ABSTRACT

YANG, YIPENG. Path Dependent Stochastic Models and Their Applications in Finance and Communications. (Under the direction of Dr. Tao Pang and Dr. Robert T. Buche.)

In this work we studied the path dependent stochastic models and their applications to finance and wireless communication systems. Due to the difficulty of finding the analytical solutions of such problems, the numerical approach is the best way so far to apply. We are first concerned the controlled Markov Chain approximation to solve stochastic control problem with time delay. So far little has been done about the implementation of this numerical method in stochastic control problems with time delay. This method is applied to a wireless communication queuing system. The system state follows a stochastic differential delay equation (SDDE), which is obtained from a heavy traffic model. Inspired by the separation principle for control systems with partial information, we add a state estimation procedure in the discrete controlled Markov Chain approximation, which has not been done by other researchers. Our simulation shows that this procedure does improve the control affect compared to the control policy where delayed information is used directly in the transition probability. In order to reduce the huge memory required by this approach and better predict the present state, we present an idea to keep the control history in the state representation. In this way, we are able to find the accurate probability distribution of the present state. It also makes the optimality proof possible. This extension of controlled Markov Chain approximation to solve stochastic control problems with time delay is also new.

For some long path dependent stochastic models, the Monte Carlo simulation is the best approach so far. We examined the pricing of mortgage-backed securities using Monte Carlo simulation. By doing the Option Adjusted Spread (OAS) analysis, we found that in stead of achieving the absolute convergence, we should better use the relative convergence to calculate the duration or convexity of the bond. The computation is speed up dramatically.

Path Dependent Stochastic Models and Their Applications in Finance and Communications

by

Yipeng Yang

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fullfillment of the
requirements for the Degree of
Doctor of Philosophy

Operations Research

Raleigh, North Carolina

2008

APPROVED BY:

Dr. Tao Pang Chair of Advisory Committee	Dr. Robert T. Buche Co-Chair of Advisory Committee
Dr. Min Kang	Dr. Zhilin Li

DEDICATION

To my parents and my wife.

BIOGRAPHY

Yipeng Yang was born in Shandong, China. He received the B.S. degree from the Department of Control Theory and Control Engineering in 2000 and the second B.S. degree from the Department of Applied Mathematics in 2001, both in Shanghai Jiaotong University, Shanghai, China. He received the M.S. degree from the Department of Control Theory and Control Engineering, Shanghai Jiaotong University, Shanghai, China, in 2003. From the year 2003 to 2008, he has been pursuing the Ph.D. degree major in Operations Research, and minor in Department of Mathematics, North Carolina State University, Raleigh, North Carolina, USA. His research interests include stochastic control with memory, the computation and simulation methods in stochastic control problems and their applications in financial engineering and wireless communication systems.

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor, Dr. Tao Pang, and Co-advisor, Dr. Robert T. Buche, for the great and generous help they offered to me for the past four years. They not only taught me the right way of being a careful and devoted researcher, but also a lot of merits of a good personality, like patience, kindness, forgiveness, encouragement, from which I will benefit throughout my whole life. I had never noticed that I was so lucky to have Dr. Pang and Dr. Buche as my advisors until I am graduating and find that I have achieved so much and learned so much from them. Sometimes I am even surprised how I have become a modest and assiduous researcher from a presumptuous and innocent student a few years ago. It will be the happiest thing for me to share my future achievements with Dr. Pang and Dr. Buche. I also want to thank Dr. Min Kang for her wonderful lectures on probability and stochastic process theories, and Dr. Zhilin Li for his great suggestions and instructions on my research.

There are two great professors who are extremely important to me, and they are Dr. Shu-Cherng Fang and Dr. Xiuli Chao. I always consider them as my mentors, best friends and my ultimate academic goal. Without their encouragement, I could not have finished my Ph.D. study. In addition, I am very grateful to Dr. Bibhuti B. Bhattacharyyya, for the rich measure theory and time series knowledge he taught me. People told me that he was a good instructor, but I had never known that he was that good until I took his course and was deeply impressed.

I could not remember how much help Dr. Yahya Fathi and Dr. Negash Medhin, who are Co-Directors of Operations Research, and Ms. Barbara Walls, who is our program assistant, have offered me, nor do I remember how much convenience they have brought to me. But I do know that they are always behind.

Special thanks to my best friends, like Xinbing Wang, who helped me tremendously in my Ph.D. study, Chuan Lin, who gave me a lot of advices in my research, and especially Kun Sun and Yifan Zhu, who took care of me like a younger brother. A bunch of thanks to all my Operations Research classmates, like Dong-Kyoung Choe, Chung-chien Hong, Xiang Zhou, etc.

Finally and certainly, I want to thank my parents and my dear wife. Their love is my priceless fortune. Thanks for sharing it with me, a very, very, lucky man.

TABLE OF CONTENTS

LIST OF FIGURES ..... **vii**

LIST OF TABLES ..... **x**

1 Introduction **1**
1.1 Motivation 2
1.2 Contributions 2
1.2.1 Numerical Solutions of Stochastic Control Problems with Time Delay in Wireless Communication Queueing Models 2
1.2.2 Numerical Solutions of Stochastic Control Problems with Delayed Control Information 3
1.2.3 Strong Path Dependent Stochastic Models and the Monte Carlo Simulations 3
1.3 Outline 4

2 Background **5**
2.1 The Markov Chain Approximation Method 6
2.2 Construction of the Approximating Markov Chains 8
2.3 Computational Methods for Controlled Markov Chains 12
2.4 Weak Convergence Proofs 15

3 Numerical Solutions for Nonlinear Stochastic Control Problems with Time Delay and Its Application to Wireless Communication Systems **18**
3.1 Numerical Methods for Nonlinear Stochastic Control with Time Delay 19
3.2 Controlled Markov Chain Approximation with State Estimation 23
3.3 Application to the Power Allocation Problem in Wireless Communication Queueing Systems 27
3.3.1 Background of the Wireless Communication Queueing Model 27
3.3.2 Introduction of Time Delay and the Stochastic Control Problem-Power Allocation 30
3.3.3 Implementation of Controlled Markov Chain Approximation 32
3.3.4 Algorithms 37
3.3.5 Simulations 39
3.3.6 Discussion of the simulation results: control policy and cost relations 55
3.4 Future Topics 57

4 Numerical Solutions with Delayed Control Information **59**
4.1 Introduction 59
4.2 Markov Chain Approximation Model with Control History 63
4.3 Finding the Control using Receding Horizons 66

4.4 Application to Wireless Communication Queueing Models 70
4.4.1 Model construction 71
4.4.2 The Algorithm 73
4.4.3 Simulations 74
4.5 Summaries and Further Topics 79

5 Convergence Studies on Monte Carlo Methods for Pricing Mortgage-Backed Securities 81
5.1 Introduction 82
5.2 Mortgage-backed Securities 86
5.2.1 MBS Bonds 86
5.2.2 MBS Pricing 88
5.2.3 Option Adjusted Spread Analysis 95
5.2.4 MBS Greeks 96
5.3 Monte Carlo Simulations 98
5.4 Pseudo Random Number Generators 100
5.5 Numerical Results and Analysis 102
5.5.1 Convergence of the OAS 103
5.5.2 Greeks 104
5.5.3 Relative Convergence and Absolute Convergence 110
5.5.4 Robustness 112
5.6 Summaries and Further Topics 113

6 Conclusions and Future Work 117

Bibliography 119

LIST OF FIGURES

Figure 2.1 Continuous interpolation $x^h(\cdot)$ 8

Figure 2.2 Approximating Grid with Reflecting Boundaries 11

Figure 3.1 Approximating the Delay Path 23

Figure 3.2 State Estimation based on Delayed Information 25

Figure 3.3 State Prediction 26

Figure 3.4 Forward Link Wireless System 28

Figure 3.5 Representation of A Sparse Matrix 35

Figure 3.6 Cost convergence of the no-delay system without reserved power 40

Figure 3.7 Running cost of the no-delay system without reserved power 41

Figure 3.8 Running cost difference of the no-delay system with and without control . 41

Figure 3.9 Power allocation in Channel state 1-no time delay 42

Figure 3.10 Power allocation in Channel state 2-no time delay 42

Figure 3.11 Running Cost Difference between Simple Control and Policy I($D = 1$) 43

Figure 3.12 Running Cost Difference between Policy I and the Optimal Control($D = 1$) 44

Figure 3.13 Cost Convergence of Simple Control, Optimal Control and Policy I($D = 1$) 45

Figure 3.14 Control difference between Simple Control and Optimal Control($D = 1$) .. 45

Figure 3.15 Control difference between Optimal Control and Policy I($D = 1$) 46

Figure 3.16 Running Cost of the Optimal Control (Increased volatility, $D = 2$) 47

Figure 3.17 Difference of Running Costs between Policy I and the Optimal Control (Increased volatility, $D = 2$) 48

Figure 3.18 Difference of Running Costs between the Simple Control and Policy I(Increased volatility, $D = 2$) 48

Figure 3.19 Running Cost Convergence(Increased volatility, $D = 2$) 49

Figure 3.20 Difference of control between the Simple Control and Optimal Control(Increased volatility, $D = 2$) 50

Figure 3.21 Difference of control between the Optimal Control and Policy I(Increased volatility, $D = 2$) 50

Figure 3.22 Running Cost Convergence(Increased time delay, $D = 2$) 51

Figure 3.23 Difference of Running Costs between Policy II and the Optimal Control (Increased time delay, $D = 2$) 51

Figure 3.24 Difference of Running Costs between Simple Control and Policy II (Increased time delay, $D = 2$) 52

Figure 3.25 Difference of control between the Simple Control and Optimal Control(Increased time delay, $D = 2$) 52

Figure 3.26 Difference of control between Policy II and the Optimal Control(Increased time delay, $D = 2$) 53

Figure 3.27 Percentage of Cost Reductions over Simple Control (increased σ) 54

Figure 3.28 Percentage of Cost Reductions over Simple Control (increased τ) 54

Figure 4.1 Approximating the Delay Path 64

Figure 4.2 Controlled Markov Chain Approximation of the System with Delay 65

Figure 4.3 Determine the Control using P steps of Decision Horizon 68

Figure 4.4 Running Cost (No delay case) 75

Figure 4.5 Power allocation for Queue 1 (No delay case) 76

Figure 4.6 Running Cost (One step of delay case) 77

Figure 4.7 Running Cost Difference between the One step of delay case and the no-delay case 77

Figure 4.8 Power Allocation (One step of delay) 78

Figure 4.9 Control Difference Between the delay and no-delay cases 78

Figure 5.1 PAC OAS-Convergence using Method I 104

Figure 5.2 Support OAS-Convergence using Method I 104

Figure 5.3 FNCL OAS-Convergence using Method I 105

Figure 5.4 PAC Price with Interest Rate Shifted Down by 100bps-Convergence using Method I 105

Figure 5.5 Support Price with Interest Rate Shifted Down by 100bps-Convergence using Method I 106

Figure 5.6 FNCL Price with Interest Rate Shifted Down by 100bps-Convergence using Method I 106

Figure 5.7 PAC Price with Interest Rate Shifted Up by 100bps-Convergence using Method I 107

Figure 5.8 Support Price with Interest Rate Shifted Up by 100bps-Convergence using Method I 107

Figure 5.9 FNCL Price with Interest Rate Shifted Up by 100bps-Convergence using Method I 108

Figure 5.10 PAC Duration-Convergence using Method I 108

Figure 5.11 Support Duration-Convergence using Method I 109

Figure 5.12 FNCL Duration-Convergence using Method I 109

Figure 5.13 PAC Convexity-Convergence using Method I 110

Figure 5.14 Support Convexity-Convergence using Method I 110

Figure 5.15 FNCL Convexity-Convergence using Method I 111

LIST OF TABLES

Table 3.1 Simulation Parameters 39

Table 3.2 Average (over initial state) cost 53

Table 4.1 Simulation Parameters 74

Table 5.1 MT-BM, Method I 113

Table 5.2 MT-INV, Method I 113

Table 5.3 NR3-BM, Method I 114

Table 5.4 NR3-INV, Method I 114

Table 5.5 MT-BM, Method II 114

Table 5.6 MT-INV, Method II 115

Table 5.7 NR3-BM, Method II 115

Table 5.8 NR3-INV, Method II 115

Chapter 1

Introduction

Stochastic models are a popular topic in modern research [63]. Recently, the stochastic models with path dependence or time delay are becoming more important and popular [45] [14] [25] [64]. These models are the extension of general stochastic differential equations (SDE), based on the observation that in some situations, not only the present system state has effect on the drift, but some forms of history information will affect the growth.

Because of various forms of path dependency, the closed form of Ito's formula or Dynkin formula are generally hard to find, hence the analytical solutions to the stochastic control problem with time delay. Therefore, the numerical approach becomes a popular tool. There are two main numerical methods, the controlled Markov Chain approximation [47][45], or analogously the finite difference method [64], and the Euler scheme. The controlled Markov Chain approach utilizes a discrete approximation of the continuous time system, which comes from the finite difference method of the Hamilton-Jacobi-Bellman (HJB) equation, and constructs the transition probabilities among the states. By letting the interpolation interval go to zero, it has been proved that the discrete solution converges to the continuous time solution weakly [47]. This method excels at and was first used in solving stochastic control problem with bounded state space [47]. Later it was extended to stochastic control problems with time delay [45] [64]. The Euler scheme discretizes the underlying stochastic process directly [30], and the computation relies on Monte Carlo simulation. This method is widely used in the pricing problem of financial products [11], and deals with path dependent models with unbounded state space.

We turn to the numerical methods in order to facilitate us to find the theoretical

and analytical solutions of the continuous time stochastic problem with path dependence or time delay. It has been found that some stochastic models with certain forms of path dependence have closed form of Ito's formula [25] [49] [50]. Some of its applications are also given there.

1.1 Motivation

The extension of the controlled Markov Chain approximation to stochastic control problem with path dependence or time delay is a newly presented topic. It has been shown that if only the delayed information is used in the discrete scheme, the solution converges optimally if the cost in the control problem is also a function of the delay path [44]. However, in real time control system, the cost is usually a function of the present state, but the time delay is introduced in the observing or control process. That is, the underlying controlled diffusion has time delay or path dependency, but the cost function has no time delay. In this case, the optimality of the formal controlled Markov Chain approximation remains a question. This problem with its application in wireless communication systems will be treated in this work.

The capability of the controlled Markov Chain in handling time delay is limited due to the huge simulation memory requirement. For problems with long path dependence, the Euler scheme with the Monte Carlo simulation becomes a better choice. However, the Monte Carlo simulation is known to have a slow convergence speed generally. In the financial trading system where faster computation is preferred, long computation time is not tolerable. We examined the pricing problem of Mortgage-backed securities in the real market, and presented a new pricing procedure that has fast convergence rate.

1.2 Contributions

1.2.1 Numerical Solutions of Stochastic Control Problems with Time Delay in Wireless Communication Queueing Models

High-speed wireless networks with large capacity requirements are becoming prevalent and system delays need to be considered for good performance. We analyze a power control problem for a forward-link wireless queueing model where only delayed queue state information is available to the controller. The queue-size model is a controlled diffusion

with delay obtained from a modification of an established heavy traffic limit model with no delay.

We consider two approaches for solving the associated stochastic (power) control problems. In the first approach the problems are solved using an application of the numerical methods developed in Kushner [47]. In the second approach, motivated by separation principle ideas, we add a prediction component, using an a priori control policy obtained by another application of the numerical methods of Kushner but where the current queue-state information is available to the controller. A simulation study shows good results: in particular, convergence of the cost function are observed in both approaches and the addition of the prediction component leads to some cost reduction. This motivates further investigation of prediction techniques in the numerical methods for the stochastic control problems.

1.2.2 Numerical Solutions of Stochastic Control Problems with Delayed Control Information

The control policies we derive in Section 1.2.1 give an upper bound on the system running cost. However, it is hard to say if this is the optimal control for the system with time delay. One reason is that, the real system behaves according to the control variable which has been actually applied. Therefore, to estimate the present system state assuming the system is controlled under optimal control for no-delay system, an estimation bias is expected. We present an idea to record a short path of the control history. That means, we should keep in mind what we have done to the system. From the delayed state information which is only available in the realtime control, and knowing all the controls we have already applied, we expect to estimate the present system state more accurately. A major algorithmic improvement is made in this work, where we no longer need to keep the huge transition probability matrix, so that we are able to solve larger scale problems. The simulation results show the effectiveness of this algorithm.

1.2.3 Strong Path Dependent Stochastic Models and the Monte Carlo Simulations

For some strong path dependent models, like pricing mortgage backed securities (MBS), the Monte Carlo simulation is broadly implemented. In this work, we examined

the procedure of Monte Carlo simulations in pricing MBS. Because this method relies on computer generated pseudo random numbers, we are interested in finding out the performance of different random number generators in pricing MBS. We examined the absolute convergence and relative convergence of the pricing procedure based on the Option Adjusted Spread analysis. By using the interim result of OAS which has not shown good convergence yet to compute the duration and convexity of the MBS, we found that the results show early convergence. That means, although the OAS has not achieved its absolute convergence, the relative convergence of the duration and convexity has already been there. In this way, our pricing procedure is speed up dramatically.

1.3 Outline

In this work, we first give the background of classical numerical approach of solving stochastic optimal control problems in continuous time in Chapter 2. Then the extension of the numerical method to solve stochastic control problems with time delay is discussed in Chapter 3. The application to wireless communication queueing models is also studied, where the state estimation effect is tested. In Chapter 4, we present a new control policy based on the discrete controlled Markov Chain approximation. The path of the control history is utilized to estimate the present state. In Chapter 5 where the problem involves long path dependence, we examine the absolute and relative convergence of Monte Carlo simulations in pricing MBS, and present the pricing method using OAS analysis. The conclusions and further topics are given in Chapter 6.

Chapter 2

Background

It is known that some stochastic control problems can be formed into a nonlinear partial differential equations [63]. A basic impediment is that the PDE's often have only a formal meaning, and standard methods of numerical analysis might not be usable to prove convergence of the numerical methods [47]. The convergence here means the numerical solution to be convergent to the analytical solution of the continuous model in some sense. Often in many other models, we can not even derive the PDE at all. And many important classes of new problems appear which are not covered in the analytical area. All these concerns make us turn to the controlled Markov Chain approximation, which is an important and easy-to-use numerical method. The basic idea is to approximate the original problem with a controlled Markov Chain on a finite state space, which is the discretized form of the original state space. A cost function for the Markov Chain model, which is an appropriate analogue to that for the original model is then found. Finally the discrete optimal solution and the optimal cost function are determined by dynamic programming based on this Markov Chain model.

Besides the controlled Markov Chain method, there are some other numerical approaches, like the Euler approximation [30] [4] and the binomial tree approximation [20]. All these methods have difficulties dealing with the stochastic control problem and the convergence to continuous time solutions. In this chapter, we will give an outline of the numerical approach solving stochastic optimal control problems using the controlled Markov Chain approximation, an approach presented by Kushner [47], for the completeness of this work. An extension of this method to stochastic system with time delay will be discussed in the next chapter. The details of this method can be found in [47]. For the convenience of

later chapters which are topics based on this method, we think it is necessary to redescribe the highlights of this procedure.

2.1 The Markov Chain Approximation Method

The analytical solution of a nonlinear stochastic control problem is generally hard to find. Consider the diffusion process model

$$dx(t) = b(x(t), u(t))dt + \sigma(x(t))dw, \tag{2.1}$$

where w is standard Wiener process. Let G be a compact set which is the closure of its interior G^0. For a discounting factor $\beta > 0$, we consider the cost function

$$W(x, u) = E_x^u \int_0^T e^{-\beta t} k(x(t), u(t))dt + E_x^u e^{-\beta T} g(x(T)), \tag{2.2}$$

where $T = \inf\{t : x(t) \notin G^0\}$ is a stopping time that indicates the first escape time of $x(\cdot)$ from G^0. If we consider a diffusion process with reflection

$$dx(t) = b(x(t), u(t))dt + \sigma(x(t))dw + dz(t), \tag{2.3}$$

where $z(\cdot)$ is continuous and keeps the process $x(\cdot)$ from leaving G, then T can be infinity. We also noticed the application of Euler scheme for the reflected stochastic differential equations [51], where the control problem was not concerned. Korf [42] also did some research in the infinite horizon stochastic optimal control problem, but in a discrete time case. We use Kushner's [47] numerical method in order to find the optimal solution to the continuous time system.

Define

$$V(x) = \inf_u W(x, u), \tag{2.4}$$

where the infimum is over all admissible controls. Let $\mathcal{U}$ be the set of the admissible controls.

A general restriction on the model is as follows

Assumption 1 *$b(\cdot)$ and $\sigma(\cdot)$ are bounded, continuous, and Lipschitz continuous in x, uniformly in u. Both $k(\cdot)$ and $g(\cdot)$ are bounded and continuous.*

Pick an interpolation interval h, then use the discrete parameter Markov Chain $\{x_n^h, n < \infty\}$ on the discrete state space S_h to approximate the diffusion process, with

the transition probabilities given by $p^h(x, y|\alpha)$, where α denotes the admissible control. Sometimes we use u_n^h to denote the control action for the chain at discrete time index n.

The interpolation interval for the time dimension is denoted by $\Delta t^h(x, \alpha) > 0$, and let $\Delta t_n^h = \Delta t^h(x_n^h, u_n^h)$. It should satisfy $\sup_{x,\alpha} \Delta t^h(x, \alpha) \to 0$ as $h \to 0$, and $\inf_{x,\alpha} \Delta t^h(x, \alpha) > 0$ for each $h > 0$. Let G_h^0 denote the components of the state space which are interior to the set G, then $G_h^0 = S_h \bigcap G^0$. Thus G_h^0 is the finite state space of the chain until it escapes from G^0.

Define the difference $\Delta x_n^h = x_{n+1}^h - x_n^h$, and let $E_{x,n}^{h,\alpha}$ denote the conditional expectation given $\{x_i^h, u_i^h, i \le n, x_n^h = x, u_n^h = \alpha\}$. An essential condition for this controlled Markov Chain approximation is called the "local consistency" condition, as given below

$$\begin{aligned}
&E_{x,n}^{h,\alpha} \Delta x_n^h = b_h(x, \alpha)\Delta t^h(x, \alpha) = b(x, \alpha)\Delta t^h(x, \alpha) + o(\Delta t^h(x, \alpha)),\\
&E_{x,n}^{h,\alpha}[\Delta x_n^h - E_{x,n}^{h,\alpha}\Delta x_n^h][\Delta x_n^h - E_{x,n}^{h,\alpha}\Delta x_n^h]' = a_h(x)\Delta t^h(x, \alpha)\\
&\qquad = a(x)\Delta t^h(x, \alpha) + o(\Delta t^h(x, \alpha)),\\
&a(x) = \sigma(x)\sigma'(x),\\
&\sup_{n,w} |x_{n+1}^h - x_n^h| \xrightarrow{h} 0, \sup_{x,\alpha} \Delta t^h(x, \alpha) \xrightarrow{h} 0.
\end{aligned} \tag{2.5}$$

We say a control policy $u^h = \{u_n^h, n < \infty\}$ is admissible for the chain if the chain has the Markov property under that policy. In particular, the policy is admissible if

$$P\{x_{n+1}^h = y | x_i^h, u_i^h, i \le n\} = p^h(x_n^h, y|u_n^h). \tag{2.6}$$

Because the control u is a function of the state, i.e., feedback control, sometimes we write $u^h = u(x_n^h)$ to denote the control of a discrete state, and for simplicity, we just use $u()$ and whether it is a continuous function of discrete function depends on the system state representation. Let $E_x^{u^h}$ be the expectation given that $x_0^h = x$ and the admissible control u^h is used.

To better approximate the continuous time diffusion (2.1), we use the continuous time interpolation for convenience, as shown in Figure 2.1. Let $u_n^h, n < \infty$ be an admissible control, and let $t_n^h = \sum_{i=0}^{n-1} \Delta t_i^h$, then the continuous parameter interpolations $x^h(\cdot)$ and $u^h(\cdot)$ are defined by

$$x^h(t) = x_n^h, u^h(t) = u_n^h, t \in [t_n^h, t_{n+1}^h]. \tag{2.7}$$

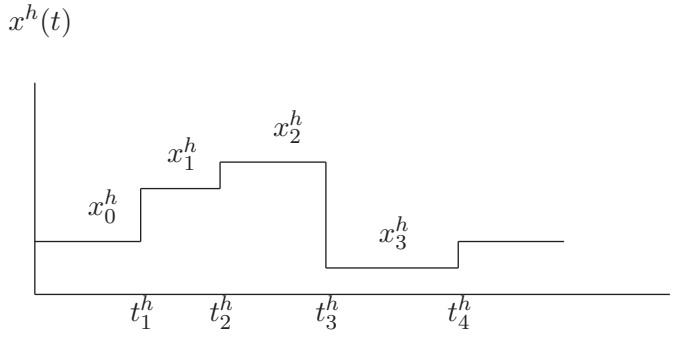

Figure 2.1: Continuous interpolation $x^h(\cdot)$

Let N_h be the first time that the process $\{x_n^h, n < \infty\}$ leaves G_h^0. Now we are ready to approximate the cost function $W(x,u)$. For simplicity, suppose the discount rate is constant, then the approximation is

$$W^h(x,u^h) = E_x^{u^h} \sum_{n=0}^{N_h-1} e^{-\beta t_n^h} k(x_n^h, u_n^h)\Delta t_n^h + E_x^{u^h} e^{-\beta t_{N_h}^h} g(x_{N_h}^h). \tag{2.8}$$

The optimal cost $V^h(x) = \inf_u W^h(x,u^h)$ thus satisfies the following dynamic programming equation

$$V^h(x) = \begin{cases} \min_{\alpha\in\mathcal{U}} \left[\sum_y e^{-\beta\Delta t^h(x,\alpha)} p^h(x,y|\alpha)V^h(y) + k(x,\alpha)\Delta t^h(x,\alpha)\right] & x \in G_h^0, \\ g(x) & x \notin G_h^0. \end{cases} \tag{2.9}$$

2.2 Construction of the Approximating Markov Chains

After setting up the representations and conditions for this procedure, we are ready to illustrate the steps constructing the controlled Markov Chain approximation. A simple one dimensional example is introduced for illustration purpose only. Detailed constructions can be found in [47].

Assume the system satisfies

$$dx = b(x,u(x))dt + \sigma(x)dw, \tag{2.10}$$

where the control is just in the feedback form. Let the stopping time $T = \inf\{t : x(t) \notin (0,B)\}$, where B is the bound of feasible state space. The cost function, which is first

assumed to have no discounting, is

$$\begin{aligned} W(x,u) &= E_x^u\left[\int_0^T k(x(s),u(x(s)))ds + g(x(T))\right], \\ W(x,u) &= g(x), \text{ for } x = 0, B. \end{aligned} \tag{2.11}$$

Apply formally the Itô's formula to the function $W(x,u)$, we get the PDE

$$\mathcal{L}^{u(x)}W(x,u) + k(x,u(x)) = 0, x \in (0,B), \tag{2.12}$$

where $\mathcal{L}^\alpha$ is the infinitesimal generator [47] [63] when the control is fixed at α, and the boundary conditions are $W(0,u) = g(0), W(B,u) = g(B)$. For this example, the PDE is just

$$\frac{1}{2}\sigma^2 W_{xx}(x) + bW_x(x) + k(x,\alpha) = 0, \tag{2.13}$$

where $W_x(\cdot), W_{xx}(\cdot)$ denote the first and second derivatives with respect to x. Suppose that $\sigma^2(x) \geq h|b(x,\alpha)|$ for all x, α, apply the central difference method in finite difference algorithms to $W(\cdot)$,

$$\begin{aligned} f_{xx}(x) &\to \frac{f(x+h) + f(x-h) - 2f(x)}{h^2}, \\ f_x(x) &\to \frac{f(x+h) - f(x-h)}{2h}. \end{aligned} \tag{2.14}$$

Then for $x \in G_h$, it leads to the finite difference approximation

$$W^h(x) = p^h(x, x+h|\alpha)W^h(x+h) + p^h(x, x-h|\alpha)W^h(x-h) + \Delta t^h(x,\alpha)k(x,\alpha), \tag{2.15}$$

where

$$W^h(x,u) = E_x^u \sum_{n=0}^{N_h-1} k(x_n^h, u(x_n^h))\Delta t^h(x_n^h, u(x_n^h)) + E_x^u g(x_{N_h}^h), \tag{2.16}$$

and N_h is the discrete stopping time. Thus the transition probabilities are given as

$$p^h(x, x \pm h|\alpha) = \frac{\sigma^2(x) \pm hb(x,\alpha)}{2\sigma^2(x)}, \tag{2.17}$$

and

$$\Delta t^h(x,\alpha) = \frac{h^2}{\sigma^2(x)}, x \in G_h. \tag{2.18}$$

The local consistency condition (2.5) holds for this form of transition probabilities in (2.17), and this controlled Markov Chain can serve as the approximation to the original system.

Now the dynamic programming equation, or the Bellman equation can be derived as

$$V^h(x) = \inf_{\alpha \in \mathcal{U}^h} \left\{ \sum_y p^h(x, y|\alpha) V^h(y) + k(x, \alpha) \Delta t^h(x, \alpha) \right\}, \tag{2.19}$$

with the boundary conditions

$$V^h(-h) = V^h(0) + c_1 h, \ V^h(B + h) = V^h(B) + c_2 h, \tag{2.20}$$

where c_1, c_2 are the penalty cost for the infeasible states, which is used for the infinite time horizon diffusion with reflecting boundaries.

The finite difference approximation is used here only to get the transition probabilities of a Markov Chain which is locally consistent with (2.10).

To accommodate the discounting factor in the cost function, like in (2.8), $W^h(x, u)$ should satisfy

$$W^h(x, u) = \sum_y e^{-\beta \Delta t^h(x, u(x))} p^h(x, y|u(x)) W^h(y, u) + k(x, u(x)) \Delta t^h(x, u(x)) \tag{2.21}$$

for $x \in G_H^0$, and with the boundary condition

$$W^h(x, u) = g(x), \text{ for } x \in \partial G_h. \tag{2.22}$$

The dynamic programming equation for the optimal cost is just

$$V^h(x) = \inf_{\alpha \in \mathcal{U}^h} \left\{ e^{-\beta \Delta t^h(x, \alpha)} \sum_y p^h(x, y|\alpha) V^h(y) + k(x, \alpha) \Delta t^h(x, \alpha) \right\}. \tag{2.23}$$

Here we extend the example to the multiple dimensional case. The dynamic programming equation is just like (2.9). If the interpolation intervals are sufficiently small, we can replace the exponential terms in this case by $1 - \beta \Delta t^h(x, \alpha)$ or $1/[1 + \beta \Delta t^h(x, \alpha)]$ for convenience.

This procedure can be written in a simpler vector form. Let $u(\cdot)$ be a feedback control. Define the vectors $W^h(u) = \{W^h(x, u), x \in G_h^0\}$ and $V^h = \{V^h(x), x \in G_h^0\}$. Define the cost vector $C^h(u) = \{C^h(x, u), x \in G_h^0\}$ with components

$$C^h(x, u) = k(x, u(x)) \Delta t^h(x, u(x)) + e^{-\beta \Delta t^h(x, u(x))} \sum_{y \in \partial G_h} p^h(x, y|u(x)) g(y) \tag{2.24}$$

for $x \in G_h^0$. Define the discounted transition probability matrix $R^h(u) = \{r^h(x, y|u(x)); x, y \in G_h^0\}$, where

$$r^h(x, y|u) = e^{-\beta \Delta t^h} p^h(x, y|u) \tag{2.25}$$

for $x \in G_h^0$, and G_h^0 is the set of interior points of the feasible region of the discretized state. Then the cost equation (2.21) can be written as in the following dynamic programming form

$$W^h(u) = R^h(u)W^h(u) + C^h(u), \tag{2.26}$$

$$V^h = \min_{u \in \mathcal{U}} [R^h(u)V^h + C^h(u)]. \tag{2.27}$$

We may further extend this method to accommodate problems with reflecting boundaries. Suppose the process follows

$$x(t) = x + \int_0^t b(x(s), u(s))ds + \int_0^t \sigma(x(s))dw(s) + z(t), \tag{2.28}$$

where the reflecting term $z(\cdot)$ is continuous and keeps the process $x(\cdot)$ from leaving G. Thus for each $x \in \partial G$, we are given a set of direction vectors $r(x)$ whose members are of unit length. It is required for $z(\cdot)$ to satisfy

$$|z|(t) = \int_0^t I_{\partial G}(x(s))d|z|(s), z(t) = \int_0^t \gamma(s)d|z|(s), \tag{2.29}$$

where $\gamma(s) \in r(x(s))$ almost surely (w, s) with respect to the random measure induced by $|z|(\cdot)$. Further technical requirements on the reflection term can be found in [24].

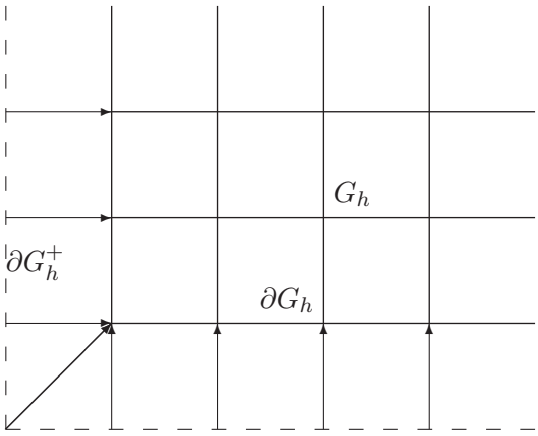

Figure 2.2: Approximating Grid with Reflecting Boundaries

The term ∂G_h^+ indicates the states of one unit length (not at corners technically) outside of the feasible region. If $x \in \partial G_h^+$, we know the state has jumped out of the feasible state space, then the reflecting term $z(\cdot)$ becomes active, which will bring the state back into the feasible region with probability one. A penalty cost will thus be incurred. In this scenario, the cost function is

$$W(x,u) = E_x^u \int_0^\infty e^{-\beta t}[k(x(t),u(t))dt + c'(x(t))dz(t)], \tag{2.30}$$

where $c(\cdot)$ is bounded and continuous, which defines a reflecting direction, and $c'(x)\gamma \geq 0$ for any $\gamma \in r(x), x \in \partial G$. Notice that this is an unstopped process with boundary reflection.

The discrete approximation for this cost function is

$$W^h(x,u) = E_x^u \left\{ \sum_{n=0}^{\infty} e^{-\beta t_n^h} \left[k(x_n^h, u_n^h)\Delta t_n^h + c'(x_n^h)\Delta z^h(x_n^h) \right] \right\}, \tag{2.31}$$

for an admissible control sequence $u = \{u_n^h, n < \infty\}$ and $x \in G_h$. Let the control be in a general feedback form, $u_n^h = u(x_n^h)$, then if the sum in (2.31) is well defined and bounded, $W^h(x,u)$ satisfies (2.21). Analogously the dynamic programming equation for the optimal cost for $x \in G_h$ is given in (2.23).

If $x \in \partial G_h^+$, then

$$W^h(x,u) = \sum_y p^h(x,y)W^h(y,u) + c'(x)\Delta z^h(x), \tag{2.32}$$

and the dynamic programming equation for the optimal cost is

$$V^h(x) = \sum_y p^h(x,y)V^h(y) + c'(x)\Delta z^h(x). \tag{2.33}$$

The reflection properties of the approximating Markov Chain models the "instantaneous" reflection of the underlying diffusion. The interpolation time interval for the reflections steps equals zero. Hence there is no discounting factor in (2.33).

2.3 Computational Methods for Controlled Markov Chains

Recall our problem formulation

$$\begin{aligned} W^h(x,u) &= \sum_y r^h(x,y|u(x))W^h(y,u) + C^h(x,u(x)), \\ V^h(x) &= \min_{\alpha \in (U)} \left[\sum_y r^h(x,y|\alpha)V^h(y) + C^h(x,u(x)) \right], \end{aligned} \tag{2.34}$$

for appropriate $r^h(x, y|\alpha)$ and $C^h(x, \alpha)$. The vector forms are

$$\begin{aligned} W^h(u) &= R^h(u)W^h(u) + C^h(u), \\ V^h &= \min_{u(x)\in\mathcal{U}} \left[R^h(u)V^h(u) + C^h(u)\right]. \end{aligned} \tag{2.35}$$

Besides the Assumption 1 in Section 2.1, further restrictions on the problems are as follows

Assumption 2 *$r^h(x, y|\alpha), C(x, \alpha)$ are continuous functions of α for each x, y in S, where S is the feasible region.*

Assumption 3 *There is at least one admissible feedback control $u_0(\cdot)$ such that $R^h(u_0)$ is a contraction, and the infima of the costs over all admissible controls is bounded from below. $R^h(u)$ is a contraction for any feedback control $u(\cdot)$ for which the associated cost is bounded.*

Assumption 4 *If the cost associated with the use of the feedback controls*

$$u^1(\cdot), u^2(\cdot), \ldots, u^n(\cdot), \ldots$$

in sequence, is bounded, then

$$R(u^1) \cdots R(u^n) \xrightarrow{n} 0.$$

We also write $u_{(n)}(\cdot)$ of $u^n(\cdot)$ to indicate the controls obtained in the nth iteration.

The solving procedure, in brief, follows an iteration algorithm. Given an initial guess of the value function $W^h(\cdot)$, a control policy is first derived, then a new Markov transition probability matrix R^h is calculated. Next a new cost vector $W^h(\cdot)$ is obtained, and so forth until convergence exists.

The following theorem is from Theorem 2.1 in [47]:

Theorem 1 *Assume the assumptions 2 and 3, then there is a unique solution to (2.35), and it is the infimum of the cost functions over all time independent feedback controls. Let $u_{(0)}(\cdot)$ be an admissible feedback control such that the cost $W(u_{(0)})$ is bounded. For $n \geq 1$, define the sequence of feedback controls $u_{(n)}(\cdot)$ and costs $W^h(u_{(n)})$ recursively by (2.35) together with the formula*

$$u_{n+1}(x) = \arg\min_{\alpha\in\mathcal{U}} \left[\sum_y r^h(x, y|\alpha)W^h(y, u_{(n)}) + C(x, \alpha))\right]. \tag{2.36}$$

Then $W^h(u_{(n)}) \to V^h$.

Under the assumption 4, V^h *is the infimum of the costs over all admissible control sequences.*

After the control is determined, we can approximate the value vector $W^h(\cdot)$. The following theorem is from Theorem 2.2 in [47]:

Theorem 2 *Let* $u(\cdot)$ *be an admissible feedback control such that* $R^h(u)$ *is a contraction. Then for any initial vector* $W^h_{(0)}$*, the sequence* $W^h_{(n)}$ *defined by*

$$W^h_{(n+1)}(x,u) = \sum_y r^h(x,y|u(x))W^h_{(n)}(y,u) + C(x,u(x)) \tag{2.37}$$

converges to $W^h(u)$*, the unique solution to (2.35). Assume the assumptions 2-4, then for any vector* $V^h_{(0)}$*, the sequence recursively defined by*

$$V^h_{(n+1)} = \min_{u(x)\in\mathcal{U}}[R^h(u)V^h_{(n)} + C(u)] \tag{2.38}$$

converges to V^h*, the unique solution to (2.35). In detail, (2.38) is*

$$V^h_{(n+1)} = \min_{\alpha\in\mathcal{U}}\left[\sum_y r^h(x,y|\alpha)V^h_{(n)}(y) + C(x,\alpha)\right]. \tag{2.39}$$

$V^h_{(n)}$ *is the minimal cost for an* $n-$*step problem with terminal cost vector* $V_{(0)}$.

The iterations in (2.37) and (2.38) calculates all the values of the components of the $(n+1)$st iterations $W^h_{(n+1)}(u)$ and $V^h_{(n+1)}$ respectively, directly from the components of $W_{(n)}(u)$ and $V_{(n)}$ respectively. An advantage is that it needs little computer memory in the simulations. A disadvantage is its longer computing time. If the memory restrict is tight, and larger scale problems are considered, this algorithm is a better choice than the Gauss-Seidel iteration, which is introduced as follows.

It is argued that in the above algorithm, the newly calculated values $W^h_{(n+1)}(u)$ and $V^h_{(n+1)}$ are not used until they are available for all $x \in S$. If the computation uses successive substitutions, in the sense that each newly calculated term $W^h_{(n+1)}(u)$ and $V^h_{(n+1)}$ is immediately substituted for the $W^h_{(n)}(u)$ and $V^h_{(n)}$ respectively, the procedure is then referred to as the Gauss-Seidel procedure. Suppose the states in the feasible space is ordered, then we may let the inequality sign $<$ denote the ordering. And the Gauss-Seidel iteration algorithm is in the following, which is from Theorem 2.3 in [47],

Theorem 3 *Let $u(\cdot)$ be an admissible feedback control for which $R^h(u)$ is a contraction mapping. For any given $W^h_{(0)}$, define $W^h_{(n)}$ recursively by*

$$W^h_{(n+1)}(x,u) = \sum_{y<x} r^h(x,y|u(x))W^h_{(n+1)}(y,u) + \sum_{y\geq x} r^h(x,y|u(x))W^h_{(n)}(y,u) + C(x,u(x)). \tag{2.40}$$

Then $W^h_{(n)}$ converges to the unique solution to (2.35).

Assume the assumptions 2-4, for any $V^h_{(0)}$, define $V^h_{(n)}$ recursively by

$$V^h_{(n+1)}(x) = \min_{\alpha\in\mathcal{U}} \left[\sum_{y<x} r^h(x,y|\alpha)V^h_{(n+1)} + \sum_{y\geq x} r^h(x,y|\alpha)V^h_{(n)}(y) + C(x,\alpha) \right]. \tag{2.41}$$

Then $V^h_{(n)}$ converges to the unique solution of 2.35.

The Gauss-Seidel iteration solves the problem in the vector form, or linear system form. If the problem's scale is small, this algorithm gives faster convergence. However, if the problem requires huge computer memory, this approach can not be applied since it needs to store the huge linear system, which includes the value vectors and the transition matrix. Both algorithms will be studied in later chapters.

2.4 Weak Convergence Proofs

In this section, we will highlight the convergence proof of the controlled Markov chain approximation to the continuous time solution. This is a general advantage of this numerical scheme. Details can be found in [47] Chapter 9 and 10. An analogous numerical method using binomial tree, but without a stochastic control component, can be found in [84] [70], where the convergence was studied. The convergence of the Euler scheme, also without the control component, is seen in [90] [55].

The approach to proving the convergence of numerical schemes will be based on proving the weak convergence of a sequence of stochastic process to an appropriate limit process. Let S denote a metric space with metric d and let $C(S)$ denote the set of real valued continuous functions defined on S. Let $C_b(S)$ and $C_0(S)$ denote the subsets of $C(S)$ given by all continuous functions that are bounded and have compact support, respectively.

Definition 1 *Let $X_n, n < \infty$ and X be $S-$valued random variables. Let E_n and E denote expectation on the probability spaces on which X_n, X are defined. Then the sequence*

$\{X_n, n < \infty\}$ converges in distribution to X if $E_n f(X_n) \to Ef(X)$ for all $f \in C_b(S)$. Let $P_n, n < \infty$ and P denote the measures defined on $(S, \mathcal{B}(S))$ that are induced by $X_n, n < \infty$ and X, respectively. Then for any $f \in C_b(S)$,

$$E_n f(X_n) \to Ef(X) \Leftrightarrow \int_S f(s) P_n(ds) \to \int_S f(s) P(ds).$$

This form of convergence of probability measures is called weak convergence, and use the notation $P_n \Rightarrow P$.

We also write the above convergence as $X_n \Rightarrow X$. More properties on weak convergence can be found in [9].

The convergence is first proved for the uncontrolled diffusion.

$$dx(t) = b(x(t))dt + \sigma(x(t))dw(t), \tag{2.42}$$

with $x(0) = x$ given. Recall the continuous time interpolation $x^h(\cdot)$ in Fig 2.1 and assume the local consistency holds. Kushner first showed that the collection $\{x^h(\cdot), h > 0\}$ is tight, see Theorem 4.1 in [47]. Then it is shown that the process $x^h(\cdot)$ converges weakly to solution $x(\cdot)$ of equation (2.42), see Theorem 4.2 in [47]. This implies $W^h(x) \to W(x)$ (Theorem 4.3 in [47]) by the property of weak convergence.

For the analogous problem involving controlled processes (2.1), it will be necessary to consider sequences of controlled Markov Chains. In this case, besides the convergence properties of the process, we must also deal with the convergence of the controls. The condition on the proof is to require that the functions $b(\cdot), \sigma(\cdot), k(\cdot)$ and $g(\cdot)$ are bounded and continuous. By considering the admissible control as a measure, which is called relaxed control, Kushner [47] showed the weak convergence of the cost function W^h and the control as well. And finally the optimal cost $V^h(x) \to V(x)$ as $h \to 0$. Interested reader may refer to Chapter 10 in [47].

We notice that the determination of the discrete control policy (2.36) does not play a key role in the convergence proof. It only guarantees an optimal policy for the discrete controlled Markov Chain approximation. And as $h \to 0$, the control and cost converges to the optimal continuous solutions weakly. In other words, if we choose another policy instead of (2.36), and follow the same construction procedure, this new discrete control policy and cost should converge to a certain continuous time control policy and cost function, though rigorous proof is necessary. For the no-delay problem as (2.1), equation

(2.36) gives the best choice for sure, because it converges to the optimal solution. However, for the diffusion process with time delay, different discrete control policies may yield different control results, each may converges to a certain continuous time solution. How to find the optimal control policy in the time delay case remains an interesting issue. More work on the study of different discrete control policies in the diffusion system with time delay and their applications will be presented in later chapters.

Chapter 3

Numerical Solutions for Nonlinear Stochastic Control Problems with Time Delay and Its Application to Wireless Communication Systems

The numerical schemes solving stochastic control problems using controlled Markov Chain approximations are powerful enough to be extended to stochastic control problem with delay [44]. In this chapter, we will first introduce the classical approach using this method to solve stochastic system control problems. As the interpolation interval goes to zero, the numerical solution converges to a certain continuous time solution, and the optimality is proved if only the delayed information is used. This control policy is called simple control policy in this paper. By introducing a state estimation procedure, we are expected to find better control policies. In this chapter, a separation principle is applied in the state estimation. We showed that this control policy yields better control result than that of the simple control. Finally, the application to wireless communication queueing system is studied, where the time delay is a common issue.

3.1 Numerical Methods for Nonlinear Stochastic Control with Time Delay

The stochastic diffusion process with time delay is a recent model [88] [81]. The delay or history information can be introduced in two general ways. One is the delay in the path of the underlying process, and the other is the time delay in control. Let the maximum delay be $\tau > 0$. Let $\mathcal{U}$ denote the compact control valued space, and the controls $u(\cdot)$ are $\mathcal{U}-$valued, measurable. Let $\bar{x}(t,\theta)$ denote the path segment of the process $x(t+\theta), \theta \in [-\tau, 0], \tau > 0$. We also write $\bar{x}(t) = \bar{x}(t,\cdot)$. The control $\bar{u}(t)$ is defined analogously from $u(\cdot)$. The solution $x(t)$ is confined to a finite interval $G = [0, B]$ by reflection, which is denoted by $z(\cdot)$. Its purpose is to assure that $x(t)$ does not escape from the feasible interval G. The effect of $z(\cdot)$ is to bring the simulated state into its feasible region instantly whenever the state is beyond the boundary. Since the states of the real system will never be infeasible, independent of whether there is time delay or not, the reflecting term $z(\cdot)$ must have no delay. In other words, although the current system state is unknown to the controller, $z(\cdot)$ works whenever the unknown current state is beyond its boundary.

Let $w(\cdot)$ be a standard Wiener process, then the diffusion model with path delay is

$$dx(t) = b(\bar{x}(t), u(t))dt + \sigma(\bar{x}(t))dw(t) + dz(t), \bar{x}(0) \text{ given.} \tag{3.1}$$

Let $D[S; t_1, t_2]$ denote the set of $S-$valued functions that are right continuous on $[t_1, t_2)$, have left hand limits on $(t_1, t_2]$ and with the Skorohod topology [8]. An application of this model is in the financial market. Some traders would analyze the performance of a certain stock over a short path and then make the decisions. Their actions would in turn affect the stock price. That is why the path information exists in the diffusion model.

An assumption on the model is

Assumption 5 *$b(\cdot)$ is bounded and measurable and is continuous on $D[G; -\tau, 0]\times\mathcal{U}$ at each point $(\bar{x}(\cdot), \alpha)$ such that $\bar{x}(\cdot)$ is continuous. The function $\sigma(\cdot)$ is bounded and measurable and is continuous on $D[G; -\tau, 0] \times \mathcal{U}$ at each point $\bar{x}(\cdot)$ that is continuous.*

An extension where the control is also delayed is given below

$$dx(t) = b(\bar{x}(t), \bar{u}(t))dt + \sigma(\bar{x}(t))dw(t) + dz(t), \bar{x}(0), \bar{u}(0) \text{ given.} \tag{3.2}$$

We consider a particular form of (3.2)

$$dx(t) = b(\bar{x}(t), u(\bar{x}(t)))dt + \sigma(\bar{x}(t))dw(t) + dz(t), \bar{x}(0) \text{ given.} \tag{3.3}$$

This is the model where the control is a function of path information, for example, $u(x(t-\tau))$. This is a general diffusion form to model systems where there is a time delay in the state observation and the control is in feedback form. In wireless communication systems where the queue size is modeled as a diffusion, time delay is a common issue due to the mechanical design or noises.

For certain forms of diffusion with time delay, the analytical solutions are possible to find [49] [50]. But for general solutions of (3.2), little is known of the analytical solution, and numerical methods come into place. Most of the numerical methods deal with the discrete stochastic models, especially the Euler method. Li [74] showed that the Euler method converges in solving stochastic delay differential equations with jumps under weaker conditions than linear growth or global Lipschitz. Bauer [6] studied the infinite dimensional path dependent model. Buckwar [17] applied the Euler's method in SDE with one step of delay. The Euler's method in SDE can also be found in [43]. However, little was done with the control problem. After reducing the model to finite dimensional model, the Hamiltonian principle is used to find the optimal solution of the reduced model as an approximation of the original model. As pointed by Kushner [47], most of the numerical methods have difficulty dealing with the proof of convergence to continuous time diffusions, where the controlled Markov Chain approximation has such advantage. Pang [64] [65] showed the theoretical feasibility of the finite difference method to solve stochastic control problems with time delay, yet the implementation or simulation has not bee doen. Besides this, not much result is known about the controlled Markov Chain method solving stochastic control problem with delay in continuous time, especially the numerical applications.

Consider the form (3.3), which is the base model of our interest. Again, construct a controlled process $x_n^h, n \geq 0$ and interpolation intervals $\Delta t_n^h, n \geq 0$, in much the same way as was done for the no delay case. Let $\bar{x}_0^h \in D[G_h; -\tau, 0]$ be the piecewise constant approximation and that converges to $\bar{x}(0)$ uniformly on $[-\tau, 0]$. G_h is the feasible space of the discrete approximating process. Given the $x_n^h, \Delta t_n^h$, define $t_n^h = \sum_{i=0}^{n-1} \Delta t_i^h$. Define $x^h(\cdot)$ such that on $[0, \infty)$, it is the continuous time interpolation of x_n^h with intervals $\{\Delta t_n^h\}$, and the segment on $[-\tau, 0]$ is $\bar{x}_0^h$, where $x_0^h = \bar{x}_0^h(0)$. Define

$$\bar{x}^h(t, \theta) = x^h(t + \theta), \text{ for } \theta \in [-\tau, 0],$$

and set $\bar{x}_n^h(\cdot) = \bar{x}^h(t_n^h, \cdot)$, the interpolated path on $[t_n^h - \tau, t_n^h]$. We also write $\bar{x}_n^h = \bar{x}_n^h(\cdot)$.

Analogously to the no-delay case, the chain and intervals are assumed to satisfy the following properties. There is a function $\Delta t^h(\cdot)$ on $D[G_h; -\tau, 0] \times \mathcal{U}$ such that $\Delta t_n^h = \Delta t^h(\bar{x}_n^h, u_n^h)$ where u_n^h is the control applied at time n. The distribution of x_{n+1}^h, given the initial data and $x_i^h, u_i^h, i \le n$, will depend only on $\bar{x}_n^h, u_n^h$ and not otherwise on n. Let $E_{\bar{x}_0^h}^{h,\alpha}$ denote the expectation under control value $u_0^h = \alpha$, and define

$$\beta_n^h = [\Delta x_n^h - E\{\Delta x_n^h | x_i^h, u_i^h, i \le n\}] I_{\{x_n^h \in G_h\}},$$

then the local consistency for $x_n^h, \Delta t_n^h$ is said to hold if, for x_0^h,

$$\begin{aligned}
&E_{\bar{x}_0^h}^{h,\alpha} \Delta x_0^h = b^h(\bar{x}_0^h, \alpha)\Delta t^h(\bar{x}_0^h, \alpha) = b(\bar{x}_0^h, \alpha)\Delta t^h(\bar{x}_0^h, \alpha) + o(\Delta t^h(\bar{x}_0^h, \alpha)),\\
&E_{\bar{x}_0^h}^{h,\alpha} \beta_0^h [\beta_0^h]' = a^h(\bar{x}_0^h)\Delta t^h(\bar{x}_0^h, \alpha) = a(\bar{x}_0^h)\Delta t^h(\bar{x}_0^h, \alpha) + o(\Delta t^h(\bar{x}_0^h, \alpha)),\\
&a(\bar{x}_0^h) = \sigma(\bar{x}_0^h)\sigma'(\bar{x}_0^h),\\
&\sup_{n,w} |x_{n+1}^h - x_n^h| \xrightarrow{h} 0, \sup_{\bar{x}_0^h, \bar{u}_0^h} \Delta t^h(\bar{x}_0^h, \bar{u}_0^h) \xrightarrow{h} 0.
\end{aligned} \tag{3.4}$$

The reflecting boundary is treated exactly as for the no-delay case. In particular, if $x_n^h = -h$ or $B + h$, then $x_{n+1}^h = 0$ or B respectively.

The control objective is again to minimize a certain cost function. After discretization, and let $E_{\bar{x}_0^h}^{h,u^h}$ denote the expectation given initial condition $\bar{x}_0^h$ and control sequence $u^h = \{u_n^h, n < \infty\}$, the approximated cost function is

$$\begin{aligned}
W^h(\bar{x}_0^h, u^h) &= E_{\bar{x}_0^h}^{h,u^h} \sum_{n=0}^{\infty} e^{-\beta t_n^h} [k(\bar{x}_n^h, u_n^h)\Delta t^h(\bar{x}_n^h, u_n^h) + c'\delta z_n^h],\\
V^h(\bar{x}_0^h) &= \inf_{u^h} W^h(\bar{x}_0^h, u^h),
\end{aligned} \tag{3.5}$$

where $c'\delta z_n^h$ gives the penalty at the boundary. Assume the cost function is well defined. As in Chapter 2, because the control u is a function of the state, i.e., feedback control, sometimes we write $u^h = u(x_n^h)$ to denote the control of a discrete state, and for simplicity, we just use $u()$ and whether it is a continuous function or discrete function depends on the system state representation. Suppose $\sigma^2(\bar{x}_0^h) \ge h|b(\bar{x}_0^h, \alpha)|$. Kushner [47] showed that for the systems like (3.3), the transition probabilities are given by

$$\begin{aligned}
&P_{\bar{x}_0^h}^{h,\alpha}\{x_1^h = x_0 \pm h\} = \frac{\sigma^2(\bar{x}_0^h) \pm hb(\bar{x}_0^h, \alpha)}{2\sigma^2(\bar{x}_0^h)},\\
&\Delta t^h(\bar{x}_0^h, \alpha) = \frac{h^2}{\sigma^2(\bar{x}_0^h)}.
\end{aligned} \tag{3.6}$$

The cost rate becomes $k(\bar{x}_0^h, \alpha)$. If $\sigma(\cdot)$ is a constant, then the intervals Δt_n^h are all h^2/σ^2.

Theorem 4 *Let $x_n^h, \Delta t_n^h$ be locally consistent, whose initial condition is $\bar{x}(0)$, a continuous function. Let $\bar{x}_0^h \in D[G_h; -\tau, 0]$ be any piecewise constant sequence that converges to $\bar{x}(0)$ uniformly on $[-\tau, 0]$. Assume the Assumption 5, and assume there is a weak sense unique weak sense solution to the system, and the transition probabilities are given as in (3.6), then $V^h(\bar{x}_0^h) \to V(\bar{x}(0))$.*

This is Theorem 4.1 in [45], where the weak sense convergence of the numerical method is proved. The optimality is also proved [44], by using a semigroup representation of the model [83]. Detailed information can be found in [46]. We need to point out that the method guarantees the optimality if only the delayed information is used. That is to take the delayed state as the present state and apply simply the same transition probability form as in the no-delay case. We are interested in examining the control effect by estimating the present system state.

In the transition probability formula for the simple control policy, the control is a function of the delayed state. The system behaves just as described by (3.3). If the cost function is also a function of the path, then by taking the discrete approximation of the path, which is a sequence of discrete states, as the state representation, it follows that this approximating system is just a standard controlled Markov Chain approximation, where the state is not just a single state but a sequence of states. This is the standard controlled Markov Chain approximation with extended state representation. Kushner [47] showed that for this kind of problem, the numerical method with the transition probabilities given by (3.6) guarantees the optimality of the result.

However, in most cases, the cost function is a function of the most recent state but not of the path. Time delay is only introduced to the underlying process. Then by simply applying the transition probabilities (3.6) and taking the delayed path information as the present system state may not guarantee the optimality of the result. Because of the time delay, the system is controlled under partial information, and the problem becomes stochastic control problem with partial information. State prediction or estimation is expected to yield a better control affect, where the separation principle is a common tool. Little is known so far of applying the separation principle or state estimation to the controlled Markov Chain approximation to solve continuous nonlinear stochastic control problem with time delay.

3.2 Controlled Markov Chain Approximation with State Estimation

In this section we are concerned again with the stochastic control problem with partial information,

$$dx(t) = b(\bar{x}(t), u(\bar{x}(t)))dt + \sigma(\bar{x}(t))dw(t) + dz(t), \bar{x}(0) \text{ given}, \tag{3.7}$$

but with the no-delay cost function

$$W(x,u) = E_x^u \int_0^\infty e^{-\beta t}[k(x(t),u(t))dt + c'(x(t))dz(t)], \tag{3.8}$$

whose discrete form is

$$W^h(x,u) = E_x^u \left\{ \sum_{n=0}^{\infty} e^{-\beta t_n^h} \left[k(x_n^h, u_n^h)\Delta t_n^h + c'(x_n^h)\Delta z^h(x_n^h) \right] \right\}. \tag{3.9}$$

Figure 3.1 illustrates the way we approximate the delay path $\bar{x}(t)$ in one dimension assuming the present time is $n\Delta t^h$. D is the number of interpolations in the delay path of lenght τ, so in Fig 3.1 we approximated $2D\Delta t^h$ length of the path. Vector $\mathbf{x}_c = [x_{n-D}^h, \ldots, x_n^h]$ contains all the recent information, called complete information, and $\mathbf{x}_d = [x_{n-2D}^h, \ldots, x_{n-D}^h]$ contains the old information with the same length. More discussion is given in the application to the wireless queueing system.

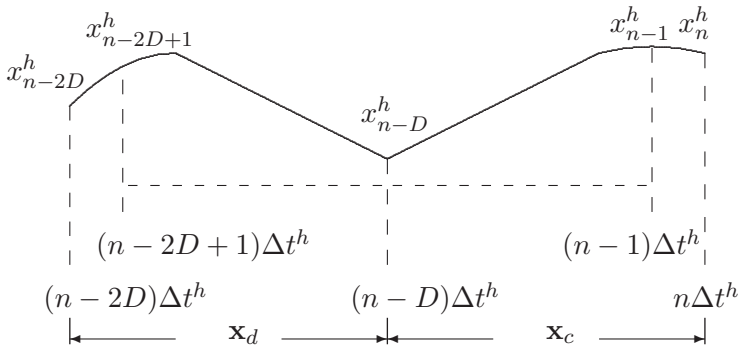

Figure 3.1: Approximating the Delay Path

In particular, we consider a simpler expression of $\bar{x}(t)$, say, $\bar{x}(t) = x(t-\tau)$, i.e., a fixed time delay process, to illustrate our state estimation procedure. For this kind of control system with delay, where the current system state x_n^h is not known, the control

strategy is derived from the delayed information x^h_{n-D}. A simple control strategy is to assume x^h_{n-D} to be the true system state, although the control variable is actually applied to x^h_n. We call this control simple policy.

As we mentioned, the classical controlled Markov Chain approximation with the transition probability form (3.6), simple policy, may not necessarily yield the optimal result. Here we develop procedures for adding a prediction component in the numerical methods. The addition of the prediction components in various control problems with delays loosely motivates our general approach. For example, Altman [3] applied the discrete stochastic control with time delay to a network queueing congestion control problem, where the current system state is predicted based on delay information. The simulation results, which used Monte-Carlo-like methods, show the efficiency of the control derived from predicted state, which was suboptimal. Koole [41] studied finite dimensional discrete stochastic control problems with partial observations. The idea is to transform the model to an equivalent system with full observation, where the separation principle was shown to hold under some conditions. A robust state-predictive control strategy was proposed by Marinescu and Bourles [56], where the method is based on a separation property between control and prediction. Ge [33] applied the separation technique to a nonlinear MIMO time-delay systems. The idea in [57] is to construct another system that approximates the original system. By finding a δ optimal control of the approximated system, it can be shown that it is also an ϵ optimal solution of the original system. Similar ideas can be found in [75]. The state estimates and separation principle idea can also be found in [76], [80].

A summary of the separation principle ideas for a general discrete linear system driven by white noise is described in the following. Let x_n be the state of the system at discrete time n with delay D in the observation of the state; i.e., at n the most recent state observed is x_{n-D}, giving a partial state information case. Let u^* be the optimal control for the system where there is no delay in observation of the state, i.e. the full state information case. In the system simulation, we may just derive the optimal control policy, assuming that there is no time delay, though it may not be optimal for the delayed real time system control. The separation principle result is that the optimal control in the partial state information case is given by $u^*(\hat{x}_n)$ where $\hat{x}_n = E^\alpha[x_n|x_{n-D}]$ for some control policy α. In other words, the optimal control uses u^* from the full state information case with the prediction $\hat{x}$, there is a separation of obtaining the optimal control u^* and obtaining an estimate of the current state, x^h_n. See [7] and [63] for more information.

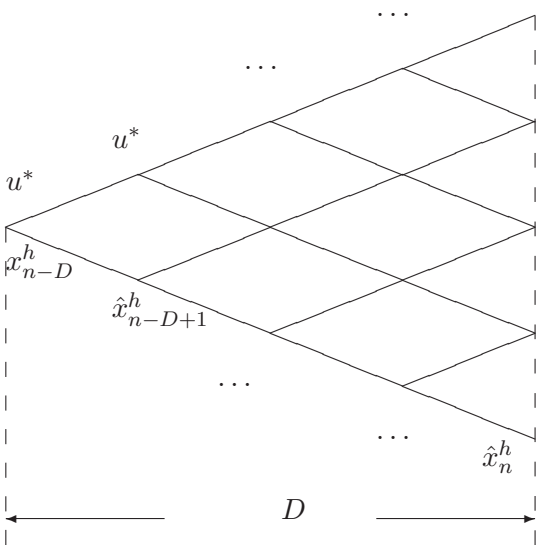

Figure 3.2: State Estimation based on Delayed Information

Following the above separation principle description, we add a prediction component in the numerical methods for the stochastic control problem when only $\bar{x}(t) = x(t-\tau)$ is observable. For the stochastic models considered here, we predict using u_c^* , the optimal control under complete state information case. That means, we predict the present system state x_n assuming that the system is controlled under u_c^*, and $\hat{x}_n = E^{u_c^*}[x_n|x_{n-D}]$. Under the controlled Markov Chain approximation scheme, the states are discrete. So we write them as $\hat{x}_n^h = E^{u_c^*}[x_n^h|x_{n-D}^h]$. The present state x_n^h is a random variable based on available information and can take the value $v_1, v_2, \ldots, v_m$, which are a finite number of outcomes. Under the control u_c^*, each outcome $v_i, i = 1, 2, \ldots, m$ has a certain probability $p_i, i = 1, 2, \ldots, m$. Therefore, $\hat{x}_n^h$ takes the form $\hat{x}_n^h = \sum_{i=1}^m v_i p_i$. This prediction brings convenience for theoretical proof, however, it is hard to apply numerically. One reason is that $\hat{x}_n^h$ may not be a feasible state in the discrete space G_h such that we are unable to correspond it with the controls. Another concern is that for the problem with higher dimensions, the average of the vectors $\sum_{i=1}^m v_i p_i$ does not give a meaningful physical state, as shown in Fig 3.3.

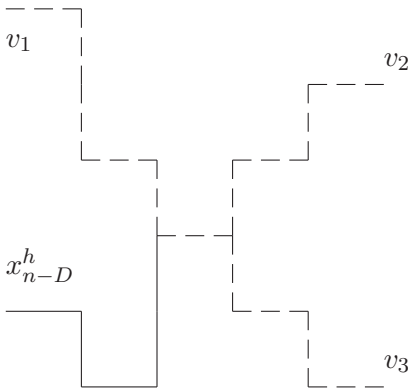

Figure 3.3: State Prediction

Therefore, we try two different methods of prediction, both of which are based on estimates $\hat{x}^h_n$ of x^h_n. (Without prediction, x^h_{n-D} is used by the controller, giving by the simple control policy, in the transition probability form). There will be an optimal control corresponding to each v_i, i.e. $u^*_c(v_i)$. A possible strategy is to apply the control $\sum_{i=1}^m u^*_c(v_i)p_i$, i.e. the average control. We call this control Policy I. Another control strategy is determined by $j = \arg\max_i p_i, i = 1, 2, ..., m$ one then selects the corresponding v_j and then applies the control $u^*_c(v_j)$. That is, take the possible outcome with the highest probability as the estimation of the present state, and apply the optimal control corresponding to it. We call this Policy II. The control strategy using policy I or II is denoted by u^*_p. The simple control policy which takes the delayed information as the present system state and applies the form 3.6 is denoted by u^*_s.

Although the prediction methods are heuristic, a justification of using these prediction and control strategies is as follows. Let $W^h(\bar{x}^h_0, u^*_p)$ denote the cost in either Policy I or Policy II. Then for every $h > 0$,

$$\begin{aligned} V^h(\bar{x}^h_0) &\triangleq \inf_\alpha W^h(\bar{x}^h_0, \alpha) \le W^h(\bar{x}^h_0, u^*_p) \\ &\le W^h(\bar{x}^h_0, u^*_s) \triangleq V^h_s(\bar{x}^h_0), \end{aligned}$$

where V^h_s is the running cost using the simple control policy.

The first inequality follows from the fact that u^*_c is a minimizing control with complete information. That it holds for each $h > 0$ follows from $V^h(\bar{x}^h_0) \to V(\bar{x}_0)$, established in Theorem 4.1 of [45]. The second inequality is comparing to the delay (i.e., partial) state

information case. It follows from the fact that $W^h(\bar{x}_0^h, u_p^*)$ uses a close prediction of the current state but in $W^h(\bar{x}_0^h, u_s^*)$ only delayed state information is used. That it holds for each $h > 0$ from $V_s^h(\bar{x}_0^h) \to V_s(\bar{x}_0)$, also established by Theorem 4.1 of [45].

In our simulations in Section 3.3.5, we illustrated the benefits of prediction over the delay state information case and suboptimality compared to the complete state information case.

3.3 Application to the Power Allocation Problem in Wireless Communication Queueing Systems

In this section, we will study the application of controlled Markov Chain approximation on wireless communication queueing systems with time delay. High-speed wireless networks with large capacity requirements are becoming prevalent and system delays need to be considered for good performance. We analyze a power control problem for a forward-link wireless queueing model where only delayed queue state information is available to the controller. The queue-size model is a controlled diffusion with delay obtained from a modification of an established heavy traffic limit model with no delay. We consider two approaches for solving the associated stochastic (power) control problems. In the first approach the problems are solved using the standard controlled Markov Chain approximation developed by Kushner [47], where the delayed information is used directly in the transition probability form. In the second approach, motivated by separation principle ideas, we add a prediction component, using an a priori control policy obtained by the standard approach assuming the system has no time delay. A simulation study shows good results: in particular, convergence of the cost function are observed in both approaches and the addition of the prediction component leads to some cost reduction. This motivates further investigation of prediction techniques in the numerical methods for the stochastic control problems.

3.3.1 Background of the Wireless Communication Queueing Model

High-speed wireless systems with large capacity requirements (e.g. for multimedia) are becoming a prevalent reality. We investigate a power control problem for a simplified forward-link wireless model with a finite set of queues at the base station and model delays in obtaining queue-state information for use in a feedback controller. Our approach is to

obtain an approximating model for the queue dynamics that incorporate the delay and use numerical methods for solving the associated stochastic control problem using the model. The queue model is obtained from a heavy traffic limit model for a system with no delay modeled (the no-delay system). The heavy traffic method scales time and the state space where the scaling is such that in the heavy traffic limit model inessential detail is averaged out. For the no-delay system, the limit model for the queue dynamics is a controlled diffusion, also called a stochastic differential equation with reflection (SDER). Obtaining a queue model incorporating the delay in queue state information utilizes the SDER limit model obtained for the no-delay system and is given by a delayed controlled diffusion, also called a stochastic differential delay equation with reflection (SDDER).

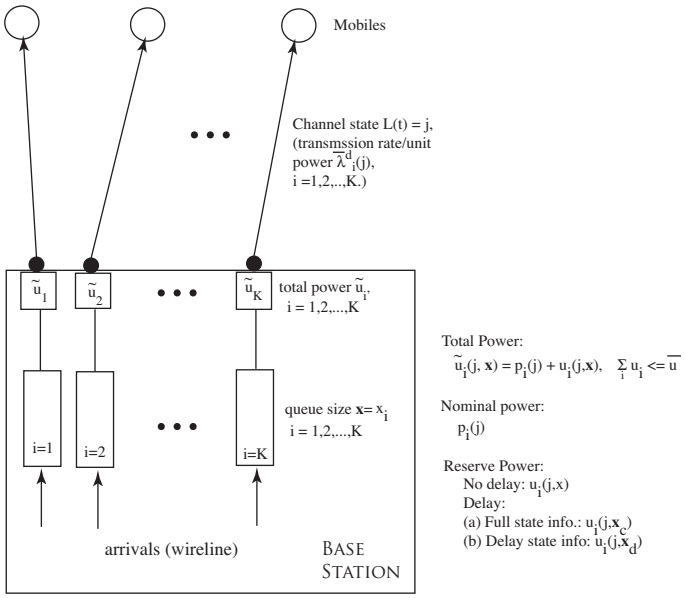

Figure 3.4: Forward Link Wireless System

The physical model considered is a one-cell, forward-link system shown in Figure 3.3.1, with finite-dimension K. Data arrivals are via wireline and enter a queue dedicated to a particular mobile. Power $\tilde{u}$ is applied at the base station antennas for transmissions of data to the mobiles through the random wireless medium modeled by the (vector) channel process $L()$. At each time t, the channel process $L(t)$ is in a state indexed by $j \in J$, $|J|$ finite (i.e., $L(t) = j$); each j specifies the transmission rate per unit power $\bar{\lambda}^d_i(j)$ for each queue $i = 1, 2, ..., K$. Typically (and used here) the channel process is modeled by a Markov or semi-Markov process with stationary distribution given by $\pi(j)$.

The cumulative arrival process for queue i, $A_i(\cdot)$, are via wireline to the base station and we consider both a batch arrival process, following [13], and an infinite source Poisson model as in [14]. The number of packets in the batch arrivals is assumed to have finite mean and second moment. The second model is an infinite source Poisson model which is a standard choice to model long range dependence (LRD) and heavy tailed (HT) characteristics often observed in high speed wireline systems transmitting large data files [59]. In our wireless model aggregate arrivals to the base station have LRD/HT characteristics and these characteristics are assumed to be retained at the individual queues after a flow control mechanism is applied to the aggregate arrivals. Details of the arrival rates can be found in [16].

The services (departures) are accomplished by applying power $\tilde{u}$ to the antennas at the base station. The power is divided into two parts: for queue i, channel j, and canonical queue state χ,

$$\tilde{u}_i(j,\chi) = \bar{p}_i(j) + u_i(j,\chi)$$

where most of the power is preallocated (nominal power p) to the queues for balancing the mean arrival rates, and the reserve power u is the power left over for handling stochastic variations from the mean process and is subject to control. We will interchangeably call the $u_i(,), i = 1, 2, ..., K$ the controller or reserve power.

In the most general form, the reserve power depends on the channel state j and queue state χ. For the no time delay case, we have $\chi(t) = \mathbf{x}(t)$, and for the fixed delay case, $\chi(t) = \mathbf{x}(t-\tau)$, where $\mathbf{x}(\cdot)$ denotes a vector of states. In all cases the reserve power is constrained to be less than the amount available at the base station.

The departure rate depends on the power applied and the channel state; the rate for queue i has the form

$$\bar{\lambda}_i^d(j)(\bar{p}_i(j) + u_i(j,\chi)). \tag{3.10}$$

The variations in the departure rate clearly depend on the variations in the channel process $L(t) = j$. As there can be no services when the queue is empty, the cumulative departure process is given by

$$D_i(t) = \int_0^t \sum_{j \in J} I_{\{L(s)=j\}} \bar{\lambda}_i^d(j)(\bar{p}_i(j) + u_i(j,\chi(s))) I_{\{x_i(s)>0\}} ds,$$

where $I_{\cdot}$ is the indicator function and $I_{x_i(s)>0}$ constrains the queues from being negative.

Applying the heavy traffic limit model to the arrivals and departures [13][14][16] and assuming there is no time delay, one can apply weak convergence analysis and obtain

$$x_i(t) = x_i(0) + \int_0^t b_i(u, \mathbf{x}(s))ds + \sigma_i w_i(t) + z_i(t), i = 1, 2, \ldots, K, \tag{3.11}$$

modeling the queue dynamics where the drift term is given by

$$b_i(u, \mathbf{x}(s)) = -\sum_j u_i(j, \mathbf{x}(s))\bar{\lambda}_i^d(j)\pi(j), \tag{3.12}$$

and w_i is the standard Wiener process, σ_i the volatility, and z_i is a reflection term increasing when $x_i(t) = 0$ keeping the queue size nonnegative. We see the Brownian motion in the limit model is due to the stochastic variations of the channel process. Mild weak dependence assumptions are imposed on the channel process and the perturbed test function technique is applied in the weak convergence analysis, establishing the Wiener process in the limit (details of the method applied to our wireless model is in [14]).

The channels are assumed to be mutually independent so that the covariance of the Brownian motion is diagonal with entries defined by [13]

$$\begin{aligned}\sigma_i^2 &= Ew_i(1)w_i(1) \\ &= 2E\left[\sum_j I_{\{L(0)=j\}}\bar{\lambda}_i^d(j)\bar{p}_i(j) - \Lambda_i^a\right] \times \int_0^\infty \left[\sum_j I_{\{L(s)=j\}}\bar{\lambda}_i^d(j)\bar{p}_i(j) - \Lambda_i^a\right] ds,\end{aligned} \tag{3.13}$$

where Λ_i^a is the average arrival rate for queue i, and the expectation is taken with respect to the stationary distribution π for the channel. Extending to higher dimensional reflected Brownian motion case, our case here, can readily be done using the perturbed test function method.

3.3.2 Introduction of Time Delay and the Stochastic Control Problem-Power Allocation

We are more interested in the queueing system with time delay, which is a common issue in wireless communications due to the mechanical system design. The delay is usually caused by the state observer, which only makes the lagged state information available at present time. So by modifying the drift term in the queue model (3.11), and assume fixed

time delay τ for simplicity, we end up with the following model

$$x_i(t) = x_i(0) + \int_0^t b_i(u, \chi(s))ds + \sigma_i w_i(t) + z_i(t), i = 1, 2, \ldots, K, \tag{3.14}$$

with initial data, and $\chi(s) = \mathbf{x}(s - \tau)$. The drift term analogous to (3.12) is

$$b_i(u, \mathbf{x}(s)) = -\sum_j u_i(j, \chi(s))\bar{\lambda}_i^d(j)\pi(j), \chi(s) = \mathbf{x}(s - \tau), \tag{3.15}$$

and the reflection $z_i(t)$ is as before, keeping the queue-size nonnegative and works whenever $x_i(t) = 0$ as discussed in 3.1. This is a straightforward way to get a model incorporating the type of delays in queue state information we are interested in, but it is not without a couple of caveats. First, we assume the conditions for the strong or weak existence and uniqueness of solutions are satisfied [49]. Second, the model is not obtained by a limit of a sequence of prelimit models with a corresponding delay component; in fact, to the best of our knowledge there are no results for the heavy traffic method with direct modeling of the delays in the prelimit. Nonetheless, this does not detract from our interest in applying the numerical methods in [47] for obtaining solutions of stochastic control problems along with analyzing the affects of adding a prediction component to these methods.

With the models (3.14) and (3.15), the control problem is to allocate the reserve power of the base among the queues in order to minimize a certain running cost. We consider an infinite-horizon discounted cost function with cost rates of the general form $k(\chi)$ specialized to a weighted linear combination of powers of the queue-size, where $\chi(\cdot)$ can be $\mathbf{x}(t)$ for the no-delay case and $\mathbf{x}(t - \tau)$ for the time delay case. We notice that the cost of the real system is only about the system performance, for example how much power it consumes, how many services it provides, etc., no matter whether there is time delay in state observer. Therefore, it is reasonable to use the cost rate as

$$k(\chi(s)) = \sum_{i=1}^{K} a_i x_i(s)^{p_i}, \tag{3.16}$$

where a_i is the weight among the queues, and p_i is the power index.

The general form of the cost function is given by

$$W(\chi_0, u) = \int_0^\infty e^{-\beta t}[k(\chi(t))dt + c'dz], \tag{3.17}$$

where β is the discounting rate, and $c' = [c_u, c_o]$ are weighting cost penalty of buffer underflow or overflow, respectively. The stochastic control problem is then to find the optimal

cost

$$V(\chi) = \inf_{u \in \mathcal{U}} W(\chi_0, u), \tag{3.18}$$

and optimal policy $u^*() \in \mathcal{U}$, where $\mathcal{U}$ is the set of admissible controls, and where χ is an element of the state-space S. The numerical method [47] is illustrated in the following sections.

3.3.3 Implementation of Controlled Markov Chain Approximation

First, the simple no-delay case is studied [15] [52]. Then the delay case is considered, where the simple control policy which uses the delayed information without state estimation is first tried. After that, we detail adding a prediction component in the delay state information case where only delayed information is available to the controller at time t. Loosely stated, we obtain predictions over $[t-\tau, t]$ using the dynamics of the Markov chain under the controlled transition probability using an a priori optimal control for an analogous stochastic control problem, i.e., for the complete state information case. The results of the prediction are used to specify two different controlled transition probabilities applied to the system (i.e. Policy I and II, described in Section 3.2). An objective of adding the prediction component is to reduce the cost and this is seen in the simulations in Section 3.3.5. A more practical point of view for studying the prediction component is that it gives upper bounds on the cost for the optimal practical control. The simulation results, nonetheless, motivate further investigation of incorporating prediction methods for the numerical solution of the stochastic control problem.

Numerical Implementation: No-delay Case

When the system has no time delay, $\chi(t) = \mathbf{x}(t)$ denoting the queue size, and for the purpose of illustration, we consider a one dimensional problem. The queue is supposed to be bounded, $x(t) \in [0, B]$ for all t and denote the underlying probability space for the Markov chain $(\Omega, \mathcal{F}, P)$. The state space for the Markov chain approximation is obtained by a discretization of $[0, B]$ with parameter h: the number of points in the discrete state space G_h is $[B/h] + 1$, where h is the interpolation level. In order to incorporate the reflection, we need two more states beyond G_h to reflect from giving ∂G_h^+ , so the total number of points in the state space is $N = [B/h] + 3$. The states in the Markov chain approximation are denoted x_n^h and the interpolated process $x_n^h()$ (notice we use the same denotation for

simplicity) is formed from the piecewise constant interpolations of x_n^h over the interpolation intervals Δt_n^h. The value of the control applied at time n is given by u_n^h.

It can be shown that the local consistency discussed in Chapter 2 is held under the transition probabilities and interpolation intervals obtained by using the methods in [47], and suppose $\sigma^2 > h|b(x_n^h, u)|$, then the central-difference approximation can be used in the first-order derivatives and we obtain

$$\begin{aligned} P^h(x_n^h, x_n^h \pm h|u) &= \frac{\sigma^2 \pm hb(x_n^h, u)}{2\sigma^2}, \\ \Delta t^h &= \frac{h^2}{\sigma^2}, \end{aligned} \tag{3.19}$$

for $x_n^h \in G_h$. When outside the feasible region G_h, i.e. $x_n^h \in \partial G_h^+$, there is an instantaneous reflection back to the nearest state in G_h. Note that the time step is constant and not part of the Markov chain which lives in the queue state space. This is a case of an explicit approach to the Markov chain approximation. The discrete cost function is again

$$W^h(x,u) = E_x^u \left\{ \sum_{n=0}^{\infty} e^{-\beta t_n^h} \left[k(x_n^h, u_n^h)\Delta t_n^h + c'(x_n^h)\Delta z^h(x_n^h) \right] \right\}, \tag{3.20}$$

where $t_n^h = \sum_{i=0}^{n-1} \Delta t_n^h$ and $\Delta z^h(x_n^h)$ is the reflection term. The dynamic programming equation for the optimal cost and the determination of the optimal control is shown in Chapter 2. The solving procedure just follows the steps described in Chapter 2. The higher dimensional cases readily follow from the one-dimensional discussion above with the exception of constraints on the reflection directions in order to satisfy some technicalities for the convergence to a SDER as $h \to 0$ (see [13]). We assume the reflection is orthogonal to the boundary, then the reflection directions fall within these constraints (see Fig 2.2 in Chapter 2). Orthogonal reflection corresponds to the simple case where power is not reallocated from empty queues to non-empty queues.

Numerical Implementation: Delay Case with Simple Control Policy

Now suppose the system has time delay, and for illustration purpose, we assume it is a fixed time delay, say τ, and one dimensional problem again. Then $\chi = x(t-\tau)$. The running cost is still (3.20), but the system becomes

$$dx(t) = b(x(t-\tau), u(x(t-\tau)))dt + \sigma dw(t) + dz(t), \tag{3.21}$$

indicating the system time delay in state observation.

After discretization by choosing an interpolation level h, the delayed path is denoted by a sequence of states

$$\mathbf{x}_n^h = [x_{n-D}^h, x_{n-D+1}^h, \ldots, x_{n-1}^h, x_n^h] \in G_h,$$

approximating the path information $\bar{x}(t) = x(t+\theta), \theta \in [-\tau, 0]$, whose piecewise-constant interpolation are delay paths

$$\bar{\mathbf{x}}_n^h \in D(G_h, [-\tau, 0]),$$

where $D(G_h, [-\tau, 0])$ is the space of right-continuous functions with left-hand limits taking values in G_h and with the Skorohod J1-topology (see [9]). In this representation, it is easy to know the states of the next time step, which is

$$\mathbf{x}_{n+1}^h = [x_{n-D+1}^h, x_{n-D+2}^h, \ldots, x_{n-1}^h, x_n^h, x_{n+1}^h] \in G_h,$$

where $x_{n+1}^h = x_n^h + h$ or $x_n^h - h$, assuming all states are feasible.

We might use the representation $\mathbf{x}_n^h$ as the states in our controlled Markov Chain approximation scheme, and derive the control as a function of $\mathbf{x}_n^h$. However, notice that because of the time delay τ, the information $\mathbf{x}_n^h = [x_{n-D}^h, x_{n-D+1}^h, \ldots, x_{n-1}^h, x_n^h]$ is not observable at time step n. The only available information is

$$\mathbf{x}_{n-D}^h = [x_{n-2D}^h, x_{n-2D+1}^h, \ldots, x_{n-D-1}^h, x_{n-D}^h],$$

states form a longer time ago. The control u_n^h can be derived only through this delayed information, although once it is applied to the control system, it works on the present state x_n^h. Therefore, a practical control policy u_n^h is a function of $\mathbf{x}_{n-D}^h$. Nevertheless it requires us to keep all the information and use $[\mathbf{x}_{n-D}^h, \mathbf{x}_n^h]$ as the state representation in the controlled Markov Chain approximation. That is, the delay information of length $2D$ should be kept, where the most current D points are not available to the controller.

Computer memory requirements become an issue in applications of the method, and some techniques are required to reduce the memory. Towards obtaining an efficient representation of the state $\mathbf{x}_n^h$, we note that the difference between two consecutive states can only be h or-h, so we may use $\mathbf{x}_n^h = [d_{-D}, d_{-D+1}, ..., d_{-1}, x_n^h]$ to represent the delay path, where $d_{-i} = x_{n-i+1}^h - x_{n-i}^h = h, i = 1, 2, ..., D$. Furthermore, if we let $I_{-i} = 1$

denote the case $d_{-i} = h$, and $I_{-i} = 0$ denote the case $d_{-i} = -h$, then this is a one-to-one mapping between the integers and the delay path; we may represent the delay path as $\mathbf{x}_n^h = [I_{-D}, I_{-D+1}, ..., I_{-1}, x_n^h]$. The part $[I_{-D}, I_{-D+1}, ..., I_{-1}]$ corresponds to a unique binomial number, and together with x_n^h, we are able to use integers to represent the system states $\mathbf{x}_n^h$. So all together there will be $2^{2D} \cdot N$ discrete states, where $N = [B/h] + 3$. We note that increasing D (the interpolating points in the delay path) will rapidly increase the number of states leading to severe memory requirements.

In the controlled Markov Chain approximation, the transition probability matrix R^h takes the most memory, having $2^{4D}N^2$ elements. However, we notice that this matrix is extremely sparse, so if we only record the nonzero entries, the scale of the matrix can be reduced dramatically. Here, in Figure 3.5, a data queue technique is used in recording the sparse matrix R^h, which is used in the linear system using Gauss-Seidel method (Section 2.3). In the simulation cases in Section 3.3.5, where two queues are considered, the scale would be only $4 \cdot 2^{2D} \cdot N$.

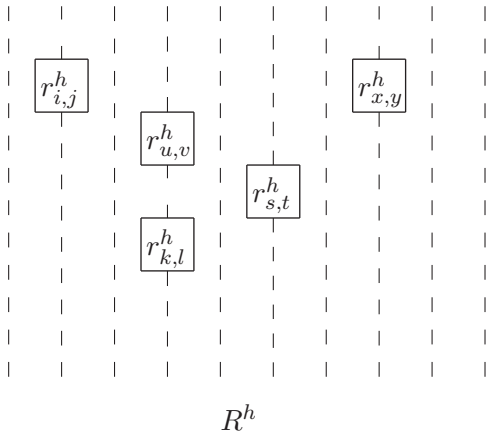

Figure 3.5: Representation of A Sparse Matrix

The issues in memory requirements are also addressed in [45] [44] [46] and, in fact, are the motivation for the implicit numerical methods in [45][46] based on a representation of systems with delay in Vinter and Kwong [83]. However, interesting future research would include simulations under the numerical models in [45][46] and their comparison to the results here.

It can be shown that the local consistency (3.4) holds under mild conditions pointed

out in [45]. Assume $\sigma^2 > h|b(\alpha, \bar{\mathbf{x}}_n^h)|$, then follow (3.6), the simple control policy uses the following transition probabilities in the controlled Markov Chain approximation

$$p^h(x_n^h, x_n^h \pm h|u) = \frac{\sigma^2 \pm hb(u, x_{n-D}^h)}{\sigma^2},$$
$$\Delta t^h = \frac{h^2}{\sigma^2}, \tag{3.22}$$

for the simple fixed time delay case, that is, it uses the delayed information directly.

When in ∂G_h^+ , the effect of the reflection $z(s)$ is to instantaneously bring the simulated state into the feasible region G_h^+. As mentioned in Section 3.1, since the states of the real system will never be infeasible, independent of whether there is time delay or not, we impose the condition that the reflecting term $z(s)$ has no delay.

Numerical Implementation: Delay Case with State Estimation

The simple control policy uses the delayed state information directly in the transition probability form. We are interested in finding the effect of state estimation, and use $b(u, \hat{x}_n^h)$ in (3.22) instead of $b(u, x_{n-D}^h)$.

In the simulation model, it is possible to assume there is no time delay and find the true optimal control policy u^* for the system. But for the real time control system, u^* is not a practical control policy because only delayed information is available. However, u^* is a priori control policy and is useful for estimating the present system state. The estimation procedure generally follows the two control policies presented in Section 3.2, namely, Policy I and Policy II. In Policy I, the possible outcome of present state x_n^h are predicted with certain probabilities based on the delayed information x_{n-D}^h and the control u^*, then an averaged control variable is applied to the system. In Policy II, the outcome with the highest probability is considered as the estimation of x_n^h, say $\hat{x}_n^h$, and the control $u^*(\hat{x}_n^h)$ is applied. The state representation $\mathbf{x}_n^h = [x_{n-D}^h, x_{n-D+1}^h, \ldots, x_{n-1}^h, x_n^h] \in G_h$ enables us to find what the next state is in the prediction procedure. That is, based on the information $x_{n-2D}^h, x_{n-2D+1}^h, \ldots, x_{n-D}^h$, we are able to predict $x_{n-D+1}^h, \ldots, x_n^h$ step by step, analogous as in Fig 3.2, and finally a sequence of prediction is determined. In the vector form, we predict $\mathbf{x}_{n-D+1}^h, \mathbf{x}_{n-D+2}^h, \ldots, \mathbf{x}_n^h$ step by step from the delayed information $\mathbf{x}_{n-D}^h$, where this vector is the state in the controlled Markov Chain approximation, and the practical control policy is a function of this state vector, as discussed in Section 3.3.3. The details of Policy I and

II are given in Section 3.2. In the simulation, the control effect of u^*, simple control policy, and Policy I and II are given and compared. It shows that the state estimation does yield a better result than the simple control policy.

3.3.4 Algorithms

The solving steps just follow the procedure in Chapter 2. After the construction of the controlled Markov Chain scheme, several cases are considered. We use the subscript (i) to denote the result at the ith iteration of the algorithm. First for the no delay case, an initial guess for the cost vector $W^h_{(0)}(\cdot)$ is made, then by applying the one step dynamic programming, the control $u^h_{(1)}$ and the transition probability matrix $R^h_{(1)}$ is determined. After that, a new cost vector $W^h_{(1)}(\cdot)$ is calculated using Gauss-Seidel algorithm to solve the linear system. This process goes on until convergence exists. In this way, the final control policy is the optimal control u^*. Another advantage to run the no-delay case first is that we can use the cost result W^h as an initial guess for later runs, because it is expected that this cost is not far away from the result of other control policies, so a close initial guess would speed up the computation dramatically.

Now if the system has time delay, and the simple control policy is used, then the delayed state information is used directly in the transition probability form in finding R^h, and other steps are analogous to the no-delay case. If the state prediction is applied, then at each delayed state $\mathbf{x}^h_{n-D}$, Policy I or II are implemented to determine the transition probabilities. And the iteration is similar to that of the no-delay case. The detailed algorithm is shown below:

Algorithm

Input simulation settings and validate.
Allocate memory for cost function W^h, transition matrix R^h and control u.
Initialize W^h.
if Using state estimation **then**
  Set initial guess of $W^h_{(0)}$ the cost obtained from optimal control if the no-delay case has been run.
  Allocate memory for the optimal control.
end if

while $W^h_{(n)}$ not converges **do**
  if No delay case **then**
    For each point $\mathbf{x}^h_n$ in the structure
    if x^h_n which is the last entry of the vector $\mathbf{x}^h_n$ is beyond the boundary **then**
      Set the transition probability $p(x^h_n \to x^h_{n+1}) = 1$, where x^h_{n+1} is a neighbor point of x^h_n and is also an interior point.
      Add the boundary penalty cost C to $W^h_{(n)}(\mathbf{x}^h_n)$. Update W^h, R^h.
    else
      Apply the reserved power (control) to minimize the cost increment.
      Compute the transition probabilities $p(\mathbf{x}^h_n \to \mathbf{x}^h_{n+1})$, update u, W^h, R^h.
    end if
  else if Delayed case **then**
    if Simple control **then**
      Take $\mathbf{x}^h_{n-D}$ as the true system state and proceed as in the no-delay case.
    else
      Using optimal control u^* obtained before to predict all possible present states.
      if Policy I **then**
        Apply the averaged control to the system, and proceed as in the no-delay case.
      else if Policy II **then**
        Apply the control $u^{*h}(\hat{\mathbf{x}}^h_n)$, where $\hat{\mathbf{x}}^h_n$ has the highest probability of outcome among all possible predictions, proceed as in the no-delay case.
      end if
    end if
  end if
  Using updated W^h, R^h, solve the system $W^h_{(n+1)} = R^h_{(n)} W^h_{(n)} + C$ for new $W^h_{(n+1)}$.
  The Gauss-Seidel iteration is used to solve this linear system.
  if $||W^h_{(n+1)} - W^h_{(n)}||/||W^h_{(n)}|| >$stop-criterion **then**
    Use $W^h_{(n+1)}$ as the new cost vector.
  else
    Break
  end if
end while
Record W^h, u, R^h.

3.3.5 Simulations

In this section, we give the simulation results, the controls and corresponding running costs of these different control policies, and compare them with the no-delay optimal control results. The convergence of the costs at each iteration for both no-delay case and delay case are also shown, which is crucial to the numerical method. The simulations give a first application of the numerical methods in [45] for obtaining optimal control policies with delays in the state, complementing applications in the no-delay case described in [52] which use the methods in [47]. By comparing the different results, we will also see that the improvement of Policy I and II over the simple control is accomplished by small differences in the control policy.

As noted in Section 3.3.3, the large memory requirements are due to the need to keep track of the state information comprising the paths and there are practical limits on the size of the simulations. In our case we had 8 gigabytes of memory and were limited to modeling small delays. In particular, we set $\tau = N \cdot \Delta t^h$ where $\Delta t^h = h^2/\sigma^2$, a constant, and N = 1 or 2, the number of interpolation steps in the delay. We model two affects when varying N. In the first case, N and σ are adjusted together for τ to remain fixed in order to study the affect of changing volatility (σ) in the system. In the second case, only N is varied in order to study the affect of delay length, in which case the delay τ varies.

The simulating system is a wireless communication system with two queues, and the control problem is to allocate optimally the reserved power u between these two queues in order to minimize the expected running cost W. It is assumed that there are two channel states for this system, and the service rate is different under these channel states. It is desirable to avoid overflow and even underflow, i.e., the queue size should stay away from its boundaries, hence the costs c_o and c_u are employed. The value for the queue size B and interpolation interval h were influenced by our available memory.

Table 3.1: Simulation Parameters

No. of Queues	2	No. of Channel State j	2
Queue Weights a_1, a_2	$0.3, 0.2$	Channel State Distrib. π_1, π_2	$0.7, 0.3$
Reserved Power u	1	Discount Rate β	0.05
$\bar{\lambda}_1^d(j), [j = 1, j = 2]$	$[0.3, 0.2]$	$\bar{\lambda}_2^d(j), [j = 1, j = 2]$	$[0.2, 0.1]$
Queue Size B	8	Interpolation Level h	0.1
Penalty on Empty Queue c_u	10	Penalty on Blocked Queue c_o	500
Cost Rates p_1, p_2	$2, 2$	Base Volatility σ^2	0.3

It is also interested to find out the performance of the numerical algorithms. So some parameter analysis of the simulation result are first investigated, say the control effect. After that, the effect of time delay, effect of different control policies will be given, especially the result of state estimation policies.

Effect of Control in the No-delay Case

The first point of interest is to know the control effect, i.e., the effect of the small amount of the reserved power. To illustrate it, we use the no-delay system. In this case, there will be 80×80 grid points, each corresponds to a physical state of the system. Suppose the system has no reserved power, i.e., $u = 0$, the following graph shows the convergent property of the simulation, and the running cost corresponding to different starting points.

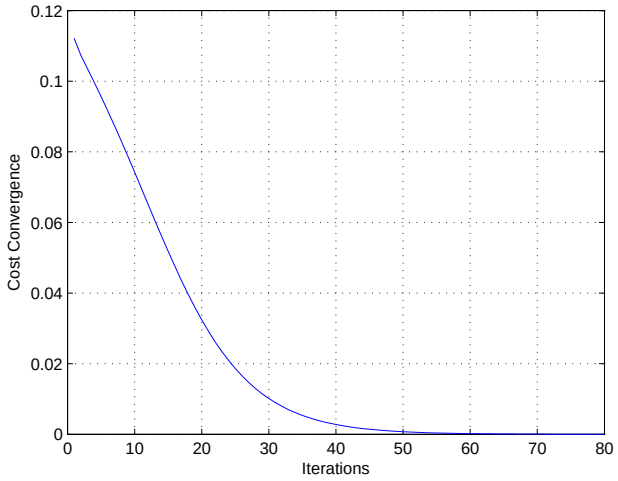

Figure 3.6: Cost convergence of the no-delay system without reserved power

In Fig 3.6, it shows the error of two consecutive $W^h_{(n)}$ of the algorithm iterations, as illustrated in Section 3.3.4. We can show that the convergence is achieved, guaranteed by the contraction mapping R^h.

Fig 3.7 shows the discounted infinite time running cost starting from each system state, i.e., the grid point, without reserved power allocation. If at time zero, the queue sizes are small, the total discounted running cost tends to be small, and vice versa.

Now to show the effect of control, i.e., the power allocation, let $u = 1$, which is a small amount of power, ignorable compared to the nominal power. The cost curve is similar

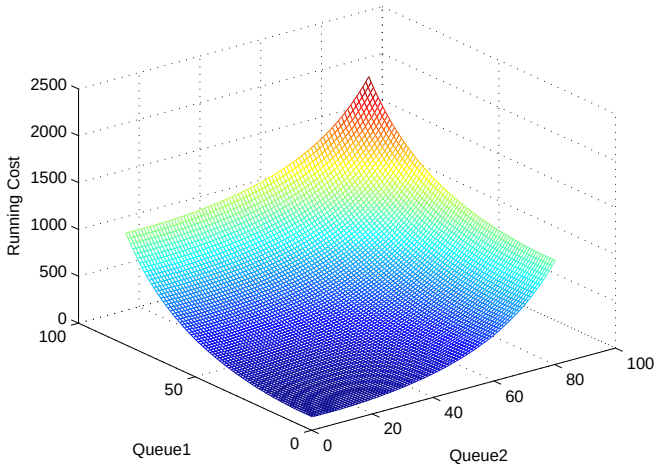

Figure 3.7: Running cost of the no-delay system without reserved power

to Fig 3.7, and we give the cost difference between the control and no-control case, namely, the cost reduced by the control.

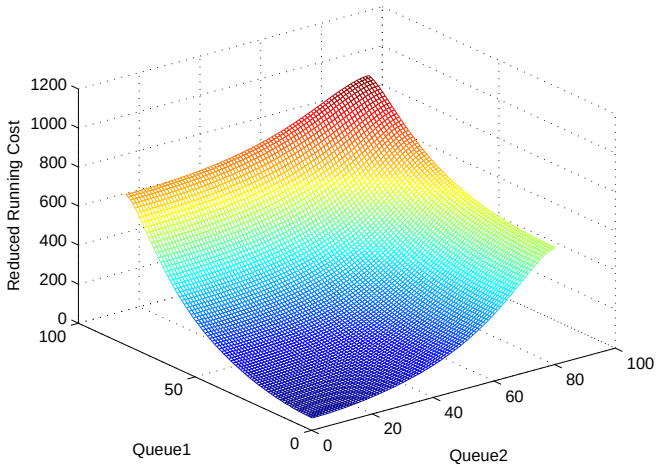

Figure 3.8: Running cost difference of the no-delay system with and without control

From Fig 3.8, we can see that a large amount of cost is reduced. The reserved power is a small amount of power that is used to reduce the queue size. The purpose of control is to allocate this reserved power between these two queues optimally in order to minimize the expected running cost. Although only a small amount of the total power is

subject to control, it has a significant effect on the queue levels [13].

The cost convergence curve for the system with control is similar to Fig 3.6, and we will show some of them wherever we think necessary.

The power allocation policy is shown in Fig 3.9 and 3.10, which are the controls under channel state 1 and 2.

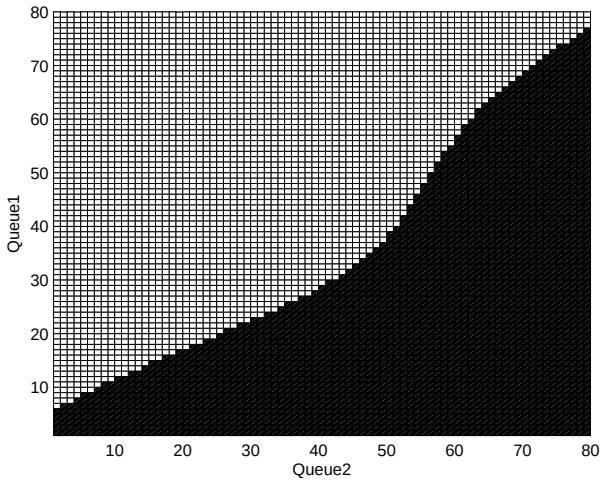

Figure 3.9: Power allocation in Channel state 1-no time delay

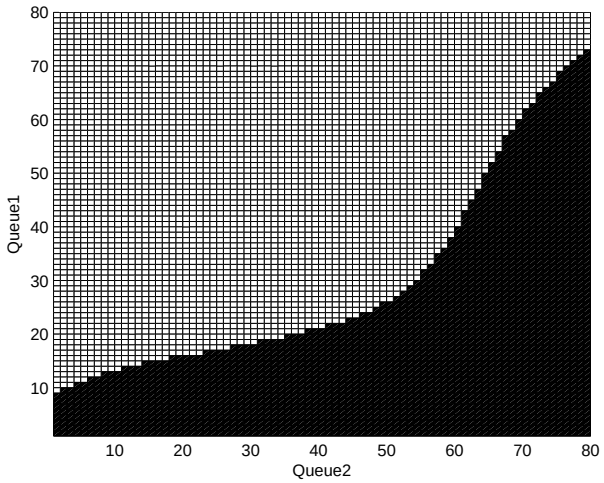

Figure 3.10: Power allocation in Channel state 2-no time delay

The power allocated to queue 2 under these channel states are the complements

of Fig 3.9 and Fig 3.10 respectively. Notice that the queue weights $a_1 > a_2$, so more power is assigned to queue 1 instead of queue 2 in order to reduce the total running cost.

One Step of Delay $D = 1$

In this section we investigate the system with the smallest delay $\tau = \Delta t^h$. Then the path representation $\bar{x}_n^h$ is formed from the interpolation of two states. If delay exists, i.e., suppose there are D steps of delay for each queue state, then corresponding to each grid point in the 8080 matrix, there will be 2^{2D} different states representing different delay paths. Because these 2^{2D} states play equivalent roles for the purposes of illustrating the costs, we arbitrarily pick one of them, namely, the first one of these 2^{2D} choices to represent the results.

There are three control strategies. The first one is the control based on delayed information, or the simple control policy. The second is the optimal control, which is not realizable in practice. The last one is the control using state estimation. Here we have two ways of propagation, namely Policy I and II which were discussed before. We first talk about strategy I, which applies the averaged control of its predictions. The comparison of strategy I and II will be discussed in details later.

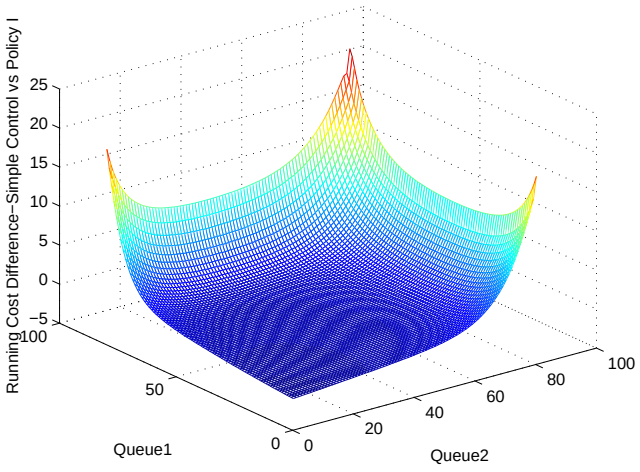

Figure 3.11: Running Cost Difference between Simple Control and Policy I($D = 1$)

Fig 3.11 shows the cost difference between simple control policy and Policy I. It showed that the simple control yields a higher cost than Policy I, which is an evidence of

the efficiency of state estimation.

The performance of the optimal control, which has the complete system information, is shown in Fig 3.12, where we can see that it costs even less than Policy I.

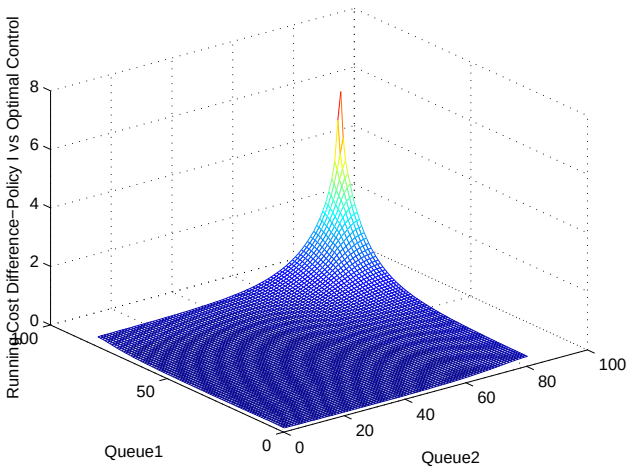

Figure 3.12: Running Cost Difference between Policy I and the Optimal Control($D = 1$)

Recall that the optimal control is not applicable. The simple control policy derives control variable only from the delayed state. The cost is higher than the optimal one, which agrees with our expectation. Policy I applies the state estimation, where the control is an average of its prediction values. The cost is between the optimal one and that of the simple control.

In Figure 3.13 the convergence behavior of the simulation for obtaining the costs of these three control methods is illustrated, as one iterates in the numerical method. Simple control and the optimal control have almost the same speed of convergence. For Policy I, since it starts iteration with the optimal control obtained before, so at first it converges faster than the other two. An important point is that Policy I, with its added complication from the prediction component using averaged control, shows similar convergence properties as in the Optimal Control and Simple Control cases.

The difference between the simple control and the optimal control under channel state $j = 1$ is shown in Fig 3.14. The results for the controls under channel state 2 are quite similar, which are not shown here.

In Fig 3.14, to the right of the curve, the control value is one $u = 1$ for both

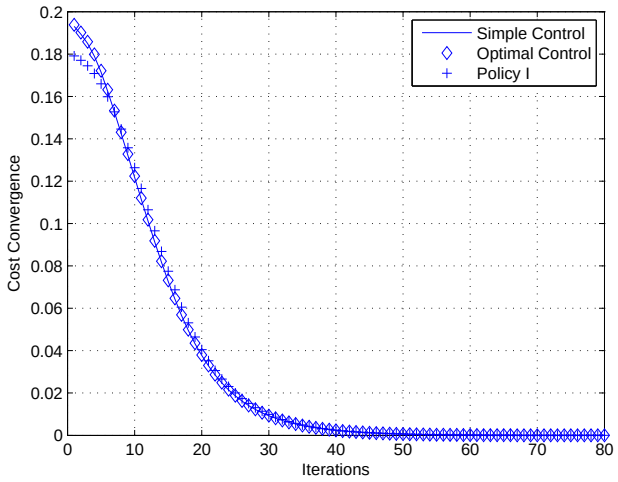

Figure 3.13: Cost Convergence of Simple Control, Optimal Control and Policy I($D = 1$)

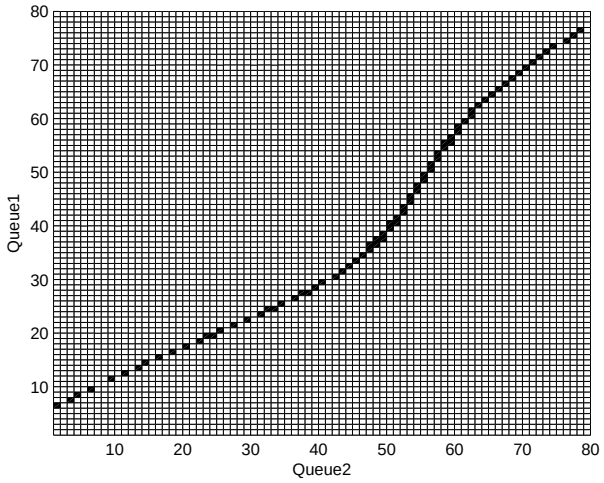

Figure 3.14: Control difference between Simple Control and Optimal Control($D = 1$)

the simple control and the optimal control and it is applied to queue 2; to the left of the curve no power is assigned to queue 2 in either case (in other words, all power is applied to queue 1). The differences lie along the curve and have magnitude 1. This small region of difference, which we will call the switching region, leads to the cost relations between the Simple Control and the Optimal Control which can be deduced from Figures 3.11 and 3.12.

The control difference between the Optimal Control and Policy I under channel

state $j = 1$ is shown in Fig 3.15.

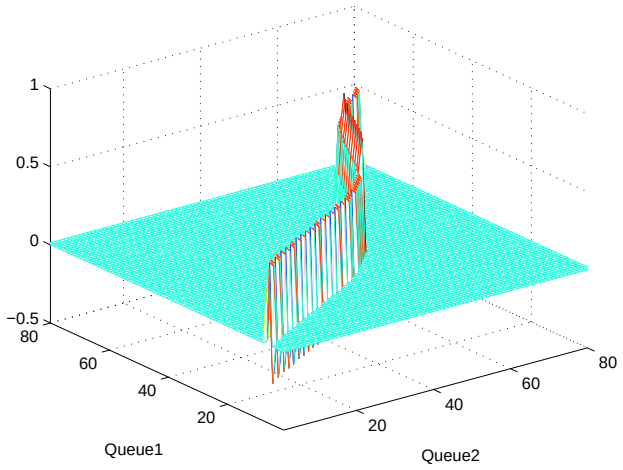

Figure 3.15: Control difference between Optimal Control and Policy I($D = 1$)

Again, we see the difference gives a small switching region, where now magnitude can be less than one since the average control is used in Policy I. As before, in fact for all control results in our simulations, the differences in the control policy for channel state $j = 2$ shows similar results. Even further, there is a similar switching region (not shown) and cost characteristics when comparing Policy II to the simple control and Optimal control policy, although, of course, the Optimal Control policy gives the lowest cost, and the simple control yields the highest.

Two Steps of Delay $D = 2$

In this section we increase the number of interpolations in the delay path to $D = 2$. While keeping the interpolation level h fixed, there are two ways to increase D. From the equation $\Delta t^h = \frac{h^2}{\sigma^2}$, we may increase σ^2 and keep τ unchanged, so that the number of discrete points in the delay path D would increase. Another method is that we increase the length of the system time delay, τ. If all others are unchanged but τ is doubled, then D is doubled. Although both cases correspond to different physical characteristics, they give similar simulation results.

Increased volatility σ To make $D = 2$, the volatility σ is increased by a factor $\sqrt{2}$ ($\sigma = 0.3\sqrt{2}$). In this way, the volatility of the system changed, which means the state varies 'faster', and it requires finer grid to simulate this system, hence larger D. This gives another motivation for studying the affects of increasing the variance. Since the simulation results under channel state $j = 1$ and $j = 2$ are analogous, we only give the result under channel state $j = 1$.

The running cost of the optimal control is shown in Fig 3.16,

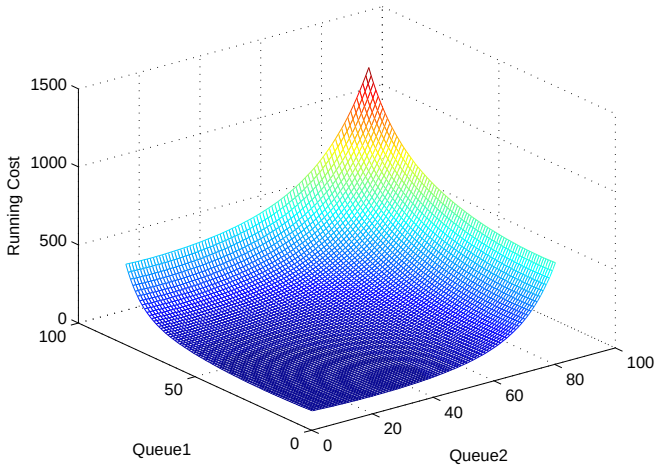

Figure 3.16: Running Cost of the Optimal Control (Increased volatility, $D = 2$)

This is the control with complete system information. Although it is not practical in real time control, it gives the lower bound on the cost. Again, we first consider the control policy I, which uses the averaged control. The cost increment of this policy over the optimal control case is given in Fig 3.17,

As is expected, the state estimation control policy can not outperform the control with full system information case. Note the larger cost difference than that of the one step delay case. However, this control does give lower cost than the simple control policy, which uses the delayed information directly in the transition probability formula in the controlled Markov Chain approximation, and this can be seen in Fig 3.18.

For the control Policy II, which takes the estimation with the highest probability of outcome as the present state, we also notice the same phenomenon that Policy II gives a cost that is higher than the full information but lower than the simple control case. In

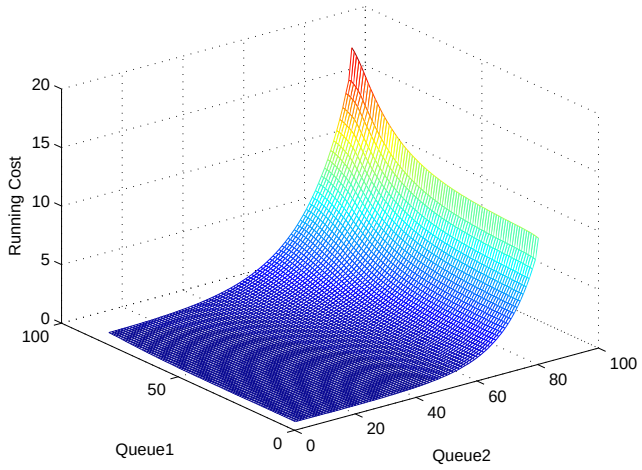

Figure 3.17: Difference of Running Costs between Policy I and the Optimal Control (Increased volatility, $D = 2$)

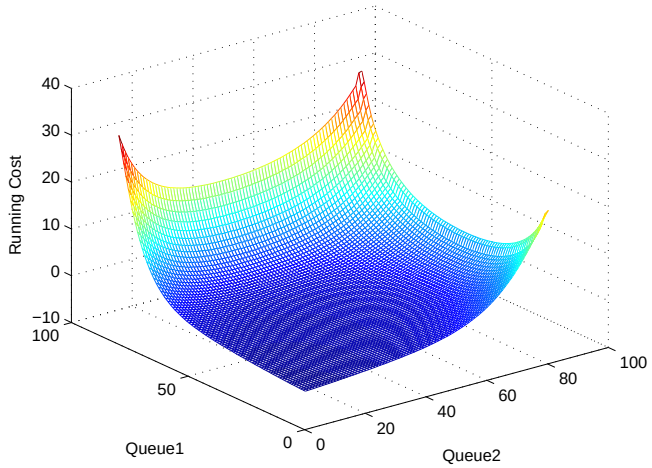

Figure 3.18: Difference of Running Costs between the Simple Control and Policy I(Increased volatility, $D = 2$)

order to make the topic complete but not lengthy, we give the performance of Policy II in the two steps of delay in the case of increased τ.

As in Figure 3.13 for $D = 1$, Figure 3.19 shows the convergence of the costs of simple control, control with complete information and Policy II. We note that the jumps in the figure are due to stopping and restarting of the simulation to accommodate memory

constraints. At these times, the code clears the history of each iteration, as well as the transition probability matrix in order to save memory, but it records the newest cost matrix. The purpose is to find a better starting point for the iterations and clear redundant memory so that we may do more iterations with limited computer memory. If successive iterations yield insignificant improvement, then the iteration terminates.

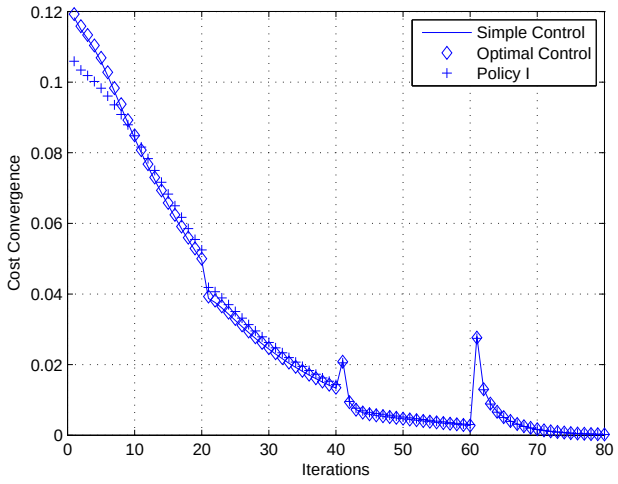

Figure 3.19: Running Cost Convergence(Increased volatility, $D = 2$)

The control policies have similar characteristics as in the $D = 1$ case: Figure 3.20 is analogous to Figure 3.14, where again the control difference is along the switching region and has magnitude 1. However, note the thicker switching region in Figure 3.20 indicates greater occurrences of control differences. Figure 3.21 is analogous to Figure 3.15 showing the difference in the policies between optimal control and Policy I.

Increased Time Delay τ In this Section $D = 2$ again but with volatility σ unchanged (hence Δt^h unchanged) from Section 3.1 so that τ is doubled. The control Policy II, which uses the estimation with the highest probability as the present state, is investigated here since not much result about this method were given before.

The convergence behavior of the simulation method for the various strategies is good (see Figure 3.22) and is similar to the $D = 2$ case with increased volatility (see Figure 3.19).

The running cost difference of Policy II over the control with complete information

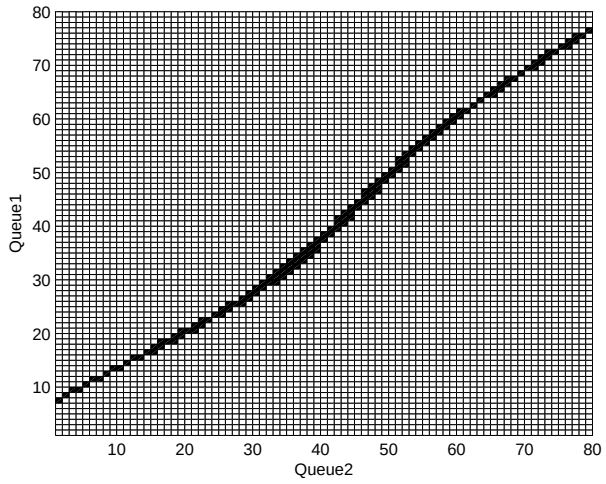

Figure 3.20: Difference of control between the Simple Control and Optimal Control(Increased volatility, $D = 2$)

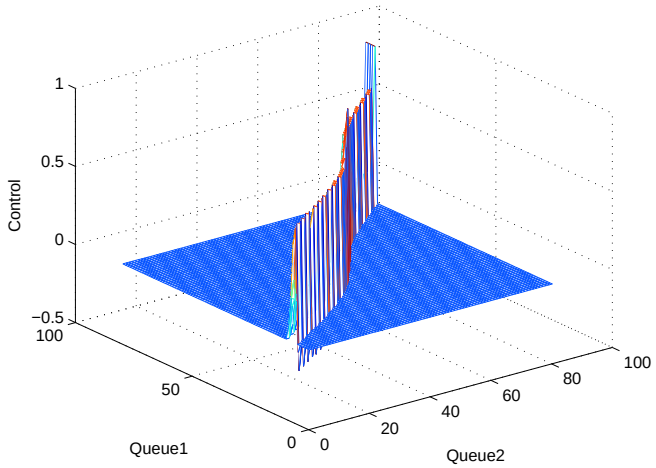

Figure 3.21: Difference of control between the Optimal Control and Policy I(Increased volatility, $D = 2$)

case is shown in Fig 3.23, and the cost difference of Simple control over Policy II is shown in Fig 3.24. Again, we observe the fact that the state estimation policies yield a lower cost than the Simple control, but no better than the optimal control which has access to full system information. Similar conclusion is deduced for control Policy I, as in the increased volatility case.

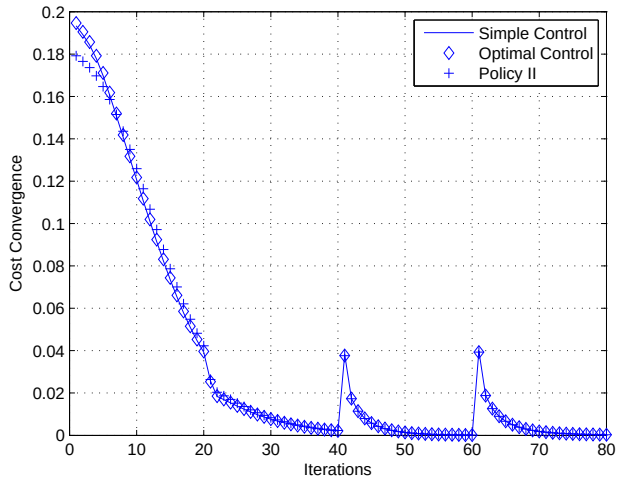

Figure 3.22: Running Cost Convergence(Increased time delay, $D = 2$)

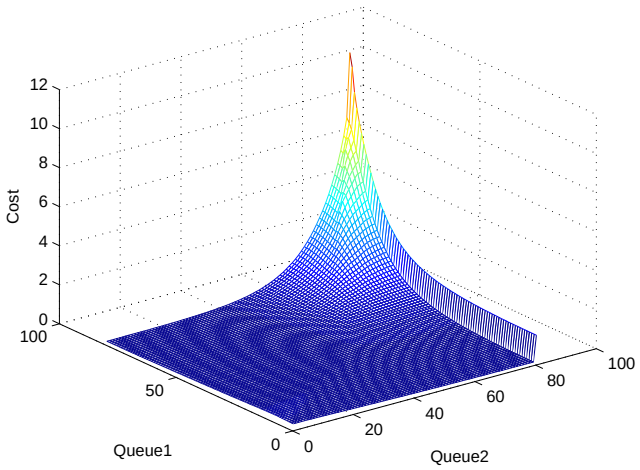

Figure 3.23: Difference of Running Costs between Policy II and the Optimal Control (Increased time delay, $D = 2$)

For the difference of the Optimal Control case and Simple control, we also have similar trends for both $D = 2$ cases (increased volatility and increased delay). In Figure 3.25 we see similar characteristics in the difference in control policies between Simple control and the Optimal Control for the increased delay case as in the increased volatility case. And Figure 3.26 is the control difference between Policy II and Optimal control. Since in Policy II, it does not use the averaged control, instead it applies $u^*(\hat{x}_n^h)$ where $\hat{x}_n^h$ is chosen by the

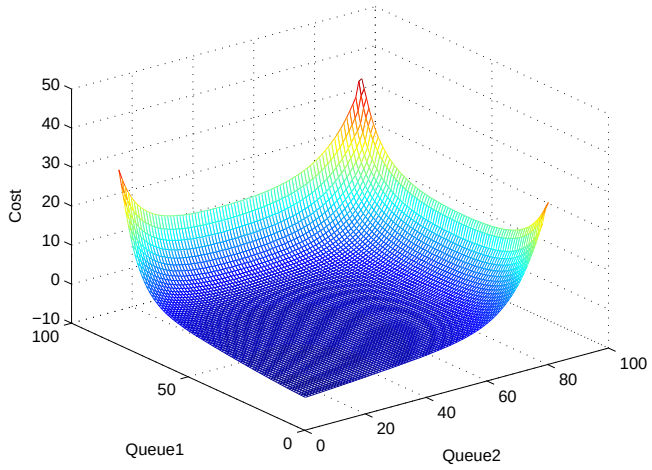

Figure 3.24: Difference of Running Costs between Simple Control and Policy II (Increased time delay, $D = 2$)

estimation with the highest probability of outcome, so the magnitude is 0 or 1.

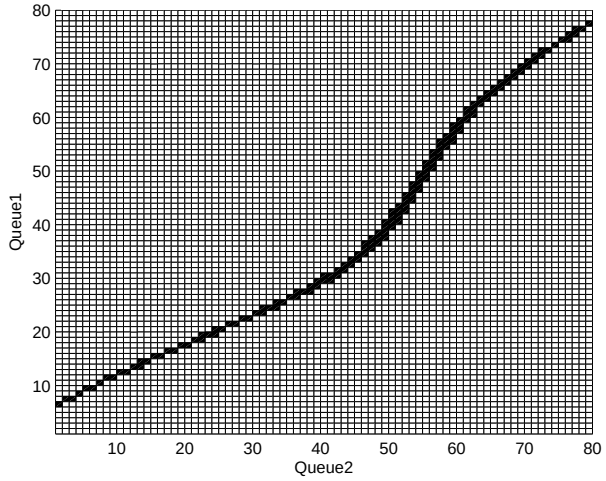

Figure 3.25: Difference of control between the Simple Control and Optimal Control(Increased time delay, $D = 2$)

Comment on Policy I and II in the increased volatility and increased delay cases If consider the grid points in the state matrix with equal weights, we can compute the average running costs for the no delay, one step of delay and two steps of delay case,

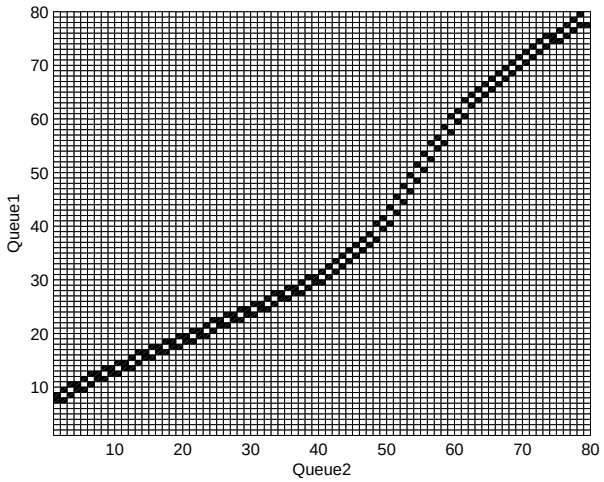

Figure 3.26: Difference of control between Policy II and the Optimal Control(Increased time delay, $D = 2$)

both for the increased volatility and increased time delay scenario. Table 3.3.5 gives the results. Recall that for different number of delay steps D, the state representations are different, in other words, the states from the one and two steps of delay case may not represent the same system state because of the delay path information, even though they have the same queue size. Therefore, the comparison between different delay steps does not make much sense. However, for the same scenario, we can compare the control results of different control policies.

Notice the $D = 2$ case with increased volatility (σ), and recall the formula for

$$p^h(x_n^h, x_n^h \pm h|u) = \frac{\sigma^2 \pm hb(u, x_n^h)}{2\sigma^2}.$$

If σ increases, the transition probability $p^h \to \frac{1}{2}$, which diminishes the control effect $b(\cdot)$. In other words, the control plays less role in this situation. So for all the control policies, the running costs are higher than other cases.

Table 3.2: Average (over initial state) cost

Delay Steps	Simple Control	Optimal Control	Policy I	Policy II
$D = 1$	202.64	200.45	200.71	200.61
$D = 2$, increased σ	270.20	264.41	266.81	265.40
$D = 2$, increased τ	203.87	199.95	201.47	200.41

Figure 3.27 shows the percentages of cost reduction of Optimal Control, Policy I

and II over the Simple control method along with the increase of delay steps, where the two steps of delay is due to increased volatility. Figure 3.28 gives the similar result, where the two steps of delay is due to increased time delay.

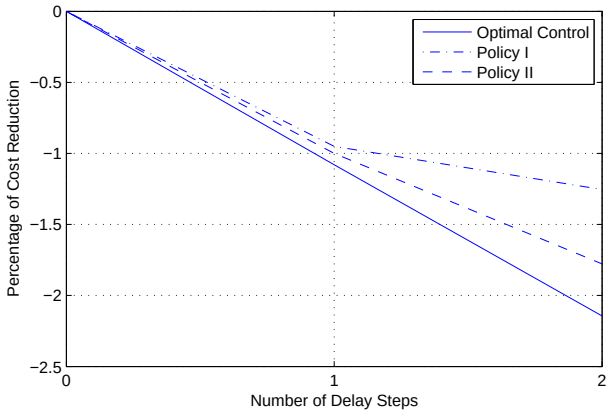

Figure 3.27: Percentage of Cost Reductions over Simple Control (increased σ)

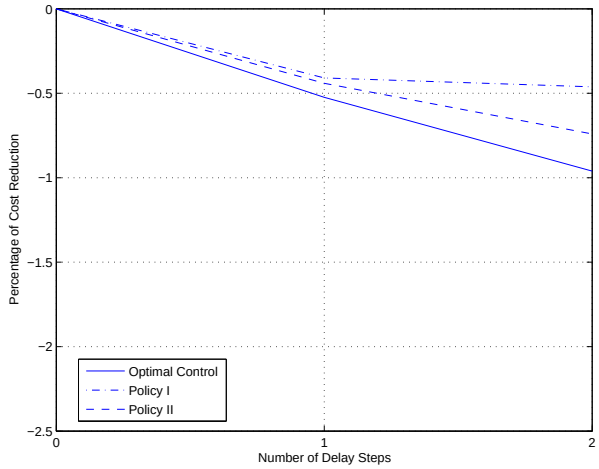

Figure 3.28: Percentage of Cost Reductions over Simple Control (increased τ)

We have considered two state estimation methods, which are Policy I and II. When the length of delay is short, so that the number of interpolation points in the delay path is small, Policy II performs better than Policy I, since the highest probability among the predictions can be large. This means the short prediction should be a good estimation

of the true system state. As the number of interpolation points D increases, the possible choices of the predictions will be 2^{DN}, where N is the dimension of the problem, or number of queues specifically.

There are two factors that can affect D. One is the length of delay τ. If τ increases while other parameters unchanged, D increases linearly with τ. But in this case, the number of predictions, which is 2^{DN}, is much higher. In other words, the average of the controls is more stable and is hard to get close to the optimal control. It happens that Policy II shows a better control result than Policy I as D increases in this case. Interested reader should refer to [15].

In high-speed systems with fast arrivals rates and departures rates, compared to slower systems, we expect the volatility of the system to increase, and consequently the interpolation time interval Δt^h decreases. Then even the length of delay τ keeps the same, the number of interpolation points D also becomes bigger. However, in high speed system with larger $\bar{\lambda}^d$ note the control effect through the drift term ($b_i(u,\chi) = -\sum_j u_i(j,\chi)\bar{\lambda}_i^d(j)\pi(j)$) would be stronger and also note the fact that this term has a large effect on the transition probabilities. Therefore, although the number of predictions 2^{DN} increases, the highest probability among these choices can still remain large. So Policy II still turns out to yield good result. If both τ and σ change, the comparison of Policy I and II will be complicated.

3.3.6 Discussion of the simulation results: control policy and cost relations

The Optimal Control, Simple control, and Policy II control policies have the characteristics that given a channel state j, at each element of the queue state space the control takes the lower bound 0 for one queue and the upper bound 1 for the other queue. Furthermore, the cost rate weighting the relative queue sizes divides the state space. In particular, for the states on one side of a dividing the curve in the state space, the control value takes its upper bound for one queue and a lower bound for the other queue; on the other side of the switching curve, the control values are reversed on the queues.

In the following we consider the differences between the Optimal Control policies and Simple Control and Policy II (Policy I is briefly considered at the end of the section). In Section 3.3.5 we saw the control differences form a switching region. The differences of the control policies in the switching region explain, of course, the corresponding differences

of the costs.

Looking more carefully at these differences, first consider Figure 3.14 which is for a system with one step of delay. For the states next to the switching region, the Simple Control does not agree with the Optimal Control since the Simple Control is derived based on delayed state information, but the Optimal Control which is based on the actual system state, switches from one bound to the other. We have a similar switching region for $N = 2$, but we note that in general with a wider width (see Figures 3.25 and 3.20).

Now consider Policy II; more precisely the difference with the Optimal Control case. For the states next to the switching region, due to the small delay, there is a significant probability that the propagation method used in Policy II predicts the actual system state. That means, there is a significant probability that Policy II agrees with the Optimal Control for some states close to the switching curvesee Fig 3.26, the switching region is narrow. For the states along the switching region where the queue sizes are large, where the right control policy is more important to the system for cost reduction, Policy II does a better job than Simple Control in getting the optimal control (compare Figures 3.25 and 3.26 for channel state $j = 1$). This gives an explanation of the lower cost of Policy II than Simple Control.

Figure 3.27 3.28 and Table 3.3.5 show how these control policies perform in reducing the cost. The cost reductions (averaged over the initial states) are relative to Simple Control and shows the Optimal Control giving the best results (of course) with Policy II second best, up to $D = 2$. The improvement is small but this is a consequence of the small delays considered.

Policy I applies the averaged control to the system. As the number of delay steps increases, the number of possible outcomes of the prediction increases too. By averaging the controls associated with these predictions, the mean is more and more unsensitive to the change of delay steps. Then it is more unlikely that the mean is close to the optimal control. This explains the performance of Policy I in Fig 3.27 and 3.28, where for one step of delay, it still yields a similar cost reduction as Policy II, but as the number of delay steps increases, its performance can not match Policy II.

In summary, we have found propagation or state estimation control policy that is better than the control under delayed information for our control system. This propagation policy utilizes the optimal control under full state information, which can be obtained through simulation but not applicable in real time control system due to the time delay. Now we can claim that the propagation policy gives a new upper bound for the averaged

running cost of the stochastic control system with time delay.

3.4 Future Topics

We have shown that by estimating the present system state, we found better control methods than the simple control which uses the delayed information directly in the approximation. We estimate the state by two approaches, one is to pick the prediction with the highest probability of outcome, the other applies the average, and both methods are based on the assumption that the system is controlled under the optimal control u^*, which has full system information. Several issues remain. One is that, is the assumption that the system is controlled under u^* a proper assumption? This assumption is motivated from the separation principle used in deterministic control system, yet whether it is proper to apply in stochastic control systems with time delay still remains an interesting topic.

The control effect of Policy I and II along with the number of delay steps is shown up to $D = 2$. How do these controls perform as the number of delay increases? Do they still have large cost reductions or the reductions diminish as D increases? All such kind of problems are hard and need further study, especially the problem of dealing with computer memories in order to handle large delays.

The performance of Policy I and II motivated us to use the same state estimation methods, especially Policy II, in later Chapters. However, the optimality of these state estimating methods is not guaranteed yet. What state estimating method yields the optimal policy for the controlled Markov Chain approximation and how to show the optimality should be the followed topics in the near future. In Chapter 4 we found it is convenient to have access to the control history in the delay path in order to have accurate present state estimation, yet the optimal control policy is still in search.

We used the controlled Markov Chain approximation to solve stochastic control problems with time delay in continuous time. As the interpolation interval $h \to 0$, the discrete results converge weakly to the continuous time solutions. However, this conclusion holds for the system with no-delay or the delayed system using the simple control [46]. We claim that the state estimation methods, Policy I and II, do not alter the structure of the theoretical proof, as in Chapter 2, so we expect them to converge to certain continuous time control policies as well. However, it does need rigorous proof.

We have only considered the stochastic control problem with fixed time delay.

The extended problem with general time delay is still a hard problem. We made the first try applying the numerical method solving such kind of problems in this area, yet future research should focus on finding the solutions of stochastic control problem with general time delay, and finding the analytical form inspired by the numerical results.

Chapter 4

Numerical Solutions with Delayed Control Information

4.1 Introduction

In the previous chapters, we discussed the nonlinear stochastic control problems with time delay, where a short path of the system state is taken as the representation of the numerical state. The system is simulated to run as if there were no time delay, and the optimal control policy for the no-delay system is derived. Then for the realtime control system, applying the separation principle for the deterministic system control problem with delay, the present system state is estimated based on the delayed information and the assumption that the system is controlled under the optimal control policy for the no-delay system. Once the present state is estimated, the optimal control variable associated with this estimation is applied to the system. The simulation results showed that this control policy gives a better result than the Simple control policy where the time delay was ignored, i.e., using the delayed state information directly in the transition probability.

The control policies we derived, Policy I and II in Chapter 3, give an upper bound on the system running cost. However, it is hard to say this is the optimal control for the system with time delay, which we will call delayed optimal control. One reason is that, the real system behaves according to the control variable which has been actually applied. That is, the system with time delay does not behave exactly under the optimal control for no-delay system (u^* in Chapter 3). Therefore, to estimate the present system state

assuming the system is controlled under optimal control for no-delay system, an estimation bias is expected. We have shown that, for our control system, it is better to estimate the present state than not doing so, and because the system runs according to the control actually applied, we present an idea to record a short path of the control history. That means, we should keep in mind what we have done to the system. From the delayed state information which is only available in the realtime control, and knowing all the controls we have already applied, we expect to estimate the present system state more accurately. If this is a deterministic discrete time control system with delay, then knowing the delayed state information and the control variables already applied is enough to estimate the present system state exactly, assuming there is no noise input. The difference here is that we have a controlled Markov Chain as the simulation model, and we can estimate each possible outcome of the present state with accurate probabilities.

This idea is also motivated by the separation principle, where a good state estimation is very important to the control effect. Some earlier applications of separation principle to stochastic control systems can be found in [78] [12] [23] [21] [53]. Recently Grimble [34] applied the separation principle to a discrete stochastic control system in frequency domain. Rishel [73] studied the separation principle in linear quadratic system with hidden Markov drift. Most of the research applied the state estimation procedure in linear systems in discrete time. For nonlinear stochastic control systems with time delay or partial information in continuous time, the analytical solution is hard to find. Little is known about the numerical methods and its convergence to continuous time solutions. That motivates us to use the controlled Markov Chain approximation with state estimation, where the convergence proof is an advantage of this numerical approach. We also noticed that the state estimation in the discrete system with time delay makes the optimality proof possible [77] [61], so by adding the state prediction in our numerical approximation, we are in hope to achieve the optimality of the numerical methods and hence the optimal continuous time solution which the numerical result converges to.

Another merit of this control policy is that the dimension of the simulation system is reduced dramatically. Recall that in the previous control strategy, 2τ of the state history is recorded for each queue, where τ is the length of time delay. So for the two queues, we need to keep 4τ of path information. The dimension of the Markov Chain simulation will increase exponentially accordingly. And if there are more queues, the computer memory needed to run the simulation would be huge. In the policy discussed in this chapter, we

only need to keep τ length of the control history, since there is only one control variable.

In the solution iteration step in this new policy, we avoid solving the huge linear system, as what we did in the previous method. In fact, we update the running cost associated with each state node by node. Then the huge computer memory requirement of the transition probability matrix for the Markov Chain is freed. Therefore, larger scale and finer interpolation grid problems can be solved.

The control system is a reflected controlled diffusion system with delay, which is identical to what we considered before (equation (3.7) in Section 3.2).

$$dx(t) = b(x(t-\tau), u(x(t-\tau)))dt + \sigma dw(t) + dz(t), \tag{4.1}$$

where $x(t)$ is the state of the system, $b(\cdot)$ and σ are the drift and volatility term respectively, and $w(t)$ is the standard Wiener process. The solution $x(t)$ is confined to a finite interval $G = [0, B]$. $z(t)$ is the reflection term which is nonzero only when $x(t)$ is on the boundary, in order to keep the state feasible. The time delay length is denoted by τ. We can note that the drift term is a function of the delayed information $x(t-\tau)$. $u(t)$ is the admissible control chosen to minimize a certain cost function $W(x, u)$, e.g.,

$$W(x, u) = E_x^u \int_0^\infty e^{-\beta s} k(x(s), u(s)) ds, \tag{4.2}$$

where β is the discount factor, and $k(\cdot)$ is the cost rate.

To represent the system state $x(t)$, the Markov Chain approximation is again denoted by $x_n^h(\cdot)$, a piecewise constant function which takes values on a discretization of the state space where the distance between adjacent state is h. The choice of h also gives an associated interpolation time interval $\Delta t^h = \frac{h^2}{\sigma^2}$ over which the state x_n^h is held constant. We use x_n^h to denote a particular state, and use $x_n^h \in G_h^0$ to denote some general feasible state in G_h^0, where G_h^0 denotes the set of interior points of the discrete feasible region of the system states.

All the conditions on $b(), u(), \sigma$ that are needed for the local consistency and those needed for the controlled Markov Chain approximation are assumed to be held, as discussed in previous chapters, say Chapter 2 and 3.

Kushner [45] showed that for this kind of stochastic delay control problems with reflection, the controlled Markov Chain approximation can be applied, and under that scheme, the state approximation x_n^h will jump to $x_n^h + h$ with probability p^h and jump to

$x_n^h - h$ with probability $1 - p^h$, where p^h is given below

$$p^h(x_n^h, x_n^h \pm h|u) = \frac{\sigma^2 \pm hb(u(x_n^h(t-\tau)))}{2\sigma^2}, \tag{4.3}$$

where u is the control policy associated with the system states, and $x_n^h(t-\tau)$ is the discrete approximation of the delayed information $x(t-\tau)$.

Let $W^h()$ denote the discounted cost associated with the discretized process, then $W^h(x)$ is the approximation of the expected running cost starting from the point x,

$$W^h(x) = E_x \sum_{n=0}^{\infty} e^{-\beta n \Delta t^h} \left[\Delta t^h k(x_n^h, u_n^h) + c(x_n^h)\delta z_n \right], \tag{4.4}$$

where x_n^h, u_n^h are the states and controls at time step n, and $c(\cdot)$ is the cost increment if the state hits the boundary of the feasible region, and δz_n is the discrete approximation of the process $z(t)$. So $W^h(x)$ is the expected discounted running cost of the system starting from the state x, given the control variables u_n^h.

The iteration for updating $W^h(\cdot)$ for the interior points is thus

$$\begin{aligned} W^h(x) &= e^{-\beta \Delta t^h} p^h(x, x+h|u) W^h(x+h) \\ &+ e^{-\beta \Delta t^h} p^h(x, x-h|u) W^h(x-h) + \Delta t^h k(x^h, u), \end{aligned} \tag{4.5}$$

and the optimal running cost is

$$V^h(x,u) = \inf_{u \in U} W^h(x), \tag{4.6}$$

where U is the set of admissible controls.

Define the matrix $R^h(u) = \{r^h(x, y|u(x)); x, y \in G_h^0\}$, where

$$r^h(x, y|u) = e^{-\beta \Delta t^h} p^h(x, y|u) \tag{4.7}$$

for $x \in G_h^0$, and G_h^0 is the set of interior points of the feasible region of the discretized state. Then the cost equation (4.5) can be written as in the following dynamic programming form

$$W^h = R^h(u) W^h + C^h(u), \tag{4.8}$$

$$V^h = \min_{u \in U} [R^h(u) V^h + C^h(u)], \tag{4.9}$$

where $C^h(u)$ is the cost increment, which is $\Delta t^h k(x_n^h, u)$ plus possible boundary cost $c(\cdot)$ in our example.

The simple control policy, which we also call delayed control policy, assumes the delayed information as the present state, and takes the transition probability as in (4.3). Using the state estimation as in previous chapters, we have shown that if we can estimate the present state and apply the control accordingly, we can achieve a better control policy, which gives a smaller running cost than that of the simple control. That is, to use the estimation $\hat{x}_n^h$ to replace the delayed information $x_n^h(t-\tau)$, and use the transition probability

$$p(x_n^h, x_n^h \pm h|u) = \frac{\sigma^2 \pm hb(u(\hat{x}_n^h))}{2\sigma^2}. \tag{4.10}$$

As pointed out in the introduction of this chapter, it remains a question whether the separation principle can guarantee the optimality of the control policy obtained, because it estimates the present state assuming that the system is controlled under optimal control of the no-delay system u^*, where the system does not. Using the state prediction procedure in our example gives us an upper bound of the running cost, which is lower than that of the delayed control.

Notice that the system is run under the control variable which is actually applied to it, so if we know the controls that have been applied by the present time, we can, based on the information we have, estimate the system state more accurately. Here we present our idea to find the control policy while keeping the path information of control history.

4.2 Markov Chain Approximation Model with Control History

Assume the system has time delay τ, and by choosing the interpolation level h for the state space, we also get the time interval $\Delta t^h = \frac{h^2}{\sigma^2}$ for the time space. Let $D = [\tau/\Delta t^h]$, i.e., there are D intervals in the delay path of the discrete approximating system. Denote these states by

$$x_{n-D}^h, x_{n-D+1}^h, \ldots, x_{n-1}^h, x_n^h,$$

shown in Fig 4.1, similar to the Figure 3.1 in Chapter 3, but with much less memory requirement.

In this Figure x_n^h is just the present system state, and x_{n-D}^h is the approximation of the delayed state $x(t-\tau)$. Because of the time delay, the states $x_{n-D+1}^h, \ldots, x_{n-1}^h, x_0^h$ are not observable at present time, i.e., time zero. For the delayed control policy, the transition

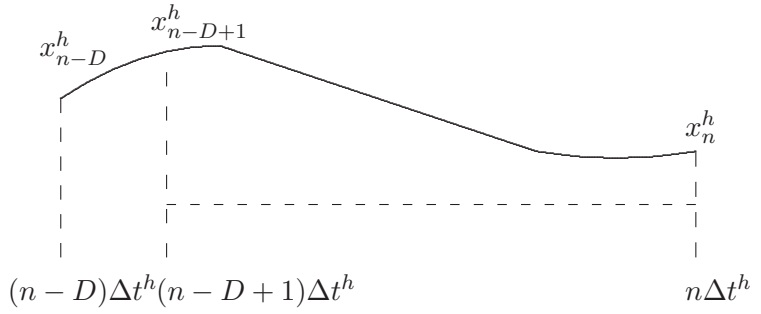

Figure 4.1: Approximating the Delay Path

probability is in the following form by Kushner [45]

$$p(x^h_n, x^h_n \pm h|u) = \frac{\sigma^2 \pm hb(u(x^h_{n-D}))}{2\sigma^2}. \tag{4.11}$$

If only the delayed information $x(t-\tau)$ or x^h_{n-D} is available, the construction of controlled Markov Chain approximation using (4.11) gives the best choice. It just takes the delayed system state x^h_{n-D} as the present system state. However, as we mentioned that, where the state will jump to in the next time period only depends on where the present state is and what control variable we apply to it. So if we could estimate the present system state, we are expected to find a better control policy. This kind of result was shown in previous chapters, where we estimate the present state using Policy I or II and replace x^h_{-D} by an estimation $\hat{x}^h$ in (4.11). There we found a better control policy than the Simple Control, hence found an upper bound on the optimal system running cost.

Notice that assuming the system to be controlled under the optimal control for no-delay system cannot guarantee the optimality of the result for the delayed system, and although the system has time delay τ in observation, which means only x^h_{n-D} is available, we do have access to the control variables that have been applied, that is, the information $u_{n-D}, u_{n-D+1}, \ldots, u_{n-1}$ are known, which are the control variables applied at time $(n-D)\Delta t^h, (n+1-D)\Delta t^h, \ldots, (n-1)\Delta t^h$. Our task is to determine the control variable u_n to apply at present time.

Under the controlled Markov Chain approximation model, the information $(n-D)\Delta t^h, (n+1-D)\Delta t^h, \ldots, (n-1)\Delta t^h$ together with the delayed state x^h_{n-D} along with control function are enough to estimate the present state x^h_n, as shown in the figure below.

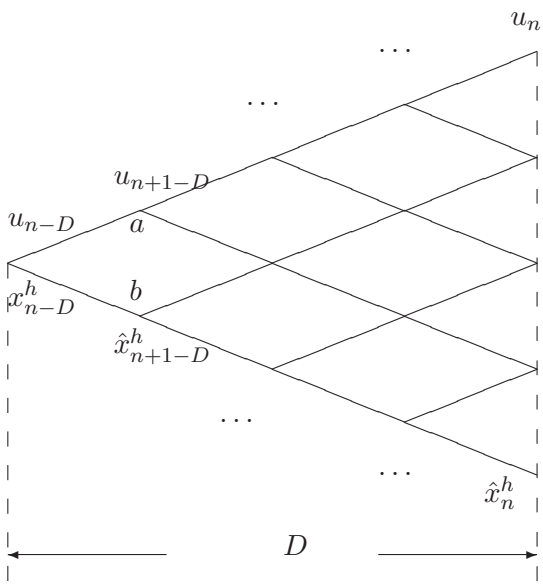

Figure 4.2: Controlled Markov Chain Approximation of the System with Delay

Suppose the present time is at the nth step, the information

$$(u_{n-D}, u_{n-D+1}, \ldots, u_{n-1}, x^h_{n-D})$$

are known. $(\hat{x}^h_{n+1-D}, \ldots, \hat{x}^h_{n-1}, \hat{x}^h_n)$ are the state estimations, and the variable u_n is the one that needs to be determined.

Because now the state in our controlled Markov Chain structure becomes $(u_{n-D}, u_{n-D+1}, \ldots, u_{n-1}, x^h_{n-D})$, we may write the cost function in the simulation as

$$W^h(u_{n-D}, u_{n-D+1}, \ldots, u_{n-1}, x^h_{n-D})$$

or $W^h(\mathbf{u}, x^h_{n-D})$, where $\mathbf{u} = (u_{n-D}, u_{n-D+1}, \ldots, u_{n-1})$. And the control u_n which we seek can be written as $u(\mathbf{u}, x^h_{n-D})$.

The estimates $(\hat{x}^h_{n+1-D}, \ldots, \hat{x}^h_{n-1}, \hat{x}^h_n)$ can be determined one by one. For example, the points a, b are two possible outcomes of the state x^h_{n+1-D}, since we know x^h_{n-D} and the control u_{n-D} was applied, the probability that

$$P(x^h_{n+1-D} = a = x^h_{n-D} + h) = \frac{\sigma^2 + hb(u_{n-D})}{2\sigma^2},$$

and similarly,

$$P(x^h_{n+1-D} = b = x^h_{n-D} - h) = \frac{\sigma^2 - hb(u_{n-D})}{2\sigma^2}.$$

Start from the nodes a, b, the probabilities of the outcomes of state x^h_{n+2-D} can be calculated, and so forth, until we reach the estimation of the state x^h_n. That is, if let $v_1, v_2, \ldots, v_m$ be the possible outcomes of x^h_n, then we have computed the exact probabilities of $p^h_i = P(x^h_n = v_i), i = 1, 2, \ldots, m$.

If the system is a discrete controlled Markov Chain system with time delay in the state observation, then using this estimating method based on delayed information and control history, we can predict the outcome of the present system state with exact probability. If the system is just a discrete deterministic dynamic system with time delay, then the present state x^h_n can be determined, which is no longer a random variable for us.

The controlled Markov Chain model, as an approximation to the stochastic control system in continuous time, is our underlying model, and the system is assumed to run following the discrete controlled Markov Chain structure. The utilization of the control path history helps us determine the probability of every possible outcome of the present state x^h_n. Since the states are discrete, to find a good match of x^h_n, we choose the outcome that has the highest probability to occur. That is, choose

$$\begin{aligned} \hat{x}^h_n =& v_j, \\ \text{where } p^h_j =& P(x^h_n = v_j) = \max_{1 \le i \le m} P(x^h_n = v_i). \end{aligned}$$

This is one of the estimating method we used in previous chapters, i.e., Policy II in Chapter 3. As we discussed earlier, it is convenient to find one specific match of x^h_n than using the average of all possible outcomes, because the states are usually multidimensional, the average does not have an explicit meaning, for example, it may not necessarily be a feasible state in the controlled Markov Chain scheme. Another advantage using the specific match is that, we can apply a control associated with this estimation, and form the control policy of the approximating system easily.

4.3 Finding the Control using Receding Horizons

After finding the estimation $\hat{x}^h_n$ of the present state x^h_n, we use it as an approximation of x^h_n and determine the control from there. The cost update formula is given in (4.5), and (4.8) gives the matrix form.

By considering $\hat{x}_n^h$ as an approximation of x_n^h, we can take the system as if there were no time delay. That is, we study the system as if we know the present system state.

Recall Kushner's algorithm using the controlled Markov Chain approximation in Chapter 2. The solution of cost matrix W^h and optimal control policy u are determined iteratively [47]. At the kth iteration, the control for the next iteration $u_{(k+1)}$ is given in the following form

$$u_{(k+1)}(x) = \arg\min_{\alpha \in U} \begin{bmatrix} e^{-\beta \Delta t^h} p^h(x, x+h|\alpha) W^h_{(k)}(x+h) & \\ +e^{-\beta \Delta t^h} p^h(x, x-h|\alpha) W^h_{(k)}(x-h) & +\Delta t^h k(x,\alpha) \end{bmatrix}, \tag{4.12}$$

where the subscript k is the index of iteration number.
A general form is as follows

$$u_{(k+1)}(x) = \arg\min_{\alpha \in U} \left[\sum_y r(x, y|\alpha) W^h_{(k)}(y) + C(x,\alpha) \right], \tag{4.13}$$

for $x, y \in G_h^0$, and $r(x, y|\alpha)$ is in (4.7).
The formula for updating the cost function $W^h_{(k)}$ is

$$W^h_{(k+1)}(x) = \sum_y r(x, y|u_{(k+1)}(x)) W^h_{(k)}(y) + C(x, u_{(k+1)}), \tag{4.14}$$

where $u_{(k+1)}(x)$ is in (4.13).
The recursive equation for the optimal cost function $V^h_{(k)}$ is defined by

$$V^h_{(k+1)}(x) = \min_{\alpha \in U} \left[\sum_y r(x, y|\alpha) V_{(k)}(y) + C(x,\alpha) \right], \tag{4.15}$$

or in the general dynamic programming equation

$$V_{(k+1)} = \min_{\alpha \in U} \left[R(\alpha) V_{(k)} + C(\alpha) \right]. \tag{4.16}$$

Let V be the infimum of the cost over all admissible control sequences, Kushner [47] showed that under some general conditions, $W^h_{(k)} \to V$ as $k \to \infty$.

Consider the control updating equation (4.12), and notice that it is in fact a one-step dynamic programming equation. There are some reasons we increase the decision horizon to P steps. The first is, if the initial guess for $W^h(\cdot)$ is far from the final solution, the convergence would be very slow and the computation time would be very long. If the horizon is longer, by adding up the discounted cost at each step, we can obtain a closer

approximation to the running cost $W^h_{(k)}(\cdot)$, rather than using an arbitrary initial guess. Consider again the equation for $W^h(\cdot)$ in (4.4). Because of the discounting factor, the cost of the first several steps would account for a considerable portion of W^h. Use this value as the initial guess for W^h would speed up the computation quite a bit.

The second reason is that, because of the initial cost array, for the first several iterations, the determined control policy might be quite different from the final solution. Longer decision horizon is expected to yield the control which converges faster. The following figure shows the scheme that starts from the estimation $\hat{x}^h_n$ and uses P steps of decision horizon.

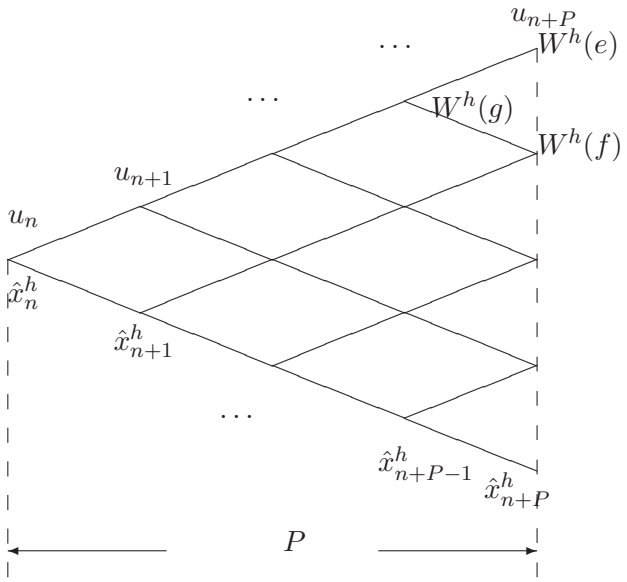

Figure 4.3: Determine the Control using P steps of Decision Horizon

Suppose we are at the kth iteration of the algorithm. Start from the estimation $\hat{x}^h_n$, we need to find a sequence of controls $(u_n, u_{n+1}, \ldots, u_{n+P})$ for the P steps of decision horizon. That is, we perform a P steps of dynamic programming. The values at the Pth step are the boundary conditions for the dynamic programming, i.e., $W^h(e), W^h(f)$, etc.

According to the dynamic programming principle (4.15), the values are chosen as follows

$$W^h(e) = \min_{\mathbf{u}_e} W^h_{(k)}(\mathbf{u}_e, e), \tag{4.17}$$

which is just the minimum running cost so far starting from the node e, given that the control $\mathbf{u}_e$ has already been applied. In fact, the control $\mathbf{u}_e$ is the control sequence

$$u_{n+P+1}, u_{n+P+2}, \ldots, u_{n+P+D}$$

which are not determined within our P step decision horizon but will be applied at time $n+P+1, n+P+2, \ldots, n+P+D$. At time $n+P+D$, the state e or f or other possible outcome at time P will be observable, and the control $u_{n+P+1}, u_{n+P+2}, \ldots, u_{n+P+D}$ has been applied. The sequence $(u_{n+P+1}, u_{n+P+2}, \ldots, u_{n+P+D}, e)$ or $(u_{n+P+1}, u_{n+P+2}, \ldots, u_{n+P+D}, f)$ or other possible outcome forms our standard system state in the controlled Markov Chain approximation. Therefore, the cost representation in the P step dynamic programming horizon is consistent.

After the boundary condition is set, the dynamic programming can be performed. For example, from the point e, f we work backward to calculate the cost $W^h_{(k)}(g)$ and the associated control u_{n+P-1} at time $n + P - 1$, using the one step dynamic programming principle (4.12). The procedure continues until we reach the node $\hat{x}^h_n$ and control u_n. Denote this control at the kth iteration $u_{n(k)}$. This control $u_{n(k)}$ is the control that we want to apply to the system at present time, based on delayed information x^h_{n-D} and a path of control history $(u_{n-D}, u_{n+1-D}, \ldots, u_{n-1})$. Notice that $\hat{x}^h_n$ is just an estimate of the present system state x^h_n. The states $v_1, v_2, \ldots, v_m$ are all possible outcomes. Therefore, to calculate the expected cost, the control $u_{n(k)}$ is actually applied to the states $v_1, v_2, \ldots, v_m$ with corresponding probabilities $p^h_1, p^h_2, \ldots, p^h_m$. This is where the extra cost is brought by the time delay.

Discount all the costs in the $D + P$ delay and decision tree Fig 4.2 and 4.3 to the node x^h_{n-D}, we get an updated cost $W^h(u_{n-D}, u_{n+1-D}, \ldots, u_{n-1}, x^h_{n-D})$, and this is the cost in the next iteration, $W^h_{(k+1)}(u_{n-D}, u_{n+1-D}, \ldots, u_{n-1}, x^h_{n-D})$.

The idea using a P decision horizon is similar, but not quite, to the Model Prediction Control (MPC) strategy [27]. In every iteration we calculated a sequence of controls $u, u_1, \ldots, u_P$ and apply the first one to the system. The MPC is a powerful realtime control policy, which was first presented to solve control problems of deterministic systems. Recently researchers extended this policy to discrete stochastic system control problems, see

[31] [19] [85] [10]. Wei [86] used the prediction procedure in a nonlinear stochastic discrete control system. Ren [71] and Kumar used this method in stochastic adaptive filtering. The underlying systems are all discrete and stochastic. The MPC is also founded in dealing with system delays. Marinescu [56] used the MPC to solve a deterministic control system with time delay. Here we are trying to solve a continuous time stochastic control problem with time delay, using the controlled Markov Chain approximation as our discrete model, where the state estimation and a decision horizon is performed.

To set up the algorithm, we may start the simulation with the no-delay system. Then the boundary condition can be just set zero. Because we use a larger horizon P, the running cost accumulates fast and the convergence is thus speed up. After obtaining the cost for no-delay system, we can add time delay to the system. For the first iteration in this case, the boundary conditions can just be set the no-delay cost. This can be a close approximation for the running cost of the delay system. Then for further iterations, the boundary conditions are calculated according to (4.17).

Another advantage of this algorithm, as we mentioned earlier, is that we do not need to keep a huge memory of the transition probability matrix of the controlled Markov Chain, as we did in Chapter 1 or 2. The size of the transition probability matrix increases exponentially as the number of states and length of delay increase. To update the cost function $W^h(\cdot)$, the Gauss-Seidel method is used, which involves solving the huge linear system. Those all bring difficulty for the simulation. The algorithm presented here updates the cost function node by node. At each node update, it only needs to keep the $D+P$ steps of the delay and decision horizon. Therefore, the problem with larger scale and finer grid can be solved.

4.4 Application to Wireless Communication Queueing Models

We will study the same heavy traffic model used in Chapter 3, derived from wireless communication queueing systems presented by Buche and Kushner [13]. This is an optimal power allocation problem for a two dimensional wireless communication queueing system. The base station maintains a certain amount of power that meets the average running cost of the system. A small portion of extra power called reserved power is applied among the

queues in order to reduce the queue length of certain queue. The control problem is to allocate the reserved power optimally in order to have the minimum expected running cost of this stochastic system. We notice that the channel state factor does not bring the key point, so we only consider the system with one channel state, and solve for the optimal power allocation policy. Time delay of length τ is introduced to the system as we did in chapter 3.

4.4.1 Model construction

Under the assumptions given in [13], and by applying the weak convergence analysis, the queue size x_i for the ith queue follows a diffusion process

$$x_i(t) = x_i(0) + \int_0^t b_i(u, \chi(s))ds + \sigma_i w_i(t) + z_i(t), i = 1, 2, \tag{4.18}$$

where $\mathbf{x}$ denotes the two dimensional system state, σ_i is the volatility of queue i, and w_i is a standard Wiener process. z_i is the reflection term that keeps the state feasible, i.e., within the feasible region $[0, B]$. Here for simplicity, we assume each queue has the same bound. The drift term b_i is given by

$$b_i(u, \chi(s)) = -\sum_j u_i(j, \chi(s))\bar{\lambda}_i^d(j)\pi(j),$$

for queue i and channel state j at time s, where $u_i \in \{0, u_r\}$ is the power allocated to queue i, i.e., reserved power, $\bar{\lambda}_i^d(j)$ is the average departure rate or processing rate of queue i under channel state j, and $\pi(j)$ is the stationary probability distribution of channel state j. Notice that $u_i \in \{0, u_r\}$, which means our control policy is to allocate the reserved power as a whole to queue i or apply nothing to queue i. Since we only consider one channel state, the term is simplified as

$$b_i(u, \chi(s)) = -u_i(\chi(s))\bar{\lambda}_i^d, i = 1, 2. \tag{4.19}$$

The delay part is introduced in the observation term $\chi(s)$ due to the mechanical design of the system base or other disturbances. At present time, the system state $\chi(s)$ is not observable, and in stead, we can only observe the delayed information $\chi(s - \tau)$. Thus the problem becomes

$$b_i(u, \chi(s)) = -u_i(\chi(s - \tau))\bar{\lambda}_i^d, i = 1, 2. \tag{4.20}$$

Choose a small h as the numerical interpolation level for $\chi(s) \in [0, B]^q$, where q is the number of queues, and for our example $q = 2$. Then the time interpolation is $\Delta t^h = \frac{h^2}{\sigma^2}$. In the state space, there will be $[B/h] + 1$ discrete node in the controlled Markov Chain approximation. In order to represent the reflection term, for each queue, there are two more nodes that are outside of the feasible region, which are the states $-h$ and $B + h$. If the system is in these states, the reflection term becomes active and will bring the state into feasible region immediately. Let $D = [\tau/\Delta t^h]$ be the number of interpolation steps in the delay path, then the path representation is

$$\chi^h_{n-D}, \chi^h_{n+1-D}, \ldots, \chi^h_n.$$

Suppose the present time is $n\Delta t^h$, then due to time delay, only the state χ^h_{n-D} is observable. However, the control history $(u_{n-D}, u_{n+1-D}, \ldots, u_{n-1})$ is a known factor. So the state representation of the controlled Markov Chain is $(u_{n-D}, u_{n+1-D}, \ldots, u_{n-1}, \chi^h_{n-D})$, and the cost function $W^h(\cdot)$ is just $W^h(u_{-D}, u_{1-D}, \ldots, u_{-1}, \chi)$, which is

$$\begin{aligned} &W^h(u_{-D}, u_{1-D}, \ldots, u_{-1}, \chi) = \\ &E_\chi \sum_{n=0}^{\infty} e^{-\beta n \Delta t^h} \left[\Delta t^h k(\chi) + c(\chi) \delta z_n \right], \end{aligned} \tag{4.21}$$

where the subscript n is the index of time interval.

The cost associated with the interior point χ is a convex and increasing function in χ. For our two-queue model, we choose the form

$$k(\chi) = w_1 (x^h_{[1]})^{p_{[1]}} + w_2 (x^h_{[2]})^{p_{[2]}},$$

which is a polynomial of the queue size $x^h_{[1]}, x^h_{[2]}$. w_1, w_2 are the weights of the two queues, and $p_{[1]}, p_{[2]}$ are the powers of the queue size. We put brackets [] for the queue index $1, 2$ in order to distinguish from the subscript n which the time index in 4.21. At the boundary of the feasible region, a penalty cost is introduced as the term $c(\chi)$. When the queue is empty, a small cost penalty c_u is introduced, and when the queue is full, a bigger cost penalty c_o is introduced, as we did in Chapter 3.

The underlying system follows a controlled Markov Chain where the transition probability is given by,

$$p^h(x^h_{[i]}, x^h_{[i]} \pm h | u) = \frac{\sigma^2 \pm h b_i(u(\chi))}{2\sigma^2}, i = 1, 2. \tag{4.22}$$

Here the canonical state χ is $(x^h_{n[1]}, x^h_{n[2]})$, i.e., the queue sizes for the two queues. Use the state representation and (4.22), the present system state can be estimated as in Fig 4.3, which is denoted by $(x^h_{n[1]}, x^h_{n[2]})$. Start from this point $(x^h_{n[1]}, x^h_{n[2]})$, a sequence of controls $u, u_1, \ldots, u_P$ can be found using the receding decision horizon, where P is the length of the horizon. In short, the control history helps us estimate the present system state, and then the system is treated as a no-delay system. A decision horizon is then used to determine the control u_n. Apply the control u_n to the system and update the cost $W^h_{(k)}(\cdot)$. The detailed algorithm is given in the following section.

4.4.2 The Algorithm

Set the parameters for queue bound B, interpolation level h, queue cost weight w_1, w_2, cost power $p_{[1]}, p_{[2]}$, boundary cost penalty c_u, c_o, time delay τ and hence the delay steps D, prediction horizon length P, discount rate β, system volatility σ, queue average departure rates $\bar{\lambda}^d_1, \bar{\lambda}^d_2$, and the reserved power u_r. Set the iteration number N and the stop criteria ϵ, set subscript $k = 1$.

if D=0 **then**

Set the initial guess zero for $W^h_{(1)}$.

while $k \leq N$ or not convergent **do**

For each node, construct the decision tree as in 4.3, where the transition probabilities follow 4.22.

Apply dynamic programming principle to determine the control $u_{(k)}$.

Update $W^h_{(k)}$ to $W^h_{(k+1)}$, and update the error $|W^h_{(k+1)} - W^h_{(k)}|_\infty$ as well for the stopping criterion.

Increase k.

end while

Record the running cost and control policy.

else

Input the no-delay cost as the initial guess for $W^h_{(1)}$.

while $k \leq N$ or not convergent **do**

Use the control history information to estimate the present system state $\hat{x}^h_n$ as in 4.2, where the transition probabilities follow 4.22.

Construct the decision tree as in 4.3, and apply dynamic programming principle to

determine the control $u_{(k)}$.

Update $W^h_{(k)}$ to $W^h_{(k+1)}$, and update the error $|W^h_{(k+1)} - W^h_{(k)}|_\infty$ as well for the stopping criterion.

Increase k.

end while

Record the running cost and control policy.

end if

4.4.3 Simulations

The state representation is $(u_{n-D}, u_{n+1-D}, \ldots, u_{n-1}, x^h_{n-D})$, where $u_{-j}, j = 1, 2, \ldots, D$ can be $1, 2$ or 0 indicating that the reserved power was applied to queue $1, 2$ or was not applied. As an illustration, we just give the result for the states where $u_{-j} = 0$. That is, we give the control policy and running cost for the system where we have not applied any control to the system for the past τ time of delay. Or in other words, before we observe any delayed state of the system, we apply no control to the system. Other results are similar.

Only one channel state is considered, as we mentioned in Section 4.4 that the channel state does not play the key role in this problem. Table 4.4.3 gives the simulation settings. We first use the length of prediction horizon $P = 1$, and for larger P, the discussion is made near the end of this section. We notice that we are handling a much larger scale problem, where the interpolation level is $h = 0.02$, whereas in Chapter 3 we used $h = 0.1$. And the queue size here is also larger than before.

Table 4.1: Simulation Parameters

No. of Queues	2	No. of Channel State j	1
Queue Weights w_1, w_2	$1, 0.5$	Base Volatility σ^2	0.3
Reserved Power u	1	Discount Rate β	0.05
$\widetilde{\lambda}^d_i(j), [j = 1, i = 1, 2]$	$[1, 1]$	Cost Rates p_1, p_2	$2, 2$
Queue Size B	10	Interpolation Level h	0.02
Penalty on Empty Queue c_u	10	Penalty on Blocked Queue c_o	3000
Iteration number N	15000	Stopping Criterion	0.001

No delay case $D = 0$

The system with no time delay is first considered. Remember that the algorithm used here (See Section 4.4.2) is quite different from that used in Chapter 3. We no longer have to record the transition probability matrix, which enables us to free much computer memory required, nor do we use the Gauss-Seidel algorithm to solve the huge linear system. Instead, we update the cost vector $W^h()$ node by node. That is why we put a large iteration number and strict stopping criterion. The system is much larger than we could solve before, and the only disadvantage is that the computation time is longer. Because we are not computing a real time control policy, and the interpolation interval h plays the fundamental role in the accuracy of this numerical algorithm, we tend to use this algorithm (Section 4.4.2) in future research.

Fig 4.4 shows the running cost of this algorithm.

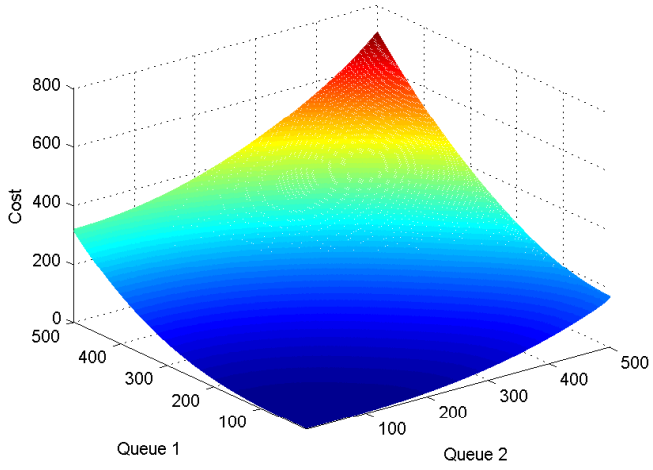

Figure 4.4: Running Cost (No delay case)

Interesting enough, or is expected, the cost function is quite similar to the cost figures in Chapter 3. If the simulation starts with high queue sizes, the total discounted running cost tends to be large. Notice that queue 1 weighs more than queue 2, and we see that if the starting point has big queue 1 size and low queue 2 size, the cost is greater than that of the starting point with low queue 1 size but high queue 2 size.

The power allocation is given in Fig 4.5. Again, it has similar properties of the control figures in Chapter 3. There is a narrow switching curve that divides the state space.

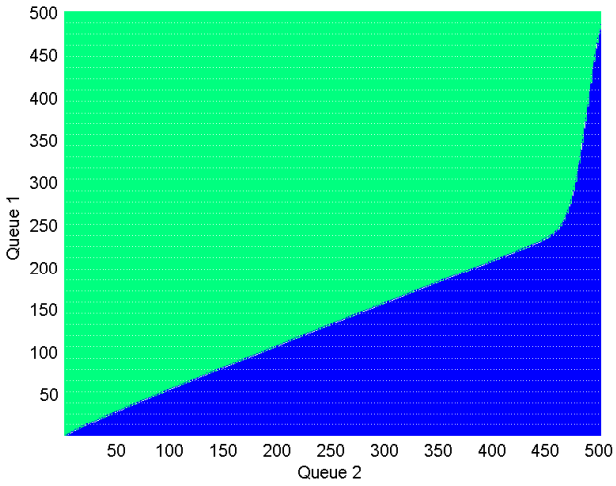

Figure 4.5: Power allocation for Queue 1 (No delay case)

To the left side of the switching curve, all the power is allocated to Queue 1, and to the other side of the curve, all the power is assigned to Queue 2. The shape of the switching curve is close to that of the control figures in Chapter 3. Because Queue 1 has higher weights than Queue 2, we can see that the power assignment region for Queue 1 is bigger than Queue 2, in order to reduce the total discounted cost.

One Step of Delay $D = 1$

In this section, we introduce one step of delay, that is, $\tau = \Delta t^h$. Then in the simulation, the state representation would be $(u^h_{n-1}, \chi^h_{n-1})$ at time interval n, where χ^h_{n-1} is the one step delayed queue sizes, i.e., queue 1 and Queue 2, and u^h_{n-1} is the control we have applied at time $(n-1)\Delta t^h$. u^h_{n-1} can indicate the power allocation to Queue 1 or Queue 2, or no power assigned. Therefore, there will be 3 different scenario associated with the state χ^h_{n-1}. For illustration purpose, we give the result of $u^h_{n-1} = 0$, and other results are analogous to this one.

Fig 4.6 is the running cost for one step of delay.

Unlike that in Chapter 3, the state representation in this chapter has a physical meaning. In other words, χ in the state represents the true queue sizes. The delay path information only contains the control history. Therefore, it makes sense to compare the result with the same queue sizes but different steps of time delay, and in this way, we are

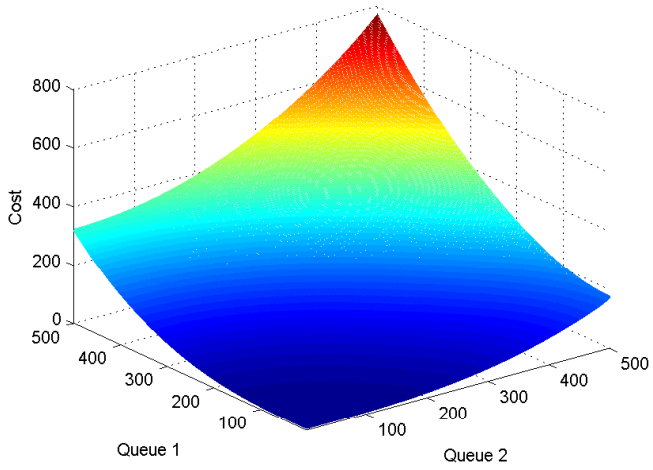

Figure 4.6: Running Cost (One step of delay case)

able to analyze the effect of time delay. So here we take a look at the cost increment by the one step delay case over the no-delay case.

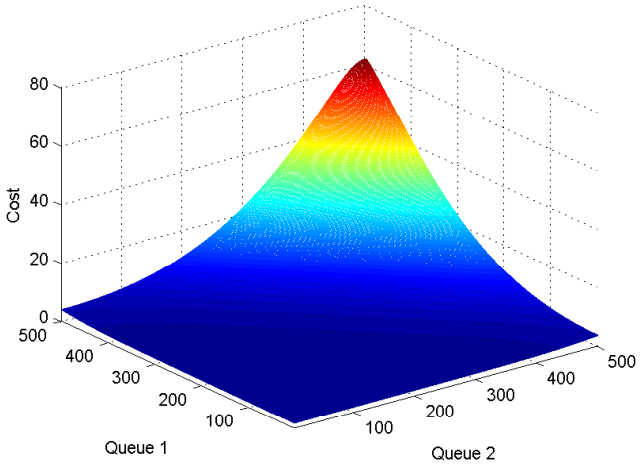

Figure 4.7: Running Cost Difference between the One step of delay case and the no-delay case

From Fig 4.7 we can see that the running cost increases due to the time delay and of course the control policy we used here.

The control for this one step of delay case is given in Fig 4.8. Again the narrow

switching curve is seen.

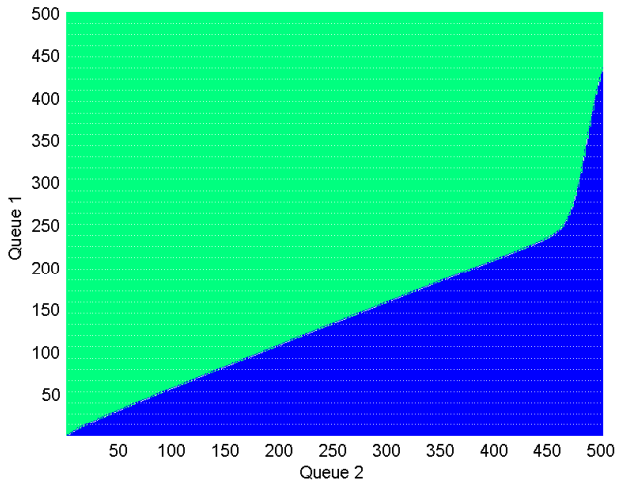

Figure 4.8: Power Allocation (One step of delay)

To compare the control differences between the delay and no-delay cases, we give the error in Fig 4.9.

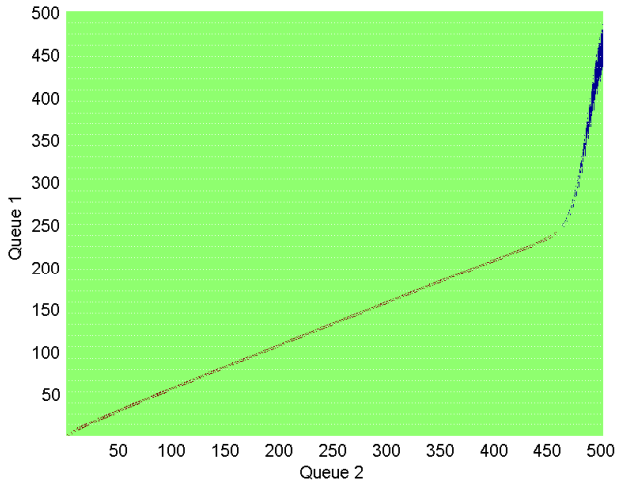

Figure 4.9: Control Difference Between the delay and no-delay cases

Interesting enough, we notice the small difference along the switching curve again, as we saw in the simulation result in Chapter 3. This small difference is due to the one step of time delay and the control policy we applied here. And this small amount of control

error causes the running cost increment in Fig 4.7, same conclusions as in Chapter 3.

Because of the state representation in the controlled Markov Chain approximation here, for the one step of delay case, all the states χ are the observed states which are τ time delayed. In the simulation, we have no connection to the present system state because of this representation. Therefore, we can not apply the Optimal Control policy which has complete system information and not practical in real time system control, as we did in Chapter 3, hence the lack of the control result comparison. However, the state representation we used here enables us to predict the present system state with actual probabilities. This provides the possibility to find and prove the optimality of the control policies which are practical for the delayed system. The optimal control is not necessarily Policy II, which we applied in this section and which takes the prediction with the highest probability of outcome as the present state estimation, but this state representation does provide the convenience for finding the optimal practical control.

4.5 Summaries and Further Topics

The algorithm presented in this chapter gives a good control result, where the graphs look similar to previous results. However the state representation here, which accommodates the control history, is quite different from previous chapters, where the system delayed path is part of the state representation. Therefore, it does not make much sense to compare these results. However, it does not prevent us from obtaining a feasible control policy for the stochastic system with time delay.

Though the result is decent, it still remains a question that under what condition this policy is optimal for the discrete controlled Markov Chain model with time delay. And also, how does the time delay affect the control effect? What role does the length of receding horizon play in obtaining the control policy?

We use the discrete controlled Markov Chain model as the approximation to the stochastic system in continuous time. It is known that as the interpolation level h goes to zero, the discrete control policy converges to the continuous time control policy weakly, provided that the system has no time delay. Later Kushner [45] showed that if the system has time delay, then for the simple discrete control policy, the convergence still holds. It is not known yet if this convergence holds for our control policy presented in this chapter. We notice that our policy is a feasible control policy, and it does not alter the structure of the

proof of convergence as in the no-delay or simple control policy case. However, it still needs rigorous proof to show the convergence to a feasible control strategy in continuous time.

Chapter 5

Convergence Studies on Monte Carlo Methods for Pricing Mortgage-Backed Securities

The stochastic model we considered in Chapter 3 and 4 is a kind of stochastic delay differential equations (SDDE), a particular form of path dependent stochastic models. Although the controlled Markov Chain approximation (analogously called finite difference method), as one of the powerful numerical approaches, gives decent result for the SDDEs we studied, it is hard to apply where the path dependency is complicated, for example, very long path dependence. We have already seen that from Chapter 3 and 4, for a small amount of time delay, the computer memory requirement is huge, even though we have tried many technical methods to reduce this restriction. For some computations which relies heavily on the path of the underlying stochastic process, and not much control issue is concerned, the controlled Markov Chain approximation is not, in general, the best choice. For example, the pricing of Mortgage-backed securities (MBS) which depends on the entire path of the interest rate. In other words, because of the following reasons, normal numerical methods like lattice methods, or finite difference method for solving PDEs are hard to apply: 1) the path dependence of mortgage pool cash flows. 2) the embedded American call option to prepay. 3) the American put option to default. 4) the fact that mortgage borrower do not/cannot exercise these option optimally [18]. In this case, the Monte Carlo simulations are the dominant tools solving such kind of problems. The underlying stochastic process is

discretized, then for each path generated, the computation is performed and the average of the runs is taken as an approximation of the theoretical solution. This is the idea of Monte Carlo simulations.

In this chapter, we examined the procedure of Monte Carlo simulations in pricing MBS. Because this method relies on computer generated pseudo random numbers, we are interested in finding out the performance of different random number generators in pricing MBS, and by doing the Option Adjusted Spread analysis, we found that the pricing procedure can be speed up dramatically.

The idea is first introduced in Section 5.1. The background of MBS is given in Section 5.2. Then the Monte Carlo Simulation is introduced in Section 5.3. Because the computer generated random number plays the fundamental role in the Monte Carlo method, we give some brief discussion in Section 5.4. The simulations are shown in Section 5.5, and finally we present some future topics on this problem.

5.1 Introduction

Mortgage-backed securities (MBS) are financial instruments that are financed by the cash flows from residential mortgage payments and commercial mortgage payments. It is an asset-backed security whose cash flows are backed by the principal and interest payments of a set of mortgage loans. Payments are typically made monthly over the lifetime of the underlying loans. Residential mortgagors in the United States have the option to pay more than the required monthly payment or pay off the loan in its entirety (prepayment). Because prepayment affects the remaining loan principal, the monthly cash flow of an MBS is not known in advance, and therefore presents an additional risk to MBS investors.

Commercial mortgage-backed securities (CMBS) are secured by commercial and multifamily properties (such as apartment buildings, retail or office properties, hotels, schools, industrial properties and other commercial sites). The properties of these loans vary, with longer-term loans (5 years or longer) often being at fixed interest rates and having restrictions on prepayment, while shorter-term loans (1-3 years) are usually at variable rates and freely prepayable.

The MBS market is one the fastest growing and one of the largest financial markets in the United States. As of June 2000, the MBS market amounted to almost 2 trillion US dollars, only second to the US Treasury issuance, which is about 3.5 trillion US dollars.

There are three major housing finance agencies, Ginnie Mae, Fannie Mae and Freddie Mac, in the US to provide MBS products in the secondary mortgage market. Ginnie Mae is part of the US government. Fannie Mae and Freddie Mac used to belong to the government, but now they are private entities. Although they do not have explicit US government guarantees, they continue to maintain close ties to the US government. Therefore, MBS products issued by those three agencies are considered to have no credit risk or to have negligible credit risk. Those three agencies provide the majority of the MBS market. However, there are also private-labeled MBS which are issued by other agencies and they do bear credit risk. In this chapter, we will only consider the MBS products issued by those three aforementioned agencies. In other words, we assume that there is no credit risk for the MBS products considered in this thesis.

Owners of MBS can receive cash flow from the principal and/or the interest of the mortgage payments. Usually, all mortgages have a given payment schedule and often allow prepayment without penalty. Therefore, all MBS bear interest risk and the prepayment risk. In addition to the classical MBS product, the pass-through, investors also want instruments that provide choices on a variety of maturity and prepayment profiles. Therefore, collateralized mortgage obligations (CMO) were introduced in the market first in 1983. In CMO's the total pool is divided into several tranches, with different prepayment and interest risk. Since then, CMO is more and more popular than the classical MBS pass-through.

Pricing MBS, especially the CMOs, has never been an easy task. Pricing a vanilla corporate bond is based on two sources of uncertainty: default risk (credit risk) and interest rate (IR) exposure. The MBS adds a third risk: early redemption (prepayment). The number of homeowners in residential MBS securitizations who prepay goes up when interest rates go down. One reason for this phenomenon is that homeowners can refinance at a lower fixed interest rate. Since these two sources of risk (IR and prepayment) are linked, solving mathematical models of MBS value is a difficult problem in finance. The level of difficulty rises with the complexity of the IR model, and the sophistication of the prepayment IR dependence, to the point that no closed form solution exists. In models of this type numerical methods provide approximate theoretical prices. These are also required in most models which specify the credit risk as a stochastic function with an IR correlation. While we have to take the interest risk and the prepayment risk into consideration, the price of a MBS or CMO is usually path dependent. Technically, the problem of pricing MBS is known as evaluating high dimensional integrals. The Monte Carlo method is the best way so far

to apply [18].

Before we talk more about Monte Carlo simulation, we would like to point out that, in practice, the evaluation of the Greeks associated with a MBS or CMO, such as Duration and Convexity, is also very important. Investors would like to know the Greeks of a MBS bond or CMO bond for hedging purpose. The pricing procedure is usually like the following: first, we use the market bond price to obtain the option-adjusted spread (OAS); then we use the implied OAS to evaluated the Greeks of the bond.

When we use the Monte Carlo simulation, we first generate the interest rate paths. The real cash flows can then be generated using the interest rate paths and certain prepayment schedule model. Once we have the cash flows, we can obtain the time 0 prices of the MBS or the CMO for the given paths. By taking average, we can obtain the price. As the number of paths increases, the average will converge to the theoretical price. This convergence is guaranteed by the Law of Large Numbers. Next, we compare it to the market price to find the OAS. Then, in the steps followed, we use this OAS as an input to obtain the prices when the interest is up or down by 100 base point. Then we can calculate the duration of the bond. Similar ideas can be used to obtain the convexity. This method has been a useful tool for many of these financial calculations, evidenced in part by the voluminous literature of successful applications. For a brief sampling, the reader is referred to the valuation of path-dependent options in Kemna [2], the valuation of mortgage-backed securities in Schwartz [26], the portfolio optimization in Worzel [39], the valuation of interest rate derivative claims in Carverhill [1], and in the stochastic volatility applications in Duan [37] and Hull [36].

One drawback of the Monte Carlo simulation is the slow convergence [11]. Usually, people use standard deviation to describe the corresponding accuracy. This is also the case in our study. To obtain the desired accuracy, 10^4 paths are needed usually and it can take hours to compute, which is a headache for practitioners. In this chapter, we are going to investigate the convergency of this procedure, and we find that, to obtain the desired accuracy of the Greeks (duration and convexity), usually hundreds of paths are enough. By adapting this method, the practitioners can save time up to 95%.

Before we go further, we would like to define two type of convergence: Absolute Convergence and Relative Convergence. For many practitioners, the final goal is to find the associated Greeks of the MBS bonds, while OAS is just an intermediate vehicle. So far, the typical method is to obtain the OAS first by Monte Carlo methods, and then calculate

the Greeks using the OAS obtained. People will use a big number of paths to achieve the desired accuracy for the OAS. Then the OAS can be used to obtain the Greeks with desired accuracy. In other words, both OAS and the Greeks converges in the similar degree. We call this **absolute convergence**. On the other hand, in our investigation, we find that it is possible that the Greeks converge faster than the OAS. In other words, even if the OAS does not have the desired accuracy, the Greeks may already have the desired accuracy derived from this OAS that are not so accurate. While the Greeks reaches the desired accuracy, but the OAS does not, we call this **relative convergence**.

Let us give a simple example to illustrate why this is possible. Assume that we want to estimate the expectation of $Y = f(X)$, which is a random variable of another random variable X. By virtue of the Monte Carlo method, we first generate $X_1, X_2, \cdots, X_N$ which are random realizations of X. Then we can estimate $E[X]$ and $E[Y]$ with

$$\hat{X} = \frac{X_1 + X_2 + \cdots + X_N}{N}$$

and

$$\hat{Y} = \frac{f(X_1) + f(X_2) + \cdots + f(X_N)}{N}$$

respectively. To estimate the accuracy, variance is usually used. For a given number N, it is possible that the two estimator, $\hat{X}$ and $\hat{Y}$ may have different convergence speed, i.e. their variance may have significant difference. For example, if $Y = f(X) = 0.5X$, then, we must have

$$Var(\hat{Y}) = \frac{Var(f(X))}{\sqrt{N}} = 0.25\frac{Var(X)}{\sqrt{N}} = 0.25Var(\hat{X}).$$

In other words, in terms of variance, the convergence speed of $\hat{X}$ is just one-fourth of the convergence speed of $\hat{Y}$.

Of course, the situation for OAS and the Greeks of MBS bonds are much more complicated than the example given above. However, as the example indicated, it is possible that the convergence speed of the Greeks is faster than the convergence speed of the OAS. That being said, if the values of Greeks are the final goals of the simulation, we may reach the desired accuracy for a relative small number of paths, even though the OAS may not be very close to the theoretical value.

Our main result in this chapter is that the relative convergence does exist for MBS and MBS derivatives. In other words, to achieve the desired accuracy of the Greeks, it is not necessary to achieve a good accuracy for OAS first. The error of OAS does not have

too much influence on Greeks. Usually, more than 10^4 interest rate paths are need to see the convergence of the OAS. However, according to our study, only hundreds of paths are needed to see the convergence of the duration and convexity, even though we use the not so accurate OAS as our input in the computation of duration and convexity. Given this results, practitioners can save an significant time of computation up to 95%.

5.2 Mortgage-backed Securities

5.2.1 MBS Bonds

Passthroughs MBS bonds are financed by cash flows from residential mortgage or commercial mortgage payments. There are many different types of MBS bonds. The basic MBS first introduced into market is the Pass-through. It represents a share of an investment pool consisting of multiple mortgages. In a pass-through MBS, an issuer collects monthly payments from homeowners and then passes on a proportionate share of the collected principal and interest to the investor. A pass-through bond holder faces the same interest risk and prepayment risk faced by the mortgage lenders.

Collateralized Mortgage Obligations A collateralized mortgage obligation is a security backed by a pool of pass-throughs, whole loans or stripped mortgage-backed securities. It is split into pieces called classes or tranches, which are assigned different priorities. Each tranche is treated as a separate security, the only distinction between them being how they are affected by prepayments. Prepayments contribute towards reducing the balance of the tranche with the highest priority first. When this tranche's balance is completely diminished, additional prepayments then go towards reducing the balance of the next tranche. Prepayment exposure is greatest for the highest priority tranche, and is smallest for the lowest priority tranche. Constructing a collateralized mortgage obligation in this way helps mitigate prepayment risk, and reduces uncertainty over the maturity of the mortgage-backed security. The development of collateralized mortgage obligations was motivated by extension and contraction risks. The former is the risk that, as interest rates rise, prepayments decline; while the latter is the risk that a fall in interest rates speeds up prepayments.

The tranches of a collateralized mortgage obligation can be structured in a number of ways. Sometimes a number of tranches with different structures will be represented in

a single collateralized mortgage obligation. This allows for tranches which accommodate different investor needs to be present in a single instrument.

Planed Amortization Class (PAC) bonds, introduced in August 1986, are the MBS that can produce more stable cash flows by redirecting prepayments from the underlying securities to other classes called companion or support classes [5]. They are a type of CMO bond, and are designed to largely eliminate prepayment risk for investors. They do this by transferring essentially all prepayment risk to other bonds in the CMO. Appropriately, those other bonds are called support or companion bonds.

PAC bonds offer a fixed principal redemption schedule that will be met so long as prepayments on the underlying mortgage collateral remain within a specific PSA (Public Securities Association which introduced a metric for projecting prepayments over the life of a pool.) range, which is called a prepayment protection band. Fixed principal payments are received over a predetermined schedule. For two prepayment rates which form a band or collar, the schedule is met only if the underlying securities prepayment rate is within the range. For faster prepayments, the companions will absorb the principal payments in excess of the PAC schedule in order to support it. On the other hand, if the prepayment rate is slow and there is not enough principal for the PAC, the amortization of the companions is delayed and as a consequence the average life of the class is extended. Another similar bond is called Target Amortization Class (TAC), which do not provide protection against extension of average life. They offer one-sided protection, shielding investors from high prepayment rates up to a specified PSA. They do not protect against low prepayment rates. TAC bonds are appealing because they offer higher yields than comparable PAC bonds. The unaddressed risk from low prepayment rates generally does not concern investors as much as risk from high prepayment rates, which the TAC structure does address.

Support Support Structure bonds are also called companion bonds that absorb the excess prepayments of PAC or TAC. It helps maintain the ability of PAC or TAC to stay on their schedules by absorbing excess prepayments or receiving less prepayments to prevent extension of average life.

In this chapter, we will consider three bonds: a Pass-through bond, a Planed Amortization Class (PAC) bond and a Support Structure bond. The bonds we considered are all market-traded bonds, and we will specify the names of the bonds in next section. Those bonds are choose randomly from market-traded bonds to test the robustness of our

approach. Later we will see that the numerical results for all three bonds do show similar convergence patterns which indicates the robustness of our approach.

5.2.2 MBS Pricing

The problem of pricing MBS is known as high dimensional integral, where the Monte Carlo method is the best way so far to apply. A standard MBS pricing framework would consists the following parts:1) Interest Rate model. 2) Prepayment model, which consists house turnover model and refinance model. 3) OAS model, which captures risk factors from the market price. The basic idea is to generate the future cash flow of the MBS and get the total present value of the future cash flow. Since the future cash flow depends on interest rates in the future with some uncertainty, the expectation of the total present value is used to give the price of the MBS. Therefore, we can use Monte Carlo simulation to estimate the expectation.

To generate the cash flow, we need to know the interest rate path and the prepayment schedule. Once the interest rate path and prepayment schedule are determined, we can price the MBS by calculate its the Net Present Value (NPV). However, the future interest rate and prepayment schedule are not known in advance. Usually, people have to use some models to predict the cash flows. This basically involves two fundamental models, interest rate model, which determines the discounting factor and the overall interest rate environment, and the prepayment model, which determines the prepayment behavior of the mortgage borrower as the interest rate changes [18]. Theoretical pricing models must take into account the link between interest rates and loan prepayment speed. Mortgage prepayments are most often made because a home is sold or because the homeowner is refinancing to a new mortgage, presumably with a lower rate or shorter term. Prepayment is classified as a risk for the MBS investor despite the fact that they receive the money, because it tends to occur when floating rates drop and the fixed income of the bond would be more valuable.

The interest rate model is usually stochastic to catch the randomness of the future. The prepayment model is also needed to predict the prepayment rate. The prepayment rate is highly correlated with the interest rate. Usually, when the interest rate is low, the borrowers may have stronger intension to refinance. Therefore, the prepayment will usually increase when the interest rate is decreasing. At the same time, historical data indicate that the prepayment rate also depends on other factors such as the mortgage age, season,

and burn-out. For more information about interest rate models and prepayment models, we refer the readers to Fabozzi [29] and Ivan [5].

Prepayment Model Mortgages have the unique characteristic of giving the borrower the right to prepay a mortgage at any time during the life of the loan. This right means the investor in a mortgage cannot know the maturity of the loan or the amount of interest that will be received with absolute certainty. There are three primary reasons for prepayment: homeowners may sell their properties; they may obtain cheaper financing; or they may elect simply to repay a larger portion than necessary of the outstanding balance. Prepayments are the main difficulty in pricing mortgage-backed securities, due to the uncertainty they introduce into the behavior of these instruments. For instance, a high prepayment rate will reduce the maturity of a mortgage-backed security.

There are four main factors that explain about 95% of prepayment behavior [72]: refinancing incentive (RI); burnout (BM-Burnout Multiplier); seasoning (AGE); and seasonality (MM-Month multiplier). A typical prepayment model comprises the above four factors, usually a weighted combination.

The refinancing incentive happens when the market mortgage rate drops below the rate of a particular home-loan, thus inducing the homeowner to prepay his existing loan and to refinance his property at the cheaper rate. It is generally expressed as the percentage of monthly savings that can be realized by refinancing. The refinancing incentive is measured by either taking the difference or the ratio between the mortgage coupon rate and the mortgage refinancing rate.

Seasoning or aging reflects the observation that newer loans tend to prepay slower than older loans. Prepayment rates are low after origination, and increase after some seasoning. When they level off, a pool of loans is considered to be fully seasoned. This industry convention adopted by the Public Securities Association models mortgage prepayment rates as increasing linearly from 0.2% CPR (Conditional Prepayment Rate) at issue to 6% CPR at thirty months and then remaining constant. An example is as follows

$$AGE(t) = \min\left(\frac{t}{30}, 1\right), \text{ where } t \text{ is in terms of months.}$$

Seasonality or month of the year takes into consideration the time of year. It is believed that prepayments peak in the summer and decrease in the winter. It declines in fall and winter with consequent low prepayment rates during the corresponding months.

One of the sources of prepayments due to seasonality is housing turnover, and this could be due to weather and school schedules.

Burnout is the tendency for prepayment to diminish over time, even when refinancing incentives are favorable. Refinancing burnout is directly linked to the path of the market mortgage rate-since if it falls to a particular level, rises and then falls to the same level again, there should be fewer prepayments the second time around. Factors leading to this effect include inadequate home equity to qualify for refinancing, and/or a negative change in the credit status of the borrower. Richard and Roll [72] suggest the more the prepayment option has been deep in-the-money, the more burned out the pool is, and the smaller prepayments are, all other things being equal.

There are four main types of prepayment functions [18][28]

- Arctangent Model: (An example from the Office of Thrift Supervision (OTS).)

$$CPR(t) = 0.2406 - 0.1389 \arctan\left(5.9518\left(1.089 - \frac{WAC}{r_{10}(t)}\right)\right),$$

  where WAC is the weighted average coupon.

- $CPR(S, A, B, M)$ Model

$$CPR(t) = RI(t)AGE(t)MM(t)BM(t);$$

- Prepayment models incorporating macroeconomic factors, i.e., the health of economics, housing market activity, etc.

- Prepayment models for individual mortgages.

For the last two types of prepayment models, we do not have any explicitly stated functional forms, mainly because they are proprietary models in the mortgage industry. But since our approach is general for any type of prepayment function, we can derive the derivatives once we are given an explicit form for the prepayment function.

Interest Rate Model For the purpose of valuating MBSs and CMOs, any arbitrage-free model of the term structure of interest rates can be used [5]. The general form of the interest rate models is described in terms of changes in the short rate,

$$dr_t = \kappa(\theta - r_t)dt + \sigma r^{\alpha} dB_t, r_0 \text{ given} \tag{5.1}$$

where dr_t represents an infinitesimal change in r_t, and B_t is a standard Brownian motion. κ is the speed of mean-reversion, θ is the long-run mean of the interest rate process, α is the proportional conditional volatility exponent, and σ is the volatility of r_t.

The Vasicek [82] model is an equilibrium model, which attempt to give an accurate account of the dynamics of the infinitesimal interest rate, but do allow for the possibility of arbitrage, in the sense that model values of some traded instruments may differ from their market values. This model is based on the Ornstein-Uhlenbeck (OU) process for the spot interest rate r_t. It is a one-factor model, where all rates ultimately depend on the shortest-term interest rate.

$$dr_t = \kappa(\theta - r_t) + \sigma dB_t.$$

The solution to this SDE is as follows

$$r_t = r_0 e^{-\kappa t} + \theta(1 - e^{-\kappa t}) + \sigma \int_0^t e^{-\kappa(t-s)} dB_s.$$

In order to simulate interest rates using this model, the discretized form is applied as follows

$$\Delta r = \kappa(\theta - r)\Delta t + \sigma B\sqrt{\Delta t}.$$

A drawback of this process is that negative interest rates can be simulated. In most simulations, absolute values of the Vasicek model are used.

The Hull and White [35] model is an extension of the Vasicek model, which assumes that the short rate is modeled by

$$dr_t = \alpha(\beta - r_t)dt + \sigma dB_t,$$

where α, β and σ are constants. Thus $r_t|_{t\geq 0}$ follows a mean-reverting process.

The CIR model considers an interest rate process of the type

$$dr_t = \kappa(\theta - r_t) + \sigma\sqrt{r_t} dB_t.$$

Volatility is now dependent on the level of interest rates. Interest rates will never reach zero if $\sigma^2 < \kappa\theta$. This was first pointed out by Beaglehold and Tenney [22] that r_t can still reach zero during simulation in discrete time.

The model of Dothan [48] increases the volatility exponent to 1.0, which considers the type

$$dr_t = \kappa r_t dt + \sigma r dB_t.$$

It does not include a mean reverting term in the drift. Courtadon [32] extends Dothan's model to include mean reversion, which is as follows

$$dr_t = \kappa(\theta - r_t)dt + \sigma r dB_t.$$

The Black-Karasinski (BK) model is a modification of the Hull and White model [35]. In this model, interest rates are lognormal,

$$r_t = exp\left(\log r_0 e^{-at} + \int_0^t e^{-at} b(u)du + \int_0^t \sigma e^{-at} dB_u\right).$$

This dynamics can be split as follows

$$r_t = exp(\alpha(t) + X_t),$$

where $\alpha(t)$ is a function of time

$$\alpha(t) = \log r_0 e^{-at} + \int_0^t e^{-at} b(u)du,$$

and X_t is a stochastic process satisfying

$$dX_t = -aX_t dt + \sigma dB_t, X_0 = 0,$$

and the solution is

$$X_t = X_s e^{-a(t-s)} + \int_s^t \sigma e^{-a(t-u)} dB_u.$$

The first stage of the procedure consists in constructing a trinomial tree for X_t and then use $\alpha(t)$ to displace the tree nodes so as to retrieve the initial zero coupon curve. Let define T as the time horizon of our tree and fix evaluation times $0 = t_0 < t_1 < t_2 < \cdots < t_N = T$. Set $\Delta t_i = t_{i+1} t_i$ for each i. The time instants are not necessary equally spaced. The tree nodes are indexed by (i, j). Where the time index i runs from 0 to N and the index j ranges from a lower bound lb_i to an upper bound ub_i . Let $M_{i,j}$ and V_t^2 be defined by:

$$\begin{aligned} M_{i,j} =& E(X_{t_i+1}|X_{t_i} = x_{i,j}) = x_{i,j} e^{-a\Delta t_i} \\ V_i^2 =& Var(X_{t_i+1}|X_{t_i} = x_{i,j}) = \frac{\sigma^2}{2a}(1 - e^{-2a\Delta t_i}) \end{aligned}$$

If we set $x_{i,j} = j\Delta x_j$, then

$$\Delta x_i = V_{i-1}\sqrt{3} = \sigma\sqrt{\frac{3}{2a}(1 - e^{-2a\Delta t_{i-1}})}.$$

Since we are interested in the performance of our pricing method, we just pick one interest model, say BK model in our simulation. The comparison of different interest models will be another big topic. For a given interest rate model and prepayment model, we can use Monte Carlo method to price the MBS. First, we generate a number of paths of interest rate. Based on the interest rate, we can determine the prepayment schedule based on the prepayment model. Secondly, once the interest rate and the prepayment schedule are given, we can generate the cash flows. Discounting the cash flows back to present time using the same interest rate path, and we can obtain the net present value as a price for the given interest rate and prepayment schedules. For each interest rate path, we do the same. Finally, we take the average as the price of the MBS. We can proceed as the follows.

Let P be the price of the MBS, V be the value of the MBS, which is a random variable, $PV(t)$ be the present value for cash flow at time t, and $df(t)$ be the discounting factor at time t, then under the risk neutral probability measure,

$$P = E[V] = E\left[\sum_{t=0}^{M} PV(t)\right] = E\left[\sum_{t=0}^{M} df(t)cf(t)\right], \tag{5.2}$$

where $cf(t)$ is the cash flow at time t, M is the maturity of the MBS.

The discounting factor is found from the short-term (risk-free) interest rate process

$$df(t) = \Pi_{k=1}^{t} \frac{1}{1+r_k},$$

where r_k is the short term rate generated from one of the interest rate models discussed.

Since we are generating cash flows for an MBS and not just a risk-free zero coupon bond, generating $cf(t)$ is more complicated as the cash flows depend not only on interest rates, but also on prepayment behavior. The cash flows can be calculated using the following formulas [28][5]

$$\begin{aligned} cf(t) &= MP(t) + PP(t) = TPP(t) + IP(t); \\ MP(t) &= SP(t) + IP(t); \\ TPP(T) &= SP(t) + PP(t); \end{aligned}$$

where

$MP(t)$ is the scheduled mortgage payment for period t

$TPP(t)$ is the total principal payment for period t

$IP(t)$ is the Interest payment for period t

$SP(t)$ is the scheduled principle payment for period t

$PP(t)$ is the principal prepayment for period t,

where t denotes the month for all variables.

To evaluate cash flows, the only uncertainty lies with calculating $PP(t)$. This is calculated using $CPR(t)$, which was derived from our prepayment model.

Let $MB(t)$ be the mortgage balance in the beginning of month t, WAC be the weighted average coupon rate and WAM be the weighted average maturity for the MBS, then the formula for computing $MP(t)$ is

$$MP(t) = MB(t)\left(\frac{WAC/12}{1-(1+WAC/12)^{-WAM+t}}\right),$$

and the formula for $IP(t), PP(t)$ are given as follows

$$IP(t) = MB(t)\frac{WAC}{12}$$
$$PP(t) = SMM(t)(MB(t) - SP(t)),$$

where

$$SMM(t) = 1-(1-CPR(t))^{1/12}, \text{ and } SP(t) = MP(t) - IP(t).$$

The reduction in the mortgage balance for each month is given by

$$MB(t+1) = MB(t) - TPP(t).$$

In the Monte Carlo simulation, several paths of cash flows are generated, then by strong law of large numbers, the MBS price is given by

$$E[V] = \lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N} V_i,$$

where V_i is the calculated present value on path i. More details can be found in [18].

5.2.3 Option Adjusted Spread Analysis

The valuation model gives theoretical price of a MBS security. While we use the Monte Carlo simulation, the theoretical price is the average of the total present values of the cash flows under a number of interest rate scenarios. In practice, the bond market participants tend to compare the yield spread instead of the bond prices in dollar term–a cheaper bond means a higher yield and vice versa. The Option Adjusted Spread (OAS) for an MBS bond is a constant spread that when added to the forward yield curve will make the theoretical value equal to the market value. The spread is referred as an 'option-adjusted' spread for that the spread takes the embedded option into the consideration. Please refer to Fabozzi [29] for more information about OAS.

OAS measures the yield spread that is not directly attributable to the fixed income characteristics. It nets out the impact of prepayments of MBS. It is used to evaluate the relative cheapness or richness of securities with imbedded options [5]. When the option component is removed from the bond's total spread, the bonds can be compared primarily on the credit risk.

Because of the imbedded options, the calculation of definite maturity date of the bond and yield-to-maturity become impossible. OAS provides a way measuring the return of a bond by comparing it with the risk-free term structure of the interest rates. It provides a means of comparing bonds with different cash flow characteristics. Equation (5.3) represents the discounting factor modified for OAS. By using this discounting factor for $df(t)$ and the current market price P, equation (5.2) can be solved for OAS[5]. See [54] for an idea solving OAS.

$$df_{OAS}(t) = \Pi_{k=1}^{t} \frac{1}{1 + r_k + OAS}. \tag{5.3}$$

In other words, when we apply the interest rate and prepayment model to price an MBS, it is generally not in agreement with the market price. In order to adjust our model, such that it could produce a price equal to market price, we need to introduce the option-adjusted spread (OAS). If we believe our prepayment model is accurate, i.e., the cash flow is correct, in order to change the price, we can only adjust the discounting factor, and that is exactly how OAS plays the role [18]. OAS could be viewed as excess return beyond the risk free return, adjusted for the prepayment option. It captures the return required by the investor community, to compensated for risks associated with MBS, after adjustment for prepayment. It could be viewed as the premium for a tiny portion of credit risk from

the MBS issuer, and the model uncertainty in the pricing framework, or market liquidity premium.

We first study the convergence property of these random number generators in calculating OAS of the MBS. That is, based on a given market price of the MBS, we calculate the OAS using different random number generating methods with different number of paths in the Monte Carlo simulation. Then using the OAS we obtained, we price the MBS with different interest rate shifts. For example, use the OAS obtained by MT-BM with 500 paths, calculate the MBS price with interest rate shifting up and down by 100 base points, still using MT-BM with 500 paths. In this way, we are able to compute the effective duration and convexity of the pricing. The duration and convexity are important because from them we can tell how the MBS price will change if the interest rate changes.

It turns out that, at several hundred of paths where the calculations of the OAS have not shown good convergence property, some random number generators have already shown very good convergence in calculating the effective duration and convexity, and the MBS pricing can thus be much improved. We will discuss the details of Monte Carlo method in Section 5.3.

5.2.4 MBS Greeks

In practice, the Greeks of the MBS, especially the effective duration and the convexity, are needed for hedge purpose. The effective duration gives the estimation of how much the portfolio price or the bond price will change if the interest rate term structure curve shift for a small amount. In other words, the duration measures the risk of small yield change of the bond or the portfolio. If an investor wants to hedge the risk of small yield change for an interest-rate-dependent asset, he can use some other interest derivatives such as future to build a portfolio whose entire duration is 0. This is called the duration hedge.

In addition, for a relative large shift of the yield curve, the duration hedge may not give a satisfactory result. In addition, we need to build a portfolio such that both the duration and the convexity are hedged. Mathematically, the convexity is the second order derivative of the asset price to the yield. It measures the change of the duration of the asset when the yield curve shift for a small amount. For more details about duration, convexity and the hedge strategy using duration and convexity, we refer the readers to Hull [35] or Fabozzi [29].

The duration of a bond is a measure of how long on average the bond holder must wait for payment, or in other words, its a measure of the asset's price sensitivity to a change in the yield. Let D denote the duration, P be the bond price, and suppose the interest rate r is a continuous stochastic process, then $P = \sum_{i=1}^{n} c_i e^{-rt_i}$, where n is the number of payments, c_i is the cash flow at time t_i. The duration is defined by

$$D = \frac{\sum_{i=1}^{n} t_i c_i e^{-rt_i}}{P} = \sum_{i=1}^{n} t_i \left(\frac{c_i e^{-rt_i}}{P} \right),$$

which is just the weighted average of time to payments.

Notice that

$$\frac{dP}{dr} = -\sum_{i=1}^{n} c_i t_i e^{-rt_i} = -PD,$$

so

$$\frac{dP}{P} = -Ddr,$$

which means the percentage change in the bond price is the change in yield multiplied by the bond duration. The accuracy of this relation is guaranteed only for small values of dr.

In practice, when we use a valuation model to price the MBS bonds, the effective duration can be approximately obtained as the following:

$$D \triangleq Duration = \frac{P_- - P_+}{2P_0 \Delta r}, \tag{5.4}$$

where P_0 is the initial value of the bond; P_- is the estimated value of the bond if the yield is decreased by Δr; P_+ is the estimated value of the bond if the yield is increased by Δr and Δr is the change in the bond yield which is usually very small.

Similarly, the convexity of the bond is

$$C = \frac{1}{P} \frac{d^2 P}{dr^2} = \frac{\sum_{i=1}^{n} c_i t_i^2 e^{-rt_i}}{P},$$

which measures how fast dP/dr is changing with the size of dr. Use the second order Taylor series, it can be written as

$$\frac{dP}{P} = -Ddr + \frac{1}{2} C (dr)^2.$$

One can use convexity to improve a hedge and immunize against fairly large parallel shifts in the zero curve. It can be estimated by the following formula:

$$C \triangleq Convexity = \frac{P_- + P_+ - 2P_0}{P_0 (\Delta r)^2}. \tag{5.5}$$

The procedure is as the follows:

1. Use the valuation model (interest rate model, prepayment model,etc) and the current market price P_0 to obtain the OAS.

2. Use the obtained OAS in the valuation model to estimate the bond prices P_- and P_+ when the yield is decreased or increased by Δr, respectively.

3. Use the formula (5.4) to obtain the estimated duration and convexity.

While Monte Carlo Method is used, the OAS is first obtained with some accuracy. Usually, to obtain the desired accuracy, we need to use about 10^4 interest rate paths, which takes a lot of time. Then, the OAS is used to obtain the Greeks. We notice that all of the above work are performed to study the absolute convergence of the Monte Carlo simulation. The smallest number of paths needed is about 10^4. If the interest rate changes, the MBS needs to be priced again. In real time trading, faster computation is always preferred. In our work, in stead of studying the absolute convergence of the MBS pricing, we examine the relative convergence of the pseudo random number generators through Option Adjusted Spread (OAS) analysis, presented in Section 5.1.

5.3 Monte Carlo Simulations

Monte Carlo method is one of the most popular numerical methods in finance. In the problems of option pricing, securities valuation, risk analysis, etc., the Monte Carlo method has proved to be a successful tool. For a brief summary of its recent advances, the reader is referred to Boyle, Broadie and Glasserman [11].

The basic idea of Monte Carlo method is to simulate several sample paths of the underlying asset prices or interest rates under the risk-neutral measure, then evaluate the discounted cash flows according to the specific structure of the securities, and average these values over the sample paths. It is also known, however, that the main drawback of the Monte Carlo method is its slow convergence rate. In the case of conventional Monte Carlo method, the convergence speed is about $1/\sqrt{N}$ by the central limit theory, where N is the number of sample paths. Sometimes, this computing time is so long that can be intolerant for problems with complex stochastic models or more accurate solution requirements, such as pricing MBS.

The problem of pricing MBS is known as high dimensional integral, where the Monte Carlo method is the best way so far to apply. In this work, we consider the pricing

problems for three MBS bonds by virtue of the Monte Carlo Method. Those bonds are: a 10-year PAC (Planned Amortization Class) (CUSIP: 31396UY46), a support (CUSIP: 31397HLJ5) and a pass-through bond (FNCL 6.0). The OAS analysis is performed. As we mentioned earlier, once we know the interest rate paths and the prepayment schedules, the cash flows can be determined. So we can get the theoretical prices of these bond. Using the market data to calibrate our models, we can get the OAS. Then the OAS can be used to price the bonds when the interest rate is up or down by a small amount. The bond prices can then be used to calculate the durations and the convexities of the bonds.

Usually, people require that the OAS obtained achieves a certain accuracy first, before it can be used in the second steps. The convergence of the Monte Carlo simulation mainly depends on the number of paths, and it also depends on the random number generator and the stochastic generator for the interest rate process.

We first use these random number generators to price the OAS of these bonds with eight initial seeds, then based on the OAS obtained, we compute the Effective Duration and Convexity of the MBS pricing. These results are important as they tell us how the MBS prices change as the interest rates change.

Relative studies indicate that the accuracy of 1 base point can be reached by virtue of Monte Carlo simulation using 10000 paths. In our study, we use the values obtained using 10000 paths as the 'accurate' values.

For each calculation, the standard deviation σ of the pricing using these eight seeds provides a good convergence criterion, because it tells us the spread, of dispersion of the pricing at certain number of paths in the Monte Carlo simulation. Therefore, to measure the degree of convergence, for a given number of paths, we calculate the standard deviation of the results obtained using 8 different random seeds.

In our simulation, there are two method to calculate the standard deviation. As we just mentioned, the values we obtained by virtue of Monte Carlo Simulations using 10000 paths can be regarded as the 'accurate' values. So we can use the 'accurate' values as our average values when we calculate the standard variation for the given number of paths using eight random seeds. Mathematically, we can do the following: For fixed random number generator, let P_p^s be the pricing result using p paths and initial seed s, then let

$$\bar{P}_{10000} = \frac{1}{8} \sum_{s \in S_s} P_{10000}^s$$

be the convergent point, and for each number of paths p, calculate the standard deviation

$$\sigma_p = \sqrt{\frac{1}{7}\sum_{s\in S_s}(P_p^s - \bar{P}_{10000})^2}.$$

We call this method I.

The other way is to calculate the usual standard deviation as

$$\sigma_p = \sqrt{\frac{1}{7}\sum_{s\in S_s}(P_p^s - \bar{P}_p)^2},$$

where

$$\bar{P}_p = \frac{1}{8}\sum_{s\in S_s} P_p^s.$$

In other words, for the given number of paths, we use the average of the values coming from eight random seeds, instead of the values obtained using 10000 paths, to calculate the standard variation. This way is called method II.

As we will see later, the numerical results indicate that the convergence patterns are very similar when we use two different methods as we mentioned above to measure the convergence speed. More importantly, we found that by generating several hundred of paths in the Monte Carlo simulation, even though the OAS pricing does not converge well, the computation of the Effective Duration and Convexity have already showed very good convergency properties.

5.4 Pseudo Random Number Generators

The random number plays the key role in Monte Carlo simulations. Computers can not generate real random numbers, since they generate numbers in a deterministic fashion based on some arithmetic algorithm. If the sequence has no or tolerable correlation, then this computer generated sequence is called pseudo random.

There are several widely used random number generators that are used to generator uniform distributions over $[0, 1]$. The NR1(Ran1) in [69] is the usual linear congruent generators with the Minimal Standard presented by Park and Miller [66]. It was found that this algorithm has a period round 10^9. Later NR2(Ran2) [69] was often used, which has a period more than 10^{18}. NR3 (Ran3) [69] is a subtractive method with a portable routine [40][38]. Zeeb and Romero [89] performed a test to study several random number generators,

where Ran2 and Ran3 passed all the tests. It is also known that Ran2, which combines two linear congruent generators and does shuffling, is a very slow algorithm compared with Ran3 [89]. Mersenne Twister (MT) is a very recently developed pseudo random number generator [58]. It is based on a matrix recurrence over a finite binary field F_2, which provides for fast generation of very high quality pseudo random numbers. Its period can be as large as $2^{19937} - 1$, and it has very high order of dimensional equi-distribution.

There is another brunch of random number generating methods, called quasi random numbers or low discrepancy sequences, like Sobol, Halton, etc., which are actually sequences of deterministic points. The reader may refer to Niederreiter [60] for the details of random number generations and some low discrepancy sequences, and Boyle [11] for a brief introduction of quasi-Monte Carlo methods. Although it is suggested that [60] low discrepancy sequences are sometimes superior to conventional Monte Carlo algorithms, in the problem of pricing MBS, which involves high dimensional integrals, different conclusions may be drawn under different situations. Morokoff and Caflisch [87] noticed that the advantage of low discrepancy sequences disappears for problems with dimension around 30. Paskov and Traub[67] applied the Sobol and Halton points to evaluate a CMO, which is a problem with dimension up to 360. They used the pseudo random generators ran1 [68] and RAN2 in [69] for the Monte Carlo simulation, and the convergence for the Sobol method was observed for the number of paths around 2e+5.

Ninomiya and Tezuka [62] examined the low discrepancy sequences such as Halton, Sobol, Faure and generalized Niederreiter sequences and compared them with Monte Carlo methods using a generator called combined Tausworthe random number generator [79]. They found that the convergence performances of classical low discrepancy sequences such as Sobol, Faure and Halton are worse than those of the random numbers. In this problem, their results showed that in order to achieve 1 base point accuracy, generalized Faure and generalized Sobol sequences need about 2e+5 sample paths, where classical low discrepancy sequences such as Halton, Faure and Sobol need over 1e+8 sample paths, and the random number sequences needs about 3e+7. Similar conclusions are obtained in their second experiment which evaluated a look back option. An explanation for this verdict mentioned by Boyle et al [11] is that, the integrands typically found in financial applications behave better than those used by numerical analysts, and financial applications often involve discounting, which may effectively reduce the dimensionality.

We need to mention that all of the above work are performed to study the absolute

convergence of the Monte Carlo simulation. The smallest number of paths needed is about 10^4. Since we are interested in finding out the relative convergence property of our pricing method, in this chapter, we will use Ran3 (NR3) and MT as our random number generator. Numerical experiments using Ran1 and Ran2 gave us similar results to the results using Ran3, so we omit Ran1 and Ran2 here.

Those generators are used to generate random variables that are uniformly distributed over $[0, 1]$. To convert these pseudo uniform random numbers into pseudo normal random numbers, we considered two methods, namely Box-Müller (BM)[69] and inverse cumulative distribution function (hereafter referred as INV).

In summary, to test the robustness of our method, we consider the combinations of two random number generators, NR3 and MT, and two methods to convert uniform random variables normal random variables, BM and INV. The combinations referred as NR3-INV, NR3-BM, MT-INV and MT-BM. As we will see later, our method does work well for all these combinations.

5.5 Numerical Results and Analysis

In our numerical study, we randomly choose three MBS bonds, one from each category of three categories: PAC, Support and Pass-Through. Those bonds are traded on the US market and they are: a 10-year PAC (CUSIP: 31396UY46, referred as PAC hereafter), a Support bond (CUSIP: 31397HLJ5, referred as Support hereafter) and a Pass-Through bond (FNCL 6.0, referred as FNCL hereafter). The interest rate model we used is the Black-Karasinski (BK) model [35].

As we mentioned earlier, to test the robustness of our method, we consider four pseudo random number generating combinations, MT-BM, MT-INV, NR3-BM and NR3-INV

Each pricing has eight seeds
$S_s = \{-26810, -6108, -30650, -90254, -66368, -50401, -97973, -36457\}$,
which are generated using the random number generator in MS-Excel.

For a given number of paths, we use the Monte Carlo method to obtain eight values for eight difference random seeds mentioned above. Then the standard variation for the given number of paths will be calculated using the eight values obtained and the method I or the method II mentioned in Section 5.3. This procedure will be repeated for all three

market traded bonds, PAC, Support and FNCL, and for all four pseudo random number generators, MT-BM, MT-INV, NR3-BM and NR3-INV.

The numerical results are summarized in table 5.1 to table 5.8. Table 5.1 to Table 5.4 gives the convergence data by virtue of Method I for three bonds using four pseudo random number generators, MT-BM, MT-INV, NR3-BM and NR3-INV. And tables 5.5 to 5.8 give the result of Method II.

In those tables,the standard deviation (σ) is used as the convergence criterion, and the numbers of paths that are needed in order to achieve a smaller σ than the threshold are given in the tables.

5.5.1 Convergence of the OAS

For these three bonds we considered, we calculated the standard variation of the OAS using Method I and four random number generators. The number of paths we chose are

$$50, 100, 200, \ldots, 900, 1000, 2000, \ldots, 10000.$$

In order to make a better view in the graph, we scaled the label to the square root of the number of paths. The results are showed in the following figures.

Use the OAS obtained above, we price the MBS with interest rate shifting up or down by 100 base points. Then we are able to estimate the sensitivity of the MBS price against the changes in interest rate, which is described by the Duration and the Convexity. This will also help determine where the value goes and where the risk goes in the MBS structure [5]. The duration and convexity results will be shown in Section 5.5.2. The following are the bond prices with shifted interest rates.

We see from these figures that although the OAS computations have not shown good convergence, the bond prices with shift interest rate, calculated based on the OAS obtained, have already shown good convergence properties when the number of paths are around one thousand (or the square root around 31 in the figures). The early convergence of the bond prices with shifted interest rate guarantees the convergence of the duration and convexity, which are shown next.

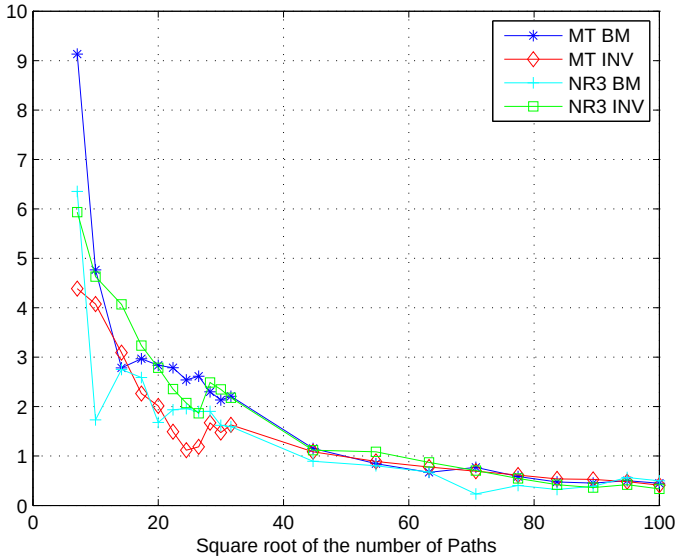

Figure 5.1: PAC OAS-Convergence using Method I

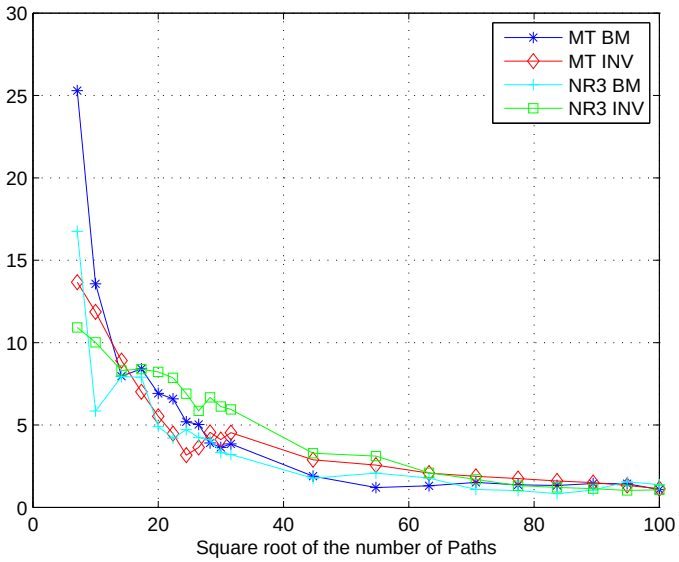

Figure 5.2: Support OAS-Convergence using Method I

5.5.2 Greeks

The effective duration measures the price sensitivity of the MBS with respect to the changes of interest rates. This can be taken as the first derivative of the MBS price

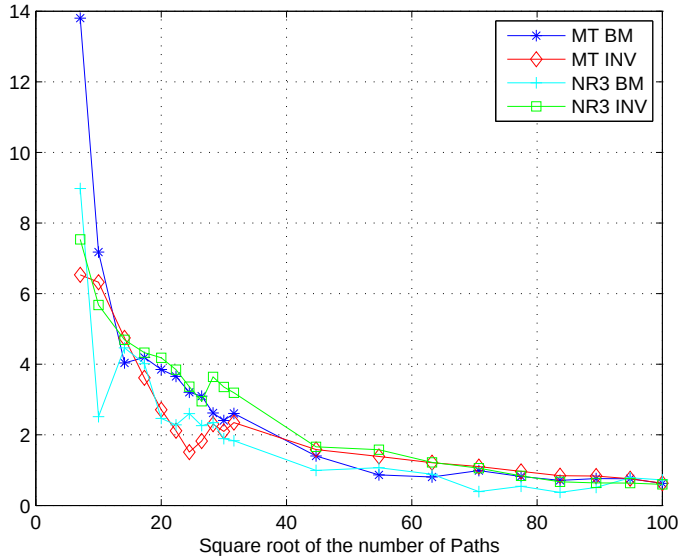

Figure 5.3: FNCL OAS-Convergence using Method I

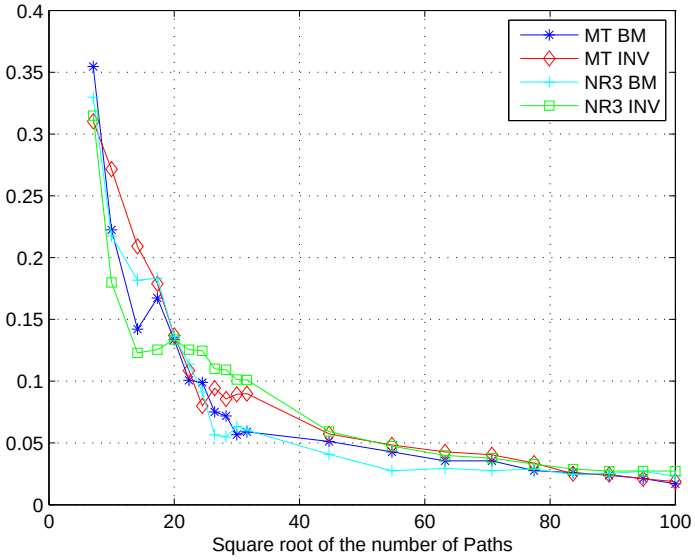

Figure 5.4: PAC Price with Interest Rate Shifted Down by 100bps-Convergence using Method I

w.r.t. interest change.

Convexity is the second derivative of the MBS price w.r.t. interest rate change. It

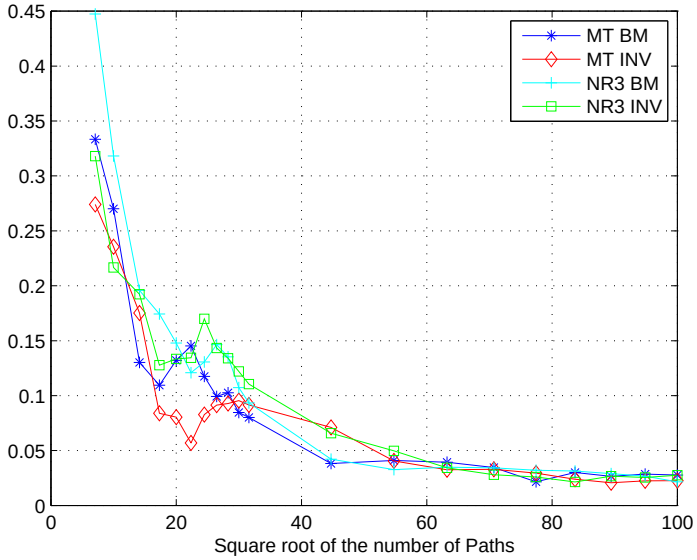

Figure 5.5: Support Price with Interest Rate Shifted Down by 100bps-Convergence using Method I

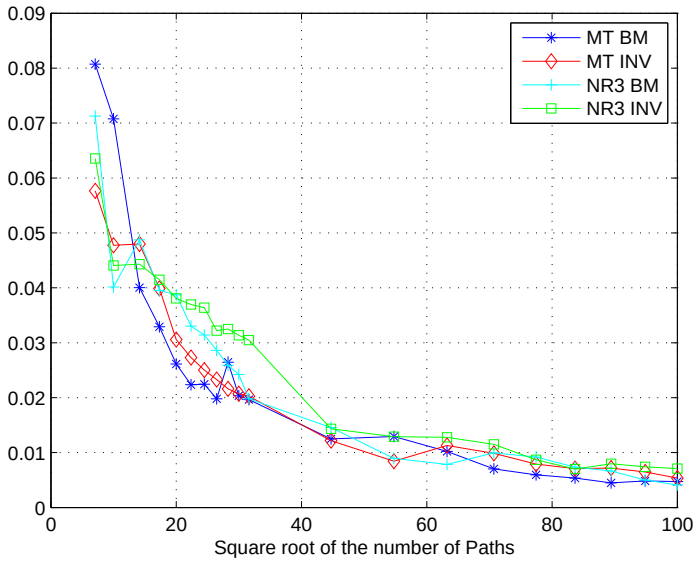

Figure 5.6: FNCL Price with Interest Rate Shifted Down by 100bps-Convergence using Method I

measures the sensitivity of the duration to the changes of interest rate.

The convergence properties of these random number generators in calculating the

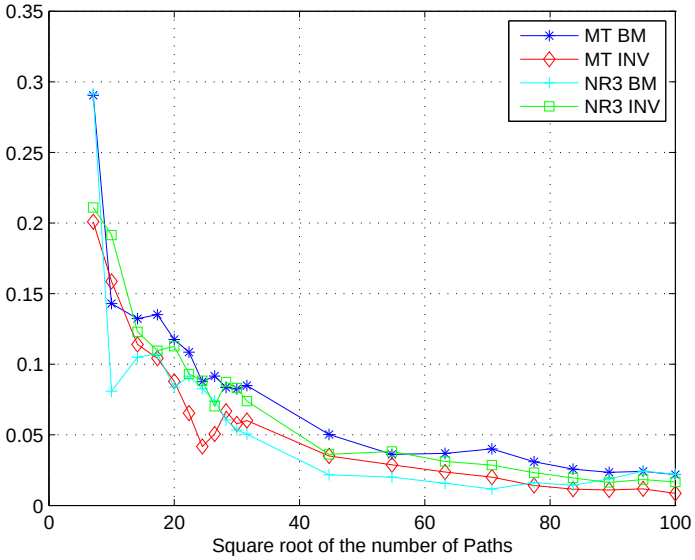

Figure 5.7: PAC Price with Interest Rate Shifted Up by 100bps-Convergence using Method I

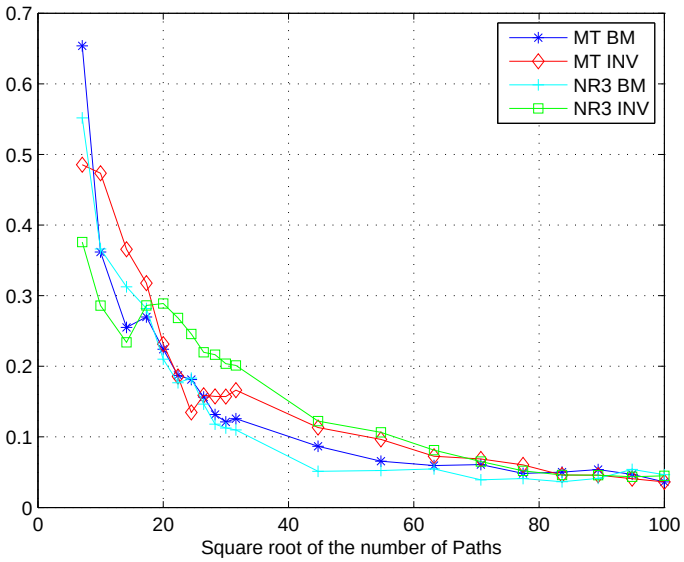

Figure 5.8: Support Price with Interest Rate Shifted Up by 100bps-Convergence using Method I

effective duration and convexity are given in the following.

From the figures, we can tell that although the OAS does show the convergence

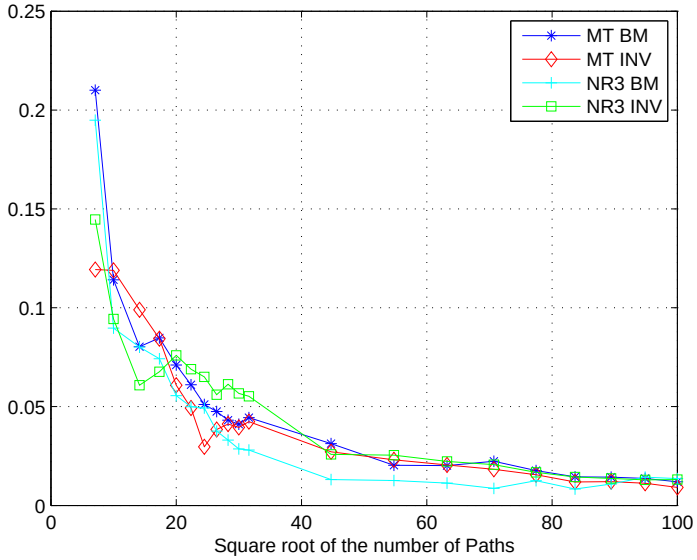

Figure 5.9: FNCL Price with Interest Rate Shifted Up by 100bps-Convergence using Method I

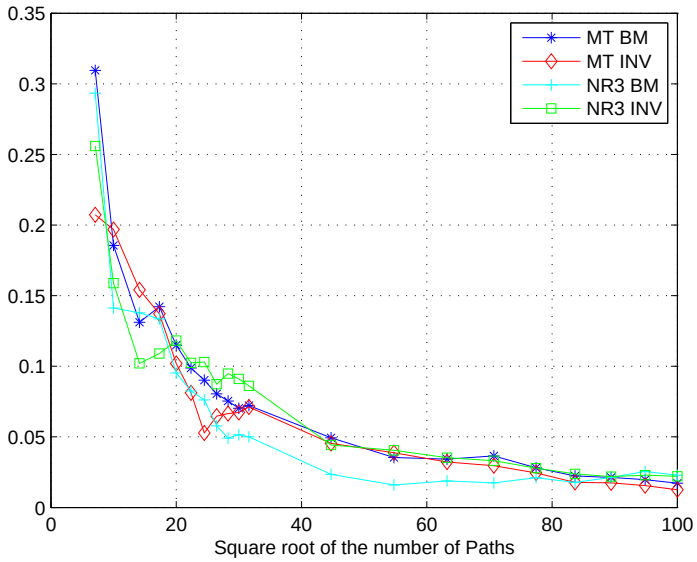

Figure 5.10: PAC Duration-Convergence using Method I

as the number of paths becomes bigger, the convergence speed is very slow if compared to the convergence of the Duration and the Convexity.

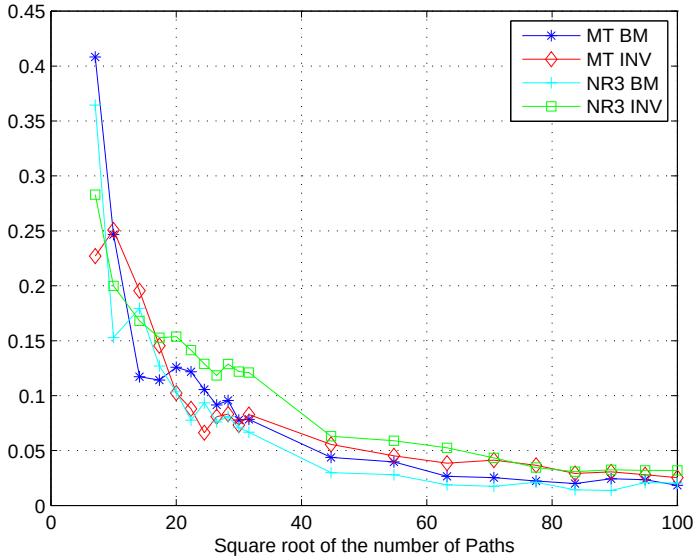

Figure 5.11: Support Duration-Convergence using Method I

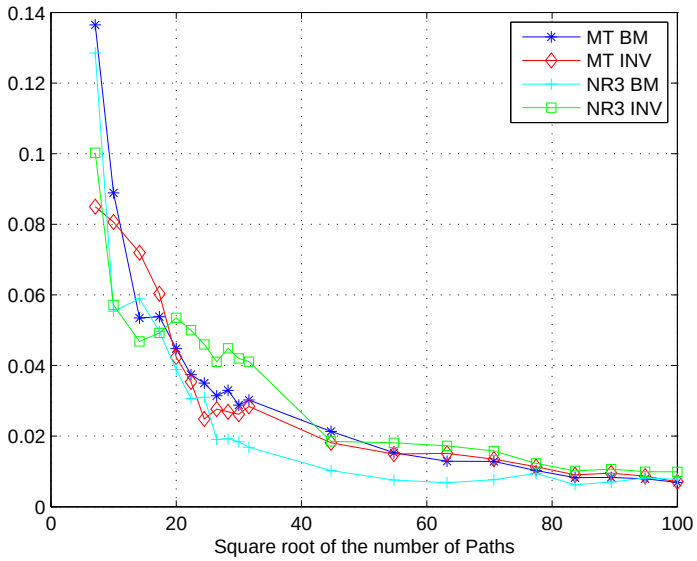

Figure 5.12: FNCL Duration-Convergence using Method I

The figures using Method II are similar to those of Method I, except a slight shift of the curves. In order not to make the chapter lengthy, we do not show them. The convergence analysis of Method I and Method II are given in Tables 5.1 to 5.8.

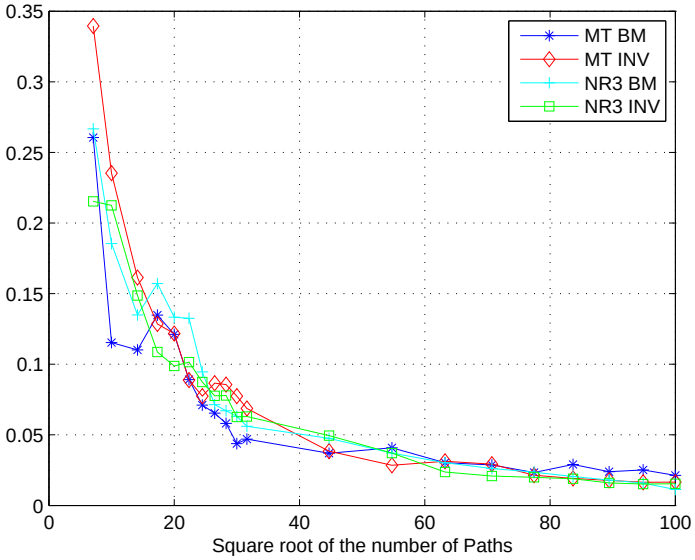

Figure 5.13: PAC Convexity-Convergence using Method I

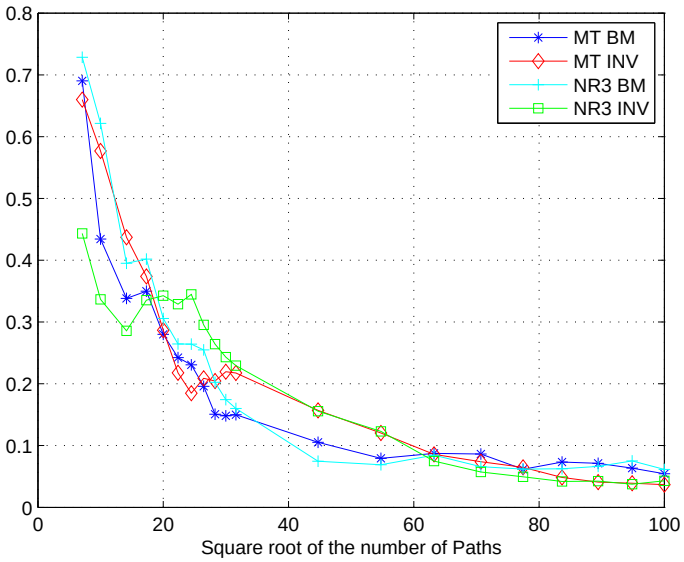

Figure 5.14: Support Convexity-Convergence using Method I

5.5.3 Relative Convergence and Absolute Convergence

As we can see from the tables, for all three bonds, the convergence speeds for Durations and Convexities are much faster than the OAS for the bonds. For examples,

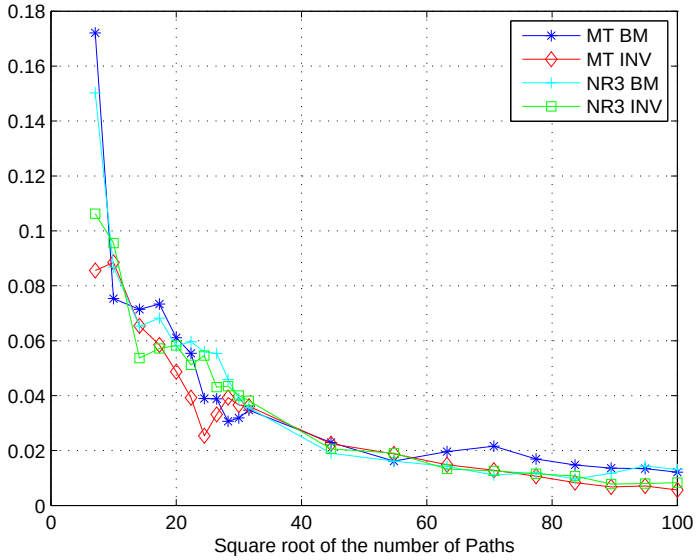

Figure 5.15: FNCL Convexity-Convergence using Method I

from Table 5.1, we can learn that, if we use the random generator MT-BM, to reach an accuracy of 0.1 for the OAS, i.e. $\sigma \leq 0.1$, the required number of paths are at least 10^4 for all three bonds we considered. On the other hand, under the same conditions, to reach the accuracy of 0.1 for Durations, the number of paths is much less than that for the OAS. From the table, we can see that, for the bond PAC, only 500 paths are needed to reach the accuracy of 0.1 for the Duration. For the bond Support, only 700 paths are needed. For the bond FNCL, only 100 paths are needed. If we look at the Convexities, to reach the accuracy of 0.1, the required numbers of paths for PAC, Support and FNCL are 500, 3000 and 100 respectively.

From Table 5.1, we can also learn that for different value of accuracy (standard variation) and for all three bonds we considered, the convergence speeds for Durations and Convexities are much faster than the convergence speeds for the OAS. In other words, for a given number of paths, if the number of paths is relatively small, the OAS may not perform the good convergence properties. Although the OAS obtained at this moment is not so accurate, we can still use it to calculate the Durations and the Convexities, which can show very good convergence properties even we use the same number of paths to calculate them.

To obtain a good approximative value for the OAS, we need to use many paths

(at least 10^4). Then we can use the OAS to calculate the Duration and the Convexity. There is no doubt that with the 'accurate' OAS and relative large number (10^4) of paths, we can get 'accurate' values for the Durations and the Convexity. This is the **Absolute Convergence**, which can always guaranteed by virtue of the Law of Large Numbers. But it takes a long time, which is not bearable in the industry.

However from our research, we found that there is a so-called **Relative Convergence** phenomenon exists for the MBS bond. In other words, for a give small number of paths, such as 1000, although the OAS still do not show very good convergence behavior, the Duration and the Convexity we obtained by virtue of the 'not-so-accurate' OAS can show very good convergence behavior. Therefore, the **Relative Convergence** does exist for the OAS and Greeks (Duration, Convexity) for MBS bonds.

Given our results, now practitioners in the industry can save tremendous of computation when they evaluate the Durations and the Convexities of the MBS bonds. For example, from Table 5.1, to reach an accuracy of 0.1 for the duration of the PAC bond, instead of using 10^4 paths to obtain a good value of the OAS, we can just use 500 paths to calculate the OAS and then use the OAS and 500 number of paths to obtain the Duration, which has our desired accuracy. Put differently, it can save the computation time by 95% in this particular example.

5.5.4 Robustness

In order to test the robustness of our method, we use four different random number generator combinations that are very popular among the industry. When Method I is used to calculate the standard variation, the results for four different random number generators are showed in Table 5.1 and Table 5.4. As we can see from the tables, the phenomenon of relative convergence can be seen for each random number generator, and the convergence patterns are very similar with each other.

In addition, we also use Method II to calculate the standard variations that are used to describe the measure of convergence. The results are given in Tables 5.5 to 5.8. From the tables, we can see that the convergence patterns are almost the same with those when we use Method I.

Therefore, it is safe to say that the relative convergence does exist for MBS bonds when we use the typical pseudo random number generators for our Monte Carlo Simulation.

Given this results, to calculate the Duration and the Convexity for a MBS bond, it is not necessary to use tens of thousands of paths to obtain an 'accurate' OAS first. In stead, in most of cases, several thousands or several hundreds of paths are enough to get very accurate values of the Duration and the Convexity.

Table 5.1: MT-BM, Method I

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	200	800	2000	7000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	200	500	2000	8000
	Convexity	50-	50-	50-	50-	100	500	900	10^4+
Support	OAS	2000	2000	5000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	200	700	2000	10^4+
	Convexity	50-	50-	50-	100	800	3000	10^4+	10^4+
FNCL	OAS	600	2000	2000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	100	400	4000
	Convexity	50-	50-	50-	50-	100	100	600	6000

Table 5.2: MT-INV, Method I

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	200	300	2000	9000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	300	400	2000	7000
	Convexity	50-	50-	50-	50-	300	500	2000	8000
Support	OAS	2000	3000	10^4+	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	300	400	3000	10^4+
	Convexity	50-	50-	50-	200	3000	4000	7000	10^4+
FNCL	OAS	400	500	3000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	50	400	2000
	Convexity	50-	50-	50-	50-	50	50	400	3000

5.6 Summaries and Further Topics

In this chapter, we investigated the relative convergence speeds of the OAS and the Greeks using the Monte Carlo simulation in pricing mortgage-backed securities, and propose the pricing procedure using OAS analysis. In our approach, only hundreds of paths are needed to obtain the desired accuracy of the Greeks, although the OAS may not reach its desired accuracies. The purpose of this research is to find a fast MBS pricing method, which is useful in real time trading. By using the OAS analysis, we speed up the computation of the duration and convexity in order to know the bond price change as fast as we can if

Table 5.3: NR3-BM, Method I

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	100	400	2000	5000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	100	400	1000	10^4+
	Convexity	50-	50-	50-	50-	400	600	2000	8000
Support	OAS	900	2000	5000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	300	400	2000	10^4+
	Convexity	50-	50-	50-	200	2000	2000	10^4+	10^4+
FNCL	OAS	400	700	2000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	100	300	700
	Convexity	50-	50-	50-	50-	50	100	800	2000

Table 5.4: NR3-INV, Method I

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	300	500	2000	6000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	200	700	2000	10^4+
	Convexity	50-	50-	50-	50-	200	500	2000	6000
Support	OAS	2000	4000	6000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	500	2000	4000	10^4+
	Convexity	50-	50-	50-	50-	3000	4000	6000	10^4+
FNCL	OAS	900	2000	4000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	50	500	2000
	Convexity	50-	50-	50-	50-	50	100	700	3000

Table 5.5: MT-BM, Method II

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	200	800	2000	7000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	200	500	2000	8000
	Convexity	50-	50-	50-	50-	50	500	700	10^4+
Support	OAS	2000	2000	2000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	200	200	2000	10^4+
	Convexity	50-	50-	50-	100	800	2000	10^4+	10^4+
FNCL	OAS	400	2000	2000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	100	300	2000
	Convexity	50-	50-	50-	50-	100	100	500	6000

Table 5.6: MT-INV, Method II

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	100	300	2000	9000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	300	400	2000	7000
	Convexity	50-	50-	50-	50-	200	500	2000	8000
Support	OAS	2000	3000	10^4+	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	300	400	2000	10^4+
	Convexity	50-	50-	50-	200	2000	4000	7000	10^4+
FNCL	OAS	400	500	3000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	50	400	2000
	Convexity	50-	50-	50-	50-	50	50	400	3000

Table 5.7: NR3-BM, Method II

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	100	100	1000	5000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	100	400	800	10^4+
	Convexity	50-	50-	50-	50-	400	600	2000	8000
Support	OAS	900	2000	5000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	300	400	2000	10^4+
	Convexity	50-	50-	50-	200	2000	2000	10^4+	10^4+
FNCL	OAS	400	500	2000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	100	300	700
	Convexity	50-	50-	50-	50-	50	100	800	2000

Table 5.8: NR3-INV, Method II

# of paths	$\sigma \leq$	3.5	2.5	1.5	0.5	0.15	0.1	0.05	0.02
PAC	OAS	300	500	2000	6000	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	100	700	2000	10^4+
	Convexity	50-	50-	50-	50-	200	400	2000	6000
Support	OAS	2000	4000	6000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	300	2000	4000	10^4+
	Convexity	50-	50-	50-	50-	3000	4000	6000	10^4+
FNCL	OAS	900	2000	4000	10^4+	10^4+	10^4+	10^4+	10^4+
	Duration	50-	50-	50-	50-	50	50	200	2000
	Convexity	50-	50-	50-	50-	50	100	700	2000

the interest rate changes. So to capture the bond price change with updated market data, especially the interest rate, is our goal. Therefore, it is important to test our pricing method with real market data. Future research should focus on testing the feasibility of our pricing approach when the interest rate changes, and compare the pricing result with real market data.

Chapter 6

Conclusions and Future Work

In this work we studied the path dependent stochastic models and their applications to finance and wireless communication systems. For the models where the analytical solutions are not known, we applied the numerical methods, namely, the controlled Markov Chain approximation and Monte Carlo simulation.

In the controlled Markov Chain approach, we have shown that by estimating the present system state, we found better control methods than the simple control which uses the delayed information directly in the approximation. We used two ways to estimate the system state. The first is to assume the system be controlled under the optimal control u^* which is derived from the no-delay system, and predict the present state using this control. Several issues remain. One is that, is the assumption that the system is controlled under u^* a proper assumption? This assumption is motivated from the separation principle used in deterministic control system, yet whether it is proper to apply in stochastic control systems with time delay still remains an interesting topic.

Later we present the idea to keep the control history in order to have an accurate prediction of the present state. Though the result is decent, it still remains a question that under what condition this policy is optimal for the discrete controlled Markov Chain model with time delay. Further questions are, how do these controls perform as the number of delay steps D increases? Do they still have large cost reductions or the reductions diminish as D increases? All such kind of problems are hard and need further study, especially the problem of dealing with computer memories in order to handle large delays.

We used the controlled Markov Chain approximation to solve stochastic control problems with time delay in continuous time. As the interpolation interval $h \to 0$, the

discrete results converge weakly to the continuous time solutions. However, this conclusion holds for the system with no-delay or the delayed system using the simple control [46]. We claim that the state estimation methods, do not alter the structure of the theoretical proof, as shown in Chapter 2, so we expect them to converge to certain continuous time control policies as well. However, it does need rigorous proof.

We have only considered the stochastic control problem with fixed time delay. The extended problem with general time delay is still a hard problem. We made the first try applying the numerical method solving such kind of problems in this area, yet future research should focus on finding the solutions of stochastic control problem with general time delay, and finding the analytical form inspired by the numerical results.

In the work of Monte Carlo simulations, we investigated the relative convergence speeds of the OAS and the Greeks in pricing mortgage-backed securities, and propose the pricing procedure using OAS analysis. In our approach, only hundreds of paths are needed to obtain the desired accuracy of the Greeks, although the OAS may not reach its desired accuracies. The purpose of this research is to find a fast MBS pricing method, which is useful in real time trading. By using the OAS analysis, we speed up the computation of the duration and convexity in order to know the bond price change as fast as we can if the interest rate changes. So to capture the bond price change with updated market data, especially the interest rate, is out goal. Therefore, it is important to test our pricing method with real market data. Future research should focus on testing the feasibility of our pricing approach when the interest rate changes, and compare the pricing result with real market data.

Bibliography

[1] Carverhill A. and Pang K. Efficient and flexible bond option valuation in the heath, Jarrow and Morton framework. *Journal of Fixed Income*, 5:70–77, 1995.

[2] Kemna A.G.Z. and Vorst A.C.F. A pricing method for options based on average asset values. *Journal of Banking and Finance*, 14:113–129, 1990.

[3] E. Altman, T. Başar, and R. Srikant. Congestion Control as a Stochastic Control Problem with Action Delays. *Automatica*, 35:1937–1950, 1999.

[4] Vlad Bally and Denis Talay. The Euler scheme for stochastic differential equations: error analysis with Malliavin calculus. *Mathematics and Computers in Simulation*, pages 35–41, 1995.

[5] Ivan Bandic. *Pricing Mortgage-backed Securities and Collateralized Mortgage Obligations*. Master thesis, University of British Columbia, 2004.

[6] Harald Bauer and Ulrich Rieder. Stochastic Control Problems with Delay. *Math. Meth. Oper. Res.*, pages 411–427, 2005.

[7] A. Bensoussan. *Stochastic Control of Partially Observable Systems*. Cambridge, 1992.

[8] P. Billingsley. *Convergence of Probability Measures,2nd ed.*

[9] P. Billingsley. *Convergence of Probability Measures*. Springer-Verlag, 1968.

[10] T.J.J. Van Den Boom and B.DE Schutter. Model predictive control for perturbed max-plus-linear systems: a stochastic approach. *INT. J. Control*, 77(3), 2004.

[11] Phelim Boyle, Mark Broadie, and Paul Glasserman. Monte Carlo methods for security pricing. *Journal of Economic Dynamics and Control*, 21:1267–1321, 1997.

[12] Richard A Brooks. Linear Stochastic Control: An Extended Separation Principle. *Journal of Mathematical Analysis and Applications*, pages 569–587, 1972.

[13] R. Buche and H.J. Kushner. Control of Mobile Communications With Time-Varying Channels in Heavy Traffic. *IEEE Transactions on Automatic Control*, 47:992–1003, 2002.

[14] R.T. Buche, A .P. Ghosh, and V. Pipiras. Heavy Traffic Scalings and Limit Models in a Wireless System with Long Range Dependence and Heavy Tails. *Preprint*, 2007.

[15] R.T. Buche and C. Lin. Heavy traffic control policies for wireless systems with timevarying channels. Cambridge, 1992.

[16] R.T. Buche, V.R. Ramezani, and Y. Yang. Power control in a wireless queueing system model with delay: an application of numerical methods for nonlinear stochastic systems. *Preprint*, 2007.

[17] Evelyn Buckwar. Introduction to the numerical analysis of stochastic delay differential equations. *Journal of Computational and Applied Mathematics*, pages 297–307, 2000.

[18] Jian Chen. Simulation-Based Pricing of Mortgage-Backed Securities. In *Proceedings of the 2004 Winter Simulation Conference, R.G. Ingalls, M.D. Rossetti, J.S. Smith, and B.A. Peters, eds.*, 2004.

[19] L. Chisci, A. Lombardi, E. Mosca, and J. A. Rossiter. State-space approach to stabilizing stochastic predictive control. *INT. J. Control*, 65(4), 1996.

[20] John C. Cox, Stephen A. Ross, and Mark Rubinstein. Option Pricing: A Simplified Approach. *Journal of Financial Economics*, pages 229–263, 1979.

[21] RUTH F. CURTAIN and AKIRA ICHIKAWA. THE SEPARATION PRINCIPLE FOR STOCHASTIC EVOLUTION EQUATIONS. *SIAM J. Control And Optimization*, (3):367–383, 1977.

[22] Beaglehold D. and M. Tenney. A Nonlinear Equilibrium Model of Term Structures of Interest Rates: Corrections and Additions. *Journal of Financial Economics*, 32:345–354, 1992.

[23] M. H. A. DAVIS. THE SEPARATION PRINCIPLE IN STOCHASTIC CONTROL VIA GIRSANOV SOLUTIONS. *SIAM J. Control And Optimization*, (1):176–188, 1976.

[24] Paul Dupuis and Hitoshi Ishii. On Lipschitz Continuity of The Solution Mapping to the Skorokhod Problem, with Applications. *Stochastics and Stochastics Reports*, pages 31–62, 1991.

[25] Ismail elsanosi, Bernt Øksendal, and Agnès Sulem. Some solvable stochastic control problems with delay. *Stochastics Stochastics Rep.*, 71(1,2):69–89, 2000.

[26] Schwartz E.S. and Torous W.N. Prepayment and the valuation of mortgage-backed securities. *Journal of Finance*, 44:375–392, 1989.

[27] D. W. Clarke et al. Generalized Predictive Control Part 1: The Basic Algorithm; Part 2: Extension and Interpretations. *Automatica*, 23(1), 1987.

[28] F. J. Fabozzi. *he Handbook of Mortgage-Backed Securities.* New York, NY: McGraw-Hill, 2000.

[29] Frank J. Fabozzi. *Fixed Income Mathematics, 2nd edn.* McGraw-Hill, 2006.

[30] Raúl Fierro and Soledad Torres. The Euler scheme for Hilbert space valued stochastic differential equations. *Statistics and Probability Letters*, pages 207–213, 2001.

[31] N.M. Filatov and H. Unbehauen. Adaptive Predictive Control Policy for Nonlinear Stochastic Systems. *IEEE Trans. On Automatic Control*, 40(11), 1995.

[32] Courtadon G. The Pricing of Options on Default-Free Bonds. *Journal of Financial and Quantitative Analysis*, 17:75–100, 1982.

[33] S.S. Ge and K.P. Tee. Approximation-based control of nonlinear MIMO time-delay systems. *Automatica*, 43:31–43, 2007.

[34] M. J. Grimble. Stochastic Control of Discrete Systems: A Separation Principle for Wiener and Polynomial Systems. *IEEE TRANSACTIONS ON AUTOMATIC CONTROL.*, (11):2125–2130, 1999.

[35] John C. Hull. *Options, Futures and Other Derivatives, 5th edn.* Prentice Hall, 2003.

[36] Hull J. and White A. The pricing of options on assets with stochastic volatilities. *Journal of Finance*, 42:281–300, 1987.

[37] Duan J.-C. The GARCH option pricing model. *Mathematical Finance*, 5:13–32, 1995.

[38] D. Kahaner, C. Moler, and S. Nash. *Numerical Methods and Software* . Englewood Cliffs, NJ: Prentice Hall, Chapter 10, 1989.

[39] Worzel K.J., Vassiadou-Zeniou C., and Zenois S.A. Integrated simulation and optimization models for tracking indices of fixed-income securities. *Operations Research*, 42:223–233, 1994.

[40] D.E. Knuth. *Seminumerical Algorithms,2nd ed.* The Art of Computer Programming, Vol 2, Chapter 3.2–3.3, 1981.

[41] G. Koole. A transformation method for stochastic control problems with partial observations. *Systems & Control Letters*, 35:301–308, 1998.

[42] Lisa A. Korf. Approximating infinite horizon stochastic optimal control in discrete time with constraints. *Ann Oper Res*, pages 165–186, 2006.

[43] Uwe Küchler and Eckhard Platen. Weak discrete time approximation of stochastic differntial equations with time delay. *Mathematics and Computers in Simulation*, pages 497–507, 2002.

[44] Harold J. Kushner. Numerical Approximations for Optimal Controls for Stochastic Systems With Delays. In *Proceeding of the 45th IEEE Conference on Decision and Control*, pages 2595–2602, Dec 2006.

[45] H.J. Kushner. Numerical Approximations for Nonlinear Stochastic Systems with Delays. *Stochastics*, 77(3), 2005.

[46] H.J. Kushner. Numerical Approximations for Stochastic Systems with Delays in the State and Control. *Stochastics*, 78(5):343–376, 2006.

[47] H.J. Kushner and P. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time,2nd ed.* Springer-Verlag, 2001.

[48] Dothan L. On the Term Stucture of Interest Rates. *Journal of Financial Economics*, 6:59–69, 1978.

[49] Bjørnar Larssen. Dynamic Programming in Stochastic Control of Systems with Delay. *Stoch.Stoch.Rep.*, 74(3,4):651–673, 2002.

[50] Bjørnar Larssen and Nils Henrik Risebro. When are HJB-equations in stochastic control of delay systems finite dimensional? *Stochastic Anal. Appl.*, 21(3):643–671, 2003.

[51] D. Lépingle. Euler scheme for reflected stochastic differential equations. *Mathematics and Computers in Simulation*, pages 119–126, 1995.

[52] Chuan Lin. *Heavy Traffic and Markov Modulated Models for Wireless Queueing Systems and Numerical Methods for Associated Resource Allocation Problems.* Ph.D. Thesis, NC State University, Dec. 2006.

[53] Anders Lindquist. On Feedback Control of Linear Stochastic Systems. *SIAM J. Control*, (2):323–343, 1973.

[54] Jian-Guo Liu and Eugene Xu. Pricing of Mortgage- Backed Se cu ri ties with Option-Adjusted Spread. *Managerial Finance*, 24(9/10):94–109, 1998.

[55] Vigirdas Mackevičius. Convergence rate of Euler scheme for stochastic differential equations: Functionals of solutions. *Mathematics and Computers in Simulation*, pages 109–121, 1997.

[56] Bogdan Marinescu and Henri Bourlés. Robust state-predictive control with separation property: A reduced-state design for control systems with non-equal time delays. *Automatica*, 36:555–562, 2000.

[57] Giovanni B. Di Masi and Wolfgang J. Runggaldier. An Approach To Discrete-Time Stochastic Control Problems Under Partial Observation. *SIAM J. Control And Optimization*, (1):38–48, 1987.

[58] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Model. Comput. Simul.*, 8(3):1192–1201, 1998.

[59] T. Mikosch, S. Resnick, H. Rootzén, and A. Stegeman. Is Network Traffic Approximated by Stable Levy Motion or Fractional Brownian Motion? *The Annals of Applied Probability*, 17:23–68, 2002.

[60] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. CBMS-NSF 63, SIAM, Philadelphia, Pa, 1992.

[61] Johan Nilsson, Bo Bernhardsson, and Björn Wittenmark. Stochastic Analysis and Control of Real-time Systems with Random Time Delays. *Automatica*, (1):57–64, 1998.

[62] S. Ninomiya and S. Tezuka. Toward real-time pricing of complex financial derivatives. *Applied Mathematical Finance*, 3:1–20, 1996.

[63] Bernt Øksendal. *Stochastic differential equations : an introduction with applications, 6th ed.* John Wiley, New York, 2003.

[64] Tao Pang, Mou-Hsiung Chang, and Moustapha Pemy. Finite difference approximations for stochastic control systems with delay. *Stochastic Analysis and Applications, to appear*, 2007.

[65] Tao Pang, Mou-Hsiung Chang, and Moustapha Pemy. Finite difference approximation for stochastic optimal stopping problems with delays. *Stochastics: An International Journal of Probability and Stochastic Processes*, (1):69–96, 2008.

[66] S.K. Park and K.W. Miller. Random number generators: good ones are hard to find. *Communications of the ACM*, 31(10):1192–1201, 1988.

[67] S. Paskov and J. Traub. Faster Valuation of Financial Derivatives. *Journal of Portfolio Management*, 22:113–120, 1995.

[68] W. Press, Teukolsky S., W. Vetterling, and B. Flannery. *Numerical Recipes in C, First Edition*. Cambridge University Press, 1988.

[69] W. Press, Teukolsky S., W. Vetterling, and B. Flannery. *Numerical Recipes in C, Second Edition*. Cambridge University Press, 1992.

[70] Xiao-Song Qian, Cheng-Long Xu, Li-Shang Jiang, and Bao-Jun Bian. Convergence of the Binomial Tree Method for American Options in a Jump-Diffusion Model. *SIAM J. Number. Anal.*, (5):1899–1913, 2005.

[71] Wei Ren and P.R. Kumar. Stochastic Adaptive Prediction and Model Reference Control. *IEEE Transactions on Automatic Control*, (10):2047–2060, 1994.

[72] S.F. Richard and R. Roll. Prepayments on fixed-rate mortgage-backed securities. *The Journal of Portfolio Management*, 15:73–82, 1989.

[73] RAYMOND RISHEL. A STRONG SEPARATION PRINCIPLE FOR STOCHASTIC CONTROL SYSTEMS DRIVEN BY A HIDDEN MARKOV MODEL. *SIAM J. Control And Optimization*, (4):1008–1020, 1994.

[74] Li Ronghua, Meng Hongbing, and Dai Yonghong. Convergence of numerical solutions to stochastic delay differential equations with jumps. *Applied Mathematics and Computationn*, pages 584–602, 2006.

[75] Wolfgang J. Runggaldier and Lukasz Stettner. Nearly Optimal Controls for Stochastic Ergodic Problems with Partial Observation. *SIAM J. Control And Optimization*, (1):180–218, 1993.

[76] C. Shieh. Robust Output-Feedback Control for Linear Continuous Uncertain State Delayed Systems with Unknown Time Delay. *Circuits Systems Signal Processing*, 21:309–318, 2002.

[77] Hu Shousong and Zhu Qixin. Stochastic optimal control and analysis of stability of networked control systems with long delay. *Automatica*, pages 1877–1884, 2003.

[78] S. C. SWORDER and D . SWORDER. Feedback Estimation Systems and the Separation Principle of Stochastic Control. *IEEE TRANSACTIONS ON AUTOMATIC CONTROL.*, pages 350–354, 1971.

[79] Shu Tezuka and Pierre L'Ecuyer. Efficient and Portable Combined Tausworthe Random Number Generators. *ACM Transactions on Modeling and Computer Simulation*, 1(2):99–112, 1991.

[80] S.C.A. Thomopoulos. Decentralized Filtering and Control in the Presence of Delays: Discrete-Time and Continuous-Time Case. *Information Sciences*, 81:133–153, 1994.

[81] Constantin Tudor. On Stochastic Functional-Differential Equations with Unbounded Delay. *SIAM J. Math. Anal.*, (6):1716–1725, 1987.

[82] O. Vasicek. An Equilibrium Characterization of the Term Structure. *Journal of Financial Economics*, 5:177–188, 1977.

[83] R.B. Vinter and R.H. Kwong. The infinite time quadratic control problem for linear systems with state and control delays: An evolution equation approach. *SIAM J. Control and Optim.*, 19:139–153, 1981.

[84] John B. Walsh. The rate of convergence of the binomial tree scheme. *Finance and Stochastics*, pages 337–361, 2003.

[85] Y. Wang, H. Wang, and J. Jiao. Nonlinear one-step-ahead predictive mean control of bounded dynamic stochastic systems with guaranteed stability. *International Journal of Systems Science*, 35(2), 2004.

[86] Chen Wei and Lei Guo. Prediction-Based Discrete-Time Adaptive Nonlinear Stochastic Control. *IEEE TRANSACTIONS ON AUTOMATIC CONTROL.*, (9):1725–1729, 1999.

[87] Morokoff W.J. and R.E. Caflisch. Quasi-Monte Carlo integration. *Journal of Computational Physics*, 122:218–230, 1995.

[88] Shengyuan Xu, James Lam, and Tongwen Chen. Robust H_∞ control for uncertain discrete stochastic time-delay systems. *Systems and Control Letters*, pages 203–215, 2004.

[89] C.N. Zeeb and P.J. Burns. *Random Number Generator Recommendation.* Technical Report, Colorado State University,Dept. of Mechanical Engineering, Fort Collins, Colorado, 1999.

[90] Qi-Min Zhang, Wei-Guo Zhang, and Zan-Kan Nie. Convergence of The Euler scheme for stochastic functional partial differential equations. *Applied Mathematics and Computation*, pages 479–492, 2004.