (x_k - \bar{x}_k)$$

### 3.3 SLQR with per-stage probabilistic constraints

---

The solution of the control is given by

$$\begin{cases} K_k = -(R + B^T Y_{k+1} B)^{-1} B^T Y_{k+1} A \\ \bar{u}_k = -(R + B^T Y_{k+1} B)^{-1} B^T \left( \frac{1}{2} s_{k+1} + Y_{k+1} A \bar{x}_k + Y_{k+1} \bar{w}_k^T \right) \\ \bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k \end{cases}$$

The recursion of the parameters of the value function are given by

$$\begin{cases} Y_N = Q + \lambda_N P_N \\ s_N = 2\lambda_N q_N \\ t_N = \lambda_N (r_N - 1 + \alpha) \end{cases}$$

$$\begin{cases} Y_k = Q + \lambda_k P_k + A^T Y_{k+1} A - A^T Y_{k+1} B (R + B^T Y_{k+1} B)^{-1} B^T Y_{k+1} A \\ s_k = A^T s_{k+1} + 2\lambda_k q_k - A^T Y_{k+1} B (R + B^T Y_{k+1} B)^{-1} B^T s_{k+1} \\ \quad + 2A^T Y_{k+1} (I - B(R + B^T Y_{k+1} B)^{-1} B^T Y_{k+1}) \bar{w}_k \\ t_k = t_{k+1} + Tr(Y_{k+1} \Sigma_k) + \lambda_k (r_k - 1 + \alpha) + \frac{1}{4} s_{k+1}^T B (R + B^T Y_{k+1} B)^{-1} B^T s_{k+1} \\ \quad + s_{k+1}^T \bar{w}_k - (\bar{w}_k^T Y_{k+1} B + \frac{1}{2} s_{k+1}^T B) (R + B^T Y_{k+1} B)^{-1} B^T s_{k+1} \end{cases}$$

where  $k = 0, \dots, N - 1$ . So for any fixed  $\lambda$ , we can easily recover the optimal control by applying the above recursions.

So far we have enough for an algorithm to solve **(P 3.3.1)** for a set of fixed ellipsoids. But we have not talked about how to update the ellipsoids. We want to make sure that every time we update, the optimal objective value of **(P 3.3.1)** is improved and its previously computed solution should remain feasible with the new ellipsoids. This can be done by minimizing the Chebyshev bounds at fixed  $(\bar{u}, K)$  ( $\Sigma$  can be computed with the pair).

$$\text{minimize: } \bar{x}^T P \bar{x} + Tr(P \Sigma) + 2q^T \bar{x} + r$$

$$\text{subject to: } \begin{bmatrix} P & q \\ q^T & r \end{bmatrix} \succeq 0, \quad (3.3.9)$$

$$\begin{bmatrix} P & q \\ q^T & r - 1 \end{bmatrix} \succeq \gamma_i \begin{bmatrix} T_i & a_i \\ a_i^T & b_i \end{bmatrix} \quad (3.3.10)$$

$$\gamma_i \geq 0, \quad (3.3.11)$$

$$i = 1, \dots, m. \quad (3.3.12)$$

Here  $(P, q, r)$  is the ellipsoid to be updated and  $(T_i, a_i, b_i)$  are the ellipsoids whose intersection contains  $(P, q, r)$ . As we can see that by relaxing the Chebyshev

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

bounds, we create more room for minimizing the objective value of **(P 3.3.1)**. Also, we will not lose feasibility as we know that the fixed  $(\bar{u}, K)$  are still feasible with the new ellipsoids.

#### 3.3.2 Model Predictive Control implementation

Model Predictive control is a class of algorithm that computes a sequence of finite horizon problems to optimize the long term behavior of a system and is one of the most studied algorithm in the industry [44] [45] [46]. In our MPC algorithm, **(P 3.3.1)** is repeatedly solved at each stage. The difficulty solving **(P 3.3.1)** is that there are two things are simultaneously being optimized: the ellipsoids and the Lagrangian multipliers and the problem as a whole is not a convex optimization problem. There are two strategies here to implement the MPC algorithm. First, we can for each set of ellipsoids compute the optimal Lagrangian multipliers and the associated optimal control. Based on the optimal control and the predicted information we then update the ellipsoids. The procedure is repeated until complementary slackness is achieved. Another strategy is that we alternatively update the ellipsoids and the Lagrangian multipliers until the ellipsoids become stable. In this paper we adopt the second strategy, which will be described in detail soon. From our experience, the first strategy is slower than the second. But it is more likely to generate reliable approximation. The second implementation is fast and is less influenced by the conservatism introduced by the Chebyshev bound and the initial set of ellipsoids. Now let us get into the details of the second strategy. We will alternate between updating the ellipsoids and the Lagrangian multipliers. Once we find that the ellipsoids are stable and can not be improved much, we will focus on computing the optimal Lagrangian multipliers and the associated control with the ellipsoids. **(P 3.3.1)** is considered solved when complementary slackness is achieved. Then the MPC algorithm moves forward to the next stage and start over to solve another SLQR.

One thing worth notice is that the Lagrangian multipliers  $\lambda$  actually carry important information about the probabilistic constraints. If a component  $\lambda_i$  is zero, it implies that the  $i$ th probabilistic constraint can be ignored. In other



### 3.3 SLQR with per-stage probabilistic constraints

---

words, only the probabilistic constraints associated with positive Lagrangian multipliers are considered and their priorities are weighted by the absolute values of the multipliers. Since using the closed-form solver as the subroutine, the sub-gradient method is fast. During the solving process, the probabilistic constraints that are detected as dangerous will be taken into account and their possibilities of being violated are reflected by the Lagrangian multipliers. At each stage, all the probabilistic constraints in the predicting horizon are under supervision and the prediction moves forward as the MPC algorithm runs. We summarize the MPC implementation in the following table

<b>Step 1:</b> Compute a set of initial ellipsoids $(P_k, q_k, r_k)$
<b>Step 2:</b> Set the Lagrangian multipliers $\lambda$ to zero.
<b>Step 3:</b> Solve <b>LRPP</b> ( $\lambda$ ) and obtain the control and the expectation $\bar{x}_k$ and covariance $\Sigma_k$ . Compute the sub-gradient.
<b>Step 4:</b> Update the ellipsoids at $\bar{x}_k$ and $\Sigma_k$
<b>Step 5:</b> Check if the change of ellipsoids is significant. If it is go to Step 7, otherwise follow through
<b>Step 6:</b> Check complementary slackness. If it is satisfied go to Step 8, otherwise follow through.
<b>Step 7:</b> Update the Lagrangian multipliers $\lambda$ and go to Step 3.
<b>Step 8:</b> Apply the control at the current stage. Go to Step 2 and start over for the next stage.

**Table 3.4:** MPC implementation

There are a couple of practical considerations when implementing the MPC algorithm. In Step 5, when testing complementary slackness  $\|\lambda_i g_i(\bar{u}, K)\| \leq \epsilon$ . The parameter  $\epsilon$  has impact on several things. A small  $\epsilon$  is crucial for the sub-gradient to correctly identify inactive/active probabilistic constraints and thus find the optimal solution of **(P 3.3.1)**. But a small  $\epsilon$  also means longer running time for the sub-gradient method. It may take a long time for it to be satisfied if  $\epsilon$  is too small. Another one is in Step 5, when we determine if an update of ellipsoids is significant. One can use the decrease of sub-gradient to measure the significance. As we can imagine the measurement is again a trade-off between

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---

efficiency and accuracy. Better ellipsoids can provide better Chebyshev bounds, which can give better optimal control. In the other hand, updating ellipsoids is time-consuming. So the threshold of judging if an update is significant is crucial to the MPC implementation.

#### 3.3.3 Numerical Example

We test our algorithm on a temperature control problem adapted from [33]. The system has three states  $x(1), x(2)$  and  $x(3)$ , which represent the room temperature, the temperature in the wall connected with another room and the temperature in the wall connected to the outside respectively. There is a temperature control device that can cool down as well as heat up the air in the room. There exist three disturbance sources  $w(1), w(2)$  and  $w(3)$ , associated with outside temperature, solar radiation and internal heat gain respectively. The objective of the problem is to conserve energy while keeping the temperature above  $20\text{ }^\circ\text{C}$  with probability at least 0.85 (this is enforced stage-wise). Here are the details regarding the system dynamics

$$x_{k+1} = Ax_k + Bu_k + Gw_k$$
$$A = \begin{bmatrix} 0.8511 & 0.0541 & 0.0707 \\ 0.1293 & 0.8635 & 0.0055 \\ 0.0989 & 0.0032 & 0.7541 \end{bmatrix}, B = \begin{bmatrix} 0.1750 \\ 0.0150 \\ 0.0100 \end{bmatrix}$$
$$G = \begin{bmatrix} 0.0222 & 0.0018 & 0.0422 \\ 0.0015 & 0.0007 & 0.0029 \\ 0.1032 & 0.0001 & 0.1960 \end{bmatrix}$$

We implemented the MPC framework with 5-hour predicting horizon and ran it with a 150-hour simulation. The testing environment was Matlab on Windows x64. The tool-sets we used were Yalmip [39] and SDPT3 [40]. The computer we used was a laptop with Intel Core i7 2630QM 2.0GHz and 8 gig ram. The following figures give us an outlook of the behavior of the controller. The data we use here is extracted from one single simulation among many others.

As we can see in Figure 3.3, the uncontrolled trajectory falls far below  $20$  degree a few times while the controlled one stays above the lower bound most of the time. The occasional violation of the constraint is because we are using the

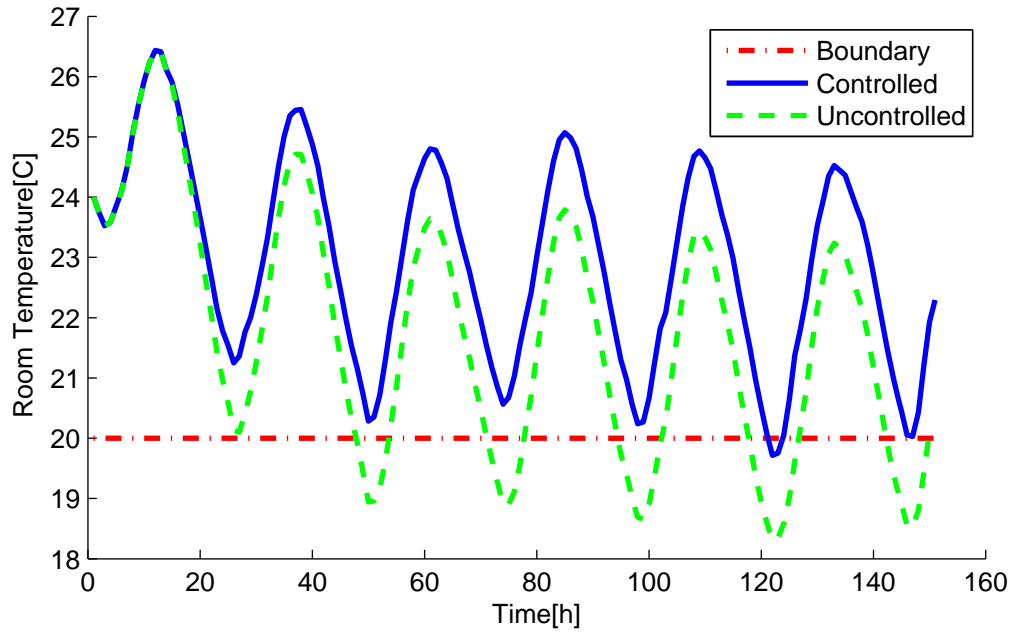


Figure 3.3: Important statistics

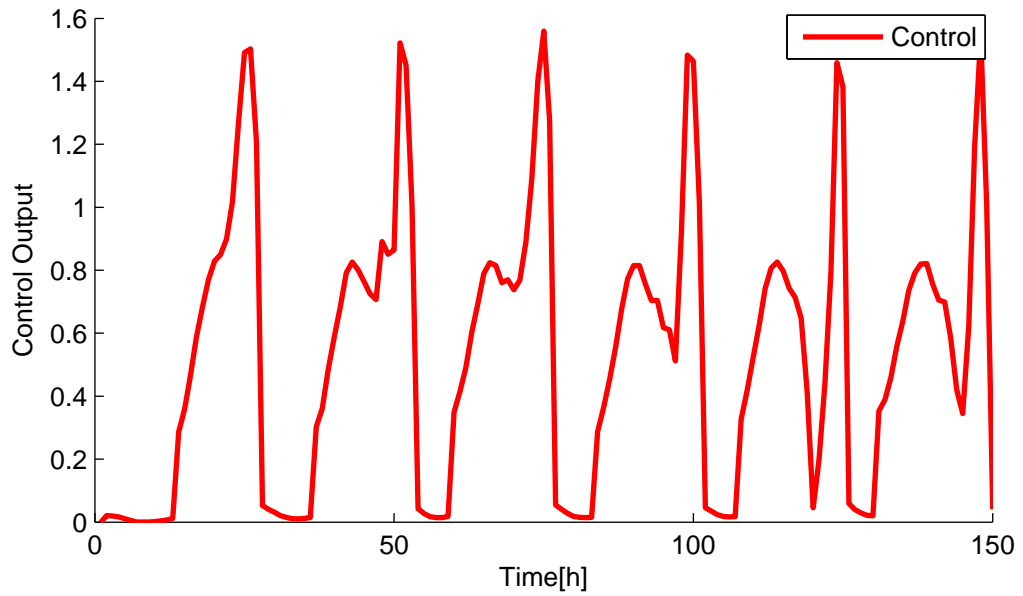


Figure 3.4: Control Output

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONstrained SLQR

---

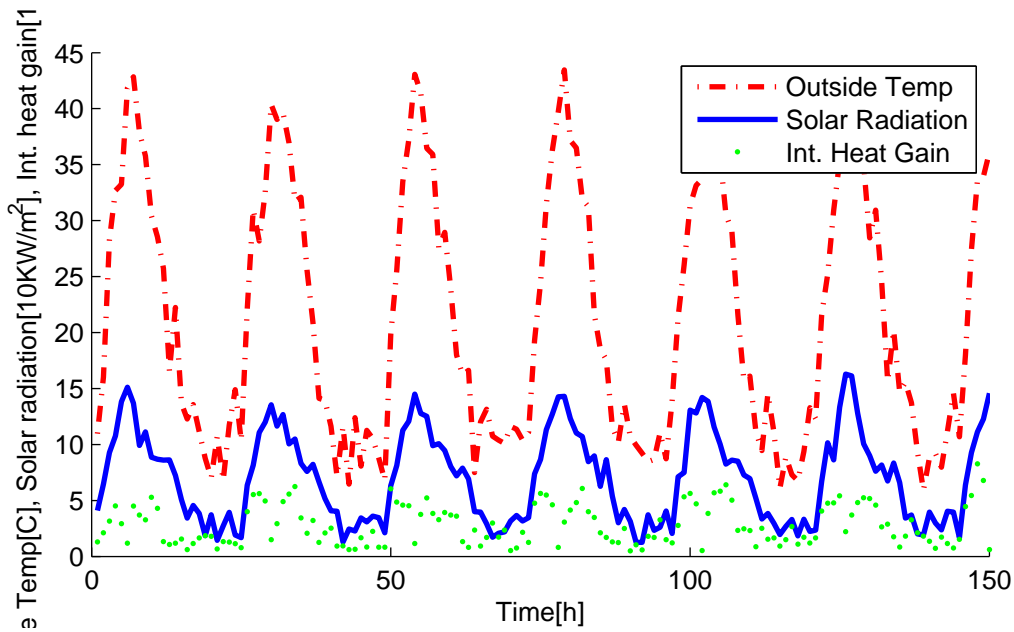


Figure 3.5: Disturbances

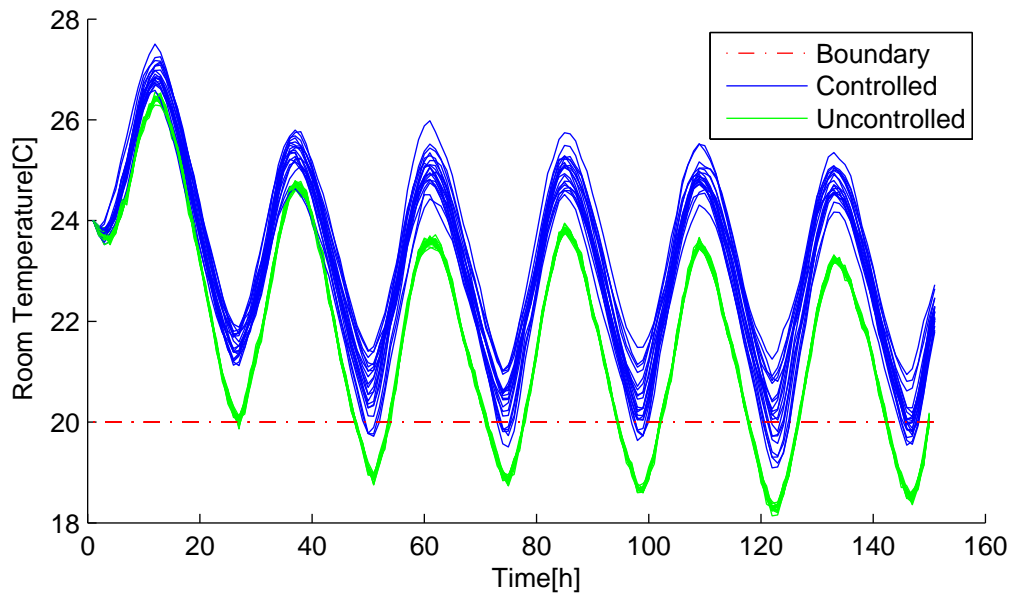


Figure 3.6: Important statistics

ellipsoids from the previous prediction to compute the control at current stage. It is the fact that we cannot perfectly predict the future trajectories that causes

### 3.3 SLQR with per-stage probabilistic constraints

---

the violation. If we look at Figure 3.4 and compare it with Figure 3.3, we can find that the controller does nothing (which is the most aggressive control strategy in this example) to conserve energy but when it decides there is no risk of leaving the safe zone temporarily. As the trajectory approaches the boundary, the controller detects the danger in advance and reacts accordingly to avoid it. Once the system is back into a safer track, the controller goes back to the aggressive mode again until it faces another potential constraint violation. Figure 3.5 is one extraction from the disturbance samples, in which we can see that although the controlled trajectory in Figure 3.3 seems to be smooth the controller was actually dealing with substantial disturbances. We repeated the simulation for 20 times and we show all the trajectories in Figure 3.6. We can see that the system under control is quite reliable speaking of staying out of the undesirable area, under the 0.85 probability requirements.

### 3.4 Conclusion and Extensions

We have presented original results so far in this chapter. We proposed a way to use multi-dimensional Chebyshev bound to approximate state probability constraints. We also showed how to use the Chebyshev bound to convert the original problem to conservative convex programs. For all-stage cases we developed a way to solve for an optimal disturbance feedback controller using convex programming. We also explored the connection between the resulting problem and the classical unconstrained SLQR and designed a recursive algorithm to efficiently find the optimal state feedback controller. For per-stage case, we similarly proposed a state feedback controller and used it in a MPC framework. The simple numerical experiments showed that our approaches have their advantages over existing results. New research results are coming out frequently in this area. As the author is writing this thesis, some of results in the literature review were updated and a couple of times the author needed to revise the manuscript to reflect the advancements. For any interested reader, we want to share our experience and thoughts on how the work here can be possibly extended or related to other similar problems.

In our problem set-up, the probabilistic constraints are on system states. However, similar results can be derived for control constrained problems. To get a state feedback controller, one can apply the Chebyshev bound on the probabilistic constraints of control. The resulting constraints will contain first and second moments of control. Assuming the probability constraint has the following form

$$\mathbf{P}(T_0 u_0 \leq b_0, \dots, T_N u_N \leq b_N) \geq \alpha$$

The resulting constraint after applying the Chebyshev bound is

$$\sum_{k=0}^N \mathbf{E} [u_k^T P_k u_k + 2q_k^T u_k] + r - (1 - \alpha) \leq 0,$$

One can imagine that it can be handled the same way as in the State-Separable case. For general cases, where the probabilistic constraint has the form

$$\mathbf{P} \left( T \begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix} \leq b \right) \geq \alpha$$

One can use the convex programming approach and let  $z$  be the concatenated control when applying the Chebyshev bound.

In our problems, we assumed that we have perfect estimation of the system states  $x_1, \dots, x_N$ . In reality this may not be the case. To be specific, say that at stage  $k$  the best we could do is to estimate system state  $x_k$  based on all the information  $I_k$  we have collected through stage  $1, \dots, k$ , i.e., we can only estimate  $E[x_k|I_k]$ . Our recursive algorithm still works in such case. One can walk through the DP algorithm to compute the closed-form optimal control for the new problem and should find that in the closed-form  $E[x_k|I_k]$  replaces  $x_k$ . However, the cost function will be different to reflect the estimation errors  $x_k - E[x_k|I_k]$ . Since a closed-form optimal control still exists in such cases, we can still use our recursive algorithm with some changes. The convex programming approach will work as long as we have accurate information about the means and covariance matrices of the disturbances  $w_1, \dots, w_{N-1}$  and it does not rely on perfect state estimation.

Another possibly useful extension is the case where in the system dynamics the uncertainty comes from coefficient matrices. To be specific, matrix  $A$  and  $B$  are random in

$$x_{k+1} = Ax_k + Bu_k$$

In such cases, if  $E[A]$  and  $E[B]$  are known and  $E[A^TQA]$ ,  $E[B^TQB]$  and  $E[B^TQA]$  can be efficiently evaluated then the DP algorithm still gives a closed-form solution for optimal control and our recursive algorithm still can be used.

In the per-stage problem we proposed a MPC framework that alternates between computing optimal controls and updating ellipsoids. This scheme also can be used in all-stage case. However, one should be noted that the cost for updating ellipsoids can be quite prohibitive when comes to time complexity for problems with long predicting horizons. One possible way to mitigate this is to pre-compute the ellipsoids and load them in on-the-fly. This is possible because the ellipsoids are only determined by the boundary constraints and system states with which they are computed. Additional structure constraints are needed when computing ellipsoids for the recursive algorithm in all-stage problems, as the structure is required for state-separable Chebyshev bounds.

### 3. FEEDBACK CONTROLLER DESIGN FOR PROBABILITY CONSTRAINED SLQR

---



# References

- [1] A. BEMPORAD, M. MORARI, V. DUA, AND E. N. PISTIKOPOULOS. **The explicit linear quadratic regulator for constrained systems.** *Automatica*, **38**(1):3–20, 2002. 6, 9, 10
- [2] J. ACEVEDO AND E. N. PISTIKOPOULOS. **An algorithm for multiparametric mixed-integer linear programming problem.** *Operations Research Letters*, **24**:139–148, 1999. 7
- [3] V. DUA AND E. N. PISTIKOPOULOS. **Algorithms for the solution of multiparametric mixed-integer nonlinear optimization problems.** *Industrial Engineering Chemistry Research*, **38**(10):3976–3987, 1999. 8
- [4] V. DUA AND E. N. PISTIKOPOULOS. **An algorithm for the solution of multiparametric mixed-integer nonlinear optimization problems.** *Annals of Operations Research*, **99**:123–139, 2000. 8
- [5] A. V. Fiacco. *Introduction to sensitivity and stability analysis in nonlinear programming.* Academic Press, 1983. 8
- [6] GAL. T. *Postoptimal analyses, parametric programming, and related topics, 2nd Edition.* de Gruyter, 1995. 8
- [7] C. ROWE AND J. M. MACIEJOWSKI. **An Algorithm for Multiparametric Mixed Integer Semidefinite Optimisation.** *Proc. 42th IEEE Conf. on Decision and Control*, pages 3197–3202, 2003. 8
- [8] A. BEMPORAD AND C. FILIPPI. **An Algorithm for Approximate Multiparametric Convex Programming.** *Computational Optimization and Applications*, **35**(1):87–108, 2006. 8

## REFERENCES

---

- [9] P. TONDEL, T.A. JOHANSEN, AND A. BEMPORAD. **An algorithm for multi-parametric quadratic programming and explicit MPC solutions.** *Automatica*, **39**(3):489–497, 2003. [10](#)
- [10] D. BERTSEKAS. *Dynamic Programming and Optimal Control*. Athena Scientific, 2007. [12](#)
- [11] A. CHARNES, W. W. COOPER, AND G. H. SYMONDS. **Cost horizons and certainty equivalents: An approach to stochastic programming of heating oil.** *Management Science*, **4**:235–263, 1958. [18](#)
- [12] L. B. MILLER AND H. WAGNER. **Chance-constrained programming with joint constraints.** *Operations Research*, **13**:930–945, 1965. [18](#)
- [13] ALEXANDER SHAPIRO, DARINKA DENTCHEVA, AND ANDRZEJ RUSZCZYNSKI. *Lectures on Stochastic Programming*. 2009. [18](#)
- [14] PREKOPA. **Logarithmic concave measures with applications to stochastic programming.** *Acta Scientiarum Mathematicarum*, **32**:301–316, 1971. [18](#)
- [15] A. NEMIROVSKI AND A. SHAPIRO. **Scenario approximations of chance constraints.** *In Probabilistic and Randomized Methods for Design under Uncertainty*, G. Calafiore and F. Dabbene, eds., Springer-Verlag, London, 2005. [19](#)
- [16] G. CALAFIORE AND M. C. CAMPI. **The scenario approach to robust control design.** *IEEE Trans. Automat. Control*, **51**:742–753, 2006. [19](#)
- [17] G.C. CALAFIORE. **Random convex programs.** *SIAM Journal on Optimization*, **20**(6):3427–3464, 2010. [19](#)
- [18] M.C. CAMPI AND S. GARATTI. **A sampling-and-discarding approach to chance-constrained optimization: feasibility and optimality.** *Journal of Optimization Theory and Applications*, **148**(2):257–280, 2011. [19](#)

- 
- [19] A. BEN-TAL AND A. NEMIROVSKI. **Robust solutions of Linear Programming problems contaminated with uncertain data.** *Mathematical Programming*, **88**:411–424, 2000. [20](#)
- [20] A. NEMIROVSKI AND A. SHAPIRO. **Convex approximations of chance constrained programs.** *SIAM Journal of Optim.*, **17**(4):969–996, 2006. [20](#), [21](#)
- [21] SCOTT R. SWENSETH DAVID L. OLSON. **A Linear Approximation for Chance-Constrained Programming.** *Journal of Operations Research Society*, **38**(3):261–267, 1987. [20](#)
- [22] MASAHIRO ONO, LARS BLACKMORE, AND BRIAN C. WILLIAMS. **Chance Constrained Finite Horizon Optimal Control with Nonconvex Constraints.** *Proceedings of the American Control Conference*, 2010. [21](#)
- [23] D. BERNARDINI AND A. BEMPORAD. **Scenario-based model predictive control of stochastic constrained linear systems.** *Proceedings of the 48 IEEE Conference on Decision and Control*, 2009. [22](#)
- [24] A.T. SCHWARM AND M. NIKOLAOU. **Chance-constrained model predictive control.** *AICHE J.*, **45**(8):1743–1752, 1999. [22](#)
- [25] P. D. COUCHMAN, M. CANNON, AND B. KOUVARITAKIS. **MPC as a tool for sustainable development integrated policy assessment.** *IEEE Transactions on Automatic Control*, **51**:145–149, 2006. [22](#)
- [26] J. SKAF AND S. BOYD. **Design of affine controllers via convex optimization.** 2010. [22](#), [45](#)
- [27] D. CHATTERJEE, P. HOKAYEM, AND J. LYGEROS. **Stochastic receding horizon control with bounded control inputs—a vector space approach.** 2010. [22](#)
- [28] YANG WANG AND STEPHEN BOYD. **Performance bounds for linear stochastic control.** *Systems and Control Letters*, pages 178–182, 2009. [22](#)

## REFERENCES

---

- [29] ATHER GATTAMI. **Generalized Linear Quadratic Control.** *IEEE Transactions on Automatic Control*, **55**(1):131–136, 2010. [22](#), [27](#)
- [30] J. A. PRIMBS AND C. H. SUNG. **Stochastic receding horizon control of constrained linear systems with state and control multiplicative noise.** *IEEE Trans. Automatic Control*, 2007. [27](#)
- [31] J. A. PRIMBS. **A Soft Constraint Approach to Stochastic Receding Horizon Control.** *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 4797–4802, 2007. [27](#)
- [32] JAMES PRIMBS AND CHANG SUNG. **A Stochastic Receding Horizon Control Approach to Constrained Index Tracking.** *Asia-Pacific Financial Markets*, **15**(1):3–24, 2008. [27](#), [29](#)
- [33] F. OLDEWURTEL, C.N. JONES, AND M. MORARI. **A Tractable Approximation of Chance Constrained Stochastic MPC based on Affine Disturbance Feedback.** *Proceedings of Conference on Decision and Control*, 2008. [30](#), [31](#), [62](#)
- [34] D. BERTSIMAS AND M. SIM. **Tractable Approximations to Robust Conic Optimization Problems.** *Math. Program., ser. B* **107**:5–36, 2006. [31](#), [32](#)
- [35] S. BOYD AND L. VANDENBERGHE. *Convex optimization.* Cambridge University Press, 2004. [36](#), [37](#)
- [36] J. DUPACOVA. **The minimax approach to stochastic programming and an illustrative application.** *Stochastics*, **20**:73–88, 1987. [37](#)
- [37] ZHOU ZHOU AND RANDY COGILL. **An Algorithm for State Constrained Stochastic Linear-Quadratic Control.** *In Proceedings of the American Control Conference*, 2011. [37](#)
- [38] PAUL J. GOULART, ERIC C. KERRIGAN, AND JAN M. MACIEJOWSK. **Optimization Over State Feedback Policies for Robust Control with Constraints.** *Automatic*, **42**:523–533, 2006. [44](#)

- [39] J. LOFBERG. **YALMIP : A Toolbox for Modeling and Optimization in MATLAB.** *Proceedings of the CACSD Conference*, 2004. [48](#), [62](#)
- [40] M. J. TODD, K. C. TOH, AND R. H. TUTUNCU. **On the implementation and usage of SDPT3: a Matlab software package for semidefinite-quadratic-linear programming, version 4.0.** 2006. [48](#), [62](#)
- [41] ZHOU ZHOU AND RANDY COGILL. **Solving Probability Constrained MPC via Sub-gradient Method.** working title. [51](#)
- [42] ZHOU ZHOU AND RANDY COGILL. **Reliable Approximations of Probability Constrained Stochastic Linear-Quadratic Control.** *Automatica*, Accepted, 2012. [53](#)
- [43] STEVE BOYD, LIN XIAO, AND ALMIR MUTAPCIC. **subgradient methods.** *Notes for EE392o*, 2003. [56](#)
- [44] S. JOE QIN AND THOMAS A. BADGWELL. **An Overview Of Industrial Model Predictive Control Technology.** pages 232–256, 1997. [60](#)
- [45] S. JOE QIN AND THOMAS A. BADGWELL. **A survey of industrial model predictive control technology**, 2003. [60](#)
- [46] B. KOUVARITAKIS, M. CANNON, AND V. TSACHOURIDIS. **Recent developments in stochastic MPC and sustainable development.** *Annual Reviews in Control*, **28**(1):23–35, 2004. [60](#)