University of Southampton

Faculty of Business & Law

Southampton Business School

# Dynamic modelling of optimal pricing and trading policies under uncertainty

by

Shahin Abbaszadeh

Thesis for the degree of Doctor of Philosophy

March 2015

# Abstract

The objective of this thesis is to present a set of useful tools for problems of sequential decision making under uncertainty. Specifically, we study three applications of dynamic planning: dynamic pricing of non-durable products in the context of Markov processes, dynamic pricing of high end fashionable products with autoregressive demand, and the dynamic trading of financial securities with added sign constraints.

Market volatility, incomplete or delayed information, and unpredictability of underlying systems are integral to real-world problems. It is important to establish methods to integrate these factors into the modelling framework of choice. In this research we study stochastic dynamic programs and their use in finding optimal or near-optimal strategies for the above problems.

In the first of three papers comprising this thesis, we examine the dynamic pricing problem in the context of Markov decision processes, and explore the structural characteristics of the model. Our results support the use of exact methods when assuming the state of the system (demand) is unobservable. The second paper is concerned with a dynamic pricing problem that assumes an autoregressive evolution model for the demand. We provide a simple but effective approximate dynamic programming method that outperforms the classic methods of solving dynamic programming problems. Finally, in the third paper, we examine the dynamic trading of large blocks of securities by extending the dynamic programming framework to include constraints and additional information. We explore the characteristics of the model to improve on the closed form solutions available in the literature, but we also utilise a heuristic approximate dynamic programming method to provide near-optimal results when the problem is augmented with necessary constraints to handle practical settings.

# List of Contents

# List of Figures

# List of Tables

# Declaration Of Authorship

I, Shahin Abbaszadeh, declare that the thesis entitled 'Dynamic modelling of optimal pricing and trading policies under uncertainty' and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: **Shahin Abbaszadeh**

Date: **March 2015**

# Acknowledgements

The research period leading up to this thesis, has been a truly immense learning experience; well beyond the mastery of the research topics within this thesis.

My supervisors – Dr Yue Wu and Dr Tri-Dung Nguyen – have helped me in thinking like a researcher and steering numerous and sometimes daunting obstacles. Their support and patience has made this work possible. Also, the fourth chapter of this thesis is based on previous work by Tri-Dung. Our collaboration has been a great vehicle for learning and understanding.

I would like to thank Professor Tolga Bektas, Dr Gunes Erdogan, Dr Maxwell Chipulu and Dr Honora Smith for their valuable insights at different stages of my study. Also thanks to my friends and colleagues Dr Yousef Ghiami, Dr Stefanos Marangos and Dr Arash Gourtani for their support and many interesting conversations.

I would also like to thank many individuals in the Southampton Business School (formerly School of Management) who have facilitated this research in different ways, especially Ms Debbie Evans who single handedly brings order to the inevitably chaotic life of PhD students.

Finally, thanks to my family for their inspiration and encouragement.

Despite the help of many, I alone take full responsibility for any shortcomings of this work.

Helenile...

# List of Symbols

### Chapter 1

$I$ Number of items to be executed

$T$ Duration of the season

$N$ Number of periods in the season

$t$ Period number

$a_t$ Action (decision variable) at period $t$

$A$ Set of available actions (action space)

$s_t$ State of the system at period $t$

$M(.)$ State evolution function

$C(.)$ Cost or reward function

$V_t(.)$ Value function at time $t$

$S$ Set of possible states (state space)

$P(s_{t+1}|s_t, a_t)$ Probability of moving to state $s_{t+1}$ given action $a_t$ and state $s_t$

$c_t$ Number of remaining items at period $t$

$d_t$ Observed sales at period $t$

$D$ Set of possible sales observations

$Q(d_t|s_t, a_t)$ Probability of observing sales $d$ given action $a_t$ and state $s_t$

$p_t$ Price during period $t$

$\omega_t$ Random fluctuation during period $t$

$z_t$ Number of available stock

$\pi_t$ Elasticity of price based on volume during period $t$

**Chapter 2**

$N$ Number of stages

$n$ Index indicating the current stage

$\tau_n$ Length of $n_{th}$ stage

$\kappa_i$ Rate of customer arrival process (Poisson) when in state $i$

$\gamma_i$ Rate of reservation price (exponential distribution) when in state $i$

$\lambda_i$ Rate of the purchasing process (Poisson) when in state $i$

$p_n$ Price during stage $n$

$S$ Set of possible states (state space)

$P^{ij}(p_n)$ Probability of moving from state $i$ to $j$ with price $p_n$

$Q^{id}(p_n)$ Probability of observing sales $d$ while in state $i$ with price $p_n$

$\pi_{n,i}$ The belief of decision maker about (probability of) being in state $i$

$\Pi$ The set of belief vector during stage $n$

$W(p_n)$ The reward function for price $p_n$

$c_n$ Number of items in stock at stage $n$

$\alpha$ A single linear element of the piecewise value function

$\mathrm{A}_n$ Set of linear hyperplanes defining the value function at stage $n$

$T(\pi)$ Transformation of belief vector

$l(.)$ An index function assigning the optimal $\alpha$ from the value function

$\mathcal{V}$ Minimum-sized vector set representing the value function

$\oplus$ An operator for element-wise summation of two vectors

$\mathbb{PR}$ An operator to reduce a set to its minimum based on certain criteria

**Chapter 3**

$T$ Number of periods during a season

$t$ Index of a period

$\bar{S}$ Initial stock of the product

$p_t$ Price set during period $t$

$d_t$ Demand during period $t$

$\psi$ Impact of demand in the previous period on the current period

$\theta$ Impact of price on demand

$\epsilon_t$ The random volatility in the environment

$\{p_t\}$ Set of all available prices during period $t$

$z_t$ Number of unsold items at the end of period $t$

$y_t$ State variable comprised of $d_{t-1}$ and $z_{t-1}$

$A_t$ Quadratic coefficient of the approximate value function

$B_t$ Linear coefficient of the approximate value function

$C_t$ Constant value of the approximate value function

$\Omega$ Linear coefficient of the augmented state variable

$\Theta$ Impact of price on augmented state variable

$\Delta_t$ The random volatility affecting the state variable

**Chapter 4**

$S$ Total number of shares to be executed

$T$ Total number of periods in the execution period

| | |
|---|---|
| $t$ | Index of the current period |
| $p_t$ | Price of each security during period $t$ |
| $x_t$ | External or expert information about the securities during period $t$ |
| $s_t$ | Total number of stocks yet to be executed at start of period $t$ |
| $u_t$ | The volume of the trade at period $t$ (decision variable) |
| $a$ | Impact of trade volume on the price |
| $\beta$ | Impact of Information on the price |
| $\rho$ | Linear coefficient of the information evolution model |
| $\epsilon_t$ | Uncertainty in the price, a white noise process of $N(0, \sigma_\epsilon^2)$ |
| $\eta_t$ | Uncertainty in information variable, a white noise process of $N(0, \sigma_\eta^2)$ |
| $\theta_j$ | Impact factor from the price of equity in period $t - j$ |
| $A$ | Linear coefficient of the augmented state evolution model |
| $b$ | The impact of trade size on the augmented state variable |
| $\omega$ | Vector containing the uncertainty information |
| $e_m$ | $m_{th}$ column of the identity matrix |
| $U_j$ | State space area under case $j$ |
| $Q_t$ | Quadratic coefficient of the approximate value function |
| $B_t$ | Linear coefficient of the approximate value function |
| $C_t$ | Constant value of the approximate value function |
| $\Phi_{t,j}$ | Quadratic coefficient of the partial value function optimal in area $U_j$ |
| $\Pi_{t,j}$ | Linear coefficient of the partial value function optimal in area $U_j$ |
| $\Omega_{t,j}$ | Constant value of the partial value function optimal in area $U_j$ |

# Chapter 1

# Introduction

The premise of this thesis is the applications of stochastic dynamic programming in real-world situations. In this chapter we discuss the dynamic programming as a tool and its strengths and limitations. We provide a brief introduction to the problems we will be addressing in the following chapters and how we overcome the inherent limitations of the classic dynamic programming techniques.

## 1.1 An Introduction to Dynamic Programming

The dynamic programming approach can be expressed as decision making throughout stages. The system under study evolves through time from one state to another based on the decisions taken by the decision maker. It can be applied to many problems, from the control of a domestic boiler to strategic decision making undertaken by governments. Any activity in between these extremes that happens over time can represent a dynamic programming problem: managing assets, scheduling operation rooms in hospitals, planning vehicle routes, investing money or playing a simple game of backgammon. The decision making process in these problems includes observing available information, making a decision, observing the new state of the system as a result of the decision and making another decision again, and so on. Generally this sequential decision making problem is easy to formulate, but challenging to solve.

Dynamic programming has its roots in several academic fields. In the field of Economics and control theory, the problems include continuous states and deci-

sions. In the operational research and artificial intelligence literature, however, the elements are mainly of discrete nature. Continuous problems with continuous states and decision variables and continuous time are generally known as "control theory" problems. Problems with discrete state and decision variables are studied under the generic term "Markov decision processes". However, both modelling approaches rely on recursive equations, using state variable to represent the historical information. Dynamic programming problems can be roughly classified into three categories: (1) deterministic problems, where the rules governing the transition from one state to next are known to the decision maker; (2) stochastic problems, where the evolution of the state space or the effect of the decision on the state of the system contains uncertainty but the underlying probability distribution is known; and (3) information acquisition problems, where the uncertainty extends to the distribution of system parameters and the focus is on collecting information so that we can better estimate the distribution (Powell, 2011). By far the largest section of dynamic programming literature focuses on deterministic problems. However, many problems have unknown elements that pose significant difficulties on finding a solution. Stochastic dynamic programming deals with the uncertainty present in the system and the overall objective of the decision making process is to minimise expected cost or maximise expected profit.

Dynamic programming is based on a simple idea. It is based on the recursive process that describes the relationship between the value of being in a state at one point in time and the value of being in the states that we will visit next, following our decision.

Let's consider an illustrative example following the convention set in Powell (2011): assume there are $I$ items of identical properties and we want to execute them in time $T$. The term "execution" might constitute sale or purchase of physical or financial commodities or manufacturing a product or maintenance of a machine. Time $T$ is divided into $N$ stages. At each stage $t$ we can choose the decision variable $a_t$ from a certain set $A$ in order to control the execution of the items. Given the decision $a_t$ the new state of the system $s_t$ will be determined by the general function $M(s_t, a_t)$. We want to maximise or minimise $C(s_t, a_t)$ over the whole period $T$, depending on if it is a reward or cost function.

In deterministic cases the value function can be written as:

$$V_t(s_t) = \max_{a_t \in A}\{C_t(s_t, a_t) + V_{t+1}(s_{t+1})\}, \tag{1.1}$$

where $s_{t+1} = M(s_t, a_t)$ is the state we visit if we are in state $s_t$ and take action $a_t$. Equation (1.1) is the so-called Bellman's equation.

But the problems we will address in this work involve uncertainty (in terms of price, demand or market forces). In such problems the value function becomes:

$$V_t(s_t) = \max_{a_t \in A}\{C_t(s_t, a_t) + \mathbb{E}[V_{t+1}(s_{t+1})|s_t]\}. \tag{1.2}$$

As can be seen, the underlying structure of the dynamic programming is rather straightforward. However, a great number of dynamic programming problems are challenging to model and standard solution approaches to solve them are computationally intractable. This is because of the so-called three curses of dimensionality (Gosavi, 2009).

We defer a thorough study of the dynamic programming model to the following chapters. However it is useful to take a closer look at its structure and examine the curses of dimensionality in more detail.

Every stochastic dynamic program consist of the following elements at the minimum (Bertsekas, 1995):

**State variable** This represents all the information we need to make a decision.

**Decision variable** Actions that represent control of the process.

**Exogenous information** This data becomes known at the beginning of each period, e.g. the demand for the product, or the price at which it can be purchased or sold. In addition, the initial state of the system should be known to the decision maker.

**Transition function** This function captures the system evolution from state $s_t$ to state $s_{t+1}$ given our decision at time $t$ and the new information that became known between $t$ and $t + 1$.

**Objective function** This function describes the costs to be minimised, or the rewards to be maximised over the designated period.

The model defined at (1.2) has a simple solution. If we know the value of $V_{t+1}(s_{t+1})$ for each state $s_{t+1}$ we would just compute (1.2) for each $s_t$, giving us the value of $V_t(s_t)$. We can then step backwards in periods to compute all the value functions. However, this is where the exponential increase in the dimensions makes finding a solution through the classical dynamic programming approach impractical. Each dynamic programming model has at least three dimensions, and these can become very large depending on the specifics of the problem. The **state variable** $s$ can be a vector, each element of which may represent many possible values. The **decision variable** $a_t$ also can have a vector form with each instance having many potential outcomes. Finally **random fluctuations** represent the third dimension that can vary from period to period and extend the space of the problem. These three dimensions represent the three curses of dimensionality.

In this research, we are looking for different methods that overcome this explosion of possible outcomes and harness the randomness of the underlying system and possibly exploit the structure of the value function, in order to solve practical and real-world applications.

## 1.2 Background of the Problems Under Study

Dynamic programming is essentially about the control of one or more resources. It is either direct manipulation of the resource (purchase or sale of goods, controlling the output of a Hydro power plant), or controlling a specific attribute of the resource (setting the price of an asset, deciding when to upgrade the generator of a Hydro power plant).

If enough assumptions are made, we will be left with a value function (1.2) and a model defining the evolution of the state, which is not particularly difficult to solve. But in real-world applications, many issues exist that make seemingly simple problems intractable. For example, when we are faced with evolving information processes, the decision is made before the information is known. Alternatively in many situations we cannot precisely determine the current state of our system. According to Powell (2011) this is mainly because information arrives with delay (stock prices, demand data), or because it is misrepresented

at some stage (wrong reports, missing data) or the underlying system is simply unobservable (the exact location of a space probe). Another important factor is the lack of clarity with the dynamics of the system (for example the state evolution model) where we are not certain of the effects of various parameters in the direction of the system dynamics. If we cannot predict the effects of our decisions (even with uncertainty) any result will be accompanied with significant biases.

Now we briefly look at the contents of the three main chapters of this thesis. We are going to examine three real-world applications of dynamic programs discussed above in the coming chapters.

### 1.2.1 Dynamic Pricing Using POMDP

In general terms revenue management and in particular dynamic pricing which is a subset of it, deal with the problem faced by a decision maker who needs to execute a set of fixed resources opposite a price sensitive population of customers. In this framework the main objective is to find a strategy that maximises the revenue throughout the selling horizon. Bitran and Candentey (2003) argue that the price is not just a financial concept, but it is among the most effective parameters that managers can use to encourage or discourage the demand, making it an operational asset that can regulate inventory and production as well as the service level. Dynamic pricing provides the ability to match supply with demand and thus increase the revenue in industries that are facing ever increasing revenue margins. These are the industries that have high start-up costs, the capacity is perishable, the sale horizon is reasonably short and demand is both stochastic and sensitive to price (Chen and Chen, 2015)

Markov decision processes (MDP) is an elegant method for solving stochastic dynamic programs, but it is practical only in the presence of strong assumptions. If our state space is small and there is a relatively small set of decisions or actions, the MDP method not only can solve these problems, it can provide strong insights into the underlying structure of the problem.

In chapter 2 we consider a dynamic pricing problem where $I$ is the number of items at the disposal of the seller. It is a finite stage Markov decision process and

the goal is to maximise the profit by deciding on the price during each period $t$. Our decision variable $a_t$ denotes the choice of price from a predefined set of prices $A$. If price $a_t$ is chosen, the probability that we move from state $s_t$ to state $s_{t+1}$ is $P_t(s_{t+1}|s_t, a_t)$ where $s_t$ (the underlying state of the system) belongs to the set $S = \{1, 2, ..., |S|\}$. $P_t(s_{t+1}|s_t, a_t)$ is the so-called state transition matrix and replaces the function $M(s, a)$ illustrated in section 1.1. The presence of a transition matrix transforms this problem to an MDP.

The Bellman equation is:

$$V_t(s_t, c_t) = \max_{a_t \in A}\{C_t(s_t, a_t) + \mathbb{E}[\sum_{i \in S} P_t(s_{t+1} = i|s_t, a_t)\ V_{t+1}(s_{t+1}, c_{t+1})]\} \quad (1.3)$$

where state $s_t$ is assumed to be the actual demand during period $t$ and $c_t$ is the number of items during that period.

However, if we assume that in a particular problem the actual state of the demand is unknown and the sale figures are the main decisive factor for the retailer, and furthermore we assume that the probabilistic relationship between sales and actual demand is known based on previous historical data, we can understand the evolution of the state by observing the sales parameter. These classes of problems, where the state of the system is unobservable but are related to an observable parameter is called partially observable Markov decision processes (POMDP). Thus we record the sale figures as observations at the end of each period, where $Q_t(d_t|s_t, a_t)$ is the probability that we observe $d$ as the sales amount while we are at state $s$ and price for period $t$ has been $a$. $d$ belongs to a set $D$, the elements of which are finite and countable.

The revised value function, although substantially more general and difficult to solve, is at the same time very elegant in providing insights into the evolution of the problem based on our decisions.

$$V_t(s_t) = \max_{a_t \in A}\{C_t(s_t, a_t) + \mathbb{E}[\sum_{i \in S} P_t(s_{t+1} = i|s_t, a_t) \sum_{j \in D} Q_t(d_t = j|s_t, a_t) V_{t+1}(s_{t+1}, c_t - d_t)].\} \quad (1.4)$$

Although the above dynamic programming model becomes prohibitively large but for the smallest problems, we solve the POMDP model in chapter 2 by exploring the structure of the value function. We consider that the value function is piecewise linear and convex and on that basis study the exact algo-

rithms devised for general POMDP problems as a viable option for solving the dynamic pricing problem. However, the next chapter provides a solution to a similar problem that is capable of handling much larger input parameters.

### 1.2.2 Dynamic Pricing with Autoregressive Demand

The pricing problem discussed in chapter 3 is an extension to the problem we consider in chapter 2.

The retailer has an initial stock of $I$ items and must within the finite time $T$ sell them. Time $T$ is divided into $N$ stages. $p_t$ denotes the decision variable (price during period $t$) which can be chosen from the set $A$. The corresponding Bellman equation for this problem can be represented as:

$$V_t(d_{t-1}, z_{t-1}) = \max_{\{p_t\}} \mathbb{E}[p_t d_t + V_{t+1}(d_t, z_t)].$$

in which $d_t$ represents state of the system (demand), $z_{t-1}$ denotes the number of items remaining at the beginning of period $t$ and $\{p_t\}$ is the set of all available prices during period $t$ which is a finite set.

The main difference to the MDP modelling is that we define the state evolution model as a linear representation that depends on our action (price) in addition to the previous state of the system. Thus we say that the demand is autoregressive and the state of the system evolves based on the following evolution model:

$$d_t = \min\left(\psi d_{t-1} - \theta p_t + \epsilon_t, z_{t-1}\right)$$

where $\epsilon_t$ denotes the volatility in the underlying system and $\psi$ and $\theta$ are coefficients to state and action variables.

The above evolution model comprise the main novelty of our approach. Very rarely in literature the demand evolution is considered of autoregressive (AR) nature, although there is enough evidence for its benefits in certain environments. The incorporation of AR to the value function, however, poses considerable difficulty in solving this problem under the classic dynamic programming approach. This dynamic program is intractable because in addition to the curses of dimensionality, the state space will grow in each backward stage of the classic dynamic programming approach, as we would have to introduce

inequalities to the model in order to handle the inclusion of capacity and price considerations. Thus in chapter 3 we propose an approximate dynamic programming method that in addition to its ability in handling large state and action sets, is capable of handling constraints, which is necessary to provide a solution that is applicable in real-world scenarios. The proposed approximate dynamic programming algorithm captures the simplicity of the value function model and provides an exploratory element through Monte Carlo simulation to converge to near optimal solutions in a fast time-frame.

### 1.2.3 Dynamic Trading

The growth in equity trading in recent decades, especially algorithmic trading, has triggered an interest in the more effective management of trading costs. These costs are often called execution costs and include commissions, bid/ask spreads, opportunity costs of waiting, and price impact from trading. According to Chan and Lakonishok (1993) trading in equity markets is increasingly dominated by institutional investors, and due to the costliness of executing individual trade transactions, the overall execution of the order becomes more costly. This overall transaction cost prompts the traders to break their order down into smaller transaction units which is then executed over a certain time period. A trading strategy that minimises the expected execution cost of the trade is defined as best execution strategy. This problem, henceforth referred to as dynamic trading have been analysed extensively in the finance literature, a classic example of which is to be found in Bertsimas and Lo (1998). They propose an effective dynamic programming method to this real-world problem, but the model is not very practical as they ignore the constraints that are part of the actual business environment. There are considerable similarities between the dynamic pricing problem and that of dynamic trading. We will expand on the concepts we employ in dealing with dynamic pricing, to provide an effective and practical solution to the dynamic trading problem when there are constraints in the actual trade setting.

The dynamic trading problem which we will cover in chapter 4 is briefly as follows:

We have a number $I$ of shares to execute (sell) in $N$ consecutive periods of equal length amounting to time $T$ in total. We are also given the price dynamics, which is dependent on the size of trade and other related information. Based on these information the decision maker wants to optimise the number of shares traded in each period in order to minimise the overall cost of execution of $I$ within time $T$. The bellman equation for this problem in its simplest form is as follows:

$$V_t(p_{t-1}) = \max_{\{a_t\}} \quad \mathbb{E}[p_t a_t + V_{t+1}(p_t)],$$

where $\{a_t\}$ is the set of available actions at period $t$.

The price (our state variable) evolves based on

$$p_{t+1} = p_t + \pi_t a_t + \omega_t,$$

where $\omega_t$ is an independent white noise process with mean 0 and variance $\sigma_\omega^2$ and $\pi_t$ is the elasticity of price based on volume known to the decision maker from historical data.

As can be seen, the structure of the model and the state evolution process is very similar to the dynamic pricing problem of chapter 3, but there are certain limitations to this problem because of its specific business setting. We have provided the basic definition of the problem here, which we will expand further in the corresponding chapter. We will explore the characteristics of the value function to provide a near optimal solution to the problem where we have added sign constraints to this basic problem. Furthermore, many price processes in financial domain exhibit autoregressive attributes of higher order. We propose a framework that is capable of handling lags of more than one in the price evolution process. Our main extension to this problem, however, is the addition of a non-negativity constraint. Based on the variance of the information variable and also price, the optimal solution might suggest a negative trade (i.e. sell in a buy operation or vice versa). In a real situation where the short-selling option is not available, a crude constraint might be introduced to change the negative trades to zero. We propose an approximate dynamic programming approach that incorporates the non-negativity constraint as well as higher order autoregressive elements into the problem and provides near-optimal solutions.

## 1.3 Structure of the Thesis

This thesis follows a three-paper structure. As such I have included the three research papers comprising this research with minimal alteration to their original structure. They have retained their independent format, while I have changed minor elements to achieve an overarching structure for the whole thesis.

Chapter 2, titled "Dynamic Pricing and the use of exact algorithms for partially observable Markov decision processes" reviews the exact methods for a special application of MDP on the dynamic pricing problem and provides numerical experiments showing their capabilities. Chapter 3, titled "Dynamic pricing of products with autoregressive demand" deals with a dynamic pricing problem with autoregressive demand. We introduce an approximate dynamic programming method that handles the possibility of optimal policies that are negative or otherwise outside the acceptable range. Chapter 4 titled "Optimal trading under non-negativity constraints using approximate dynamic programming" discusses the dynamic programming method for achieving optimal execution policies and offers an approximate dynamic programming method that accounts for the addition of sign constraints. Finally, chapter 5 provides a summary of our contributions and a look at the limitation of this study and possible directions for future research.

# Chapter 2

# Dynamic Pricing and the Use of Exact Algorithms for Partially Observable Markov Decision Processes


### Abstract

This paper examines dynamic pricing problem in a retail setting where short-life-cycle goods are sold through a single store with no replenishment during the season. We model the problem of finding the optimal dynamic pricing policy by utilizing partially observable Markov decision processes (POMDP). The objective of the model is to maximise expected profit using multiple predefined price changes throughout the season. We employ three exact algorithms for POMDP that make use of the convexity and piecewise linearity of the value function to solve this dynamic pricing problem. We assess the performance of these algorithms with simulation runs and provide insight to the benefits we can gain from them. We demonstrate that these exact algorithms are efficient in solving problems of considerable size and complexity in the context of retail discounting.

## 2.1 Introduction

### 2.1.1 Background and Literature Review

This paper studies the dynamic pricing problem for retail products with short life cycle and with no replenishment possibility. Dynamic pricing has its roots in the more general literature of revenue management, also called yield management or perishable asset revenue management, which deals with the balancing of demand and supply. Revenue management research started in airline industry, on segmentation of the market into different classes (e.g. business versus leisure customers) and allocation of the limited capacity to these classes. Its success in the airline industry has led to its application in other transportation services and service industry (McGill and Van Ryzin, 1999). These days revenue management is applied in many industries where supply capacity is inflexible and short term, such as cruise ships, hotels, electric utilities, sporting events, health care and a significant section of retail industry.

In most applications of revenue management, prices can be changed with little or no cost at all, but in retail industry, especially the brick and mortar sector, the associated costs are more substantial; although technological advances continue to reduce these costs across many industries (Netessine, 2006). In this paper following the recent literature, we ignore the costs of price changes. The big advances in information technology (IT) and e-commerce have played a substantial role, not only in reducing the cost of price change but improving inventory management practices. Despite these big strides in inventory management, lost sales and excess inventory still impose considerable amounts of expense on retailers. For this reason retailers increasingly explore the demand side of the supply-demand equation in order to drive up the profits. Dynamic pricing explores the possibility of controlling demand through price changes and is one of the tools the retail manager may employ to alter demand behaviour (Elmaghraby and Keskinocak, 2003). Factors that drive this increased adoption among the practitioners include the availability of more demand data, technological advances that make changing the prices easier and the overall increased efficiency of the decision-support tools.

An important early contribution to the dynamic pricing literature is Gallego and Van Ryzin (1994). They investigate the dynamic pricing problem for perishable products with stochastic demand that is price sensitive. They find the optimal pricing policy in closed form for a range of demand functions while for general demand functions they find an upper bound on the expected revenue. Bitran et al. (1998) formulate a stochastic dynamic programming problem to coordinate prices for a retailer. They consider a retail chain that changes prices periodically and maximise the total discounted expected revenue over the planning horizon. They also develop heuristics to achieve near-optimal solutions. Chatwin (2000) studies the case of dynamic pricing of a fixed number of perishable items in finite horizon. He obtains structural characteristics for the problem and derive that the optimal price is non-increasing in the number of remaining unsold items.

Adida and Perakis (2007) present a continuous time optimal control model for a dynamic pricing and inventory control problem in a make-to-stock manufacturing system. The demand is a linear function of the price, the inventory cost is linear and all the coefficients are time-dependent. Feng and Gallego (1995), in the same manner as Gallego and Van Ryzin (1994) model the demand as a homogenous (time-invariant) Poisson process with an intensity that is non-increasing in price. By charging a specific price at each time period the firm controls the intensity of the demand. In these papers, the reservation price (i.e. the price threshold under which the customer is willing to buy) or its distribution remains constant over time. In contrast, Bitran et al. (1998), Bitran and Mondschein (1997), and Zhao and Zheng (2000) generalise these models by modelling the demand as a non-homogeneous Poisson process and allowing the probability distribution of the reservation price to change over time. Smith and Achabal (1998) incorporate the impact of the inventory level on demand in contrast to its dependency on price or time. Particularly, in the retail and fashion goods industry, a product's demand is usually influenced by the shelf space it occupies. The relationship between "display" area and sales is typically one sided: lower inventory levels may slow the sales rate while inventory levels above the critical "minimum" does nothing to promote further sales.

Today's technologies allow retailers to collect large amounts of data, from

sales to demographic data and customer preferences. There are, however, many uncertainties which make the interpretation of this data difficult for the retailer. The seasonality of demand or special occasions for particular products add to the complexity of forecasting process. Unpredictable behaviour of the customers which is inherently difficult to observe or formulate are obstacles in retailers' ability to form an opinion about the demand. And arguably, the most important aspect of uncertainty for a market with above-mentioned characteristics (e.g. retail market of short-life-cycle products) is the underlying economic situation, which according to Song and Zipkin (1993) may stem from either the state of the industry or a reflection of the economy as a whole. Sethi and Cheng (1997) suggest that the uncertainties can be collectively described by what they label as "states of the world", and argue that the demand process of such an environment can be described by a Markov chain i.e. the future "state of the world" is determined only by the current state of the world. As a result each core state of the system includes all the past information about the market and demand, and the transition from each state to another follows the rules of a Markov process. In reality, however, a complete and thorough knowledge of the demand, even after one has all the information pertaining to the previous seasons, is unattainable. In this paper, it is assumed that the manager can acquire partial information about the state of the demand for the product through sales and its correlation to real demand. Instead of a Markov process representing the evolution of the demand through time periods, a partially observable Markov decision process (POMDP) is assumed that determines the unobservable state of the system (demand) through its relation to the sales, which is an observable parameter in the system. POMDP is a generalised form of MDP that assumes the state of the system is unknown to the decision maker, but he/she can observe a parameter of the system which is probabilistically correlated to the core process.

For a presentation of POMDP and review of various prior algorithms refer to Monahan (1982). Lovejoy (1991a) , White and Scherer (1989) and White (1991) offer literature reviews as well as important updates on previous exact algorithms. Smallwood and Sondik (1973) were the first to prove the piecewise linearity of the recursive dynamic value function and thus provide the ground-

work for the methods developed later. They offer a step by step solution using this characteristic of the formulation which although practically infeasible, provided the required insight for the later more advanced algorithms. Monahan in his review paper offers a revised edition of Smallwood and Sondik's algorithm, which although he introduces as Sondik's, is subtly different and more intuitive. We will study both of these methods later in the paper as an insight into the foundations of the POMDP exact algorithms.

Lovejoy (1991a) develops a heuristic approach based on the above methods to utilise the overlooked aspects of these solutions more efficiently. Eagle (1984) formulates a problem of partially observable Markov decision processes and presents a finite time horizon POMDP solution technique simpler than the standard linear programming methods of the time. The context is of a search problem where a target is moving in discrete time among a finite number of cells. The problem is to find a searcher path, i.e. a sequence of search cells, that maximises the probability of detecting the target in a fixed number of time periods. This work improves the algorithm introduced by Smallwood and Sondik (1973).

Kaelbling et al. (1998) survey the exact methods developed in AI community that aim to solve problems defined in partially observable stochastic domains. They discuss different approaches taken to exactly solve the POMDP problem in a control theory setting. Of these approaches they focus on witness and incremental pruning algorithms, their complexity and different approximations one can derive from these. Hauskrecht (2000) surveys various approximation (heuristic) methods for POMDP and presents a number of new approximation methods and refinements of existing techniques. Lin et al. (2004) develop an exact algorithm based on incremental pruning and use the genetic algorithm to solve the problem by constructing the minimal set of affine functions that describe the value function. Our work follows the developed algorithms by Cassandra et al. (1997), Kaelbling et al. (1998) and Zhang and Liu (1997).

According to Lovejoy (1991b), one of the main reasons behind the restricted popularity and application of the POMDP models is their intractable nature which is a result of uncountable states. In this study we argue that the exact methods that make use of convexity and piecewise linearity of the value function

reduce the intractability, and thus are capable of handling bigger and more complex problems that were not practical before. As such we utilise POMDP to model the dynamic pricing problem and apply three exact methods to solve the problem.

### 2.1.2 Contributions and Paper Structure

Despite the extensive presence of POMDP in operational research literature, the exact methods devised for it have not been applied to any revenue management problem to the best of our knowledge. Aviv and Pazgal (2005) is the only paper in the literature that has utilized the POMDP for the dynamic pricing problem. They simplify the POMDP into a regular MDP problem through belief transformation and offer an upper bound method named Information-Structure Modification which augments the state to include additional external information. They solve the model in relation to specific probability distributions and transition matrices that results in a reduced state space. They provide upper and lower bounds for the value function and a heuristic algorithm for policy generation. However, the authors do not include any reference to the exact methods in the POMDP literature. If we assume that this oversight has been justified with the intractable nature of the model, there is new motivation in application of exact methods to POMDP with the emergence of more robust variants of exact methods to POMDP models. These algorithms take advantage of convexity and piecewise linearity of the value function and reduce the state space considerably. This, along with the advent of more powerful computers and parallel computing, place exact methods in a very strong position to be utilised in today's industrial applications.

We formulate the model following the structure set by Aviv and Pazgal (2005), but the notation is driven from Monahan (1982) as it is the notation the majority of POMDP literature has adopted. The structure of the dynamic pricing model in section 2.2 which utilises the POMDP approach is slightly different from that of classical POMDP models. Although the proof exists in the POMDP literature (Sondik (1971) and Smallwood and Sondik (1973)) to the convexity and piecewise linearity of the value function, it involves models in the

context of optimal control theory. For the purposes of consistency, we provide proof that the objective function of the model in dynamic pricing context is also convex and piecewise linear. We adopt a selection of exact methods from the literature to the dynamic pricing problem modelled in POMDP. We demonstrate the ability of these exact methods in handling of dynamic pricing problem through simulation runs and provide insight into their structural properties.

The remainder of this paper is structured as follows: In section 2.2 we present the dynamic pricing problem and the corresponding POMDP model and ascertain the convexity and piecewise linearity characteristics of the value function in a dynamic pricing setting. Section 2.3 contains the discussion of the selected exact methods and their structure. Finally, we present the results of our numerical studies and conclude with our observations.

## 2.2 Dynamic Pricing Problem

### 2.2.1 Dynamic pricing model with POMDP

We consider a retail store selling a fixed amount of short life cycle products. We assume that replenishment is not possible once the season gets under way. This is a common assumption because of the increasingly long supply processes today (Chen and Chen, 2015). For example, the design and planning stage in the case of fashionable clothing and accessories is a lengthy process. Furthermore, since the majority of the products are manufactured in far off countries with cheaper labour, the lead times are too long for reordering during the season. Although for the purposes of consistency we assume the product in this paper to be fashionable clothing, the underlying problem we are attempting to address is applicable to a large number of products that share the same characteristics. Another example of products to which this problem definition is relevant are the technologically advanced products that lose their values significantly once their season comes to an end and are replaced by new products.

The retailer sets a price at the start of the season and during the season can choose to implement a new price from a predefined finite set of prices at the start of each stage to a maximum of $N$ stages. We assume that demand is

independent over time as a sustained and informed purchase among customers is unlikely due to the short sale period. We also assume that the arrival rate of customers follows a Poisson distribution with rates $\kappa_i$ where $i$ refers to the state of the demand. Reservation price (the price below which the customer commits to a purchase) follows an exponential distribution with factor $\gamma_i$ representing the average reservation price for the state of the demand $i$. This is integrated into the arrival rate and is a given at the start of the season. The resulting distribution for the overall purchasing process is also Poisson with rate $\lambda_i = \kappa_i e^{-p/\gamma_i}\tau_n$ where $\tau_n$ is the length of $n_{th}$ stage (Aviv and Pazgal, 2005).

We assume that the actual state of the demand is unknown, and the sale is the main factor driving the decision of the retailer. The probabilistic relationship between sales and actual demand is known to the retailer based on previous historical data. Thus if we consider the process to be a Markov decision process, we assume the demand to be the state of the process which is described as a Poisson arrival process with rate $\lambda_i$. This is a finite stage Markov decision process and there are $n$ remaining stages out of the total of $N$ stages of the planning horizon. The goal is to maximise the profit by deciding on the price during each period $n$. Price during period $n$ ($p_n$) is selected from a predefined set of prices. The state belongs to a discrete space $S = (1, 2, ..., |S|)$ where $S$ is small enough to enumerate. If price $p_n$ is chosen, the probability that we move from state $i$ in period $n$ to state $j$ in period $n-1$ (demand is a Poisson process with rate $\lambda_i$ and it transforms to a Poisson process with rate $\lambda_j$) is $P^{ij}(p_n)$. However as mentioned before, we cannot observe the current state of the demand and thus we record the sale figures as observation at the end of each stage, where $Q^{id}(p_n)$ is the probability that we observe $d$ as the sales figure while we are at state $i$ and the price during the period has been $p_n$. The observed value $d_n$ which denotes the number of sales during stage $n$ is bound by the number of remaining items $c_n$. Both $P^{ij}(p_n)$ and $Q^{id}(p_n)$ are known to the retailer at the start of the season, based on historical data and expert opinion.

$$P^{ij}(p_n) = \Pr\left\{s_{n-1} = j | s_n = i, p_n\right\},$$

$$Q^{id}(p_n) = \Pr\{d_n = d | s_n = i, p_n\}.$$

We define the belief vector (belief of the seller about the state of the demand formed at the start of the stage) for stage $n \in N$ as $\pi_n$, which is a discrete probability distribution over the set of core states $S$:

$$\pi_n = (\pi_{n,1}, \pi_{n,2}, ..., \pi_{n,k}) \in \Pi,$$

where $\Pi = \{\pi \in \mathbb{R}^k : \sum_{k \in S} \pi_k = 1, \pi \geq 0\}$

$W(p_n)$ is defined as the revenue structure during stage $n$ when price $p$ is chosen. The expected revenue for period $n$ is:

$$W(p_n) = p_n \sum_{i \in S} \pi_{n,i} \sum_{d < c_n} Q^{id}(p_n)\ d.$$

$V_n(\pi_n, c_n)$ represents the total expected profit over $n$ remaining stages where $c_n$ is the number of items in the stock at the start of stage $n$. It is the sum of $W(p_n)$ plus the expected profit over the remaining $n-1$ stages maximised over price:

$$V_n(\pi_n, c_n) = \max_p \left\{ W(p_n) + \sum_{i \in S} \pi_{n,i} \sum_{d < c_n} Q^{id}(p_n) V_{n-1}(\pi_{n-1}, c_n - d) \right\}, \quad (2.1)$$

$$V_0(\pi_0, c_0) = 0.$$

$\pi_{n-1}$ is the belief vector for the stage $n-1$ where we had price $p$ during stage $n$ and observed $d$ amount of sales at the end of stage $n$. The seller updates his belief at the end of stage $n$ based on $d$. The Bayesian transformation to achieve this update based on information at hand is:

$$\pi_{n-1k} = \frac{\sum_i \pi_{n,i}\, Q^{id}(p_n)\, P^{ik}(p_n)}{\sum_i \pi_{n,i}\, Q^{id}(p_n)}. \quad (2.2)$$

This transforms the POMDP problem, which is a process defined over uncountable state space, into an equivalent observable regular MDP process over continuous belief space. As Smallwood and Sondik (1973) and consequently Monahan (1982) prove, the belief vector is a "sufficient statistic" from all the past information and is stored in the current belief vector.

### 2.2.2 Properties of the Model

Although dynamic programming model (2.1) is in theory solvable through dynamic updates, it becomes prohibitively large for even the smallest problems,

as the state space is uncountable despite the finite number of states, actions and observations. Smallwood and Sondik (1973) in their landmark work prove that the value function for the dynamic program of POMDP is piecewise linear and convex. This forms the basis of exact algorithms devised for the POMDP model. The value function can be represented as follows:

$$V_n(\pi_n, c_n) = \max_{\alpha \in \mathrm{A}_n} \{\pi\ \alpha\}, \tag{2.3}$$

where $\alpha$ is a function over state space and represents a single linear piece of the value function (i.e. each $\alpha$ is a hyper-plane dividing the state space into two half spaces).

$\mathrm{A}_n$ is the set of $\alpha$ vectors defining the value function at the end of stage $n$. The objective function in (2.1) can be decomposed through following steps:

$$\begin{aligned} V_n^*(\pi_n, c_n) &= \max_p V_n^{*p}(\pi_n, c_n), \\ V_n^{*p}(\pi_n, c_n) &= \sum_{d<c_n} V_n^{*p,d}(\pi_n, c_n), \\ V_n^{*p,d}(\pi_n, c_n) &= p_n \sum_{i \in S} \pi_{n,i} Q^{id}(p_n) d + \sum_{i \in S} \pi_{n,i} Q^{id}(p_n) V_{n-1}^*(\pi_{n-1}, c_n - d) \quad \forall d < c_n. \end{aligned} \tag{2.4}$$

The expected profit is represented by $V_n^{*p,d}(\pi_n, c_n)$, when the observed sale is $d$ for a period $n$ of price $p$ and optimal prices are chosen for all the remaining stages. Sondik (1971) shows that piecewise linearity and convexity is preserved through the decomposition from $V_n^*(\pi_n, c_n)$ to $V_n^{*p,d}(\pi_n, c_n)$. It means, if one can show that $V_n^{*p,d}(\pi_n, c_n)$ is piecewise linear and convex, then $V_n^*(\pi_n, c_n)$ is also piecewise linear and convex (It is easy to show that summation and maximisation functions over a set of piecewise linear and convex functions are themselves piecewise linear and convex; for a demonstration refer to Sondik (1971)). Although the proof exists in the literature, we formally provide the proof that $V_n^{*p,d}(\pi_n, c_n)$ is piecewise linear and convex in theorem 2.1 for the purposes of consistency and to confirm that the subtle differences between the formulation of the dynamic pricing model above and the classic formulations of POMDP in the literature does not affect this fundamental result.

**Theorem 2.1.**

$$V_n^{*p,d}(\pi_n, c_n) = p_n \sum_{i \in S} \pi_{n,i} Q^{id}(p_n) d + \sum_{i \in S} \pi_{n,i} Q^{id}(p_n) V_{n-1}^*(\pi_{n-1}, c_n - d) \quad \forall d < c_n,$$

*is piecewise linear and convex.*

*Proof.* We know $V_0(\pi_0, c_0) = 0$, as we assume there are no salvage values for any remaining items.

Thus we have

$$V_1^{*p,d}(\pi_1, c_1) = p_1 d \sum_{i \in S} \pi_{1,i} Q^{id} \quad \forall d < c_1,$$

which indicates all of $V_1^{*p,d}(\cdot)$ functions are linear and as sum of linear functions result in a linear function $V_1^*(\cdot)$.

Next we assume $V_{n-1}^*(\pi_{n-1})$ is piecewise linear and convex. Equivalent of equation (2.3) for $V_{n-1}^*$ is:

$$V_{n-1}^*(\pi_{n-1}) = \max_{\alpha \in \mathrm{A}_{n-1}} \pi_{n-1}\ \alpha.$$

If we assume

$$\alpha_{n-1}^{l(\pi,p,d)} = \arg\max_{\alpha \in \mathrm{A}_{n-1}} \pi\ \alpha,$$

where $l(\pi, p, d)$ is a convenience index function (Smallwood and Sondik, 1973) in order to establish which segment of state space the vector belongs in the following period (previous dynamic programming step), then we have

$$V_{n-1}^*(\pi_{n-1}) = \pi_{n-1}\ \alpha_{n-1}^{l(\pi_{n-1},p,d)}.$$

By putting this into (2.4) we get

$$V_n^{*p,d}(\pi_n, c_n) = p_n d \sum_{i \in S} \pi_{n,i} Q^{id} + \sum_{i \in S} \pi_{n,i}\, Q^{id} \sum_{k \in S} \pi_{n-1,k} \alpha_{n-1}^{l(\pi_{n-1},p,d)} \quad \forall d < c_n.$$

By substituting the value $\pi_{n-1}$ from (2.2) into this equation we will have:

$$\begin{aligned} V_n^{*p,d}(\pi_n, c_n) = & \quad p_n d \sum_{i \in S} \pi_{n,i} Q^{id} + \sum_{i \in S} \pi_{n,i}\ Q^{id} \sum_{k \in S} P^{ik}\ \ \alpha_{n-1}^{l(\pi_{n-1},p,d)} \\ = & \quad \sum_{i \in S} \pi_{n,i} Q^{id} (p_n d + \sum_{k \in S} P^{ik}\ \ \alpha_{n-1}^{l(\pi_{n-1},p,d)})\ \ \forall d < c_n. \end{aligned} \tag{2.5}$$

If we put

$$\alpha_{n,i}^{p,d} = Q^{id}(p_n d + \sum_{k \in S} P^{ik}\, \alpha_{n-1}^{l(\pi_{n-1},p,d)}),$$

into (2.5), we get:

$$V_n^{*p,d}(\pi_n, c_n) = \pi \alpha_n^{p,d}\ \ \forall d < c_n,$$

which is clearly of the same property as $V^*_{n-1}(\pi_{n-1})$ being piecewise linear and convex over $\pi$. As discussed in the text, a piecewise linear and convex function $V^{*p,d}_n(\pi_n, c_n)$, results in a convex and piecewise linear function $V^*_n(\pi_n, c_n)$. □

Piecewise linearity and convexity of the value function is the foundation for all the exact algorithms in the literature as it allows for a global optimal point to exist and provides a framework that minimises the search route to the optimum. Albeit many heuristic methods are developed to exploit the same attribute (Cassandra, 1998). However many researchers have refined the exact methods over the years to find more efficient and fast solutions. In the next section we will study a selection of these exact methods in order to determine the most suitable procedure for the dynamic pricing problem.

In order to visualise the piecewise linearity and convexity of the value function, assume that a value function defined as in (2.3) has a dimension of 3. Since the belief space is probabilistic, we can reduce one dimension, thus the three dimensional value function is easily depicted on two dimensions as in Figure 2.1.

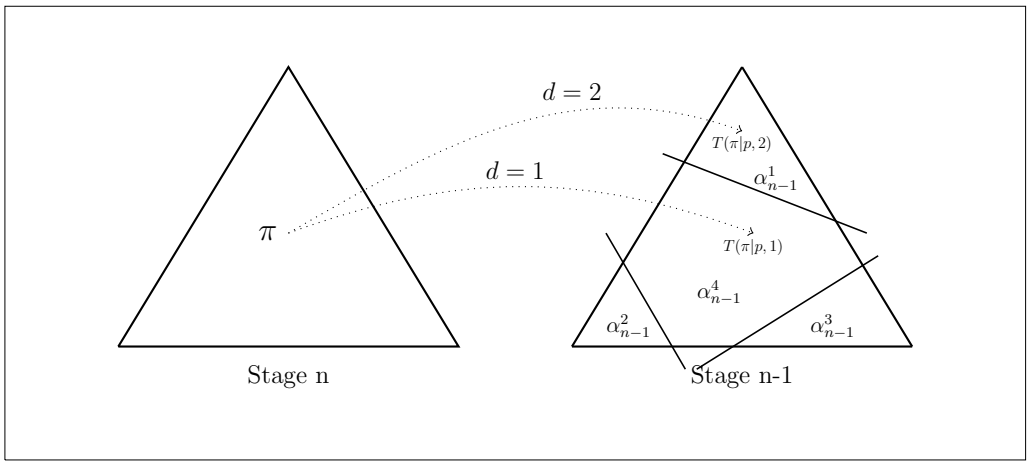


**Figure 2.1:** Belief vector transformation $T(\pi|p,d)$ for |S|=3

Smallwood and Sondik (1973) show the transformation of a belief vector through various observations from stage $n$ to stage $n-1$. Figure 2.1 is a reproduction of their description of this transformation where $T(\pi|p,d)$ stands for transformation of $\pi$ if $p$ is chosen and $d$ is observed. Figure 2.2 represents a 3 dimensional representation of value function if those four hyper-planes were to define it.

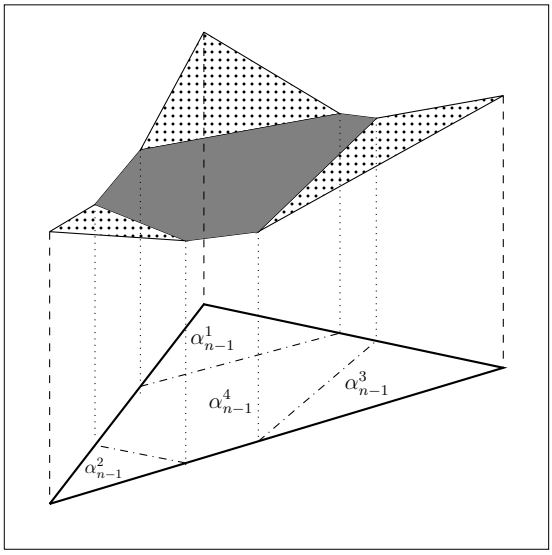


**Figure 2.2:** Sample value function for |S|=3

Sondik (1971) after proving the value function to be piecewise linear and convex, offers an algorithm to find the exact policy to maximise the profit. He shows that since there is a finite number of hyper-planes (regions dividing the belief simplex), all the exact algorithms are bound to finish and find the exact solution. The practical problem arises from the fact that the number of belief states increase exponentially after the transformation of each stage. In the next section we provide a brief description of these exact algorithms.

## 2.3 POMDP Exact Algorithms

A large part of POMDP research community have dedicated their efforts to heuristic approaches and use of approximations to overcome the complexity of the problems to be solved, but for two reasons we focus on exact methods for dynamic pricing problem. First, in recent years there have been big advances in exact methods that have increased the efficiency considerably such as incremental pruning method. Second, we believe an exact algorithm would offer a very useful benchmark in gauging the ability of proposed heuristic and approximation methods especially in the case of dynamic pricing problem where there have been no attempt to validate the results of the exact algorithms and their suitability. Our choice of the exact algorithms aims to reflect the evolution of

these algorithms over time as well as the underlying concepts that they as well as many heuristic and approximation methods in literature are based on.

### 2.3.1 Enumeration Algorithm

This algorithm first appears in Monahan (1982) in which Monahan provides a method to generate the whole spectrum of vectors that could define the value function for stage $n$ (since there is a finite number of $\alpha \in \mathrm{A}_{n-1}$ and prices and observations, there can only be finite number of vectors). Then one by one the vectors that are dominated by better vectors for the whole segment are removed. This exhaustive method is rather efficient in solving small problems, and useful as a benchmark to other more efficient exact algorithms or approximate solutions.

According to Monahan (1982), we can find a vector from a belief state for a specific action by the formula derived for $\alpha$:

$$\alpha_{n,i}^{p,d} = Q^{id}(p_n d + \sum_{k \in S} P^{ik} \alpha_{n-1,k}) \qquad \alpha_{n-1} \in \mathrm{A}_{n-1}.$$

A simple component-wise summation over observation $d$ will yield

$$\alpha_{n,i}^{p} = p_n \sum_{d<c_n} Q^{id} d + \sum_{k \in S} \sum_{d<c_n} Q^{id} P^{ik} \alpha_{n-1,k}^{l(\pi_{n-1},p,d)}, \tag{2.6}$$

where $l(\pi, p, d)$ is defined as

$$l(\pi, p, d) = \arg\max_{m} \left[ \sum_{k \in S} \sum_{d<c_n} Q^{id} P^{ik} \alpha_{n-1,k}^{m} \right],$$

and $m$ is an index over all possible $\alpha$ vectors developed through (2.6).

It is noted that although the belief space is uncountable, the $\alpha$ vectors are finite and thus each belief state can lead to only a finite number of vectors from previous stage which in turn will lead to finite number of new vectors.

And thus we can readily calculate $\alpha$ from information at hand for every period followed by the value function:

$$V_n^*(\pi, c) = \max_{p} \left\{ \sum_{i \in S} \pi_{n,i} \alpha_{n,i}^{p} \right\}. \tag{2.7}$$

### 2.3.2 Two Pass Algorithm

Although Smallwood and Sondik (1973) are the first to exploit the value function shape in the POMDP problem and have set the ground for all the later research in the area, their "two pass" algorithm is far from efficient for a few simple factors extensively discussed in Lovejoy (1991a), Mukherjee and Seth (1991) and Cassandra (1998).

Instead of generating all the possible vectors as in enumeration algorithm, in two pass algorithm it is suggested to start with finding the optimal value of the value function for an initial belief point and then through a set of constraints and a consequent application of a linear program, find the boundaries of a region which gives the value function its optimal value for all the belief points on that region. The points in the extremes, can then be added to the list of unexplored regions. When all the points have been checked and their corresponding regions have been studied, the vectors that are left are the optimal vectors.

Assuming we have calculated $\alpha_{n-1}^k$, the algorithm seeks to find the new set of vectors $\alpha_n^k$. For this, an initial belief vector ($\pi_n^0$; for example the corner of the simplex) is chosen and using equation (2.7) the optimum price and $\alpha$ vector is calculated. These are denoted by $p^*$ and $\alpha_n^*$. The next step is to identify the region on the belief space over which $\alpha_n^*$ is the optimal and dominant vector. The idea behind this is to move the $\pi$ away from initial $\pi_n^0$ and calculate $V_n(\pi_n, c_n)$ and the corresponding $\alpha$ vector for each of these $\pi$ values until we reach a point where $\alpha \neq \alpha_n^*$.

For any observation $d$, the following inequality describes the outer boundary for $\alpha_n^*$:

$$\sum_{i\in S}\sum_{j\in S} \pi_{n,i}\, Q^{id}\, P^{ij}[\alpha_{j,n-1}^k - \alpha_{j,n-1}^l] \geq 0 \quad \forall j, \tag{2.8}$$

where $l$ is short for $l(\pi^{n,0}, p, d)$.

Also the price for the region must be optimal:

$$\sum_{i\in S} \pi_{n,i}[\alpha_{i,n}^* - \alpha_{i,n}^p] \geq 0 \;\; \forall p. \tag{2.9}$$

And finally the general conditions for the belief vector:

$$\pi_{n,i} \geq 0 \quad for \quad 1 \leq i \leq N \quad and \quad \sum_{i=1}^{N} \pi_{n,i} = 1. \tag{2.10}$$

But Sondik (1971) observes that only some of these inequalities define the region. He thus introduces a linear programming algorithm to further refine the set of constraints above. If we represent the set of inequalities in (2.8) and (2.9) as $\pi_n \cdot b^m \geq 0$, then the answer to

$$\begin{aligned} \min_{\pi} \quad & \pi_n \cdot b^k \\ s.t. \quad & \pi_n \cdot b^m \geq 0 \quad m = 1, 2, ..., \\ & \pi_{n,i} \geq 0, \\ & \sum_{i=1}^{N} \pi_{n,i} = 1, \end{aligned}$$

will be zero if the inequality $k$ forms the boundary. The rest of $\alpha$ vectors can be discarded.

### 2.3.3 Incremental Pruning Algorithm

This algorithm was first suggested by Zhang and Liu (1997) as an alternative exact approach to solve the POMDP model and has been developed and analysed further by Cassandra et al. (1997), Kaelbling et al. (1998), Feng and Zilberstein (2004) and Spaan and Vlassis (2005) among others.

Incremental pruning is a method similar to the above algorithms in that it explores the convexity and piecewise linearity of the value function. Recall the decomposed value functions as in (2.4):

$$V(\pi_n, c_n) = \max_p V^p(\pi_n, c_n).$$

$$V^p(\pi_n, c_n) = \sum_{d<c_n} V^{p,d}(\pi_n, c_n).$$

$$V^{p,d}(\pi_n, c_n) = \sum_{i \in S} \pi_{n,i} Q^{id}(p_n d + \sum_{k \in S} P^{ik} \alpha_{n-1}^{l(\pi_{n-1},p,d)}).$$

As discussed earlier, each of these value functions are piecewise linear and convex and can be represented by a set of vectors. For each value function there exists a unique minimum-sized vector set. We will use the symbols $\mathcal{V}$, $\mathcal{V}^p$, and $\mathcal{V}^{p,d}$ to refer to these minimum-sized sets.

If we assume $\mathcal{U}$ and $\mathcal{W}$ denote two sets of vectors, $\mathcal{U} \oplus \mathcal{W}$ is called the *cross sum* of the two sets and is simply a point-wise summation of the two:

$$\mathcal{U} \oplus \mathcal{W} = \{u + w | u \in \mathcal{U}, w \in \mathcal{W}\}.$$

$\mathbb{PR}(\mathcal{U})$ is an operator that takes the set of vectors $\mathcal{U}$ and reduces it to its minimum set. Thus $u \in \mathbb{PR}(\mathcal{U})$ if and only if $u \in \mathcal{U}$, and for any $\pi \in \Pi$ and $u' \neq u \in \mathcal{U}$ the following condition holds: $u \cdot \pi > u' \cdot \pi$.

These two operators enable us to compute the minimum set of vectors:

$$\mathcal{V}' = \mathbb{PR}(\cup_p \mathcal{V}^p) \tag{2.11}$$

$$\mathcal{V}^p = \mathbb{PR}(\oplus_d \mathcal{V}^{p,d}) \tag{2.12}$$

$$\mathcal{V}^{p,d} = \mathbb{PR}(\{\alpha_i^{p,d} | \alpha_i \in \mathrm{A}_{n-1}\}) \tag{2.13}$$

where $\alpha_i^{p,d}$ is computed by:

$$\alpha_i^{p,d} = Q^{id}(p_n d + \sum_{k \in S} P^{ik} \alpha_{n-1,k})$$

The pruning step is the combination of a simple domination check where every vector is checked for point-wise domination over others, and solving the following linear program:

$$\begin{aligned} \max \ & b \\ & \pi \cdot (w - u) \geq b, \forall u \in \mathcal{U}, \\ & \sum_{i \in S} \pi_i = 1. \end{aligned}$$

Given a vector $w$ and a set of vectors $\mathcal{U}$ that does not include $w$, the linear program determines whether adding $w$ to $\mathcal{U}$ improves the value function represented by $\mathcal{U}$ for any belief state $\pi$. If it does, the variable $b$ optimised by the linear program is the maximum amount by which the value function is improved, and $\pi$ is the belief state that optimises $b$. If it does not, that is, if $b = 0$, then $w$ is dominated by $\mathcal{U}$.

The main focus in incremental pruning algorithm and its predecessors is the set of vectors and their transformation through stages. At each value iteration step, we are faced with a set of vectors that define our value function. But of the three steps to generate these new vector sets, the most complex is (2.12) as the size of cross sum grows exponentially by size of $d$. Incremental pruning algorithm overcomes this complexity by realising that operators $\mathbb{PR}$ and $\oplus$ can be interleaved:

$$\mathbb{PR}(\mathcal{U} \oplus \mathcal{V} \oplus \mathcal{W}) = \mathbb{PR}(\mathcal{U} \oplus \mathbb{PR}(\mathcal{V} \oplus \mathcal{W}))$$

Applying this on (2.12) we have:

$$\mathcal{V}^p = \mathbb{PR}(\mathcal{V}^{p,d_1} \oplus \mathbb{PR}(\mathcal{V}^{p,d_2} \oplus ...\mathbb{PR}(\mathcal{V}^{p,d_{k-1}} \oplus \mathcal{V}^{p,d_k})...)),$$

which reduces the number of linear programs to be solved considerably.

For a detailed account of incremental pruning methods and different variations refer to Feng and Zilberstein (2004), Cassandra et al. (1997) and Cassandra (1998). We have used the generalised incremental pruning variation developed and implemented by Kaelbling et al. (1998).

## 2.4 Performance Assessment

We program all three algorithms in C++ and run using Microsoft Visual Studio, with the academic version of the IBM ILOG CPLEX 12.2 for the linear programming sections. We design a set of sample problems in order to test the efficiency of the methods according to the following combination of parameters:

$N \in \{3, 6, 9, 12\}$ number of stages (decision periods)
$|S| \in \{2, 4, 8, 16\}$ number of demand states
$|d| \in \{2, 5, 10, 20\}$ number of possible observations
$|p| \in \{2, 5, 10\}$ number of possible price changes (actions)

The combination above amounts to 192 sample problems. These ranges were designed with dynamic pricing of fashionable products in mind, where the number of stages (price changes) and the size of the price set can be limited to single digits. Especially as we are dealing with short life cycle goods which need to be sold in less than six months, the combination of problems above provide a reasonable assessment platform for our problem. A similar concept is applicable to the number of demand states considered in the model. Interested readers are invited to refer to Feng and Gallego (2000) for a detailed discussion. Please note that the number of demand states and the number of possible observations are parameters of the sensitivity set for the model. It is possible to consider integers representing the demand at each stage to map to the demand state of the model, but we can, in order to simplify the model, consider bands of numbers as the

demand state or observation sensitivity levels. Thus when demand state is 2, we essentially have as a parameter of the model, high or low demand. The same concept applies to observation sensitivity levels.

We test the robustness of the algorithms for handling uncertainty by performing simulation runs. For each of the sample problems generated from the above combinations, we generate random demand function sets (sets of transition and observation matrices) that represent uniform demand transition probabilities. The uniformity of probabilities is an appropriate way to test the ability of these exact algorithms, as it renders the learning aspect of any procedure redundant.

As a simple comparison and to establish the scale of the efficiency of these methods, we also wrote an algorithm based on the classic dynamic programming approach. Table 2.1 depicts the solution times of all three methods and the dynamic programming approach for a subset of the above defined sample sets (all the samples with three stages and eight states). This example illustrates the efficiency of the algorithms we are employing, as they avoid the explosion of the computational time as can be seen in the case of classic dynamic programming method.

| observation | price | enum. | two pass | inc. prun. | dyn. prog. |
|---|---|---|---|---|---|
| 2 | 2 | 16 | 16 | 0 | 16 |
| | 5 | 0 | 93 | 109 | 63 |
| | 10 | 15 | 577 | 312 | 359 |
| 5 | 2 | 32 | 0 | 0 | 62 |
| | 5 | 0 | 0 | 16 | 483 |
| | 10 | 78 | 62 | 78 | 2932 |
| 10 | 2 | 25 | 6 | 9 | 289 |
| | 5 | 20 | 8 | 9 | 2870 |
| | 10 | 37 | 13 | 14 | 19778 |
| 20 | 2 | 15 | 7 | 11 | 903 |
| | 5 | 38 | 13 | 12 | 7856 |
| | 10 | 72 | 52 | 31 | 145723 |

**Table 2.1:** Solution times for problems with three time periods and eight possible states and varying number of possible observation and price sets

As the demand functions are random, the evolution of value function (set of $\alpha$ vectors) expand faster for some samples than to others, leading to longer solution times. For the following analyses, we calculate the mean processing time for each problem instance with regard to thirty different random demand functions. We also run each sample of the combination above ten times separately to acquire a reliable average CPU time for the solution. Our numerical studies were run on a PC running windows 7, with an Intel Core2 Duo CPU with 3GHz speed for each core. The internal memory (RAM) for the machine was 4GB. We report a summary of main points for each algorithm with regard to the computation time and the number of vectors they have to process at each stage.

### 2.4.1 Computation Time

For the enumeration algorithm, as the vectors are not pruned efficiently, the size of the problems grow exponentially, especially for problems where we consider higher number of observations, and this leads to very long calculation times. In Figure 2.3, for example, which shows the time taken by each algorithm based on the number of stages, many problems in 9 and 12 stage problems were aborted because they were not solved within the time limit (a one hour limit for all the execution times) which registers as an upper limit of 3600000 milliseconds.

The interesting points about enumeration algorithm is that the high times are for higher stage and observation numbers, but the performance is poor for low price numbers compared to higher numbers (2 against 10). This can be justified by the fact that the larger number of the price set offers flexibility in maximising the profit based on the evolution of the demand while a low number of prices to be set limits the ability of the manager to make use of the various opportunities risen by the demand spikes. However, a very large price set would add too much complexity that would offset the benefits of the flexibility offered.

The two pass algorithm performs significantly better than enumeration as it manages to filter more of vector projections at each step. However, although the algorithm manages to solve the most complex of our sample problems eventually (in the most extreme case, in about 50 hours) it uses significant amounts of

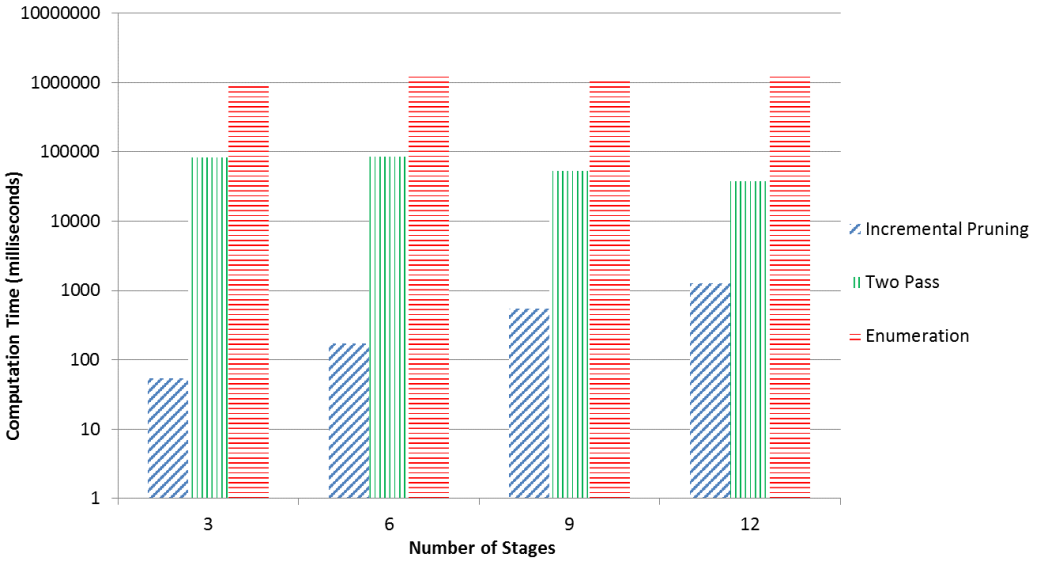


**Figure 2.3:** Average time taken by each algorithm with regard to the number of stages (price changes)

memory to achieve this. We introduced a memory limit (2 GB) as well as the one hour time limit in order to be able to finish our runs in reasonable times. Thus we have time results for all the samples by this algorithm, but as can be seen in Figure 2.6, for high observation counts the solution times are reported as the upper bound.

The two pass algorithm relies heavily on the LP solver and does not perform well for problems with high number of observations and high number of prices, as can be seen in Figure 2.7. Incremental pruning algorithm outperforms the other algorithms in all instances of the problems. The results are encouraging as the time spent on the most complex problems does not exceed mere seconds. We believe that the upper bound of the parameters in our sample problems represent what would be encountered in real life situations. As such, incremental pruning algorithm exhibits the fundamental ability to tackle most real-world problems.

### 2.4.2 Size growth

Figure 2.9 shows the growth in size (average maximum number of vectors) for each problem instance reached by each algorithm by the number of states and stages. It gives an indication of how each algorithm compares to others in terms

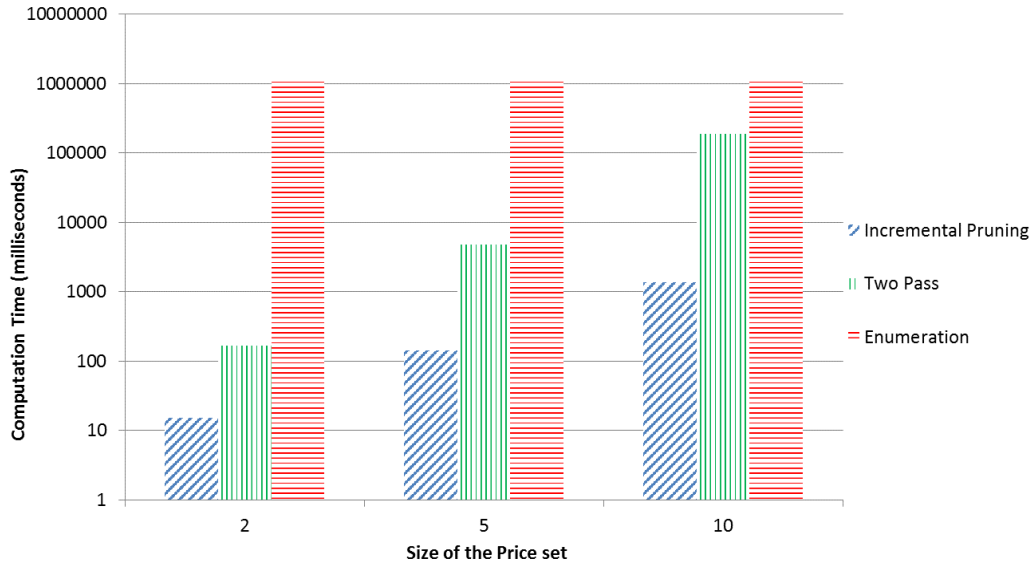


**Figure 2.4:** Average time taken by each algorithm with regard to size of the price set

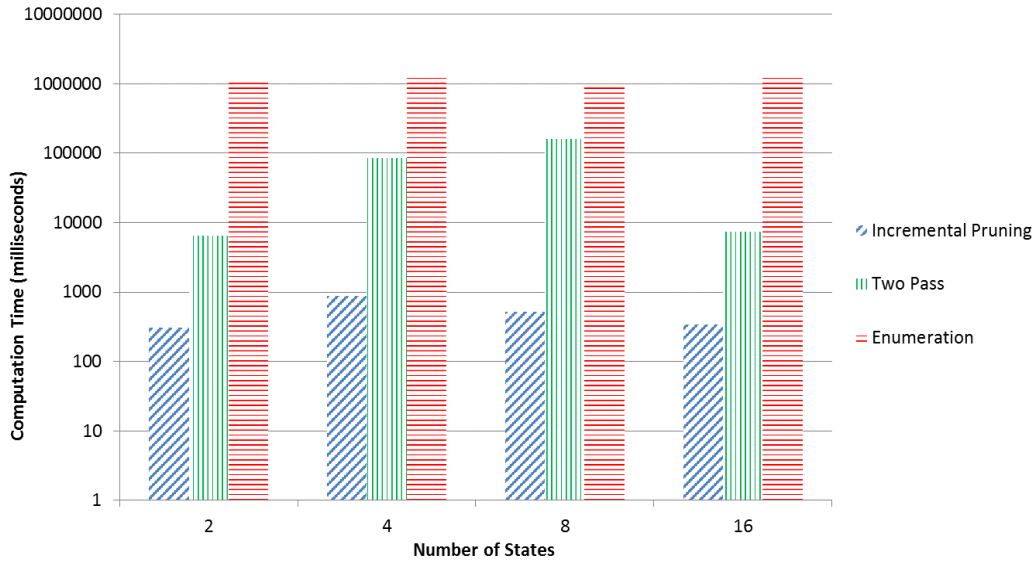


**Figure 2.5:** Average time taken by each algorithm with regard to size of the state space

of the efficiency of filtering non-dominant vectors from stage to stage. Evidently, the size of the vector sets to be processed increases exponentially by the size of the price set as can be seen in Figure 2.10.

The growth of the problem size as we discuss in this section, at the first glance corresponds to number of possible observations and number of stages in

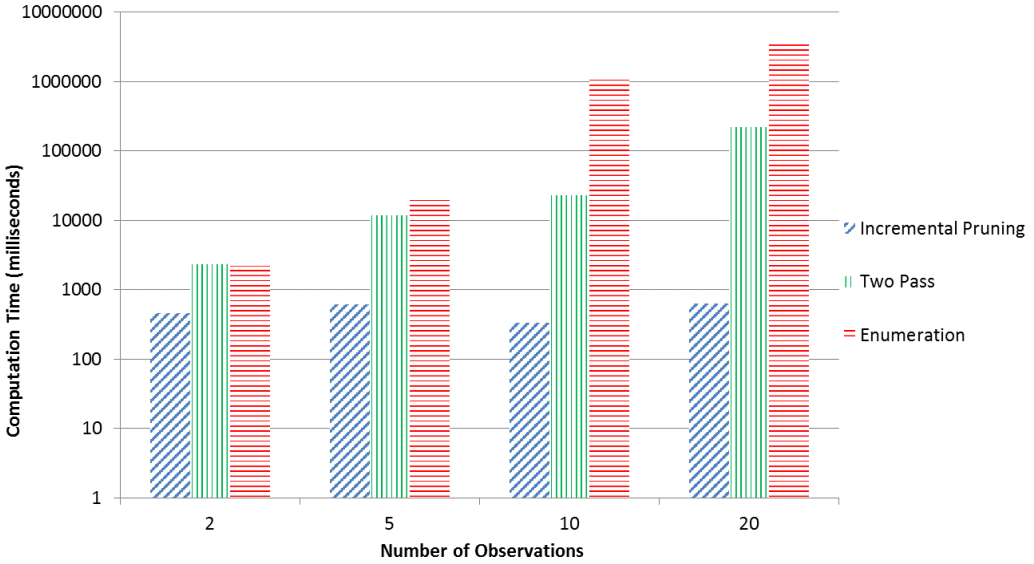


**Figure 2.6:** Average time taken by each algorithm with regard to number of observations

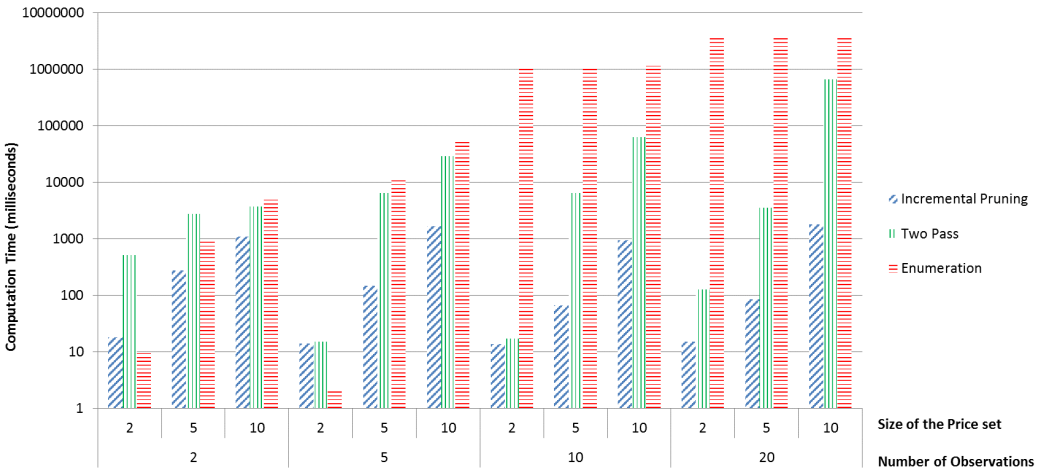


**Figure 2.7:** Average time taken by each algorithm with regard to number of observations and the size of the price set

the problem. It has negative relation to the number of possible prices, though this is more apparent in the enumeration and two pass algorithms. The number of states does not have much effect on the increase of vector numbers that define the value function. More interestingly, there are indications that for more complex problems, as in the biggest in our sample problems, the number of vectors does not follow the trend of smaller problems. This may be the result of the fact that a bigger problem yields bigger decision flexibility unlike the

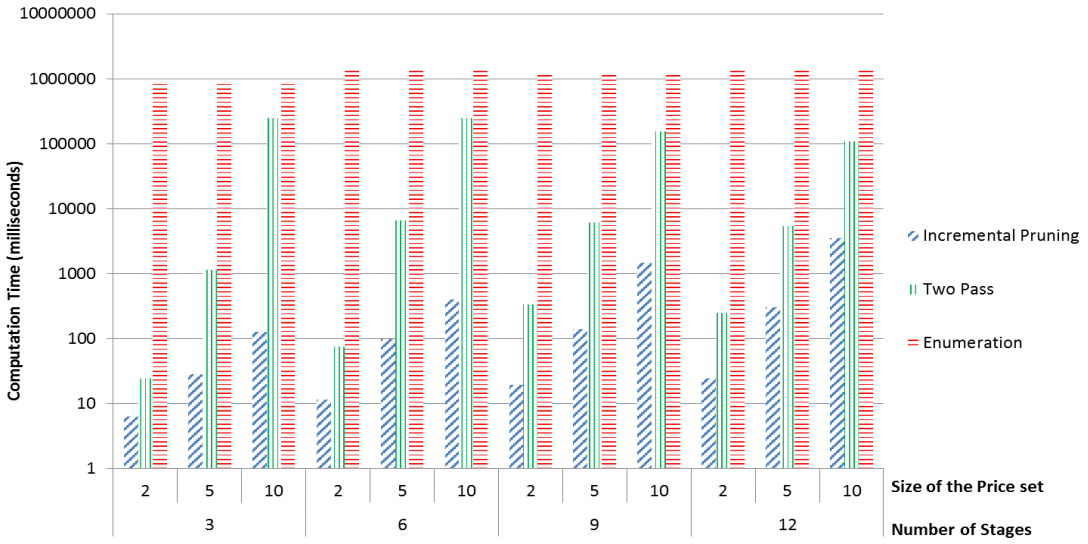


**Figure 2.8:** Average time taken by each algorithm with regard to number of price change stages and the size of the price set

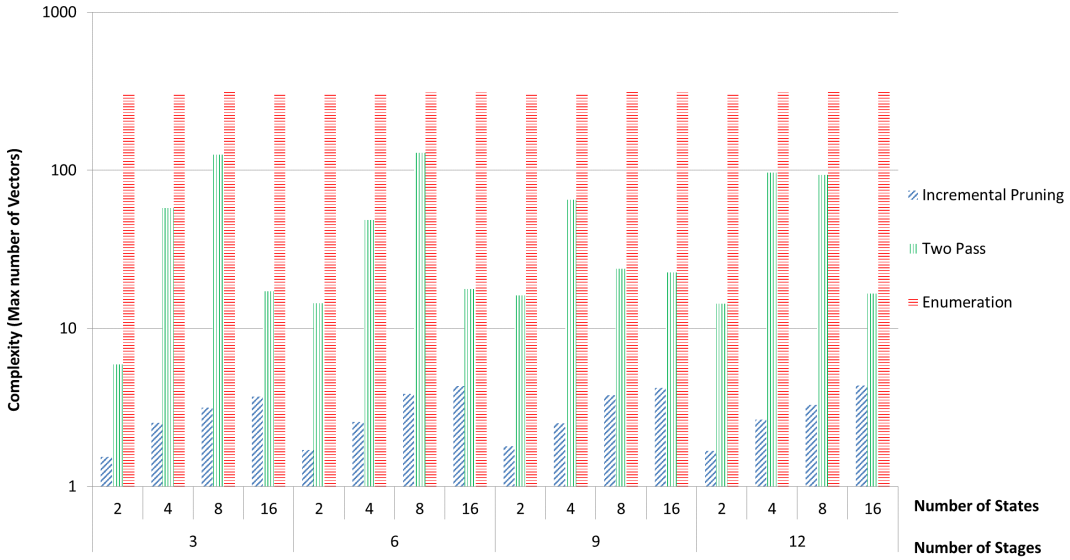


**Figure 2.9:** Average maximum number of vectors processed by each algorithm with regard to the number of stages and number of states

smaller problems.

As regards the optimal pricing policy, which is the main outcome of the algorithms, for the majority of the problems it generates a constantly declining price over time, although the exceptions vary substantially. This is also in accordance with prior knowledge, as the optimal policy for the deterministic

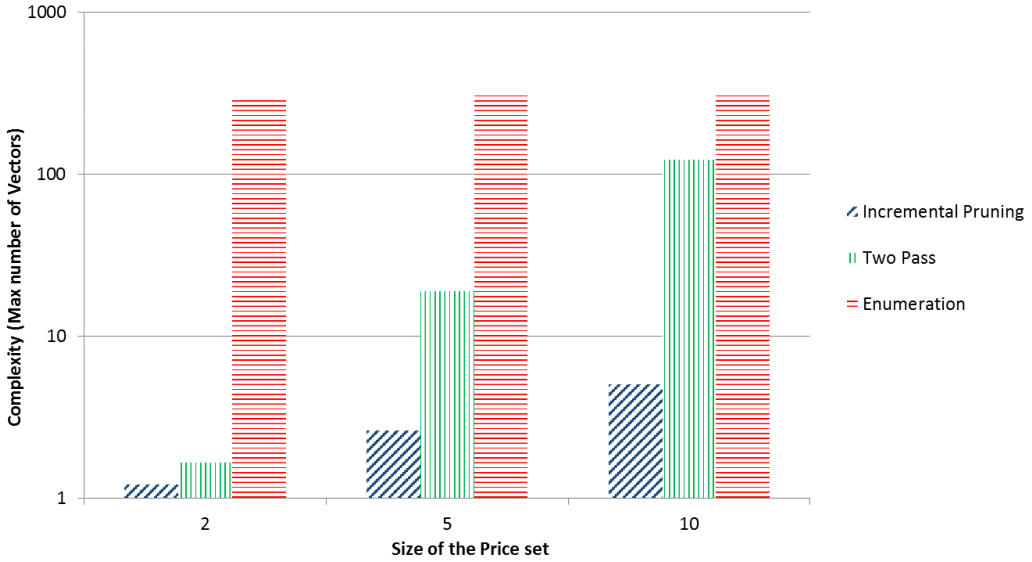


**Figure 2.10:** Average maximum number of vectors processed by each algorithm with regard to the size of the price set

instances of the problem has been shown to follow this pattern (Gallego and Van Ryzin, 1994). The most important trade-off in the pricing process is the level of starting inventory. Our results are consistent with the notion that pricing the goods at the highest is a good strategy when the time left to the end of season is long enough. Alternately it is profitable in general to set high prices initially, if the number of remaining items is high with respect to the average possible observation in the environment. We tested a number of simple policies (such as a greedy policy and constant price drop policy) against the policy drawn from the POMDP algorithms and found that the exact policy is surprisingly close to the myopic policies when the number of decision stages is few or the number of possible observations are low. However the behaviour of the optimal policy is less predictable when the number of planning stages are high. This is justified by the added uncertainty in demand evolution as number of decision periods increases.

The major results of this paper can be described in two directions. We show that the exact algorithms devised for the solution of POMDP models are effective in solving problems in accordance with retail dynamic pricing problem. In particular incremental pruning algorithm has excellent performance across the board. This performance increases the feasibility of its implementation on

more complex pricing problems or its integration with on-line decision making tools. On the other hand we gain insight into the various complexities that are added to the problem by various parameters of the problem. The vector size at each stage is highly volatile when we change the price set combination while it is rather straightforward in response to stage numbers or number of the demand states.

## 2.5 Conclusion

This paper implements exact algorithms for POMDP model applied to a dynamic pricing problem of short-life-cycle products. We consider the pricing of perishable goods in a retail environment characterised by a high degree of demand uncertainty and rapidly changing demand parameters. With the objective of maximising expected revenues, the seller needs to post prices in a way that achieves an optimal balance between current and future gains. To our knowledge this is the first attempt at implementing exact POMDP algorithms on the dynamic pricing problem. We show that not only the exact methods offer optimal results in reasonable time, they are capable of solving a large range of problems that can be readily utilised in the industry. Especially incremental pruning algorithm exhibits satisfactory performance.

A theoretical characterisation of the effectiveness of the exact methods would be a valuable step in providing a better understanding of the recently evolved exact methods in POMDP applications. We believe that further research in this direction would be a valuable input for the research in dynamic pricing field and would provide a suitable framework for industrial decision-support tools, in particular in the retail sector. Another future research direction is to consider the case of dynamic pricing of a family of products where the price of each product has cross effects on the demand of other products.

# Chapter 3

# Dynamic Pricing of Products with Autoregressive Demand


## Abstract

Dynamic pricing is an increasingly useful tool in control and manipulation of demand. In this paper, we use a stochastic dynamic programming model for dynamic pricing of products that show autoregressive behaviour. We propose a linear representation of demand evolution, in which demand evolves based on the chosen price by the seller, the demand in the previous period and the dynamics of the market. We discuss the various demand models present in the literature and examine the suitability of an autoregressive demand model for high end fashionable products. These products are characterised by strong following among consumers and critics, and a considerable word-of-mouth potential. For the basic model without any constraints, we develop a simple solution in order to understand the structure of the value function. For the extended model with constraints on price and capacity we develop an approximate dynamic programming method that provides near optimal solutions and substantial advantages over non-constrained or naive implementations of the pricing problem.

# 3.1 Introduction

## 3.1.1 Background

Revenue management is the study of demand and inventory dynamics in order to maximise revenue. It offers various tools to the decision maker, the most important of which are pricing, inventory control and marketing practices. Dynamic pricing has emerged in recent years as a more standalone research stream, although combined study of pricing with either inventory policies or marketing strategies is also significant in revenue management literature. Products such as fashionable clothing with short selling seasons and long lead times present a specific range of problems that differ from classic inventory or revenue management models because of infeasibility of replenishment during season and the behaviour of the customers involved. The demand for high end fashionable products present more complex dynamics, because they are closely followed by media and fashion critics. They also involve more dedicated customers which react to the new products and price changes in more complex ways. Furthermore, the demand model and its relation to the price in the majority of revenue management literature is assumed to be deterministic. The number of academic papers that address the underlying uncertainty and provide stochastic demand models are relatively limited.

In this paper, we provide a demand model that takes into account elements such as word-of-mouth potential and customer response to advertisements of a specific product or brand in its evolution over time. We consider high end fashionable products which represent the range of products that are non-durable (substantially decline in value at the end of the season) and are usually accompanied by advertisement campaigns. We assume an autoregressive demand process for this product range and develop a representative model. Finally we provide solutions that take into account these characteristics and analyse the implications of our assumptions.

### 3.1.2 Literature Review

Dynamic pricing literature can be found in a number of different research fields. The main field of development has been the literature on revenue management and yield optimisation which have their roots in the management science and operational research fields. McGill and Van Ryzin (1999) provide a review of revenue management and various issues in that general field. Bitran and Candentey (2003) present a summary of the pricing models in revenue management where the products are perishable and replenishment is not possible. Another main research stream derives from inventory management literature, which is concerned with joint pricing and inventory decisions. Elmaghraby and Keskinocak (2003) review the literature for dynamic pricing of perishable products with fixed inventory. Chan et al. (2004) provide a comprehensive review of the pricing literature in a supply chain context. Another stream strongly related to dynamic pricing is in the marketing literature where the attention of the research is on how the markets behave and the effects of brand loyalty. Rao (1984) and Rao (2009) provide a synthesis of research on pricing from marketing science point of view. Newsvendor problem is another field that is frequently generalised into pricing by extending single decision period to multiple periods. Petruzzi and Dada (1999) offer a review of newsvendor problem in the context of dynamic and periodic pricing. Finally, Stole (2007) provides a survey of the pricing problems studied from the view point of price discriminations.

Among the literature dealing with dynamic pricing under uncertainty, the seminal work by Gallego and Van Ryzin (1994) investigate the dynamic pricing problem for perishable products with stochastic demand that is price sensitive. They obtain structural monotonicity results for the optimal demand intensity as a function of the stock level and the length of the horizon. The authors find the optimal pricing policy in closed form for a range of demand functions. For general demand functions, an upper bound on the expected revenue is proposed, which is based on the deterministic version of the problem. This bound is used to prove that simple, fixed price policies are asymptotically optimal as the volume of expected sales tends to infinity. Besbes and Zeevi (2009) consider the same problem but with an unknown demand function that is observed as

the season progresses. They consider a single-product dynamic pricing problem over a finite sales horizon with demand functions of unknown class. The objective of the model is to minimise the revenue loss relative to the maximal revenues that can be extracted when the demand function is known prior to the start of the selling season. According to Besbes and Zeevi (2009), the Bayesian approach despite its stylised analysis of the joint learning and pricing problem, suffers from significant shortcomings, most notably the observation that prior distribution of demand has unknown parameters. Bertsimas and Perakis (2006) address this aspect of the uncertainty and offer an alternative to the Bayesian formulation using a least squares approach in the context of a linear demand model. Chatwin (2000) studies the case of dynamic pricing of a fixed number of perishable items and finite horizon. The demand follows a Poisson distribution with an inverse intensity in relation to the price. They obtain structural characteristics for the problem e.g. the optimal price declines with time as the end of product life approaches; and at any given time the optimal price is non-increasing in the number of unsold items.

Soon (2011) provides a general review of literature concerning treatment of multiple products in a dynamic pricing context. Chen and Chen (2015) survey the literature on three classes of dynamic pricing problems: problems with multiple products, problems with competition, and problems with limited demand information. Zhu and Thoneman (2009) consider a monopolistic retailer that stocks and sells two products under stochastic demand. The demand of each product depends on the price of both products. They show that the retailer can significantly increase profit by managing the products together rather than solving the pricing problem for each product individually. Similarly, Broder and Rusmevichientong (2012) discuss a dynamic pricing model over finite horizon in a monopolistic setting. Under unknown model parameters, the seller determines a pricing policy that minimises the revenue loss based on maximum likelihood estimation of the model parameters. In contrast, Perakis and Sood (2006) study dynamic pricing of a perishable product in an oligopolistic market. They obtain the non-cooperative Nash equilibrium policies for competing sellers. Levin et al. (2009) present a dynamic pricing model for oligopolistic firms facing strategic consumers. The objective of the competing firms is to

maximise total expected revenues. They provide equilibrium optimality conditions, and prove monotonicity results for special cases. Liu and Zhang (2013) extend the game theoretic approach by considering competition between two firms offering vertically differentiated products to strategic customers. They discuss the effects of strategic behaviour of customers and the unilateral and bilateral commitments of either of the two competing firms.

The above selection of literature provides an overview of dynamic pricing problems where demand evolution follows a Bayesian updating in which some level of knowledge on the distribution of the unknown parameter is assumed. We now provide a closer focus on different demand evolution models in the literature in the context of dynamic pricing.

### 3.1.3 Demand Models

The majority of dynamic pricing literature treats demand models as independent over time and dependent only on price. We argue that in many cases the demand of a product is dependent not only on price, but also on time and is furthermore influenced by other exogenous factors such as market conditions. In this section, we provide an assessment of literature on demand models and provide reasoning for the suitability of an autoregressive representation of the demand process.

The majority of demand models are deterministic models where the sellers have perfect knowledge of the demand process i.e. customer behaviour is known throughout the time horizon considered. In this research, we assume a linear demand model with stochastic elements. Stochastic demand models are mainly formulated based on a specific probability distribution and the assumption that the mean of demand is a linear function of price either additively or in a multiplicative fashion.

Aviv and Pazgal (2005) provide a Markov decision process (MDP) approach to solve a dynamic pricing problem using a heuristic method called Information-Structure Modification which takes into consideration the different shapes of demand function and transition matrix. Farias and Van Roy (2010) study a problem of dynamic pricing with limited inventory with the objective of max-

imising expected discounted revenue over an infinite time horizon. The authors study an uncertain demand function and develop a heuristic approach. A key result is that uncertainty in the customer arrival rate impacts the price positively. Fan et al. (2005) consider the dynamic pricing problem in an optimal control setting and obtain optimal prices for a demand function that depends on the price, cumulative sales up to the current period and market potential. They find structural properties for the optimal solution i.e. the unit revenue is linearly related to the demand elasticity of price and is constant over time when the demand elasticity is constant. Araman and Caldentey (2009) offer a dynamic pricing model for the non-perishable product where demand is driven by a price-sensitive Poisson process that depends on an unknown parameter as a proxy for the market. The demand is updated as time and available information (prices and sales) evolve. They derive structural properties of the optimal solution, and provide a simple and efficient approximate solution.

In general, customers are less informed about a product at the start of its life cycle. Future demand can be affected via different advertisement campaigns and various word-of-mouth potentialities of the product. According to Elmaghraby and Keskinocak (2003), for most non-durable goods, demand is independent over time, i.e. current sales do not have an impact on future sales. This is certainly true for most necessity items, such as milk and bread, where consumers make frequent repeat purchases. Also for most seasonal goods, the selling horizon is usually too short to allow for any significant knowledge acquisition by customers to have an impact on the demand. However this behaviour is reversed for high value products or products that possess habit building traits or have a knowledgeable customer base. An example pertaining to this situation is the high end fashionable items that have a dedicated following and strong media presence that act as a word-of-mouth mechanism towards the loyal and fashion aware customers. They are also accompanied by focused ad campaigns that target the new and changing customers and there is uncertainty surrounding new products in the marketplace. Another example of this kind of product is the high end technological products such as smartphones and tablets which are constantly observed by the media and experts and tend to form loyalties among customers.

According to Perakis and Sood (2006) a key assumption in the demand representation is that the demand functions and associated uncertainties are separable over time-periods. That is, "the buyers are influenced primarily by the current period prices they see..." They suggest as a next step in their research an enhanced model that considers how the history of demand will affect the present demand function (buyer behaviour and their reaction to price changes). A number of recent models have been developed which allow for dependent demand between periods. The models of Anderson and Wilson (2003), Su (2007) and Wilson et al. (2006) consider markets with strategic consumers who may wait for firms to offer lower prices. Another form of dependency between periods is caused by reference effects. Popescu and Wu (2007) examine how reference effects impact the choices made by consumers. The authors argue that customers respond to a price by comparing it to past prices they were exposed to, the so-called "reference prices". They find that if reference prices are not taken into account, firms will be faced with lost revenues because of setting the prices too low.

According to Xu and Hopp (2006) the majority of the dynamic pricing literature assumes customer arrivals to follow a Poisson distribution, implying independent demands across intervals. They argue that because in reality there is a correlation in the demand structure, the historical demand information can be very valuable in dynamic pricing. They offer a geometric Brownian motion to represent the demand evolution, which is Markovian and partially captures demand correlation. Its use in place of a Poisson process would cause pricing policies to depend on current demand information and would permit integration of demand forecasting with pricing. Startz (1989) provides empirical evidence that purchases of non-durable goods and services approximately follow an autoregressive process of level one (AR(1)). He concludes that the forecast contribution of available information for non-durable products is statistically significant. In the sphere of non-durable products, Duffy (2003) tests for the influence of advertising on the inter-product distribution of consumer demand. The long-run demand for seven categories of non-durable products is modelled. The author observes that lagged effects are important in influencing the demand of products with habit formation characteristics such as alcohol and tobacco.

This effect is often modelled by inclusion of a first-order lagged value of the consumption variable.

In this paper we consider a non-durable product that has a strong word-of-mouth potential and is accompanied by an advertisement campaign. For the purposes of clarity, in the remainder of this paper we assume the product under consideration to be a high end fashion product. We assume an autoregressive demand process for this product, in which demand is not only price elastic, but also corresponds to demand value in the previous period. As mentioned above, the demand models considered in literature do not accommodate such assumptions. We address this gap, by arguing that such assumption is justified through various real-world observations, e.g. some products may experience a lag in the intended boost in demand via advertisement or the time it takes for the word-of-mouth to spread around. We develop a demand model that represents these characteristics and discuss the framework resulting from incorporating such model with dynamic programming method. We then provide solutions to the model and analyse the implications of our assumptions through numerical analyses.

### 3.1.4 Contributions and Paper Structure

This research paper aims to contribute to the dynamic pricing literature in the following ways. We introduce a dynamic programming framework for the dynamic pricing problem with an autoregressive demand evolution model and provide a simple solution for the unconstrained model. The structural properties of the optimal pricing policy is explored through a range of numerical analyses. We discuss the relevance and application of autoregressive model in describing the demand evolution. We then expand the framework by incorporating product stock capacity and price constraints that exist in real-world applications into the dynamic programming model and study the structure of the resultant model. Finally, we provide an approximate dynamic programming solution to the extended model that utilises the structure of the problem and the uncertainty in state evolution to provide near optimal results.

The rest of this paper is organised as follows. In section 3.2 we first describe

the parameters of the dynamic pricing problem followed by description of the basic model. We then propose extensions to the model in section 3.3 and provide near optimal solutions using approximate dynamic programming. Section 3.4 provides the numerical experiments and assessment of the structural behaviour of the models discussed in sections 3.2 and 3.3. Finally in section 3.5 we conclude our discussion and propose potential research directions.

## 3.2 Problem Description

We assume the market to be a monopolistic (non-competitive) environment in which the decision maker can set prices independently for each period. We also assume a discrete time (multi-period) model in which the product is priced in each period. The season is defined as finite horizon which corresponds to non-durable products. Additionally, the salvage value is zero and any unsold item at the end of the horizon will be discarded. This assumption is widely adopted in the literature based on the observation that any problem with a fixed salvage value can be reduced to a problem with zero salvage value without loss of generality. The demand is stochastic with a white noise element that represents uncertainty. Finally, the demand is portrayed as an autoregressive behaviour pertaining to the advertisement and word-of-mouth and the delay inherent in the awareness of the customers about the product's existence and price information.

This problem definition is applicable to a wide variety of products with above characteristics; namely non-durable products with a strong base of loyal customers, but also a critical assessment in the market for their manufacturing and design quality and other merits. The high end fashionable products are usually accompanied by an ad campaign and a strong word of mouth among the loyal followers of the brand and the fashion industry in general through media and critical reviews. However, the same is true for high end technological products and modern gadgets that garner huge followings among consumers and critics. Sporting equipment that utilise the latest technologies such as bicycles also have strong following and similar demand evolution pattern to the model we are considering and all these products lose their value significantly at the

end of season and are usually cleared to discount stores.

We consider two variations of the problem. In the basic variation, a non-capacitated version of the problem is solved. In a later section we provide the framework to solve the capacitated problem to near optimal results. In the basic form of the problem, we assume unlimited supply of the product and thus at the last period there are no specific limitations on the amount of products. In the capacitated problem however, we assume the firm has a fixed stock ($\bar{S}$) of the product. It intends to sell this stock in a predefined time horizon, after which a new incarnation of the product will be introduced and the present product will be assumed decrepit. The demand in both cases is a linear function of price and previous demand, with random fluctuations representing the uncertainty in the underlying system.

The objective of the model is to maximise the sales revenue over a finite set of pricing periods. The price set at each period affects the demand for that period. However, the demand also evolves through time by an AR(1) process. Thus, the selection of the most profitable price requires an understanding of the demand model, which is acquired by observing demand at different prices and in different periods.

In addition to the above, the problem has some underlying assumptions that require special attention. The main underlying assumption is that all the random variables have a normal distribution. This will result in negative optimal prices, if the volatility of the demand is too large. In section 3.3.2 we provide an extended model that takes into account this probability. Closely related to this assumption is the consideration that the demand becomes negative following too large prices; an infinitely large price might be optimal as it would lead to a large profit, but in the following periods it will lead to very small and possibly negative demands if the demand model contains reverse relation between demand and price. The inclusion of price constraint in section 3.3.2 will curb the probability of negative demand, but we illicitly assume that the parameters are of reasonable value and demand cannot be negative. Finally, we assume a valid and accurate knowledge of the prior values of the demand model parameters. Especially if we are dealing with multi-product models with cross elasticities between product prices and their demand, the accuracy of the process of acquiring

the initial parameter values should be given a higher priority.

### 3.2.1 The Basic Model

The seller aims to maximise its revenue through maximising its sale of the product during $T$ periods. We assume the period length to reflect the common notion of season in retail of fashion products. For instance, a six month season is relatively long in this context and the assumption of period $t$ to have one week length would result in 24 decision periods. Denoted by $d_t$ is the demand in period $t$ at price $p_t$, where $t = 1, 2, ..., T$.

The objective of maximising the revenue may be expressed as:

$$\max \ \mathbb{E}\ [\sum_{t=1}^{T} p_t d_t].$$

We will deal with the constraints in a later section and build the underlying model that takes into account price and capacity constraints. At this stage, we provide the dynamic programming solution for the non-constrained and non-capacitated case as it provides the basis for these extended models. Furthermore the basic model without constraints would be suitable for certain situations where continuous production is part of the supply chain and the dynamics of the market is moderate with fairly stable price and demand. An example of this would be a seasonal product that will not depreciate and apart from the storage cost during off season will be of the same value. The demand for these range of products is also stable which limits the level of demand response the retailer can exercise with customers. Although this would result in limited benefit from the dynamic pricing, it would provide a basis for controlling the stock levels in a larger scale.

The demand evolution model has two distinct components: the dynamics of $d_t$ in the absence of our pricing (the fluctuations in demand caused by market dynamics and the delayed effects of the ad campaign and the word-of-mouth), and the impact that price $p_t$ has on the demand for this period. The former component is given by an AR(1) process, and the latter is simply a linear function of price. Thus the demand is linearly related to the demand in the previous period as well as the price. The law of motion for demand may be

expressed as:

$$d_t = \psi d_{t-1} - \theta p_t + \epsilon_t,$$

where $\psi$ is the autoregressive parameter, $\theta$ is the linear coefficient for the pricing impact on the demand (assumed throughout this paper to be positive) and $\epsilon_t$ is an independent, identically distributed variable with normal distribution and mean zero.

The basic ingredients for any dynamic programming problem are the state of the environment at time $t$, the control variable, the randomness, the value function, and the law of motion (Powell, 2011). In the context of the non-capacitated problem, the state at time $t$ consists of the demand realised at the previous period ($d_{t-1}$). The state variable summarises all the information the retailer requires in each period $t$ to make a decision. The decision variable at time $t$ is the price $p_t$. The market dynamics are characterised by the random variable $\epsilon_t$.

The compact model is:

$$\max \mathbb{E}\ [\sum_{t=1}^{T} p_t d_t] \tag{3.1}$$

$$s.t. \quad d_t = \psi d_{t-1} - \theta p_t + \epsilon_t. \tag{3.2}$$

The dynamic programming algorithm is based on the observation that a solution $p_1^*, p_2^*, ..., p_T^*$ must also be optimal for the remaining program at every intermediate period $t$. The Bellman equation which relates the optimal value of the objective function in period $t$ to its optimal value in period $t+1$ is as follows:

$$V_t(d_{t-1}) = \max_{\{p_t\}}\ \mathbb{E}[p_t d_t + V_{t+1}(d_t)],$$

where $\{p_t\}$ is the set of all available prices during period $t$.

By starting at the end (period $T$) and applying the above Bellman equation and the demand evolution model recursively, the optimal price and value function for each period can be derived as a function of the state variable that characterises the information the retailer possesses in each period. In particular

the value function $V_T(.)$ as a function of the state variable $d_{T-1}$ is given by:

$$\begin{aligned} V_T(d_{T-1}) &= \max_{\{p_T\}} \ \mathbb{E}[p_T d_T] \\ &= \max_{\{p_T\}} \ [\psi p_T d_{T-1} - \theta p_T^2], \end{aligned}$$

resulting in the optimal pricing for the last period:

$$p_T^* = \tfrac{\psi d_{T-1}}{2\theta},$$

and optimal value function:

$$V_T^*(d_{T-1}) = \tfrac{\psi^2 d_{T-1}^2}{4\theta}.$$

For the period before last we are faced with a less trivial Bellman equation:

$$\begin{aligned} V_{T-1}(d_{T-2}) &= \max_{\{p_{T-1}\}} \ \mathbb{E}[p_{T-1} d_{T-1} + V_T(d_{T-1})] \\ &= \max_{\{p_{T-1}\}} \ [p_{T-1}(\psi d_{T-2} - \theta p_{T-1}) + \frac{\psi^2(\psi d_{T-2} - \theta p_{T-1})^2}{4\theta}], \end{aligned}$$

yielding the optimal price and value function:

$$p_{T-1}^* = \tfrac{\psi(\psi^2-2) d_{T-2}}{\theta(\psi^2-4)},$$

$$V_{T-1}^*(d_{T-2}) = -\tfrac{\psi^2 d_{T-2}^2}{\theta(\psi^2-4)}.$$

By continuing in this fashion, although the formulation expands, we are able to recognise a pattern in the underlying structure of the optimal value for value function and optimal price. With the help of Mathematica software package we were able to confirm the following formulation to hold up until $t = 5$. Thus the optimal value for the value function during period $t$ is:

$$V_t^*(d_{t-1}) \cong (-1)^t \frac{\psi^2(\psi^2-4)^m}{4^j \theta(\psi^2-4)^l} d_{t-1}^2, \tag{3.3}$$

and optimal price as:

$$p_t^* \cong \frac{\psi}{(t+2)\theta} \left(\frac{(\psi^2-4)^l}{3(\psi^2-2)^l}\right)^{(-1)^t} d_{t-1},$$

where $l = \left\lceil \frac{t}{2} \right\rceil$, $m = \left\lceil \frac{t-1}{2} \right\rceil$ and $j = \left\lfloor \frac{m+1}{l+1} \right\rfloor$.

Although, as mentioned before, the applicability of the basic model on our problem is limited because we have ignored the capacity and sign constraints (for

price and demand), it would be useful to look at its potential implementation in a real-world situation. The retailer can maximise the revenue over a fixed period based on the demand during the previous period and the values for $\psi$, $\theta$ and $\epsilon_t$ from historical data. Thus in a situation where the notion of stock and capacity can be ignored (such as a continuous production where the holding costs are negligible), the model above can be used to provide an upper bound on the expected revenue under the market volatility deduced by $\epsilon_t$. The basic model also provides an easy to compute solution that could be the basis for developing efficient numerical methods where the computational speed can help explore the structure of the value function and how it responds to different parameters of the model. Furthermore, we can use the basic model as a yardstick to benchmark the computational performance and ability in handling the addition of constraints of the more complicated methods. In this paper, we use the basic model and above results for evaluation of our results from the approximate dynamic programming method (section 3.3.3) and to develop the naive method (see section 3.4) that is a heuristic application of constraints on the basic model. However, for the problem at hand, it is essential to include capacity and price constraints in the model in order for it to be applicable to real-world situations we intend.

## 3.3 Extensions to the Basic Model

In this section we present the necessary extensions to the model in section 3.2.1 and provide an approximate dynamic programming solution to the resultant model. We look at the case of predefined capacity that has to be sold by the end of the season. This is a more general problem to which a closed form solution does not exist.

Another important extension to the basic model is the inclusion of price constraints in the model described in section 3.2.1. Because of the assumptions made in the non-capacitated model, there is a positive probability that the optimal pricing policy contains prices outside the practical range. To avoid such results, a constraint on the price must be included.

### 3.3.1 The Capacitated Model

Suppose that at the start of period 1 the seller begins the season to sell $\bar{S}$ products, and this program must be completed by the end of period $T$ as a new product will be launching in period $T+1$. We call this situation capacitated because the capacity assigned to the product is a hard constraint and will affect the state of the system. At any period, if the demand is bigger than the number of products left, we will have to choose the number of products left as the actual sales. We introduce a revised version of the demand evolution to account for this constraint where $d_t$ is the actual sales at period $t$. The state of the system at time $t$ consists of the sales at the previous period ($d_{t-1}$) and the number of products that are yet to be sold at the start of period $t$ ($z_{t-1}$). The decision variable at time $t$ is the price $p_t$.

The state variable $z$ evolves based on the following equation:

$$z_0 = \bar{S},$$

$$z_t = z_{t-1} - d_t.$$

To put these considerations into a compact model we have:

$$\max \mathbb{E}\,[\sum_{t=1}^{T} p_t d_t] \tag{3.4}$$

$$s.t. \quad d_t = min(\psi d_{t-1} - \theta p_t + \epsilon_t, z_{t-1}), \tag{3.5}$$

$$z_0 = \bar{S},$$

$$z_t = z_{t-1} - d_t. \tag{3.6}$$

The Bellman optimality equation is as follows:

$$V_t(d_{t-1}, z_{t-1}) = \max_{\{p_t\}} \quad \mathbb{E}[p_t d_t + V_{t+1}(d_t, z_t)].$$

By starting at the end (period $T$) and applying the Bellman equation above and the demand evolution model recursively, the optimal price and value function for each period can be derived as functions of the state variables that characterise the information the retailer possesses in each period.

For the particular case of period $T$ we define the value function $V_T(.)$ as a

function of two state variables $d_{T-1}$ and $z_{T-1}$ which can be obtained by:

$$\begin{aligned} V_T(d_{T-1}, z_{T-1}) &= \max_{\{p_T\}} \mathbb{E}[p_T d_T] \\ &= \max_{\{p_T\}} [p_T \ min(\psi d_{T-1} - \theta p_T, z_{T-1})]. \end{aligned}$$

The existence of the *min* expression in the value function indicates that there is no closed form solution to this specific dynamic programming model. If we were to follow the classic dynamic programming method, in addition to the loop over the state and decision variables, the number of value functions to be calculated would double at each period and by the time the backward stage is completed we would be faced with $2^T$ value functions. However, this is not all the difficulty we must overcome.

### 3.3.2 Non-negativity Constraint

So far we have overlooked the possibility of negative prices. However, since the probability of a negative price with this formulation is positive, i.e. in a volatile market situation it is likely to be part of an optimal pricing policy, the decision maker in reality will replace such prices (negative or too small or large) with a price that falls into the acceptable range. Thus, if we incorporate the price constraints to the problem, we will be able to balance the revenue lost in periods with very low demand (without resorting to very small or negative prices) with higher prices in the periods with higher demand. There is strong justification for the substantial added complexity by considering the price constraints in the dynamic programming model. As such an approximation is usually necessary to arrive at near optimal results.

In order to incorporate the price constraints, we only need to add the expression $p_{min} \leq p_t \leq p_{max}$ to the list of constraints in the model. Both $p_{min}$ and $p_{max}$ are strictly positive values. The sign constraint, although simple in appearance, would expand the state space at every stage of the dynamic programming algorithm threefold, which would be computationally intractable for

problems of considerable size.

$$\begin{aligned} \max\ & \mathbb{E}\,[\sum_{t=1}^{T} p_t d_t] \\ s.t.\quad & d_t = min(\psi d_{t-1} - \theta p_t + \epsilon_t, z_{t-1}), \\ & z_0 = \bar{S}, \\ & z_t = z_{t-1} - d_t, \\ & p_{min} \leq p_t \leq p_{max}. \end{aligned}$$

Since if we augment the state in any form (as in multi-product case or with demand models in the higher order of autoregressive behaviour), the computational burden of such problem will grow exponentially (even for medium number of periods), we propose here an approximate method that will yield near optimal solutions with little computational effort.

### 3.3.3 Approximate Dynamic Programming

Approximate dynamic programming is a collection of approximation methods that are devised to address the inherent problems of dynamic programming framework in dealing with constraints and the curses of dimensionality. Classic approaches to dynamic programming are unable to deal with exponential growth of the computational requirements as the state and action spaces expand. Unless the problems are defined to very restrictive assumptions, a simulation of the system is more easily constructed than a representative model.

In this paper we employ an approximate dynamic programming method which uses an approximation of the value function in the forward stage instead of calculating exact value functions in the backward stage as is the case in the classic dynamic programming. In order to recreate the process forward in time, we solve two interconnected problems: The choice of system state most likely to happen; which we solve through Monte Carlo simulation. And to determine what is the best choice of action once we are in the next state; which we answer through the approximated value function.

Denote $\hat{V}_t(y_t)$ as the approximate value function instead of the exact value $V_t(y_t)$, where $y_t = [d_{t-1}\ z_{t-1}]^\intercal$ represents the state variable at period $t$. It should

be noted that although $\hat{V}_t(y_t)$ is considered to be a function on the full state space, it need not be, since it is an approximation. In the problem at hand, the state consists of demand at the previous period and the number of items left at the start of period. However, an approximate value function that is only dependent on the number of items left is equally valid for the purposes of approximate dynamic programming in this paper. However, since we augment the state variable to include the number of items left as well as the demand variable and since the presence of a price constraint will lead to a piecewise value function, we choose one of these value functions as our approximations after we ascertain that it is quadratic in state variable.

If we assume $\hat{V}_t(y_t)$ to be quadratic in $y_t$, we can approximate the parameters of the value function at each period through the set $[A_t, B_t, C_t]$ if we assume the value function has the form: $V_t(y_t) = y_t^\intercal A_t y_t + B_t^\intercal y_t + C_t$.

The augmented state space $y_t$ in the extended model evolves based on the following:

$$\underbrace{\begin{bmatrix} d_t \\ z_t \end{bmatrix}}_{y_{t+1}} = \underbrace{\begin{bmatrix} \psi & 0 \\ -1 & 1 \end{bmatrix}}_{\Omega} \underbrace{\begin{bmatrix} d_{t-1} \\ z_{t-1} \end{bmatrix}}_{y_t} - \underbrace{\begin{bmatrix} \theta \\ 0 \end{bmatrix}}_{\Theta} p_t + \underbrace{\begin{bmatrix} \epsilon_t \\ 0 \end{bmatrix}}_{\Delta_t}.$$

Based on above state variable, the value function at period $t$ will be:

$$V_t(y_t) = \max_{\{p_t\}} \mathbb{E}[p_t y_t^\intercal e_1 + V_{t+1}(y_{t+1})],$$

where $e_1$ is the first column of identity matrix with dimensions corresponding to the dimension of state variable.

If we plug in the state evolution equation $y_{t+1} = \Omega y_t - \Theta p_t + \Delta_t$ into $V_T(y_T)$ in the last period $(T)$ we have:

$$\begin{aligned} V_T(y_T) =& \max_{\{p_T\}} \mathbb{E}[y_{T+1}^\intercal e_1 p_T] \\ =& \max_{\{p_T\}} \mathbb{E}[(\Omega y_T + \Theta p_T + \Delta_T)^\intercal e_1 p_T] \\ =& (\Omega y_T)^\intercal e_1 p_T + \Theta^\intercal e_1 p_T^2. \end{aligned}$$

Now if we consider the value function at period $t+1$ to be in the form of $V_{t+1}(y_{t+1}) = y_{t+1}^\intercal A_{t+1} y_{t+1} + B_{t+1}^\intercal y_{t+1} + C_{t+1}$, following the Bellman procedure

we will have:

$$\begin{aligned}V_t(y_t) =& \max_{\{p_T\}} \mathbb{E}[y_{t+1}^\intercal e_1 p_t + y_{t+1}^\intercal A_{t+1} y_{t+1} + B_{t+1}^\intercal y_{t+1} + C_{t+1}] \\ =& \max_{\{p_T\}} \mathbb{E}[y_{t+1}^\intercal e_1 p_t + (\Omega y_t + \Theta p_t + \Delta_t)^\intercal A_{t+1}(\Omega y_t + \Theta p_t + \Delta_t) \\ &+ B_{t+1}^\intercal(\Omega y_t + \Theta p_t + \Delta_t) + C_{t+1}],\end{aligned}$$

resulting in the equation:

$$\begin{aligned}V_t(y_t) =& (\Theta^\intercal A_{t+1}\Theta + \Theta^\intercal e_1)p_t^2 \\ &+ ((\Omega y_t)^\intercal e_1 + (\Omega y_t)^\intercal A_{t+1}\Theta + \Theta^\intercal A_{t+1}\Omega y_t + B_{t+1}^\intercal \Theta)p_t \\ &+ [(\Omega y_t)^\intercal A_{t+1}\Omega y_t + \Delta_t^\intercal A_{t+1}\Delta_t + B_{t+1}^\intercal \Omega y_t + C_{t+1}].\end{aligned}$$

Since we have ignored the constraints on price, we would have the following optimal price if we solve above $V_t(y_t)$ for $p_t$:

$$p_t^* = -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t},$$

where

$$\phi_t = (\Omega^\intercal e_1 + \Omega^\intercal A_{t+1}\Theta + (\Theta^\intercal A_{t+1}\Omega)^\intercal),$$

$$\beta_t = B_{t+1}^\intercal \Theta,$$

$$\gamma_t = \Theta^\intercal A_{t+1}\Theta + \Theta^\intercal e_1.$$

But as we are implementing the constraint $p_{min} \le p_t \le p_{max}$, at each period the optimal price is decided based on the three possible outcomes of where $p_t^*$ is situated and consequently where $y_t$ falls in the state space:

$$p_t = \begin{cases} p_{min} & \text{if } -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t} < p_{min}, \\ p_{max} & \text{if } -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t} > p_{max}, \\ -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t} & \text{if } p_{min} \le -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t} \le p_{max} \end{cases}$$

We will have three ranges for the state variable, each of which will yield different value functions. We define $U_1 = \{y| -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t} < p_{min}\}$, $U_2 = \{y| -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t} > p_{max}\}$, and $U_3 = \{y|p_{min} \le -\frac{y_t^\intercal \phi_t + \beta_t}{2\gamma_t} \le p_{max}\}$ to be mutually exclusive domains for $y$. If we replace the $p_{t,j}$ $(j = 1, 2, 3)$ in $V_t(y_t)$ with above results, we

will have the following optimal value function for the specific region $j$ at each period $t$ $(t = 1, 2, ..., T)$:

**Case j=1** : $y \in U_1$

$$V_t(y_t) = y_t^\intercal(\Omega^\intercal A_{t+1}\Omega)y_t + (B_{t+1}^\intercal\Omega + \phi_t p_{min})y_t + \gamma_t p_{min}^2 + \beta_t p_{min} + \Delta_t^\intercal A_{t+1}\Delta_t + C_{t+1}.$$

**Case j=2** : $y \in U_2$

$$V_t(y_t) = y_t^\intercal(\Omega^\intercal A_{t+1}\Omega)y_t + (B_{t+1}^\intercal\Omega + \phi_t p_{max})y_t + \gamma_t p_{max}^2 + \beta_t p_{max} + \Delta_t^\intercal A_{t+1}\Delta_t + C_{t+1}.$$

**Case j=3** : $y \in U_3$

$$V_t(y_t) = y_t^\intercal(\Omega^\intercal A_{t+1}\Omega)y_t + (B_{t+1}^\intercal\Omega)y_t + \Delta_t^\intercal A_{t+1}\Delta_t + C_{t+1}.$$

In all three cases, this results in a quadratic equation.

We will use the value function calculated above in case $j = 3$ as an approximation to the value function in our full (capacitated and constrained) model.

$$\hat{V}_t(y_t) = y_t^\intercal A_t y_t + B_t y_t + C_t. \tag{3.7}$$

Please note the relationship between the parameters of the quadratic equation in periods $t$ and $t+1$ defined through: $A_t = \Omega^\intercal A_{t+1}\Omega$, $B_t = B_{t+1}^\intercal\Omega$ and $C_t = \Delta_t^\intercal A_{t+1}\Delta_t + C_{t+1}$. We use this relationship in initialising the parameters of the value function for the first iteration $\hat{V}_t^0(y_t)$ in algorithm 1.

The chosen approximate value function represents the areas of the state space that are inside the constraint boundaries. This means that we approximate the value of being in any state (including states that are outside the constraint bounds) with the value function that maximises the revenue when the state is inside the bounds.

We further explore the state space by following randomly selected state paths and thus avoid calculating the exponentially growing number of value functions in advance. Once we find the approximate value function (3.7) we move through periods by selecting a sample path that represents the evolution of the system. We denote by $\omega$ the sample path and the evolution of the state following this sample path by $y_{t+1} = F(y_t, p_t, W_{t+1}(\omega))$ in which $W_t(\omega)$ is the random value pertaining to period $t$ by the sample path $\omega$.

Finally we can define:

$$F(y_t, p_t, W_{t+1}(\omega)) = \Omega y_t + \Theta p_t + W_{t+1}(\omega_t).$$

The proposed approximate dynamic programming algorithm aims to capture the simplicity of the value function in a non-constrained model and provides an exploratory element through the sample path $\omega$ to converge to near optimal solutions in a fast time-frame. This is presented in algorithm 1.

**Algorithm 1:** Approximate Dynamic Programming

**Initialise** $\hat{V}_t^0(y_t)$ **for all states** $y_t$

**Establish the initial state** $y_0^1$

**Set** $n = 1$

**for** $n = 1 \rightarrow N$ **do**

  **for** $t = 1 \rightarrow T$ **do**

    **Choose a random sample of outcomes** $\hat{\Omega}^n \subset \Omega$ **representing possible realisations of the random fluctuation between periods t and t+1.**

    **Solve the following maximisation problem**

$$\hat{v}_t^n = \max[\sum_{\hat{\omega} \in \hat{\Omega}} p^n(\hat{\omega})(p_t d_t + \hat{V}_{t+1}^{n-1}(F(y_t^n, p_t, W_{t+1}(\omega))))]$$

    **Set** $p_t$ **that maximises above expression to be** $p_t^n$

    **Update** $\hat{V}_t$ **using**

$$\hat{V}_t^n(y_t) = \begin{cases} \alpha^{n-1} v_t^n + (1 - \alpha^{n-1}) \hat{V}_t^{n-1}(y_t^n) & if\ y_t = y_t^n \\ \hat{V}_t^{n-1}(y_t) & otherwise \end{cases}$$

    **Compute** $y_{t+1}$ **using**

$$y_{t+1}^n = \Omega y_t^n + \Theta p_t^n + W_{t+1}(\omega_t)$$

**Choose** $p_t^N$ **as the pricing policy.**

Algorithm 1 provides a platform by which we explore the potential gains of including the price and capacity constraints. By foregoing the backward stage of the dynamic programming we gain computational advantage while we are

well placed to take advantage of the structure of the value function. Instead of calculating the value function for each period, we assume an approximation for the value function at each period that captures the specific characteristics of the real value function according to the problem at hand. We provide numerical experiments in the next section as an overview of the results.

## 3.4 Numerical Experiments

In this section we provide numerical experiments to better understand the behaviour of each model. We show through an example that the negative prices are probable in the basic model solution, and through extending the same example provide a view of how each method behaves under volatile conditions. We then provide graphs and discussions about how each method performs based on variations in different parameters.

We programmed the models through MATLAB software package (v2014b) and execute all the experiments on a 64-bit Windows 7 workstation with 4GB of memory and quad-core Intel CPU at 2.6GHz. As the computation time of all the following experiments were very fast (any execution for any problem configuration was less than ten milliseconds) we do not include a detailed examination of the computational performance.

### 3.4.1 Dynamic Pricing Using the Basic Model

We examine the behaviour of the basic model with a numerical example of the best-pricing strategy for four simulated realisations of the demand throughout the season. Table 3.1 outlines the results of a single random run of the basic model through the season and the optimal price and realised demand at each period. The goal in this simulated example is to maximise the expected revenue over 10 periods. The decision maker chooses a price at each period to receive the best revenue over the whole season. If the prices are too high, the demand will decline, and if the price is too low, the revenue will be too little for the immediate periods. Thus the goal is to strike a balance between current prices and future demand. Recall that since the length of a period between price

changes is arbitrary, we can set the period length to reflect the smallest time that is relevant in practical terms. We believe it to be rare for a retailer of fashion products to change the prices more than once every week, prompting us to put the finest definition of period to be one week. A six month season (relatively long in this context), would comprise of maximum 24 decision periods. Thus the choice of $T = 10$ is a reasonable choice for purposes of illustration. We assume the parameters of the demand evolution model to be $\psi_t = 2$, and $\theta = 0.1$. $\psi$ is the coefficient explaining the relationship between current demand and the demand in the previous period. A value of $\psi_t = 2$ means that demand would double each period if other factors are ignored. The demand in real situations would follow subtler relation to its historical values. However, since it is affected by the price and market volatility as well as the previous demand, the actual demand evolution can be even more volatile than in this example. $\theta$ is the coefficient that represent the price elasticity of demand in (3.5). As an intuition for these parameters, the value of $\theta$ is designed to produce an optimal price of \$500 given $\psi$ and an initial demand value of 50 items. If we assume the demand to be constant over the ten periods (which would result in the optimal pricing strategy of \$500 over the whole season), the total sales revenue would amount to \$250000.

In practice, however, the optimal price will also be affected by the dynamics of the product stock. We simulate the above example for various realisations of $\sigma_\epsilon^2$ in the basic model corresponding to a variety of market volatility situations. As can be seen in Table 3.1, the occurrence of negative prices is more likely when the demand process is more volatile.

The above numerical example clearly demonstrates the necessity of the application of the sign and capacity constraints in the basic model if we are to be certain of the pricing policy under uncertain market conditions.

### 3.4.2 Effects of Incorporating Sign Constraints

We extend the above example to determine the behaviour of the approximate method that introduces constraints to the model and solves the dynamic programming model to near optimality following the procedure set out in algo-

| | $\sigma_\epsilon^2$=1 | | $\sigma_\epsilon^2$=10 | | $\sigma_\epsilon^2$=100 | | $\sigma_\epsilon^2$=1000 | |
|---|---|---|---|---|---|---|---|---|
| Period | p | d | p | d | p | d | p | d |
| 1 | 500 | 49 | 500 | 49 | 500 | 47 | 500 | 41 |
| 2 | 497 | 48 | 491 | 46 | 473 | 37 | 414 | 10 |
| 3 | 487 | 49 | 460 | 48 | 374 | 44 | 101 | 31 |
| 4 | 494 | 48 | 481 | 47 | 440 | 39 | 310 | 15 |
| 5 | 489 | 49 | 465 | 48 | 390 | 45 | 154 | 35 |
| 6 | 495 | 47 | 484 | 43 | 450 | 28 | 345 | -20 |
| 7 | 477 | 47 | 430 | 42 | 279 | 23 | -198 | -34 |
| 8 | 473 | 47 | 415 | 43 | 233 | 28 | -344 | -18 |
| 9 | 478 | 47 | 431 | 43 | 282 | 28 | -188 | -16 |
| 10 | 478 | 47 | 433 | 42 | 288 | 24 | -168 | -31 |
| Total revenue | 236083 | | 207708 | | 133441 | | 53197 | |

**Table 3.1:** Optimal pricing strategy (p), observed demand (d) and the total amount of sales revenue for various volatility rates ($\sigma_\epsilon^2$ values) obtained by the Basic method

rithm 1 in section 3.3.3. As a benchmark, we also introduce a naive implementation of the constraints after we have solved the problem through the basic model.

Table 3.2 contains the pricing policy and demand evolution in a sample run with all three methods for the case of $\sigma_\epsilon^2 = 1000$. The basic method is the method discussed in section 3.2.1 dealing with the non-capacitated case. Naive method is the basic method with a naive implementation of price and capacity constraints (i.e. the number of items left in stock). And ADP refers to approximate dynamic programming method which we explain in section 3.3.3 and incorporates the price and capacity constraints into the dynamic programming framework.

Analysing the result of Table 3.2 is straightforward. The Basic method does not provide any constraints, and thus have negative demand and price in later periods. The Naive method curtails the negativity to predefined values based on the solution of the basic model. Therefore the early pricing strategy is the same as the basic method. However, the addition of constraints in this case reduces the eventual actual revenue because the basic model produces both negative prices and negative demands resulting in positive revenue, which

| | Basic | | Naive | | ADP | |
|---|---|---|---|---|---|---|
| Period | p | d | p | d | p | d |
| 1 | 500 | 41 | 500 | 41 | 283 | 56 |
| 2 | 414 | 10 | 414 | 10 | 419 | 48 |
| 3 | 101 | 31 | 101 | 31 | 446 | 52 |
| 4 | 310 | 15 | 310 | 15 | 327 | 44 |
| 5 | 154 | 34 | 154 | 34 | 289 | 66 |
| 6 | 345 | -19 | 345 | 0 | 288 | 46 |
| 7 | -198 | -34 | 50 | 0 | 313 | 56 |
| 8 | -344 | -18 | 50 | 10 | 442 | 52 |
| 9 | -188 | -16 | 105 | 12 | 326 | 64 |
| 10 | -168 | -31 | 125 | 0 | 407 | 56 |
| Total | 53197 | | 40115 | | 190030 | |

**Table 3.2:** Optimal pricing strategy (p), observed demand (d) and the total amount of sales revenue for $\sigma_\epsilon^2 = 1000$ for the three methods

clearly is not logical. The ADP method, however, has overcome the addition of constraints and furthermore yields a better revenue than both basic and naive models. Please note that we include these results to provide a snapshot of the solutions in real-world execution. Since the choice of example is random, the better performance of ADP method is incidental. The ADP algorithm simply provides a higher probability of better performances. In later examples we will explore the performance of the methods over many simulation runs and provide better insight on the overall performance of each method. ADP will provide a larger number of better solutions in the long run compared to the naive method because we are maximising expected revenue. Important to note at this stage is the ability of the ADP to provide prices within the defined bounds and providing a mechanism to control the sways of demand.

Knowing that the ADP method provides reasonable pricing strategies, we examine the overall revenue generated through each of ADP and Naive methods to get a better picture of how these methods behave over many executions and in relation to expected revenue. The results shown in figure 3.1 are drawn from running the model with the same parameters as the example above with demand model parameters set as $\psi_t = 1.7$, and $\theta = 0.085$. The choice of

parameter values provides an optimal price of \$500 given an initial demand of 50 units as in the example above, but the demand evolution is less sensitive to price and previous demand. We run each method with the same parameters for 30 simulation runs, to negate the effect of volatility and provide an insight into the real efficiency of each method. We apply the value \$300 as lower price limit and the value \$500 as higher price limit. Although this might be rather restrictive, in this instance it is only to gauge the efficiency of the methods in handling the constraints.

Figure 3.1 offers a clear view of the methods and their efficiency. The approx-

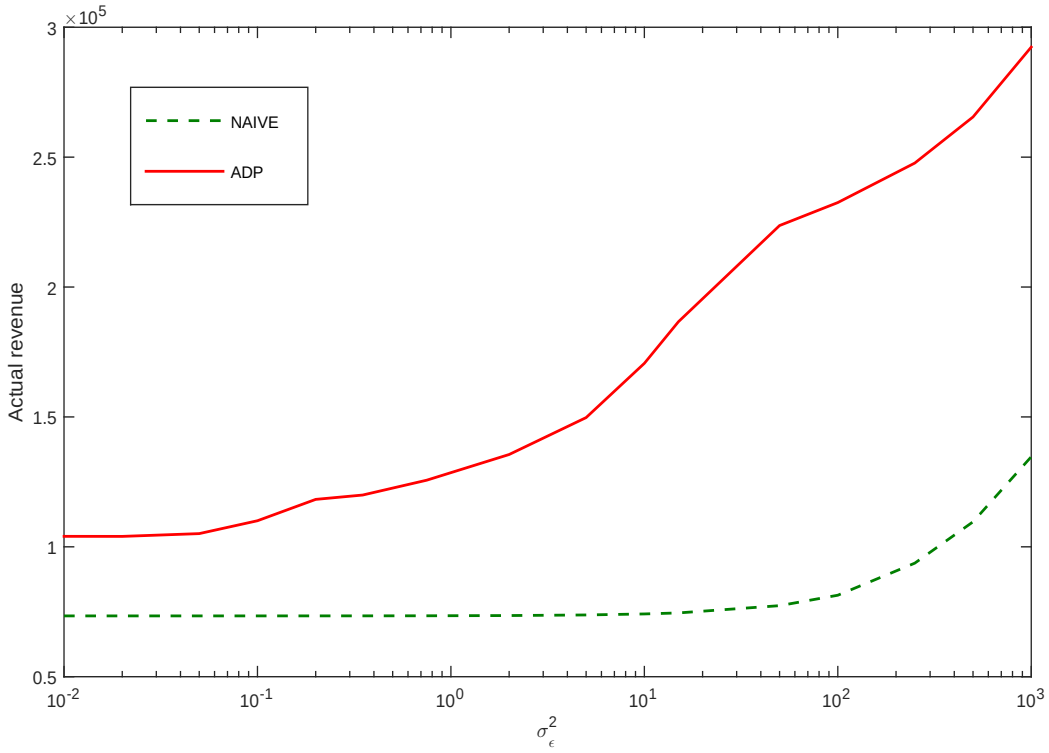


**Figure 3.1:** Actual sales revenue for the two methods with variable $\sigma_\epsilon^2$ rates

imate model (ADP) clearly outperforms the Naive method on all the volatility levels. In the naive method, where we have applied a rudimentary handling of the price range as explained above, the results are generally lower but with more consistency. We can see a marginal improvement by the Naive method as the volatility becomes very high, because the optimal prices will be closer to the defined price bounds with higher probability. Another point to consider is when $\sigma_\epsilon^2$ becomes very high, the naive method improves much faster; about the same

pace as the ADP method. This is because when the volatility is very high, the approximate method will still try to find an approximate value function that would be near optimal for many scenarios, while the optimal solution is limited to a single point in the state space. The naive method in these cases will more frequently provide the near optimal prices (price bounds). Also note that when $\sigma_\epsilon^2$ is near 1000, the demand will change more than 30 units from period to period which is a significant number in any business where the historical demand is in the region of 50 units.

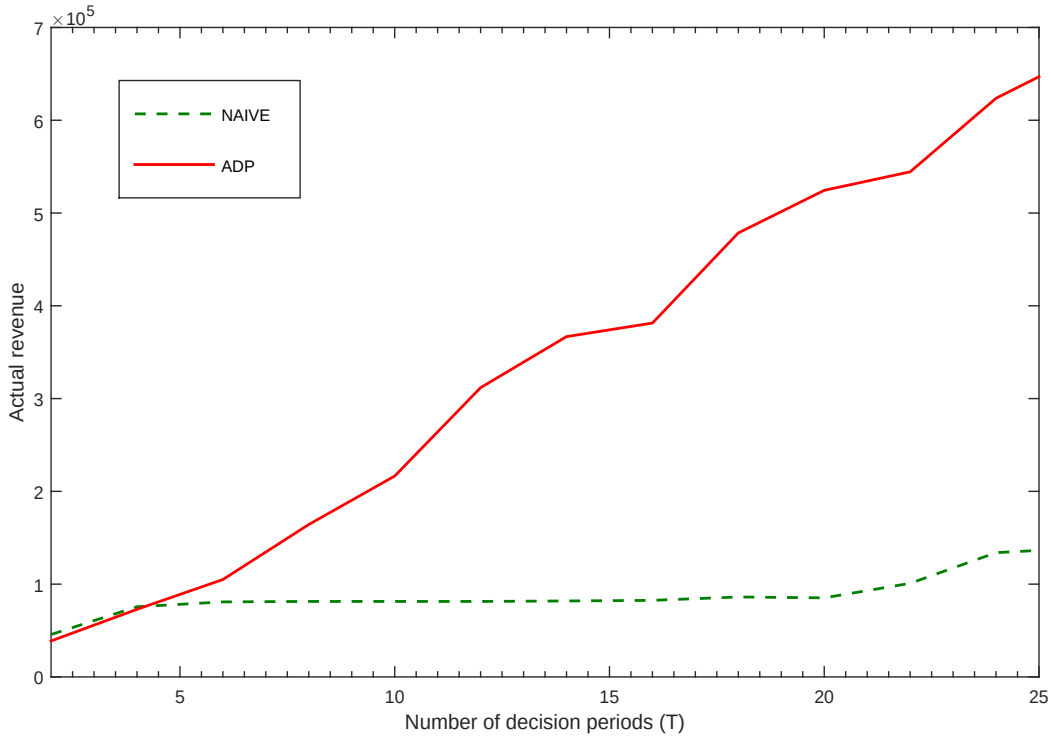


**Figure 3.2:** Actual sales revenue for the two methods for a sample of decision periods

Figure 3.2 depicts the actual revenue in relation to the variations in the number of periods in which we change prices. The parameters of the model are the same as the previous figure, with the variation in demand defined by $\sigma_\epsilon^2 = 100$. The basic model with naive implementation of the constraints outperforms the ADP for very small period numbers only. Its performance is then almost constant followed by slight improvement as the number of decision periods increase beyond twenty periods. Since the volatility is fixed for both methods, the only change in the results is because of the ability of the methods to optimise the

prices in relation to the demand evolution. The ADP method offers continuous improvement in period numbers that signifies its superior ability to utilise the increased possibility of swings in the demand.

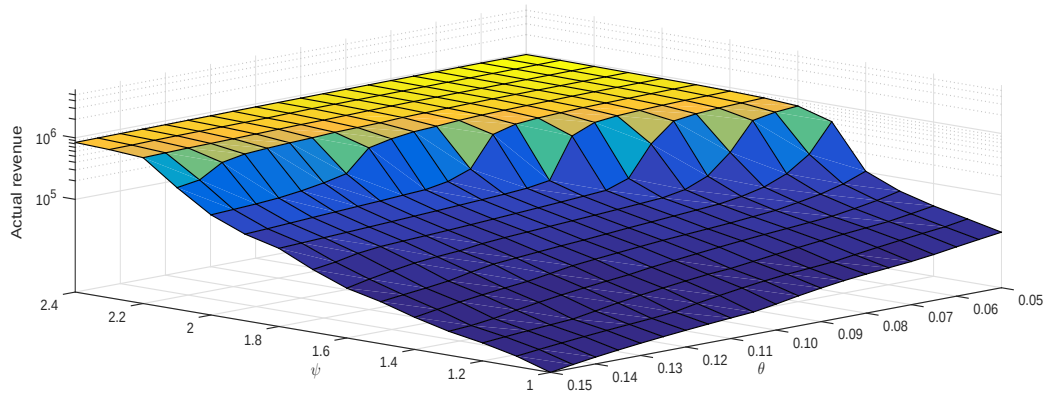


**Figure 3.3:** Actual sales revenue for the naive method with variable $\theta$ and $\psi$

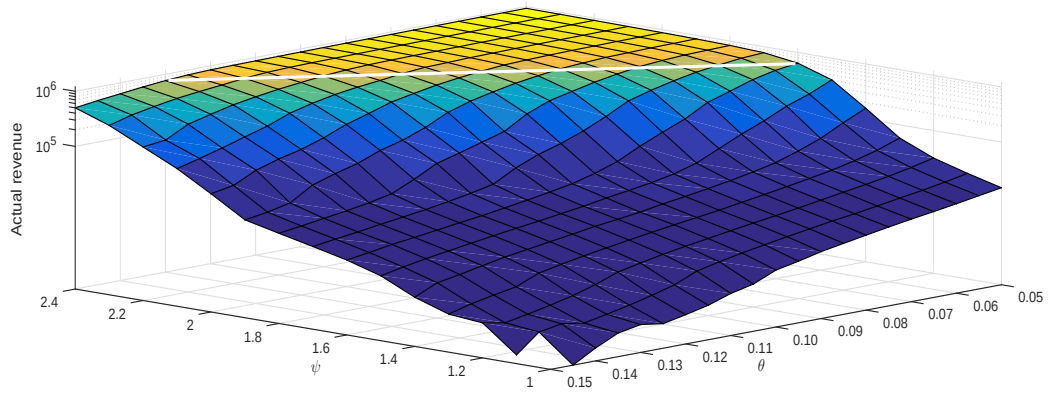


**Figure 3.4:** Actual sales revenue for the ADP method with variable $\theta$ and $\psi$

In order to gain an understanding of the parameters $\psi$ and $\theta$ and how they affect the revenue if we applied either of the methods, we provide figures 3.3 and 3.4 in which we see the average actual revenue realisations for the Naive and ADP methods respectively.

What is evident from figure 3.3 is the constant increase in revenue figures from the naive method as the rate of $\psi$ increases and the rate of $\theta$ decreases simultaneously. It results in marginally higher actual revenue when the levels of $\psi$ are high and $\theta$ is relatively low compared to ADP method. In other words the Naive method is capable of offering reasonable solutions when the demand

is highly autoregressive with low price sensitivity. In those instances the use of a naive implementation of the constraints might be justified, especially in problems dealing with moderate volatility levels. In all the other instances it is under-performing compared to the ADP method.

However, the ADP method, as is evident from Figure 3.4, performs similarly in the areas where the Naive method has good performance, while it is definitely superior in all the other instances. It generally improves as the autoregressive element $\psi$ increases and also as the price elasticity decreases, similar to Naive method, but with a higher rate.

We observe a plateau in actual revenue for both methods in the area where $\psi$ is at its biggest and $\theta$ at its lowest value. This area is marked by the line $\psi = 1.1 + 10\theta$ (the white line in figure 3.4). This is because of the limitation in the maximum price we have defined which leads to both methods yielding the maximum possible revenue. Thus the benefits of the ADP should be sought in the areas in which a Naive implementation is not as effective. Those areas, before the plateau occurs, can be said to define product characteristics that have large numbers of price aware customers. Products with such characteristics include high end fashion or technological products with a dedicated and knowledgeable following.

## 3.5 Conclusion

We study a dynamic pricing problem where sales revenue is maximised over a fixed season through price decisions. We have defined a linear demand model that is dependent on the price in the present period and demand during the previous periods. We provide a dynamic programming framework that provides near-optimal pricing strategy based on the discussed demand model. We have shown that under volatile market conditions there is a probability that the optimal price under dynamic programming setting would fall outside the reasonable price range. We address this issue with an approximate dynamic programming method. The results of our simulation runs are encouraging in the use of approximate method when the dynamic programming method must include constraints. The use of approximate dynamic programming also enables

us to extend the model even further to accommodate more complex situation that are relevant to the specific business problem. The case of multiple product pricing is a valuable addition to the pricing toolbox of the retailer of the product groups with inter-product elasticities.

This research paper signifies the importance of deriving model elements from specific product characteristics. Thus, depending on the application and the products, other models might be more appropriate for explaining the demand behaviour. Notable models worthy of attention are multiplicative or non-linear demand models as well as dynamic autoregressive models where the rate of dependence on previous demand values vary over the season. Furthermore, the inclusion of an external information variable that would represent general market condition or expert opinions in the demand model, might add to the accuracy of the demand evolution reflected in the dynamic programming model. Although any such variable, if accurately modelled, would augment the problem state and help the decision maker, the added complexity to the problem should be assessed against the benefits. As a further step it would be useful to evaluate certain complimentary information in the specific settings of the product under study. The managerial insight of such extensions would be substantial.

# Chapter 4

# Optimal Trading Under Non-negativity Constraints Using Approximate Dynamic Programming


### Abstract

In this paper we develop an extended dynamic programming (DP) approach to solve the problem of minimising execution cost in block trading of securities. To make the problem more practical, we add non-negativity constraints to the model and propose a novel approach to solve the resulting DP problem to near optimal results. We also include time lags in the problem state to account for the autoregressive behaviour of most financial securities as a way of increasing problem sensitivity to variability of prices and information. The computation times achieved for the proposed algorithm are fast and provide the possibility of live implementation. We demonstrate the benefits offered by the new approach through numerical analysis and simulation runs in comparison to the classic model without the non-negativity constraints.

# 4.1 Introduction

## 4.1.1 Background and Literature Review

The growth in equity trading in recent decades which have been largely due to the ever increasing amounts of funds available to institutional investors such as pension and mutual funds, has triggered an interest in more effective management of trading costs. These costs are often called transaction costs or execution costs, which include commissions, bid/ask spreads, opportunity costs of waiting and price impact from trading. Loeb (1983) was among the first to recognise the importance of execution costs and discusses different aspects of trading cost with relation to capitalisation and spread of the stock and the funds available to the trader. Perold (1988), among others, documents that hypothetical portfolios or passive benchmarks constantly outperform the market and the portfolio manager by almost 20% per year. Chan and Lakonishok (1993) argue that since trading in equity markets is increasingly dominated by institutional investors, the 'implementation shortfall' that Loeb (1983) and Perold (1988) discuss, may be due to the costliness of executing individual trade transactions that result in a more costly overall execution of the order. This overall transaction cost prompts the traders to break their order down into smaller transaction units which is then executed over a certain time period. A trading strategy that minimises the expected execution cost of the trade is defined as best execution strategy.

Chan and Lakonishok (1993) study the institutional effects of trade on equity prices. They argue that since trading in equity markets is increasingly dominated by institutional investors, the importance of transaction costs are increasing. They offer three explanations for price changes triggered by large trade volumes: (i) *short-run liquidity costs* that arise when there is not an immediate buyer or seller, and the attempt to attract the buyer or seller translate to a price concession, (ii) *imperfect substitution* where there is no particular stock as a substitute and buyer offers a premium for a large transaction, and (iii) *information effects* where the amount of trade reveal information about the trade that is incorporated into the subsequent prices. This and many other

studies document portfolio managers' inability to outperform various passive benchmarks, despite considerable effort to analyse and select stocks.

Bertsimas and Lo (1998) were the first to consider the use of dynamic programming in acquiring best execution strategy. They argue that the act of trading affects price dynamics that will result in changes to the future trading costs (since the demand for financial securities is not perfectly elastic, the price impact of current trades, however small, can affect the course of future prices). Also they observe that trading takes time as large trades need to be executed over numerous periods. They propose and solve the dynamic trading problem with the use of dynamic programming framework. Bertsimas and Lo (1998) offer an analytic solution to the transaction cost problem in the case of a single stock. They solve the problem using dynamic programming methods and offer numerical analysis based on simulation runs on a scenario representing a real problem. Bertsimas et al. (1999) provide a similar method to the single stock model presented by Bertsimas and Lo (1998) to address the portfolio case. They also develop a new price impact model and an approximate model to solve the problem with non-negativity constraints for a specific case.

Chakravarty (2001) suggests that the medium sized trading done by institutions (which he argues is the optimal way if the trader intends to sell big block of shares and prevent big shocks to the market i.e. adverse price impact) has a disproportionate effect on the cumulative price change compared to trading in big chunks. Alexander and Peterson (2007) study the effects of trade clustering on various parts of the market. They argue that the size of the clusters tend toward 100, 500 and 5000 clusters which is mostly used by stealth and highly informed traders. They study the effects of clustering and block trading on the price of the underlying stock and conclude that the price impact from medium sized clusters are much higher than the small or big sized clusters. Domowitz and Yegerman (2005) review some of the early algorithmic trading services and give a brief result as to their efficiency compared to manual and other types of trading considering parameters such as trade size, type of security, etc. Butenko et al. (2005) provide models and algorithms for the problem of liquidating a certain amount of securities with or without the consideration of risk as a factor in decision making. Engle and Ferstenberg (2006) propose an

extension to execution cost problem by incorporating the "risk return" trade-off into the problem. They achieve this through combination of transaction cost model with the portfolio planning problem.

Almgren and Chriss (2001) examine the relation between the risk (different levels of liquidity and the traders' idea of this) and optimal execution strategy. They obtain closed form solutions for optimal trading strategy for any level of risk aversion from the trader. Almgren (2003) offers an update on the Almgren and Chriss (2001) modelling of the portfolio/stock trading costs by introducing a more robust price impact model. He incorporates trading-enhanced risk, an additional volatility measure that corresponds to the change in price following the demand in a more rapid execution of large blocks (liquidity premium demanded by the market is not deterministic). Kissell et al. (2004) build on the work of Bertsimas and Lo (1998) and Almgren and Chriss (2001) and give a more detailed breakdown of price change sources and analyse the causal relationship between these. They offer best execution strategy for three different scenarios: cost minimisation; balancing the risk vs cost; and price improvement. Subramanian and Sherali (2010) quantify the liquidity risk which corresponds to the difference between market price and realisation price from a traders position. They find an optimality condition for block trading.

He and Mamaysky (2005) offer an optimal execution strategy which is similar to that of Bertsimas and Lo (1998), but present a different price impact model based on Merton (1971). Huberman and Stanzl (2005) consider a linear price impact and argue that in a market without arbitrage opportunity, their proposed linear price impact model is optimal. They extend the work of Bertsimas and Lo (1998) by considering a risk averse trader. Hasbrouck and Seppi (2001) study various price dynamics along with other liquidity related issues such as focusing on the market as a whole and the effect of inter-company trades on the liquidity of a security. Kissell and Malamut (2006) introduce a framework that can accommodate different aspects of trading behaviour. A system that can act aggressively/patiently at times of favourable/unfavourable price changes would benefit investors greatly. The criteria also include the change in the dynamics of price changes as well as the different initial prices the algorithms can take.

### 4.1.2 Contributions and Paper Structure

Although Bertsimas and Lo (1998) acquire the closed form solution and the resulting trade strategy is the optimal execution strategy, it does not take into account the non-negativity constraints which would lead to a short-selling situation when in practice it is not possible to do so. If the price change is severe in an otherwise normal trade scenario, during a buy operation it might be optimal to sell and vice versa, while in practice the trader will not be allowed to operate based on that insight. This shortcoming might not affect a large portion of daily trades, but since it is a significant probability when trading large volumes of securities in a volatile market, the expected cost of ignoring the non-negativity constraints is still considerable. We develop an approximate dynamic programming approach to circumvent this costly possible scenario.

Our first contribution to the literature is the inclusion of non-negativity constraints in the formulation. To the best of our knowledge, the inclusion of non-negativity constraints has not been treated in the relevant literature. This constraint, however, increases the complexity of this problem substantially, as dynamic programming does not lend itself readily to constraints. We present here an approximate dynamic programming approach that enables us to solve a large combination of problems to near optimality through a generalised platform.

Our second contribution is to expand the state space such that the price and information dynamics can include historical data such as prices and information in periods further in the past. The motivation behind this decision is that on many occasions there might be a time lag between previous price and information and their impact on the price of present period.

Finally, we contribute to the literature through development of a bespoke Approximate Dynamic Programming method that combines both above contributions into a more complex problem. Our method offers a generalised platform through which a large combination of problems can be solved fast and near optimally.

The rest of this paper is structured as follows: we first present the modelling approach and results outlined by Bertsimas and Lo followed by our proposed

extensions to their model in section 4.2. We then propose an approximate dynamic programming method in section 4.3 that handles the added sign constraints to the problem. We present the results and insights gained through numerical analysis in section 4.4 followed by conclusion and a look at the future possible directions for this research.

## 4.2 Models for Optimal Trade Execution

### 4.2.1 Basic Model

Consider a situation where a trader wants to buy a number $S$ of shares in $T$ consecutive periods of equal length. It is assumed that the price dynamics are known and the price $p$ at period $t$ follows an autoregressive process where it is related to the prices in the previous periods. It is also assumed that the effect of trade volume on the price is known to the trader. An additional information source is assumed, reflecting any complementary data the decision maker might use to infer the price behaviour. This additional source of information, denoted henceforth by $x_t$ might be an index of the market where the share is traded or expert knowledge available to the trader.

Consider $s_t$ to denote the current number of stocks available at each stage and $u_t$ to be the decision variable which is the trade volume in the current period. Based on these information the decision maker wants to optimise the number of shares traded in each period in order to minimise the overall cost of execution of the $S$ shares within the $T$ sequential periods.

The general form of the basic problem (Bertsimas and Lo, 1998) is:

$$\begin{aligned}
\min \quad & \mathbb{E}\left[\sum_{t=1}^{T} p_{t+1}u_t\right], \\
s.t. \quad & \sum_{t=1}^{T} u_t = S, && (4.1)\\
& s_1 = S, \quad s_{T+1} = 0, && (4.2)\\
& s_{t+1} = s_t - u_t, \qquad t = 1, 2, ..., T-1 && (4.3)\\
& x_{t+1} = \rho x_t + \eta_t, \qquad t = 1, 2, ..., T && (4.4)\\
& p_{t+1} = p_t + \beta x_t + a u_t + \epsilon_t, \quad t = 1, 2, ..., T && (4.5)
\end{aligned}$$

where $\eta_t$ and $\epsilon_t$ are independent white noise processes with mean 0 and variance $\sigma_\epsilon^2$ and $\sigma_\eta^2$ respectively. $a$ is assumed to be positive and $\rho$ to be bounded between 1 and -1, and $\beta$ is the effect of information on price dynamics and is a given at the start of the optimisation period. Constraints (4.1) and (4.2) explain the dynamics of the stock at hand and ensure that all the shares are executed during the $T$ periods. Constraint (4.3) is to ensure that the number of stocks (to execute) in the coming period ($s_{t+1}$) is the current number of stocks available ($s_t$) minus the trade volume in the current period. And constraints (4.4) and (4.5) express the dynamics of information and price evolution accordingly. Finally, the objective function is the expected cost over the trading period.

Throughout this study the execution is assumed to be a buy operation. However, the results would be applicable to a sell operation without any loss in generality. The objective of such problem would be to maximise revenue. Another point to be noted is that while the occurrence of negative prices in the model is possible, we implicitly assume that the parameters of the model are of such values as to make this unlikely. Thus we treat the model without such limitation.

### 4.2.2 Limitations of the Basic Model

In the original model by Bertsimas and Lo (1998) the price is assumed to follow an AR(1) process (i.e. it depends only on the price in the previous periods). However, many price processes in financial domain exhibit autoregressive attributes of higher order. We extend the assumption of the basic problem to include AR(m) in price evolution. A further possible extension is the inclusion of an autoregressive information vector ($x_t$). The assumption that the process is autoregressive is justified by the amount of time needed for the whole market to note the changes in price (and information) and assess their effects.

The other limitation in the implementation of the basic model is the absence of non-negativity constraints. Based on the variance of the information variable and also the price, the optimal solution might suggest a negative trade (i.e. sale in a buy operation). In a real situation where the short-selling option is not available, a naive strategy is to change the negative trades to zero. However,

such a solution would significantly reduce the intended benefits of the model e.g. minimisation of transaction costs.

In the coming section we address these limitations by offering a flexible framework that provides near optimal results while handling the added complexity of non-negativity constraints and offer a generalised platform through which a large combination of problems can be solved fast and near optimally.

### 4.2.3 Extended Model

We first extend the price dynamics to an AR(m) process and present the closed form solution in line with the results of Bertsimas and Lo (1998). The resulting model is:

$$
\begin{aligned}
\min \quad & \mathbb{E}\left[\sum_{t=1}^{T} p_{t+1}u_t\right], \\
s.t. \quad & \sum_{t=1}^{T} u_t = S, \\
& s_1 = S, \quad s_{T+1} = 0, \\
& s_{t+1} = s_t - u_t, && t = 1, 2, ..., T-1 \\
& x_{t+1} = \rho x_t + \eta_t, && t = 1, 2, ..., T \\
& p_{t+1} = \sum_{j=1}^{m} \theta_j p_{t+1-j} + \beta x_t + a u_t + \epsilon_t, && t = 1, 2, ..., T
\end{aligned} \tag{4.6}
$$

where $m$ is the order of autoregressiveness in AR(m) and is assumed not to be bigger than $T+1$. To apply the AR(m) in the sequence of prices we augment the state space as follows:

$$
\underbrace{\begin{bmatrix} p_{t+1} \\ p_t \\ . \\ . \\ . \\ p_{t+1-m} \\ s_{t+1} \\ x_{t+1} \end{bmatrix}}_{y_{t+1}} = \underbrace{\begin{bmatrix} \theta_1 & \theta_2 & . & . & . & \theta_m & 0 & \beta \\ 1 & 0 & . & . & . & 0 & 0 & 0 \\ . & . & & & & . & . & . \\ . & . & & & & . & . & . \\ . & . & & & & . & . & . \\ 0 & 0 & . & . & . & 0 & 0 & 0 \\ 0 & 0 & . & . & . & 0 & 1 & 0 \\ 0 & 0 & . & . & . & 0 & 0 & \rho \end{bmatrix}}_{A} \cdot \underbrace{\begin{bmatrix} p_t \\ p_{t-1} \\ . \\ . \\ . \\ p_{t-m} \\ s_t \\ x_t \end{bmatrix}}_{y_t} + \underbrace{\begin{bmatrix} a \\ 0 \\ . \\ . \\ . \\ 0 \\ -1 \\ 0 \end{bmatrix}}_{b} u_t + \underbrace{\begin{bmatrix} \epsilon \\ 0 \\ . \\ . \\ . \\ 0 \\ 0 \\ \eta \end{bmatrix}}_{\omega_t}.
$$

The above formulation can be written as:

$$y_{t+1} = Ay_t + bu_t + \omega_t.$$

The cost at period $t$ is:

$$\begin{aligned} g(y_t, u_t, \omega_t) &= p_{t+1}u_t \\ &= y_{t+1}^\intercal e_1 u_t \\ &= (Ay_t + bu_t + \omega_t)^\intercal e_1 u_t \\ &= y_t^\intercal (A^\intercal e_1)u_t + b^\intercal e_1 u_t^2 + \omega_t^\intercal e_1 u_t, \end{aligned}$$

where $e_1$ represents the first column of the identity matrix.

At the final stage ($t = T$), the value function is equal to the expected value of cost function with respect to $\omega_T$. We substitute the decision variable $u_t$ with $s_T$ which is the only available decision at $t = T$, and obtain:

$$u_T = s_T = e_{m+1}^\intercal * y_T,$$

where $m$ is the autoregressive level in AR(m). Thus,

$$\begin{aligned} V_T(y_T) &= \mathbb{E}[g(y_T, u_T, \omega_T)] \\ &= \mathbb{E}[y_T^\intercal (A^\intercal e_1)u_T + b^\intercal e_1 u_T^2 + \omega_T^\intercal e_1 u_T] \\ &= y_T^\intercal (A^\intercal e_1)e_{m+1}^\intercal y_T + b^\intercal e_1 y_T^\intercal (e_{m+1}e_{m+1}^\intercal)y_T \\ &= y_T^\intercal \left((A^\intercal e_1)e_{m+1}^\intercal + b^\intercal e_1 (e_{m+1}e_{m+1}^\intercal)\right) y_T \\ &= y_T^\intercal K_T y_T, \end{aligned}$$

where $K_T = (A^\intercal e_1)e_{m+1}^\intercal + b^\intercal e_1 (e_{m+1}e_{m+1}^\intercal)$.

For the second last period, i.e. stage $(T-1)$, we have:

$$\begin{aligned} V_{T-1}(y_{T-1}) &= \min_{u_{T-1}} \mathbb{E}[g(y_{T-1}, u_{T-1}, \omega_{T-1})V_T(y_T)] \\ &= \min_{u_{T-1}} \mathbb{E}[y_{T-1}^\intercal (A^\intercal e_1)u_{T-1} + b^\intercal e_1 u_{T-1}^2 + \omega_{T-1}^\intercal e_1 u_T + y_T^\intercal K_T y_T] \\ &= \min_{u_{T-1}} \mathbb{E}[y_{T-1}^\intercal (A^\intercal e_1)u_{T-1} + b^\intercal e_1 u_{T-1}^2 + \omega_{T-1}^\intercal e_1 u_T \\ &\quad +(Ay_{T-1} + bu_{T-1} + \omega_{T-1})^\intercal K_T (Ay_{T-1} + bu_{T-1} + \omega_{T-1})] \\ &= \min_{u_{T-1}} \Big\{ (b^\intercal e_1 + b^\intercal K_T b)u_{T-1}^2 + y_{T-1}^\intercal \left(A^\intercal e_1 + A^\intercal (K_T + K_T^\intercal)b\right) u_{T-1} \\ &\quad +y_{T-1}^\intercal A^\intercal K_T A y_{T-1} + \mathbb{E}[\omega_{T-1}^\intercal K_T \omega_{T-1}] \Big\}. \end{aligned}$$

Since $V_{T-1}(y_{T-1})$ is a quadratic equation on $u_{T-1}$ and we assume $a > 0$, we will have $(b^\intercal e_1 + b^\intercal K_T b) > 0$.

**Lemma 4.1.** *$\alpha_t = (b^\intercal e_1 + b^\intercal K_{t+1} b) > 0$ is true for all $t = 1, ..., T-1$.*

*Proof.* We demonstrate this result in appendix A. □

We acquire the optimal decision for period $(T-1)$ as:

$$u^*_{T-1} = -\frac{y^\intercal_{T-1} L_{T-1}}{2\alpha_{T-1}},$$

where $L_{T-1} = A^\intercal e_1 + A^\intercal (K_T + K^\intercal_T) b$, and $\alpha_{T-1} = b^\intercal e_1 + b^\intercal K_T b$.

We can then rewrite $V_{T-1}(y_{T-1})$ as:

$$V_{T-1}(y_{T-1}) = y^\intercal_{T-1} K_{T-1} y_{T-1} + C_{T-1},$$

where

$$K_{T-1} = \left( A^\intercal K_T A - \frac{L_{T-1} L_{T-1}\intercal}{4\alpha_{T-1}} \right),$$
$$C_{T-1} = \mathbb{E}[\omega^\intercal_{T-1} K_T \omega_{T-1}], C_T = 0.$$

To find the closed form solution for the case of extended state, we assume that:

$$V_{t+1}(y_{t+1}) = y^\intercal_{t+1} K_{t+1} y_{t+1} + C_{t+1}.$$

Then, we have:

$$\begin{aligned}
V_t(y_t) &= \min_{u_t} \mathbb{E}[g(y_t, u_t, \omega_t) + V_{t+1}(y_{t+1})] \\
&= \min_{u_t} \mathbb{E}[y^\intercal_t (A^\intercal e_1) u_t + b^\intercal e_1 u^2_t + y^\intercal_{t+1} K_{t+1} y_{t+1} + C_{t+1}] \\
&= \min_{u_t} \mathbb{E}[y^\intercal_t (A^\intercal e_1) u_t + b^\intercal e_1 u^2_t + (Ay_t + bu_t + \omega_t)^\intercal K_{t+1} (Ay_t + bu_t + \omega_t) + C_{t+1}] \\
&= \min_{u_t} \Big\{ (b^\intercal e_1 + b^\intercal K_{t+1} b) u^2_t + y^\intercal_t \left( A^\intercal e_1 + A^\intercal (K_T + K^\intercal_T) b \right) u_t \\
&\quad + y^\intercal_t A^\intercal K_{t+1} A y_t + \mathbb{E}[\omega^\intercal_t K_{t+1} \omega_t] + C_{t+1} \Big\}.
\end{aligned}$$

The above quadratic equation holds for all $t$, resulting in the optimal solution:

$$u^*_t = -\frac{y^\intercal_t L_t}{2\alpha_t},$$

where

$$\begin{aligned} L_t &= A^\intercal e_1 + A^\intercal (K_{t+1} + K_{t+1}^\intercal) b, \\ \alpha_t &= b^\intercal e_1 + b^\intercal K_{t+1} b, \\ K_t &= A^\intercal K_{t+1} A - \frac{L_t L_t^\intercal}{4\alpha_t}, \\ K_T &= (A^\intercal e_1) e_{m+1}^\intercal + b^\intercal e_1 (e_{m+1} e_{m+1}^\intercal), \\ C_t &= \mathbb{E}[\omega_t^\intercal K_{t+1} \omega_t] + C_{t+1}, \\ C_T &= 0. \end{aligned}$$

Given the values of $\theta, \beta, a$ and $\rho$, we can calculate $K_t$, $L_t$ and $C_t$ offline for all stages $t = 1, 2, ..., T$ by starting backward, after which the optimal solution is obtained in the forward stage. Algorithm 2 depicts the general steps of this procedure.

The closed form solution arrived at here is a generalised form of the results of Bertsimas and Lo (1998) and reduces to the basic form in case of an AR(1) model.

**Algorithm 2:** Closed form solution to the extended problem

**input** : $\theta$, $\beta$, $a$ and $\rho$

**output**: $u^*$

$K_T = (A^\intercal e_1) e_{m+1}^\intercal + b^\intercal e_1 (e_{m+1} e_{m+1}^\intercal)$

**for** $t = (T-1) \rightarrow 1$ **do**

- $L_t = A^\intercal e_1 + A^\intercal (K_{t+1} + K_{t+1}^\intercal) b$
- $\alpha_t = b^\intercal e_1 + b^\intercal K_{t+1} b$
- $K_t = A^\intercal K_{t+1} A - \frac{L_t L_t^\intercal}{4\alpha_t}$

$y_1 = y_0$

**for** $t = 1 \rightarrow T$ **do**

- $u_t^* = -y_t^\intercal L_t / 2\alpha_t$

### 4.2.4 Including Non-negativity Constraints

The basic model in Bertsimas and Lo (1998) does not include non-negativity constraints to avoid negative trade volumes when the variations in the prices or the market warrant negative trading. An optimal policy that contains a negative

trade in a block trade buying operation would be void in real trading situations since it is counter-intuitive and is not allowed in many systems (Bertsimas and Lo, 1998). The inclusion of non-negativity constraints would improve the performance of the trade regime substantially. However, adding non-negativity constraints to a dynamic programming problem increases the complexity of the problem considerably. Bertsimas and Lo (1998) provide a closed form solution for the problem with non-negativity constraints when price dynamics follow a special pattern:

$$p_t = \theta p_{t-1} + a x_t u_t + \epsilon_t,$$

$$log x_t = log x_{t-1} + \eta_t.$$

However, as Huberman and Stanzl (2005) conclude, a linear price impact model is the most representative formulation for price impact in most real-world situations. The above special form of price evolution formula concerns only a limited case and is not an appropriate substitute for price impact in practical situations. In this paper we assume a linear price evolution model such as (4.6).

If we assume non-negativity must hold, the optimal trade size is decided by:

$$\begin{aligned} u_t^* &= \max(0, \min(s_t, -\frac{y_t^\intercal L_t}{2\alpha_t})) \\ &= \max(0^\intercal y_t, \min(e_{m+1}^\intercal y_t, -\frac{y_t^\intercal L_t}{2\alpha_t})), \end{aligned}$$

resulting in three different possible outcomes for each period. In the context of dynamic programming, after each stage in the backward progress through algorithm, the state space of the problem grows threefold (up to $3^T$ in the final stage), which results in considerably larger problems. We discuss the solution approach for handling this in the next section.

## 4.3 Approximate Dynamic Programming

Approximate dynamic programming is a range of approximation tools that are devised to address the inherent problems of dynamic programming framework in dealing with constraints. Classic approaches to dynamic programming are unable to deal with exponential growth of the computational requirements as

the number of states increase. Unless the problems are defined to very restrictive assumptions, a simulation of the system is more easily constructed than a model. In this paper we employ a value function approximation method where we replace the piecewise value function which is resulted from the addition of non-negativity constraints with a single value function that best captures the characteristics of the three functions.

### 4.3.1 An Approximated Value Function

In order to overcome the added complexity of the non-negativity constraints, we approximate the cost to go at period $(t+1)$ with a quadratic function:

$$V_{t+1}(y_{t+1}) = y_{t+1}^{\intercal} Q_{t+1} y_{t+1} + B_{t+1}^{\intercal} y_{t+1} + C_{t+1},$$

where $Q_t$ and $B_t$ are quadratic and linear coefficients of the value function during period $t$ respectively and $C_t$ is the constant term.

Following the Bellman procedure, we have:

$$\begin{aligned}
V_t(y_t) &= \min_{0\le u_t\le S_t} \mathbb{E}\left[g(y_t,u_t) + V_{t+1}(y_{t+1})\right] \\
&= \min_{0\le u_t\le S_t} \mathbb{E}[y_t^{\intercal}(A^{\intercal}e_1)u_t + b^{\intercal}e_1u_t^2 + w_t^{\intercal}e1u_t \\
&\quad +(Ay_t+bu_t+w_t)^{\intercal}Q_{t+1}(Ay_t+bu_t+w_t) + B_{t+1}^{\intercal}(Ay_t+bu_t+w_t) + C_{t+1}] \\
&= \min_{0\le u_t\le S_t} [b^{\intercal}e_1 + b^{\intercal}Q_{t+1}b]u_t^2 + [y_t^{\intercal}(A^{\intercal}e_1 + A^{\intercal}(Q_{t+1}+Q_{t+1}^{\intercal})b) + B_{t+1}^{\intercal}b]u_t \\
&\quad +[(Ay_t)^{\intercal}Q_{t+1}(Ay_t) + B_{t+1}^{\intercal}Ay_t + \mathbb{E}(\omega_t^{\intercal}Q_{t+1}\omega_t) + C_{t+1}].
\end{aligned}$$

If there were no constraints on $u_t$, then the optimal $u_t$ would be

$$u_t^* = -\frac{y_t^{\intercal}L_t + \beta_t}{2\alpha_t},$$

where $L_t = (A^{\intercal}e_1) + A^{\intercal}(Q_{t+1}+Q_{t+1}^{\intercal})b$, $\beta_t = B_{t+1}^{\intercal}b$, and $\alpha_t = b^{\intercal}e_1 + b^{\intercal}Q_{t+1}b$.

Considering the three possible outcomes based on where $u_t^*$ is situated in the state space, we will have three possible optimal $u_t$, each of which will yield different value functions:

$$u_t = \begin{cases} 0 & \text{if } -\frac{y_t^{\intercal}L_t+\beta_t}{2\alpha_t} < 0, \\ S_t & \text{if } -\frac{y_t^{\intercal}L_t+\beta_t}{2\alpha_t} > S_t, \\ -\frac{y_t^{\intercal}L_t+\beta_t}{2\alpha_t} & \text{if } 0 \le -\frac{y_t^{\intercal}L_t+\beta_t}{2\alpha_t} \le S_t \end{cases}$$

Let us define

$$U_1 = \{y| - \frac{y_t^\intercal L_t + \beta_t}{2\alpha_t} < 0\},$$

$$U_2 = \{y| - \frac{y_t^\intercal L_t + \beta_t}{2\alpha_t} > s_t\},$$

$$U_3 = \{y|0 \leq -\frac{y_t^\intercal L_t + \beta_t}{2\alpha_t} \leq s_t\},$$

to be mutually exclusive domains for $y$. If we replace the $u_{t,j}$ $(j = 1, 2, 3)$ in $V_t(y_t)$ with above results, we will have the following optimal cost-to-go at each stage $t$ $(t = 1, 2, ..., T)$:

**Case j=1** : $y \in U_1$

$$V_t(y_t) = y_t^\intercal (A^\intercal Q_{t+1} A) y_t + B_{t+1}^\intercal A y_t + \mathbb{E}(\omega_t^\intercal Q_{t+1} \omega_t) + C_{t+1}.$$

**Case j=2** : $y \in U_2$

$$V_t(y_t) = y_t^\intercal (A^\intercal Q_{t+1} A + L_t e_{m+1}^\intercal + e_{m+1} \alpha_t e_{m+1}^\intercal) y_t + [B_{t+1}^\intercal A + \beta_t e_{m+1}^\intercal] y_t + \mathbb{E}(\omega_t^\intercal Q_{t+1} \omega_t) + C_{t+1}.$$

**Case j=3** : $y \in U_3$

$$V_t(y_t) = y_t^\intercal [A^\intercal Q_{t+1} A - \frac{3 L_t L_t^\intercal}{4\alpha_t}] y_t + [B_{t+1}^\intercal A - \frac{3 L_t^\intercal \beta_t}{2\alpha_t}] y_t + [\mathbb{E}(\omega_t^\intercal Q_{t+1} \omega_t) + C_{t+1} - \frac{3\beta_t^2}{4\alpha_t}].$$

In all three cases, this results in the quadratic form

$$V_t(y_t) = y_t^\intercal \Phi_{t,j} y_t + \Pi_{t,j} y_t + \Omega_{t,j},$$

with $\Phi_{t,j}$, $\Pi_{t,j}$ and $\Omega_{t,j}$ being defined appropriately according to the three aforementioned cases.

We now approximate the value function based on the three distinct functions resulting from each possible $u_t$ by finding a quadratic function over the whole state space where the square distance from each of the above functions to it are minimum. For simplicity and clarity, we assume a linear price impact with AR(1). These results can be extended to AR(m) without the loss in generality. If we assume linear price formulation and no time lag in the price evolution, $A$ and $b$ have the following formats:

$$A = \begin{bmatrix} 1 & 0 & \beta \\ 0 & 1 & 0 \\ 0 & 0 & \rho \end{bmatrix}, \quad b = \begin{bmatrix} a \\ -1 \\ 0 \end{bmatrix}.$$

**Lemma 4.2.** *$\Phi_{t,j}$ and consequently $Q_t$ have the following format:*

$$\Phi_{t,j} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2 \\ 0 & d_3 & d_4 \end{bmatrix}, Q_t = \begin{bmatrix} 0 & 1 & 0 \\ 0 & x_1 & x_2 \\ 0 & x_3 & x_4 \end{bmatrix}.$$

*Proof.* see appendix B. □

Under linear price impact, the approximate value function has the following form :

$$\hat{V}_t(y_t) = y_t^\intercal Q_t y_t + B_t^\intercal y_t + C_t,$$

where

$$Q_t = \begin{bmatrix} 0 & 1 & 0 \\ 0 & q_{t,1} & q_{t,2} \\ 0 & q_{t,3} & q_{t,4} \end{bmatrix}, \text{ and } B_t = \begin{bmatrix} 0 \\ r_{t,1} \\ r_{t,2} \end{bmatrix}.$$

We need to find $(Q_t, B_t, C_t)$ that minimises the least squares equations.

$$(Q_t, B_t, C_t) = \underset{(\{q_t\},\{r_t\},C_t)}{\operatorname{argmin}} \int_{y_t} (V_t(y_t) - \hat{V}_t(y_t))^2 dy_t,$$

where

$$\begin{aligned} &\int_{y_t} (V_t(y_t) - \hat{V}_t(y_t))^2 dy_t = \\ &\int_{U_1} [y_t^\intercal (Q_t - \Phi_{t,1}) y_t + (B_t^\intercal - \Pi_{t,1}) y_t + C_t - \Omega_{t,1}]^2 dy_t \\ + &\int_{U_2} [y_t^\intercal (Q_t - \Phi_{t,2}) y_t + (B_t^\intercal - \Pi_{t,2}) y_t + C_t - \Omega_{t,2}]^2 dy_t \\ + &\int_{U_3} [y_t^\intercal (Q_t - \Phi_{t,3}) y_t + (B_t^\intercal - \Pi_{t,3}) y_t + C_t - \Omega_{t,3}]^2 dy_t. \end{aligned}$$

Figure 4.1 depicts a simplified graph that shows the approximated value function based on the three value functions that are optimal in each region.

The quadratic coefficient $(Q_t - \Phi_{t,j})$ is

$$Q_t - \Phi_{t,j} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & q_{t,1} - k_{t,j,1} & q_{t,2} - k_{t,j,2} \\ 0 & q_{t,3} - k_{t,j,3} & q_{t,4} - k_{t,j,4} \end{bmatrix},$$

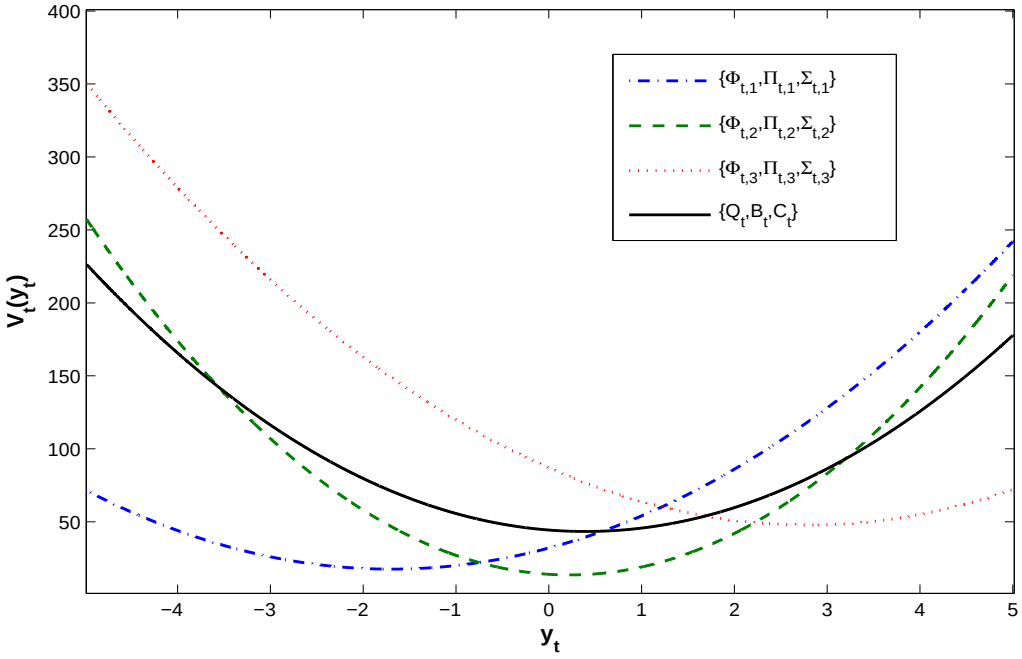


**Figure 4.1:** The three regions (representing $U$ in cases $j = 1, 2, 3$) and approximation of the three value functions that are optimal in each region.

and the linear coefficient is

$$B_t^\intercal - \Pi_{t,j} = \begin{bmatrix} 0 \\ r_{t,1} - f_{t,j,1} \\ r_{t,2} - f_{t,j,2} \end{bmatrix},$$

where $k_{t,j,i}$ is the $i_{th}$ element of $\Phi_{t,j}$ corresponding to that of $Q$ and $f_{t,j,i}$ is the $i_{th}$ element of $\Pi_{t,j}$ corresponding to elements of $B_t$. The above results also render $p$, the first element of state space, irrelevant for the rest of the calculations. Finally we have $C_t - n_{t,j}$ as the constant term.

At each stage t, we need to solve the following minimisation problem:

$$(\{q\}, \{r\}, C) = argmin_{q,r,n} \int_y (V(y) - \hat{V}(y))^2 dy$$

where

$$\begin{aligned}
& \int_y (V(y) - \hat{V}(y))^2 dy \qquad (4.7)\\
= & \int_{U_1} ([s\ x] \begin{bmatrix} q_1 - k_{1,1} & q_2 - k_{1,2} \\ q_3 - k_{1,3} & q_4 - k_{1,4} \end{bmatrix} \begin{bmatrix} s \\ x \end{bmatrix} + \begin{bmatrix} r_1 - f_{1,1} & r_2 - f_{1,2} \end{bmatrix} \begin{bmatrix} s \\ x \end{bmatrix} \\
& + C - n_1)^2 dx ds \\
+ & \int_{U_2} ([s\ x] \begin{bmatrix} q_1 - k_{2,1} & q_2 - k_{2,2} \\ q_3 - k_{2,3} & q_4 - k_{2,4} \end{bmatrix} \begin{bmatrix} s \\ x \end{bmatrix} + \begin{bmatrix} r_1 - f_{2,1} & r_2 - f_{2,2} \end{bmatrix} \begin{bmatrix} s \\ x \end{bmatrix} \\
& + C - n_2)^2 dx ds \\
+ & \int_{U_3} ([s\ x] \begin{bmatrix} q_1 - k_{3,1} & q_2 - k_{3,2} \\ q_3 - k_{3,3} & q_4 - k_{3,4} \end{bmatrix} \begin{bmatrix} s \\ x \end{bmatrix} + \begin{bmatrix} r_1 - f_{3,1} & r_2 - f_{3,2} \end{bmatrix} \begin{bmatrix} s \\ x \end{bmatrix} \\
& + C - n_3)^2 dx ds.
\end{aligned}$$

The three regions are obtained as follows:

- Case $j = 1$ ($U_1$):

$$\frac{-y_t^\intercal L_t - \beta_t}{2\alpha_t} < 0 \Leftrightarrow y_t^\intercal L_t > -\beta_t \Leftrightarrow \begin{bmatrix} s & x \end{bmatrix} \begin{bmatrix} l_2 \\ l_3 \end{bmatrix} > -\beta_t \Leftrightarrow$$

$$l_2 s + l_3 x > -\beta_t \Leftrightarrow x > \frac{-\beta_t - l_2 s}{l_3}$$

- Case $j = 2$ ($U_2$):

$$-(\frac{y_t^\intercal L_t + \beta_t}{2\alpha_t}) > s_t \Leftrightarrow y_t^\intercal L_t < -\beta_t - 2\alpha_t s_t \Leftrightarrow \begin{bmatrix} s & x \end{bmatrix} \begin{bmatrix} l_2 \\ l_3 \end{bmatrix} < -\beta_t - 2\alpha_t s_t \Leftrightarrow$$

$$l_2 s + l_3 x < -\beta_t - 2\alpha_t s_t \Leftrightarrow x < \frac{-\beta_t - 2\alpha_t s_t - l_2 s}{l_3}$$

- Case $j = 3$ ($U_3$):

$$0 \leq -(\frac{y_t^\intercal L_t + \beta_t}{2\alpha_t}) \leq S_t \Leftrightarrow -2\alpha_t S_t - \beta_t \leq y_t^\intercal L_t \leq -\beta_t \Leftrightarrow$$

$$-2\alpha_t S_t - \beta_t \leq \begin{bmatrix} s & x \end{bmatrix} \begin{bmatrix} l_2 \\ l_3 \end{bmatrix} \leq -\beta_t - \beta_t - 2\alpha_t s_t < l_2 s + l_3 x < -\beta_t \Leftrightarrow$$

$$\frac{-\beta_t - 2\alpha_t s_t - l_2 s}{l_3} < x < \frac{-\beta_t - l_2 s}{l_3}$$

where $l_2$ and $l_3$ are elements of $L_t$: $\begin{bmatrix} 0 & l_2 & l_3 \end{bmatrix}^\intercal$.

Considering both $x$ and $s$ have an additional range that they must comply with: ($x \in [-P_0, P_0]$ and $s \in [0, s_t]$), we have the following ranges to apply in the formulation. Let elements of $x$ and $s$ be parameters $a_1$, $a_2$, $b_1$ and $b_2$ such that $x \in [a_1, a_2]$ and $s \in [b_1, b_2]$.

$$U_1 = \{(x,s)|x \in [a_1 = \frac{-\beta_t - l_2 s}{l_3}, a_2 = P_0], s \in [b_1 = 0, b_2 = s_t]\},$$

$$U_2 = \{(x,s)|x \in [a_1 = -P_0, a_2 = \frac{-\beta_t - 2\alpha_t s_t - l_2 s}{l_3}], s \in [b_1 = 0, b_2 = s_t]\},$$

$$U_3 = \{(x,s)|x \in [a_1 = \frac{-\beta_t - 2\alpha_t s_t - l_2 s}{l_3}, a_2 = \frac{-\beta_t - l_2 s}{l_3}], s \in [b_1 = 0, b_2 = s_t]\}$$

Applying these ranges on (4.7) and expanding we will have the following which can be used to obtain the optimal values for Q, R and C.

$$\int_y (V(y) - \hat{V}(y))^2 dy = \int_0^{s_t} \int_{\frac{-\beta_t - l_2 s}{l_3}}^{P_0} \boldsymbol{F} \; dxds$$
$$+ \int_0^{s_t} \int_{-P_0}^{\frac{-\beta_t - 2\alpha_t s_t - l_2 s}{l_3}} \boldsymbol{F} \; dxds + \int_0^{s_t} \int_{\frac{-\beta_t - 2\alpha_t s_t - l_2 s}{l_3}}^{\frac{-\beta_t - l_2 s}{l_3}} \boldsymbol{F} \; dxds$$

where

$$\boldsymbol{F} = ((q_1 - k_{j,1})s^2 + (q_4 - k_{j,4})x^2 + ((q_2 - k_{j,2}) + (q_3 - k_{j,3}))sx$$
$$+ (r_1 - f_{j,1})s + (r_2 - f_{j,2})x + C - n_j)^2$$

The above equations are quadratic functions on $(\{q\}, \{r\}, n)$ and can be rewritten as $R^\intercal M R + O^\intercal R + N$, where $R$ is the vector $[q_1, q_2, q_3, q_4, r_1, r_2, C]$. $M$ is a symmetric matrix derived from the coefficients of the above formulation, and $O$ is the linear coefficient.

The resulting equation is of quadratic form and yields an optimal solution.

## 4.4 Numerical Analysis

Our numerical analysis consists of two studies. In section 4.4.1 we simulate the example used in Bertsimas and Lo (1998) to illustrate the comparative advantages gained through our approximate dynamic programming method with

**Algorithm 3:** Approximate Dynamic Programming

**input:** $\alpha_t$, $\beta_t$, $L_t$, $\{Q_t^0, B_t^0, C_t^0\}$ for all $t$

**output:** $\{Q_t^*, B_t^*, C_t^*\}$

**for** $i = 1 \rightarrow M$ **do**

  **for** $t = (T-1) \rightarrow 1$ **do**

    **for** $j = 1 \rightarrow 3$ **do**

      $\{\Phi_{t,j}^i, \Pi_{t,j}^i, \Omega_{t,j}^i\} =$ `F1(`$Q_{t+1}^{i-1}, B_{t+1}^{i-1}, C_{t+1}^{i-1}, j$`)`

  **for** $t = (T-1) \rightarrow 1$ **do**

    $\{Q_t^i, B_t^i, C_t^i\} =$ `F2(` $\Phi_{t,j}^i, \Pi_{t,j}^i, \Omega_{t,j}^i$`)`

  **if** $VF\{Q_t^i, B_t^i, C_t^i\} < VF\{Q_t^{i-1}, B_t^{i-1}, C_t^{i-1}\}$ **then**

    **for** $t = 1 \rightarrow T$ **do**

      $\{Q_t^*, B_t^*, C_t^*\} = \{Q_t^i, B_t^i, C_t^i\}$

---

$\{\Phi,\Pi,\Omega\}$ = **Function** `F1` $(Q, B, C, j)$

**if** $j = 1$ **then**

  $\Phi = A^\intercal Q A$

  $\Pi = B^\intercal A$

  $\Omega = \mathbb{E}(\omega_t^\intercal Q \omega_t) + C$

**if** $j = 2$ **then**

  $\Phi = A^\intercal Q A + L_t e_{m+1}^\intercal + e_{m+1} \alpha_t e_{m+1}^\intercal$

  $\Pi = B^\intercal A + \beta_t e_{m+1}^\intercal$

  $\Omega = \mathbb{E}(\omega_t^\intercal Q \omega_t) + C$

**if** $j = 3$ **then**

  $\Phi = A^\intercal Q A - \frac{3 L_t L_t^\intercal}{4\alpha_t}$

  $\Pi = B^\intercal A - \frac{3 L_t^\intercal \beta_t}{2\alpha_t}$

  $\Omega = \mathbb{E}(\omega_t^\intercal Q \omega_t) + C - \frac{3\beta_t^2}{4\alpha_t}$

---

$\{Q, B, C\}$ = **Function** `F2` $(\Phi_t, \Pi_t, \Omega_t)$

$$\begin{aligned}\underset{(\{Q\},\{B\},C)}{\operatorname{argmin}} \int_{U_1} &[y_t^\intercal (Q_t - \Phi_{t,1}) y_t + (B_t^\intercal - \Pi_{t,1}) y_t + C_t - \Omega_{t,1}]^2 dy_t \\ &+ \int_{U_2} [y_t^\intercal (Q_t - \Phi_{t,2}) y_t + (B_t^\intercal - \Pi_{t,2}) y_t + C_t - \Omega_{t,2}]^2 dy_t \\ &+ \int_{U_3} [y_t^\intercal (Q_t - \Phi_{t,3}) y_t + (B_t^\intercal - \Pi_{t,3}) y_t + C_t - \Omega_{t,3}]^2 dy_t\end{aligned}$$

non-negativity constraints. We compare our method (ADP) with their closed form solution (B&L) and a naive strategy where the trade is divided into equal sizes to be executed over the trading horizon. In this case we test the efficiency of each optimal strategy with regard to different realisations of information variable $x$. Since the variable $x$ is the main driving factor in the volatility of prices in the scenario presented by Bertsimas and Lo (1998), we test the performance of the algorithms against different rates of variance in $x$.

In section 4.4.2 we simulate trading of a security from London Stock Exchange on a specific date based on intra-day trade information of that day. This is to illustrate the effectiveness of our method in a practical setting and its advantage over the classic and naive methods that do not include the non-negativity constraints.

We code the algorithms in MATLAB and run all the experiments on a 64-bit Windows 7 workstation having 4GB of RAM and quad-core Intel CPU at 2.6GHz. All the experiments were very fast and as such we forego a detailed examination of the time performance. As a brief guideline, table 4.1 shows the time taken for a single run of each of the algorithms for various trading periods and market volatility rates.

### 4.4.1 A Simulated Example

The example involves execution (buy) of 100000 shares over $T = 20$ periods. The current price is \$50. The parameters are set as follows: $a = 5 \times 10^{-5}$, $\beta = 5$, $\rho = 0.5$ and $\sigma_\epsilon^2 = (0.125)^2$. For a full description of the values and the reasoning behind the choice of the values, interested readers are referred to Bertsimas and Lo (1998). In summary, $a$ is chosen to yield a price impact of \$500000 if the trader executes the 100000 shares in one transaction. The standard deviation of $\epsilon_t$ is calibrated to be one tick (12.5 cents) per period.

We assign different values to $\sigma_\eta^2$ (variance of the information evolution, a white noise process, representing the overall market volatility) in order to test the efficiency of the algorithms with regard to the market behaviour where it might drive the prices significantly up or down and render the non-negativity constraints relevant. For example if the optimal strategy where we do not ap-

| | $\sigma_\epsilon^2$=0.01 | | $\sigma_\epsilon^2$=0.1 | | $\sigma_\epsilon^2$=1 | |
|---|---|---|---|---|---|---|
| Period | B&L | ADP | B&L | ADP | B&L | ADP |
| 2 | 0.26 | 12.78 | 0.31 | 13.05 | 0.38 | 13.31 |
| 4 | 0.45 | 14.57 | 0.51 | 15.77 | 0.58 | 16.89 |
| 6 | 0.67 | 19.43 | 0.78 | 21.93 | 0.87 | 24.43 |
| 8 | 1.00 | 28.85 | 1.11 | 33.32 | 1.22 | 38.19 |
| 10 | 1.37 | 45.95 | 1.53 | 52.84 | 1.68 | 59.84 |
| 12 | 1.92 | 70.15 | 2.13 | 80.01 | 2.35 | 89.70 |
| 14 | 2.60 | 103.12 | 2.82 | 116.53 | 3.04 | 129.81 |
| 16 | 3.32 | 147.59 | 3.56 | 164.93 | 3.80 | 182.29 |
| 18 | 4.19 | 204.63 | 4.49 | 226.87 | 4.75 | 249.02 |
| 20 | 5.05 | 276.41 | 5.35 | 303.77 | 5.63 | 330.96 |

**Table 4.1:** Execution time (in milliseconds) of B&L and ADP algorithms for variable number of periods (T) and market volatility rates ($\sigma_\epsilon^2$ values)

ply the non-negativity constraints is to sell during a buy operation, the original method would likely offer negative trade as part of its optimal solution. However, if the assumption is that short-selling is not possible, then the original algorithm would not be able to accommodate the above cases. Naive strategy on the other hand is unable to utilise the swings in the price to minimise the incurred overall cost. The approximate dynamic programming method, as expected, provides superior results compared to the original method of Bertsimas and Lo (1998) and the naive strategy under a volatile market condition when the assumption is that a buy operation cannot include sales.

As can be seen in Table 4.2, the expected value function values improve substantially for the ADP method compared to B&L when the variance of the market volatility increases. Naive strategy is equivalent in almost all instances of $\sigma_\eta^2$ to that of B&L and hence not shown here. Table 4.2, however, depicts the expected value function as a factor for comparison. As we are dealing with the volatility of the market, the more important aspect of the improvement of the ADP method over the other methods is apparent in the actual execution cost over the whole trade period as can be seen in Figure 4.2, which depicts

| $\sigma_\eta^2$ | ADP | B&L | % Diff. |
|---|---|---|---|
| 0.001 | 5255799 | 5258643 | 0.054 |
| 0.002 | 5250526 | 5253792 | 0.069 |
| 0.005 | 5235928 | 5241587 | 0.108 |
| 0.01 | 5212885 | 5221218 | 0.159 |
| 0.02 | 5168374 | 5180444 | 0.233 |
| 0.05 | 5038702 | 5058038 | 0.383 |
| 0.1 | 4826652 | 4853935 | 0.565 |

**Table 4.2:** Expected execution cost in example execution of 100000 shares

the realized execution cost for various $\sigma_\eta^2$ rates. To achieve this, we run each algorithm with random realisations of $x$ and calculate the execution cost at each stage. This graph shows the average of total execution cost for 50 simulation runs for each method.

The performance of three methods are very similar when the $\sigma_\eta^2$ rates are very small and the probability of negative trade being optimal is relatively low. Since B&L method provides optimality with regard to the uncertainty in the market and includes short-selling in its policies, when the short-selling is prohibited, it loses that advantage. This is reflected in the average execution cost values in $\sigma_\eta^2$ rates of 1 and above. The ADP, on the other hand, takes into account the non-negativity constraints and provides optimal policies in which the majority of the trade is performed when the price has swung down and little trade when the prices are high. In other words, ADP is better equipped to take advantage of price variations without the need for short-selling. Note the sharp drop in ADP execution costs when the volatility increases beyond 1, where it is able to execute with higher probability a larger amount of the security in lower prices leading to lower overall execution cost.

### 4.4.2 An Empirical Example

The example shown in section 4.4.1 proves the suitability of the ADP approach in situations where short-selling is not allowed. However to gauge the improve-

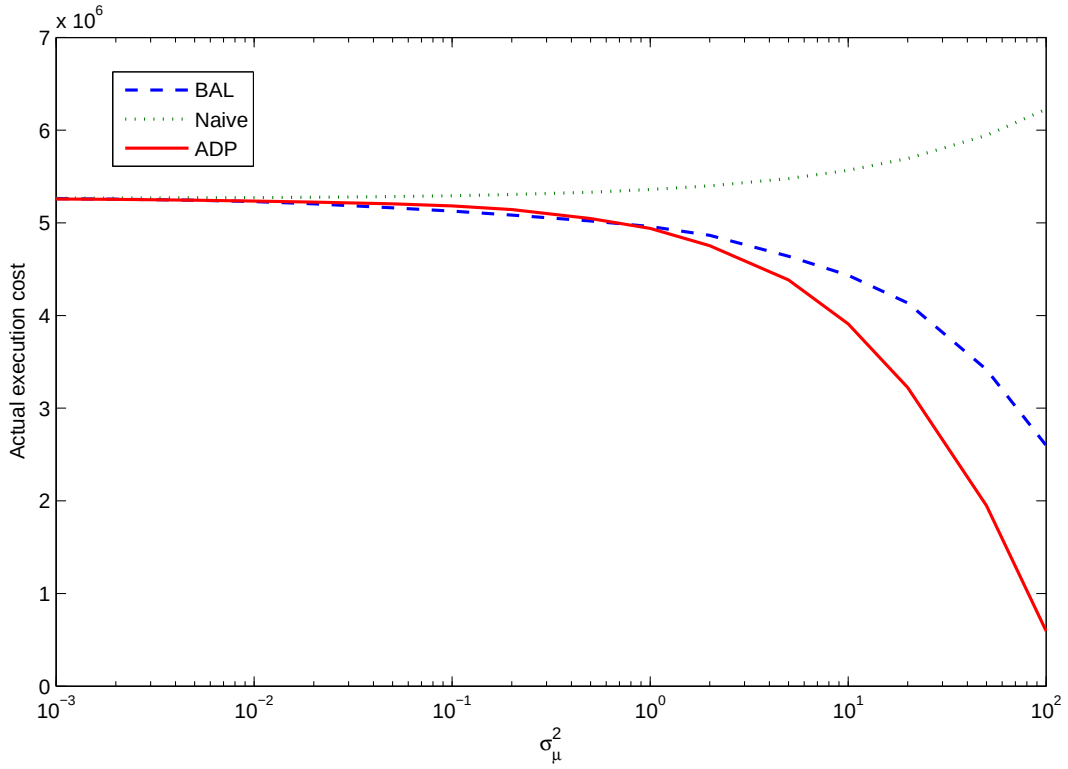


**Figure 4.2:** Actual execution cost for the three methods with variable $\sigma_\eta^2$ rates

ment that this method will yield in a real-world situation we test the algorithm on real securities traded in the market and compare it to that of B&L as a benchmark on what would be the optimal policy when we ignore the non-negativity constraints.

We considered three securities from London Stock Exchange that represent various levels of liquidity and recorded the tick price data for each of the selected stock for the duration of two weeks. FTSE100 index was considered as the information vector $x$ during the same period. We also recorded the publicly available trade and price information of each share for a particular day during the two weeks. Out of the FTSE100 companies, we chose Lloyds Banking Group shares as a high liquidity share, Rolls-Royce as a medium liquidity and Next Plc as representative of low liquidity securities. The choice of the three securities was not based on strict criteria but a loose interpretation of liquidity in order to evaluate the effects when applying the methods in this paper on stocks of varying liquidity. We also acquired intra-day price data for these securities where the information is available for the opening and closing price and the volume of

| | Lloyds | Rolls-Royce | Next Plc |
|---|---|---|---|
| Price (1 lag) | $9.805e{-}01^{***}$ | $8.776e{-}01^{***}$ | $9.159e{-}01^{***}$ |
| Std. Error | $(1.615e-02)$ | $(1.886e-02)$ | $(2.370e-02)$ |
| Trade Volume | $-1.322e-10$ | $1.227e-06^{+}$ | $6.378e-05^{*}$ |
| Std. Error | $(2.861e-09)$ | $(1.359e-06)$ | $(3.140e-05)$ |
| UK100 Index | $7.790e-05^{+}$ | $2.802e{-}02^{***}$ | $4.523e{-}02^{***}$ |
| Std. Error | $(1.573e-04)$ | $(4.223e-03)$ | $(1.335e-02)$ |
| Residual Std. Err. | 0.227 | 3.735 | 11.06 |
| $R^2$ | 0.9474 | 0.9713 | 0.9612 |
| Adj. $R^2$ | 0.9467 | 0.971 | 0.9607 |

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$, $^{+}p < 0.1$

**Table 4.3:** Regression analysis on the three chosen stocks

trade during each minute. We considered the price between the opening and closing price to be the price of the share in that time period.

The first step is to estimate the parameters of $x_{t+1} = \rho x_t + \eta_t$ in order to be able to forecast the value of $x$ from previous period. We performed a linear regression fit in R statistical software based on the historical time series data above. The parameters of information evolution ($x_{t+1} = \rho x_t + \eta_t$) which represent FTSE100 index were calculated as follows:

$$x_{t+1} = 70.91 + 0.98x_t + \eta_t,$$

where $\mu_\eta = 0 \quad \text{and} \quad \sigma_\eta = 14.15$.

Table 4.3 details the coefficients of price evolution model $p_{t+1} = p_t + \beta x_t + au_t + \epsilon_t$ in relation to each chosen security.

Based on the parameters found in Table 4.3 pertaining to the behaviour of price based on the independent variables, as well as the parameters of the information evolution, we run our algorithm against that of Bertsimas and Lo (1998) to gauge the performance of these algorithms in a real-world application. We compare the results based on different number of trade periods as well as different rates of market volatility.

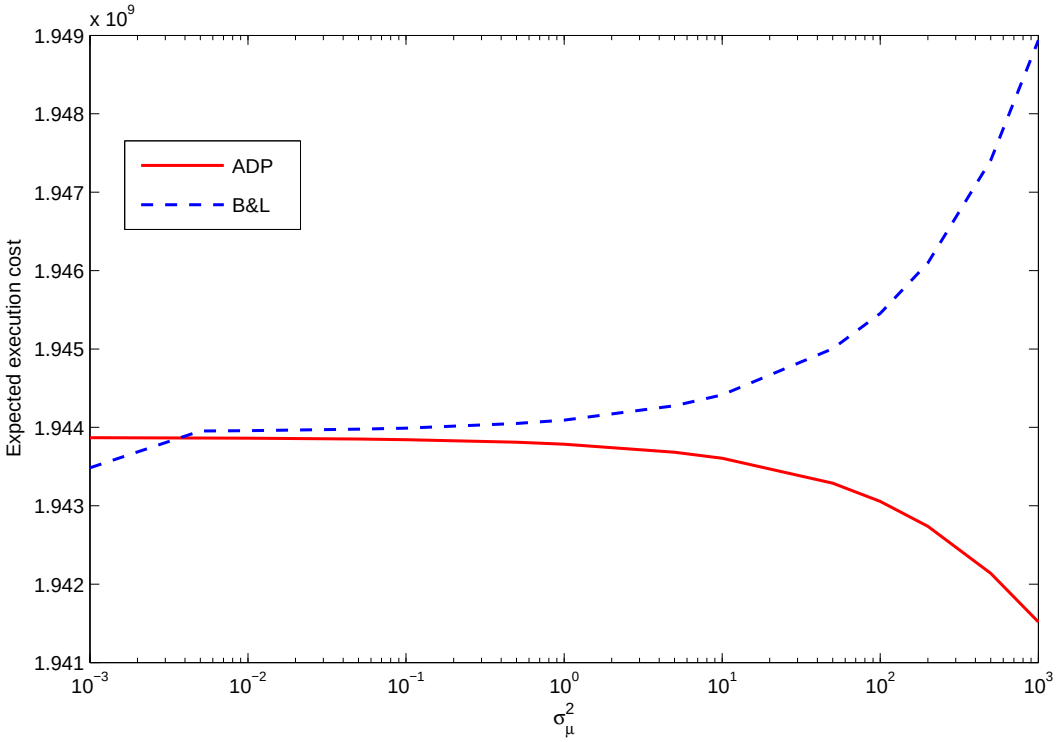


**Figure 4.3:** Expected execution cost for Lloyds share price with variable rates of $\sigma_\eta^2$

Figure 4.3 depicts the expected value function for the two methods based on 10 trade periods and varying degrees of market volatility. As can be seen, the B&L method only outperforms our method in the case of very small market volatility. As the volatility increases, the expected cost of trading the predefined number of shares increases for the B&L method. The ADP method increases in performance as the volatility increases while the performance of B&L is declining. This is indicative of the ability of the ADP method in taking advantage of price swings when the B&L falls short because of neglecting the sign constraints. If short-selling is prohibited, our method adapts the policies online by taking the non-negativity into its framework from the start.

Table 4.4 outlines the actual execution cost of the two algorithms for the Lloyds shares. The ADP algorithm outperforms the original B&L method on most cases. When the trading periods are relatively few, there will naturally be less probability of favourable prices occurring and thus the ADP algorithm is slightly under-performing.

Figure 4.4 on the other hand depicts the results of the actual execution

| # of periods | ADP | B&L | % Diff. |
|---|---|---|---|
| 10 | 8868042 | 8867250 | -0.008 |
| 20 | 10595009 | 10586747 | -0.077 |
| 30 | 11381698 | 11398890 | 0.151 |
| 40 | 11552346 | 11563438 | 0.096 |
| 50 | 11522654 | 11543933 | 0.184 |
| 60 | 11484477 | 11509757 | 0.220 |
| 70 | 11449253 | 11475887 | 0.232 |
| 80 | 11408889 | 11427038 | 0.159 |
| 90 | 11362621 | 11364245 | 0.014 |
| 100 | 11290233 | 11304324 | 0.124 |

**Table 4.4:** Actual execution cost realisation from each algorithm in execution of 100000 Lloyds shares and the percentage difference between the two algorithms based on the number of trading periods.

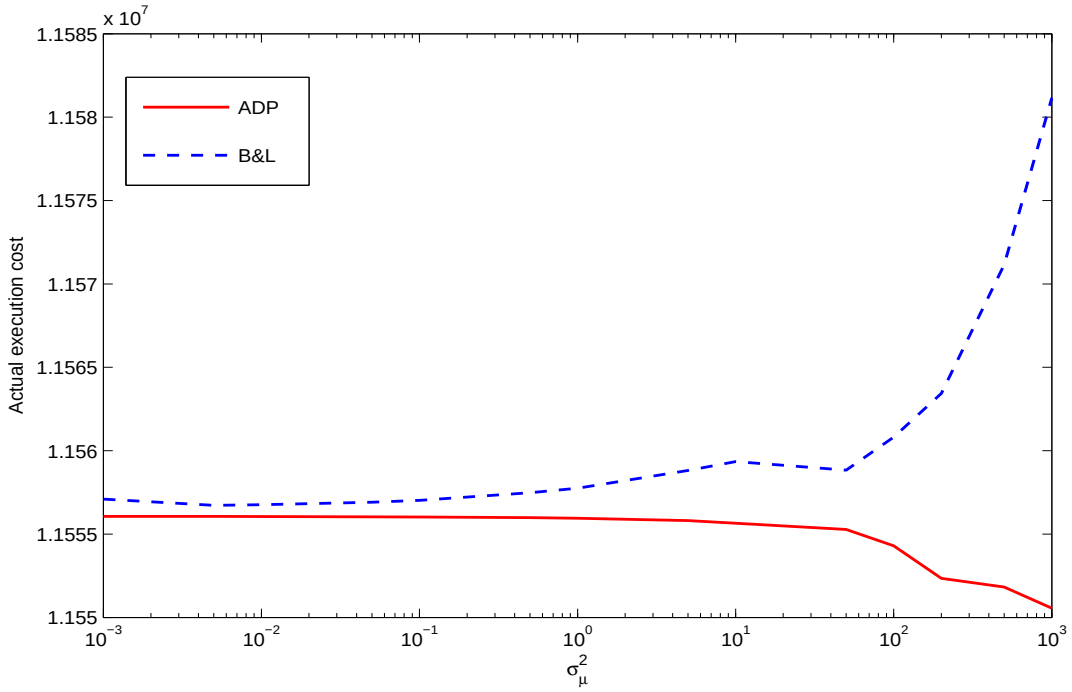


**Figure 4.4:** Actual execution cost for Lloyds share price with variable market volatility rates

cost for the case of Lloyds share prices with both methods over varying market volatility. The benefits of the approximate dynamic programming method over the B&L method is apparent from this figure when the volatility increases.

Both other stocks show similar behaviour with varying trade periods and market volatility. Due to space limitation, the interested reader can find similar graphs to above for the Next plc and Rolls-Royce shares in appendix C.

## 4.5 Conclusion

Our extensions on the work of Bertsimas and Lo (1998) is aimed at bringing the problem closer to the real-world applications. We extend the AR(1) of the classic model to AR(m) and include non-negativity constraints. These extensions add significantly to the complexity of the problem. It is a well known shortcoming of the dynamic programming method that it cannot accommodate constraints very easily. We provide an approximate dynamic programming method that circumvents this. Our simulated and empirical results support the inclusion of non-negativity constraints, when the volatility of the market, and consequently of prices, are so that the probability of negative trades is significant. It improves the expected (and actual) execution cost in real circumstances and in comparison to the model by Bertsimas and Lo (1998). The difference between the performance of the models is not substantial when applied to low volatility scenarios. However, as we raise the market and price fluctuations in the model, the significance of our method over the other two methods become apparent. The benefits of the new method is also apparent in the empirical study where we test the methods on a set of data that warrants large swings in price. In both cases we observe improvement in the results which encourage the use of the new method over the previous methods under the conditions discussed.

# Chapter 5

# Conclusions, Limitations and Future Directions

In this thesis we have focused on various applications of dynamic programming models in the presence of uncertainty. We have looked at the dynamic pricing of non-durable goods in the context of Markov decision processes. We have also considered the dynamic pricing problem with a linear representation of the demand evolution formulation. Finally, we have studied the dynamic trading of large blocks of securities where the market is sensitive to the volume of our trade. In this final chapter we summarise the contents of each chapter and recount the contributions of our study to the literature as well as the limitation of this research and potential directions for future.

## 5.1 Dynamic Pricing with POMDP

In chapter 2 we studied the dynamic pricing of non-durable products with no replenishment possibility. We studied the problem in the framework of partially observable Markov decision processes (POMDP), because the underlying demand model is unobservable during the season. Instead we used sales as a replacement parameter and explored its probabilistic relation to demand. We also provided proof that the objective function of the model in dynamic pricing context is convex and piecewise linear and utilised a number of exact methods from the literature that solve this problem to optimality. The ability of these

exact methods were demonstrated and it was shown that not only they offer optimal results in reasonable times, they are (especially incremental pruning algorithm) capable of solving full featured problems of the kind we encounter in retail industry pertaining to the problems under study.

Thus, our main conclusion was that the exact methods to solve POMDP are efficient and appropriate options, especially with the advent of more powerful computation capacity and the emergence of more robust variants of exact methods to POMDP models. These algorithms take advantage of convexity and piecewise linearity of the value function and reduce the state space considerably.

However, this research naturally contains some underlying limitations and shortcomings. Here we address these shortcomings and discuss the potential strategies for the interested reader. In this chapter we solely focused on the existing exact methods in the literature. Although this limits the extent of our contributions to the literature, we decided to limit our research to examining these exact methods instead of developing heuristic or approximate methods. This decision was taken under the consideration that the advent of more powerful computing and more advanced applications of these methods provide a remarkable capability in handling problems of considerable size encountered in real-world situations. However, the inclusion of a heuristic or approximation method would have provided some benefits which was not foreseen initially. First, such additional method would provide a benchmark against which the arguments regarding performance of the exact methods could be assessed. Second, we would be better placed to extract theoretical results in the context of the problem we want to solve. Third and finally, it would offer more robust managerial insights for the decision making process, especially in the real-world applications we are aiming to solve.

Another shortcoming of this chapter was that we decided to forego the use of real data in the examination and assessment of the proposed methods and analyses. On many occasions, the basic theoretical understanding gained by simple Monte Carlo simulations, as conducted in this work, outweigh the amount of resources spent on acquisition of real data and their eventual contribution to the result. However, as we are dealing with practical applications, the benefits of actual data in verification of results is undeniable.

The central premise of this chapter was to establish that the exact methods for POMDP problems provide sufficient performance to be used in industrial context. However, there are various research directions that would extend the performance of the available methods to solve these range of problems without compromising the quality of the solution significantly, although any of these directions would have taken this research beyond its scope.

Approximations in the context of POMDP models are rare, especially approximations built on the premise of the exact methods discussed in this work. Value function approximation models that use the piecewise linearity and convexity of the value function to further simplify the optimisation process are starting to emerge, and significant value can be gained by exploiting such approximations in addition to the adoption of exact methods. Approximate dynamic programming is another solution paradigm that can be used to solve the POMDP problems more efficiently, although significant alterations to the modelling approach is necessary.

Another alternative modelling paradigm is offered by multi-arm bandit problem (MAB) as formulated in Burnetas and Katehakis (1997), in which the authors provide a model that handles incomplete information in an MDP framework. This framework is capable of providing significant insights into the structural properties of the dynamic pricing problem. Furthermore it would lend itself for complimenting and extending the theoretical results we briefly explored in this chapter.

## 5.2 Dynamic Pricing with Autoregressive Demand

Chapter 3 expanded on the dynamic pricing problem of previous chapter. We changed the state evolution model from simple Bayesian updating to a linear model that takes into account the previous demand values in addition to price and volatility. We considered the dynamic pricing of high end fashion products that possess word-of-mouth potential and customers of which behave loyally to specific products or brands. The demand behaviour of customers was discussed

and the benefits of considering an autoregressive model compared to the demand models generally considered in literature were studied. Finally, we provided an approximate dynamic programming solution to the model that utilised the structure of the problem and the uncertainty in state evolution to provide near optimal results and explored the results through a range of numerical analyses.

The results of our simulation runs indicated that the use of the approximate method is beneficial when the inclusion of constraints such as a price limit is necessary. Another benefit of the approximate dynamic programming method was that it enabled us to extend the model to include further attributes inspired by the business problem. We extended the demand evolution model to an autoregressive linear model and offered appropriate solutions to it with the help of approximate dynamic programming. However, demonstrating the benefits of the autoregressive demand model that we introduced is less straightforward, as comparison to other demand models would not provide any insight since the underlying behaviour of the system would be altered.

The above problem is further compounded by the lack of real data from the industry to supplement the results of the research. Most of the benefits of using real-world data in verifying the results of the chapter is evident. However, such data could be further utilised to assess the underlying assumptions in formulation of the problem and inform the development of effective solution methods, especially since we are providing approximate solutions.

Classic dynamic programming method offers an elegant and efficient framework to model dynamic problems. However it is not very efficient when it comes to solving the problems. This is the main reason behind our choice of approximate dynamic programming method in providing a high quality solution without loss of performance. Ability to handle constraints is another capability of the approximate dynamic programming framework that is not easily replaceable. However, there are certain research areas that are promising in offering alternatives. Newsboy problem is a classic operational research problem that has a rich and well developed literature around it. If the periods of operation to the newsboy is extended we are faced with a multi-period dynamic model that is capable of handling similar problems to the dynamic pricing problem we discuss in chapter 3. It can further offer insights into the inventory management

side of the retail business.

Another pertinent and interesting topic regarding dynamic pricing problem is the question of when to change the prices as opposed to the rate of price changes. Various research streams have been developed to address this question, the most promising of which explore the optimal stopping problems as a starting point. A natural extension is a combined pricing problem in which the rate is set at time periods that are decided dynamically as well.

An important feature to consider in the context of dynamic pricing problems is the behaviour of the customers in the environment. There have been much advance in the literature with consideration of strategic customers in deriving the optimal strategy by the decision makers. However, it is yet another shortcoming of the dynamic programming approach that it does not lend itself readily to game theoretic concepts. It would yield great value if inclusion of strategic behaviour in the dynamic programming framework is studied further.

## 5.3 Dynamic Trading

In chapter 4 the ideas developed in the previous chapters were further extended, in that it provided an efficient alternative to the classic dynamic programming methods in situations where added restrictions in the model are necessary. We extended the work of Bertsimas and Lo (1998), which is a pioneering article that utilises dynamic programming for the first time in dynamic trading of large blocks of financial securities. Although Bertsimas and Lo (1998) acquire the closed form solution to the model, they do not take into account the non-negativity constraints which would lead to a short-selling situation. We argued that the expected cost of ignoring the non-negativity constraints is considerable in real-world situations. Furthermore we expanded the state space such that the price and information dynamics can include historical data such as price and information in periods further in the past. As we deal with stock market environment in this problem, the intuitive reason for such an inclusion is the considerable amount of time it would take for the market to observe and register past prices i.e. cases when the market is not fully efficient as often happens in real life. As classic dynamic programming solutions are not capable of handling

constraints, we developed an approximate dynamic programming approach to circumvent this costly possible scenario.

The approximate dynamic programming approach enables us to solve a large combination of problems to near optimality through a generalised platform. Our simulated numerical analysis showed that the ADP method outperforms that of Bertsimas and Lo (1998) when the volatility of the market and prices are high. The approximated method with inclusion of non-negativity constraints was also superior to the exact method without the constraints because it is capable of providing better trading strategy in situations where short selling is not allowed. Although we did not provide a comprehensive benchmark, we validated the fundamental performance of the suggested method by applying it to real data from stock market and comparing the results to that of a naive implementation that represents common practice.

Examination of alternative price evolution models in the dynamic trading context would offer considerable benefits, especially in particular contexts such as special financial products and shares of specific industries. Also depending on the specifics of the problem, we might find more appropriate models for explaining the effects of our decisions. Although we have shown in chapter 4 that the inclusion of additional variables add substantially to the accuracy of the models and their solutions, the modelling of the state evolution should be related to the specific problem and its domain.

In this chapter we offered a novel approximate dynamic programming method that takes advantage of the quadratic value function. However, the approximate dynamic programming field is rife with innovation. Adoption of methods that utilise the advances made in the field and use the specific nature of the problem in formulating the heuristic method is the natural next step in extending the performance.

Finally, the consideration of the dynamic trading problem for a portfolio of securities would be extremely beneficial, especially when there are cross elasticities between the items or strong relationship between the states of the system across periods. The application of the models discussed in this research on a portfolio of products, and in the case of dynamic trading, portfolio of stocks, is trivial in the basic format of these problems. However, the inclusion of con-

straints in our extended models render the application on portfolios much more taxing, but also more rewarding.

# Appendices

## A Proof of Lemma 4.1

**Lemma** $\alpha_t = (b^\intercal e_1 + b^\intercal K_{t+1} b) > 0$ is true for all $t = 1, ..., T-1$.

*Proof.* Since $K_T = (A^\intercal e_1) e_{m+1}^\intercal + b^\intercal e_1 (e_{m+1} e_{m+1}^\intercal)$, we have

$$
\begin{aligned}
K_T &= \begin{bmatrix} 1 \\ 0 \\ \beta \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} + a \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & 1 & 0 \\ 0 & a & 0 \\ 0 & \beta & 0 \end{bmatrix}
\end{aligned}
$$

and

$$
\begin{aligned}
(b^\intercal e_1 + b^\intercal K_T b) =& a + \begin{bmatrix} a & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & a & 0 \\ 0 & \beta & 0 \end{bmatrix} \begin{bmatrix} a \\ -1 \\ 0 \end{bmatrix} \\
=& a + \begin{bmatrix} a & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -a \\ -\beta \end{bmatrix} = a.
\end{aligned}
$$

Since $a$ is assumed to be positive, $(b^\intercal e_1 + b^\intercal K_T b) > 0$ is true.

For $K_{T-1}$ we have:

$$\begin{aligned} K_{T-1} =& A^\intercal K A - \frac{L_{T-1} L_{T-1}^\intercal}{4\alpha_1} \\ =& \begin{pmatrix} 0 & 1 & 0 \\ 0 & a - \frac{a^2}{4\,\alpha_1} & -\frac{\beta\,\rho\,a}{4\,\alpha_1} \\ 0 & \beta + \beta\,\rho - \frac{\beta\,\rho\,a}{4\,\alpha_1} & -\frac{\beta^2\,\rho^2}{4\,\alpha_1} \end{pmatrix} \end{aligned}$$

resulting in $\alpha_{T-2} = (b^\intercal e_1 + b^\intercal K_{T-1} b) = a - \frac{\alpha_{T-1}}{4}$.

Continuing in this fashion we get:

$$\alpha_t = \alpha_{t+1} - \frac{(\frac{2\alpha_{t+1}}{t})^2}{4\alpha_{t+1}}.$$

Since $\alpha_{T-1} = a$ which is assumed to be positive and since at each backward stage a value smaller than itself is deduced from it, it is concluded that $\alpha_t$ is a non-negative value for all $t = 1, ..., T-1$.

□

# B Proof of Lemma 4.2

**Lemma** $\Phi_{t,j}$ and consequently $Q_t$ have the following format:

$$\Phi_{t,j} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2 \\ 0 & d_3 & d_4 \end{bmatrix}, Q_t = \begin{bmatrix} 0 & 1 & 0 \\ 0 & x_1 & x_2 \\ 0 & x_3 & x_4 \end{bmatrix}.$$

*Proof.* We prove this by way of induction:

First of all, $K_T$ has the format $\begin{bmatrix} 0 & 1 & 0 \\ 0 & a & 0 \\ 0 & \beta & 0 \end{bmatrix}$ from the formula $K_T = (A^\intercal e_1) e_{m+1}^\intercal +$ $b^\intercal e_1 (e_{m+1} e_{m+1}^\intercal)$ which corresponds to the general format.

We assume $K_{t+1}$ to be of the format $\begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2 \\ 0 & d_3 & d_4 \end{bmatrix}$.

Since we have $K_t = A^\intercal K_{t+1} A - \frac{L_t L_t^\intercal}{4\alpha_t}$, in which $L_t$ and $\alpha_t$ are:

$$
\begin{aligned}
L_t =& A^\intercal e_1 + A^\intercal K_{t+1} b + b^\intercal K_{t+1} A \\
=& \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \beta & 0 & \rho \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \beta & 0 & \rho \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2 \\ 0 & d_3 & d_4 \end{bmatrix} \begin{bmatrix} a \\ -1 \\ 0 \end{bmatrix} \\
&+ \begin{bmatrix} a & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2 \\ 0 & d_3 & d_4 \end{bmatrix} \begin{bmatrix} 1 & 0 & \beta \\ 0 & 1 & 0 \\ 0 & 0 & \rho \end{bmatrix} \\
=& \begin{bmatrix} 1 \\ 0 \\ \beta \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \beta & 0 & \rho \end{bmatrix} \begin{bmatrix} -1 \\ -d_1 \\ -d_3 \end{bmatrix} + \begin{bmatrix} a & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2\rho \\ 0 & d_3 & d_4\rho \end{bmatrix} \\
=& \begin{bmatrix} 1 \\ 0 \\ \beta \end{bmatrix} + \begin{bmatrix} -1 \\ -d_1 \\ -\beta - \rho d_3 \end{bmatrix} + \begin{bmatrix} 0 \\ a - d_1 \\ -d_2\rho \end{bmatrix} \\
=& \begin{bmatrix} 0 \\ a - 2d_1 \\ -\rho(d_2 + d_3) \end{bmatrix},
\end{aligned}
$$

$$
\begin{aligned}
\alpha_t =& b^\intercal e_1 + b^\intercal K_{t+1} b \\
=& a + \begin{bmatrix} a & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2 \\ 0 & d_3 & d_4 \end{bmatrix} \begin{bmatrix} a \\ -1 \\ 0 \end{bmatrix} \\
=& a + \begin{bmatrix} a & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -d_1 \\ -d_3 \end{bmatrix} = d_1.
\end{aligned}
$$

Thus we have:

$$
\begin{aligned}
K_t =& A^\intercal K_{t+1} A - \frac{L_t L_t^\intercal}{4\alpha_t} \\
=& \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \beta & 0 & \rho \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & d_2 \\ 0 & d_3 & d_4 \end{bmatrix} \begin{bmatrix} 1 & 0 & \beta \\ 0 & 1 & 0 \\ 0 & 0 & \rho \end{bmatrix} \\
& -( \begin{bmatrix} 0 \\ a - 2d_1 \\ -\rho(d_2 + d_3) \end{bmatrix} \begin{bmatrix} 0 & a - 2d_1 & -\rho(d_2 + d_3) \end{bmatrix} )/4d_1 \\
=& \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 & \rho d_2 \\ 0 & \beta + \rho d_3 & \rho^2 d_4 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & \varpi & \varphi \\ 0 & \varphi & \varsigma \end{bmatrix} \\
=& \begin{bmatrix} 0 & 1 & 0 \\ 0 & d_1 - \varpi & \rho d_2 - \varphi \\ 0 & \beta + \rho d_3 - \varphi & \rho^2 d_4 - \varsigma \end{bmatrix}
\end{aligned}
$$

where $\varpi = \frac{(a-2d_1)^2}{4d_1}$, $\varphi = \frac{(a-2d_1)(-\rho(d_2+d_3))}{4d_1}$ and $\varsigma = \frac{\rho^2(d_2+d_3)^2}{4d_1}$.

The end result clearly has the same form as indicated.

□

# C Additional performance indicators for section 4.4.2

The following tables and graphs demonstrate the results shown in relation to the ADP and B&L algorithms for Rolls-Royce and Next plc. share prices.

| # of periods | ADP | B&L | % Diff. |
|---|---|---|---|
| 10 | 113196572 | 113105567 | -0.080 |
| 20 | 113150598 | 113072115 | -0.069 |
| 30 | 113128975 | 113178923 | 0.044 |
| 40 | 113092688 | 113065843 | -0.023 |
| 50 | 113045862 | 113028378 | -0.015 |
| 60 | 113002658 | 113158182 | 0.137 |
| 70 | 112949417 | 113053358 | 0.092 |
| 80 | 112852015 | 113009933 | 0.139 |
| 90 | 112723957 | 113060720 | 0.298 |
| 100 | 112684469 | 112971536 | 0.254 |

**Table C.1:** Actual execution cost realisations from running each algorithm on 100000 Rolls-Royce shares

| # of periods | ADP | B&L | % Diff. |
|---|---|---|---|
| 10 | 721966680 | 721800780 | -0.022 |
| 20 | 721755698 | 721629741 | -0.017 |
| 30 | 721653652 | 721952501 | 0.041 |
| 40 | 721365874 | 721687618 | 0.044 |
| 50 | 721052563 | 721556396 | 0.069 |
| 60 | 720865896 | 721908650 | 0.144 |
| 70 | 720765149 | 721684057 | 0.127 |
| 80 | 719697785 | 721385540 | 0.234 |
| 90 | 719122901 | 721530130 | 0.334 |
| 100 | 718250352 | 721282194 | 0.422 |

**Table C.2:** Actual execution cost realisations from running each algorithm on 100000 Next plc. shares

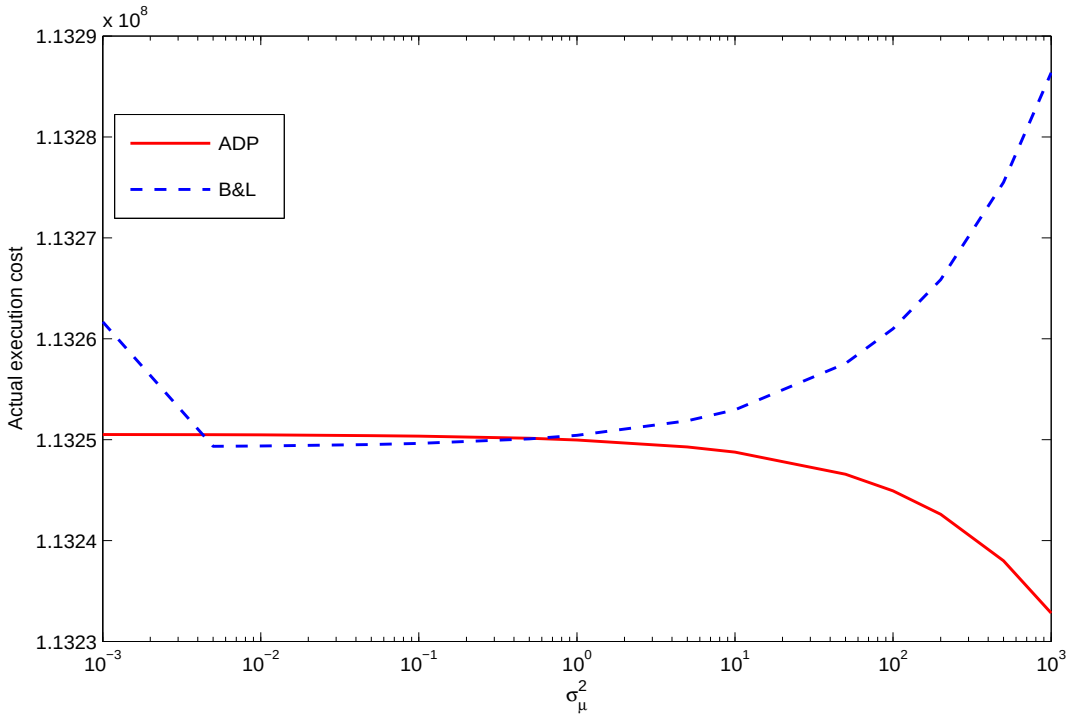


**Figure C.1:** Actual execution cost for Rolls-Royce share price with variable market volatility

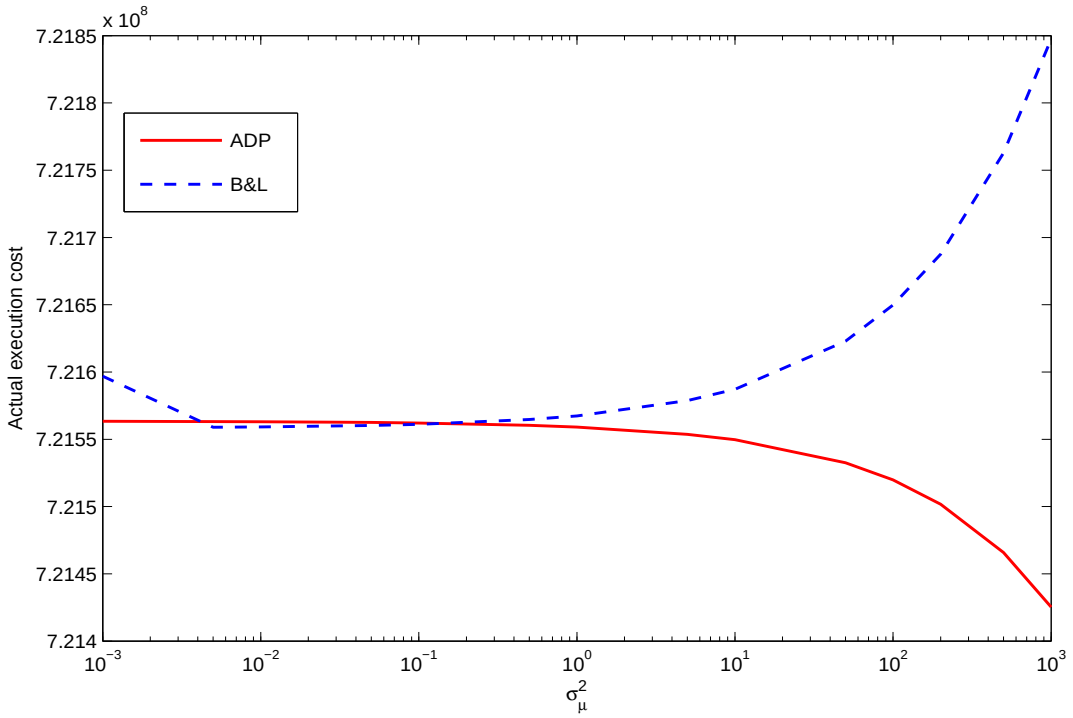


**Figure C.2:** Actual execution cost for Next plc share price with variable market volatility

# Bibliography

Adida, E., Perakis, G., 2007. A nonlinear continuous time optimal control model of dynamic pricing and inventory control with no backorders. Naval Research Logistics 54 (7), 767–795.

Alexander, G., Peterson, M., 2007. An analysis of trade-size clustering and its relation to stealth trading. Journal of Financial Economics 84 (2), 435–471.

Almgren, R., Chriss, N., 2001. Optimal execution of portfolio transactions. Journal of Risk 3, 5–40.

Almgren, R. F., 2003. Optimal execution with nonlinear impact functions and trading-enhanced risk. Applied Mathematical Finance 10 (1), 1–18.

Anderson, C. K., Wilson, J. G., 2003. Wait or buy? the strategic consumer: Pricing and profit implications. The Journal of the Operational Research Society 54 (3), 299–306.

Araman, V. F., Caldentey, R., 2009. Dynamic pricing for nonperishable products with demand learning. Operations Research 57 (5), 1169–1188.

Aviv, Y., Pazgal, A., 2005. A partially observed Markov decision process for dynamic pricing. Management Science 51 (9), 1400–1416.

Bertsekas, D. P., 1995. Dynamic Programming and Optimal Control. Vol. I. Athena Scientific.

Bertsimas, D., Lo, A. W., 1998. Optimal control of execution costs. Journal of Financial Markets 1 (1), 1–50.

Bertsimas, D., Lo, A. W., Hummel, P., 1999. Optimal control of execution costs for portfolios. Computing in Science & Engineering 1 (6), 40–53.

Bertsimas, D., Perakis, G., 2006. Dynamic Pricing: A Learning Approach. Vol. 101 of Applied Optimization. Springer US, book section 3, pp. 45–79.

Besbes, O., Zeevi, A., 2009. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. Operations Research 57 (6), 1407–1420.

Bitran, G., Caldentey, R., Mondschein, S., 1998. Coordinating clearance markdown sales of seasonal products in retail chains. Operations Research 46 (5), 609–624.

Bitran, G., Candentey, R., 2003. An overview of pricing models for revenue management. Manufacturing & Service Operations Management 5 (3), 203–229.

Bitran, G. R., Mondschein, S. V., 1997. Periodic pricing of seasonal products in retailing. Management Science 43 (1), 64–79.

Broder, J., Rusmevichientong, P., 2012. Dynamic pricing under a general parametric choice model. Operations Research 60 (4), 965–980.

Burnetas, A. N., Katehakis, M. N., 1997. Optimal adaptive policies for markov decision processes. Mathematics of Operations Research 22 (1), 222–255.

Butenko, S., Golodnikov, A., Uryasev, S., 2005. Optimal security liquidation algorithms. Computational Optimization and Applications 32 (1), 9–27.

Cassandra, A., Littman, M., Zhang, N., 1997. Incremental pruning: A simple, fast, exact algorithm for partially observable Markov decision processes. In: Thirteenth Annual Conference on Uncertainty in Artificial Intelligence. pp. 54–61.

Cassandra, A. R., 1998. Exact and approximate algorithms for partially observable Markov decision problems. Thesis, Brown University.

Chakravarty, S., 2001. Stealth-trading: Which traders' trades move stock prices? Journal of Financial Economics 61 (2), 289–307.

Chan, L. K. C., Lakonishok, J., 1993. Institutional trades and intraday stock price behavior. Journal of Financial Economics 33 (2), 173–199.

Chan, L. M. A., Shen, Z. J. M., Simchi-Levi, D., Swann, J., 2004. Coordination of Pricing and Inventory Decisions: A Survey and Classification. Vol. 74 of International Series in Operations Research & Management Science. Springer US, book section 9, pp. 335–392.

Chatwin, R. E., 2000. Optimal dynamic pricing of perishable products with stochastic demand and a finite set of prices. European Journal of Operational Research 125 (1), 149–174.

Chen, M., Chen, Z.-L., 2015. Recent developments in dynamic pricing research: Multiple products, competition, and limited demand information. Production and Operations Management 24 (5), 704–731.

Domowitz, I., Yegerman, H., 2005. The cost of algorithmic trading. Trading 2005 (1), 30–40.

Duffy, M., 2003. Advertising and food, drink and tobacco consumption in the united kingdom: a dynamic demand system. Agricultural Economics 28 (1), 51–70.

Eagle, J. N., 1984. The optimal search for a moving target when the search path is constrained. Operations Research 32 (5), 1107–1115.

Elmaghraby, W., Keskinocak, P., 2003. Dynamic pricing in the presence of inventory considerations: Research overview, current practices, and future directions. Management Science 49 (10), 1287–1309.

Engle, R., Ferstenberg, R., 2006. Execution risk. Report, National Bureau of Economic Research.

Fan, Y. Y., Bhargava, H. K., Natsuyama, H. H., 2005. Dynamic pricing via dynamic programming. Journal of Optimization Theory and Applications 127 (3), 565–577.

Farias, V. F., Van Roy, B., 2010. Dynamic pricing with a prior on market response. Operations Research 58 (1), 16–29.

Feng, Y. Y., Gallego, C., 2000. Perishable asset revenue management with Markovian time dependent demand intensities. Management Science 46 (7), 941–956.

Feng, Y. Y., Gallego, G., 1995. Optimal starting times for end-of-season sales and optimal stopping times for promotional fares. Management Science 41 (8), 1371–1391.

Feng, Z., Zilberstein, S., 2004. Region-based incremental pruning for POMDPs. In: Proceedings of the 20th conference on Uncertainty in artificial intelligence. pp. 146–153.

Gallego, G., Van Ryzin, G., 1994. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. Management Science 40 (8), 999–1020.

Gosavi, A., 2009. Reinforcement learning: A tutorial survey and recent advances. Informs Journal on Computing 21 (2), 178–192.

Hasbrouck, J., Seppi, D., 2001. Common factors in prices, order flows, and liquidity. Journal of Financial Economics 59 (3), 383–411.

Hauskrecht, M., 2000. Value-function approximations for partially observable Markov decision processes. Journal of Artificial Intelligence Research 13, 33–94.

He, H., Mamaysky, H., 2005. Dynamic trading policies with price impact. Journal of Economic Dynamics and Control 29 (5), 891–930.

Huberman, G., Stanzl, W., 2005. Optimal liquidity trading. Review of Finance 9 (2), 165–200.

Kaelbling, L. P., Littman, M. L., Cassandra, A. R., 1998. Planning and acting in partially observable stochastic domains. Artificial Intelligence 101 (1-2), 99–134.

Kissell, R., Glantz, M., Malamut, R., 2004. A practical framework for estimating transaction costs and developing optimal trading strategies to achieve best execution. Finance Research Letters 1 (1), 35–46.

Kissell, R., Malamut, R., 2006. Algorithmic decision-making framework. The Journal of Trading 1 (1), 12–21.

Levin, Y., McGill, J., Nediak, M., 2009. Dynamic pricing in the presence of strategic consumers and oligopolistic competition. Management Science 55 (1), 32–46.

Lin, Z. Z., Bean, J. C., White, C. C., 2004. A hybrid genetic/optimization algorithm for finite-horizon, partially observed Markov decision processes. Informs Journal on Computing 16 (1), 27–38.

Liu, Q., Zhang, D., 2013. Dynamic pricing competition with strategic customers under vertical product differentiation. Management Science 59 (1), 84–101.

Loeb, T. F., 1983. Trading cost: The critical link between investment information and results. Financial Analysts Journal 39 (3), 39–44.

Lovejoy, W. S., 1991a. Computationally feasible bounds for partially observed Markov decision processes. Operations Research 39 (1), 162–175.

Lovejoy, W. S., 1991b. A survey of algorithmic methods for partially observed Markov decision processes. Annals of Operations Research 28 (1-4), 47–66.

McGill, J. I., Van Ryzin, G. J., 1999. Revenue management: Research overview and prospects. Transportation Science 33 (2), 233–256.

Merton, R., 1971. Optimum consumption and portfolio rules in a continuous-time model. Journal of Economic Theory 3 (4), 373–413.

Monahan, G. E., 1982. A survey of partially observable Markov decision processes: Theory, models, and algorithms. Management Science 28 (1), 1–16.

Mukherjee, S., Seth, K., 1991. A corrected and improved computational scheme for finite-horizon partially observable Markov decision-processes. INFOR 29 (3), 206–212.

Netessine, S., 2006. Dynamic pricing of inventory/capacity with infrequent price changes. European Journal of Operational Research 174 (1), 553–580.

Perakis, G., Sood, A., 2006. Competitive multi-period pricing for perishable products: A robust optimization approach. Mathematical Programming 107 (1-2), 295–335.

Perold, A. F., 1988. The implementation shortfall: Paper versus reality. The Journal of Portfolio Management 14 (3), 4–9.

Petruzzi, N. C., Dada, M., 1999. Pricing and the newsvendor problem: A review with extensions. Operations Research 47 (2), 183–194.

Popescu, I., Wu, Y. Z., 2007. Dynamic pricing strategies with reference effects. Operations Research 55 (3), 413–429.

Powell, W., 2011. Approximate Dynamic Programming: Solving the Curses of Dimensionality, 2nd Edition (Wiley Series in Probability and Statistics). Wiley.

Rao, V. R., 1984. Pricing research in marketing: The state of the art. The Journal of Business 57 (1), S39–S60.

Rao, V. R., 2009. Handbook of pricing research in marketing. Edward Elgar Publishing.

Sethi, S. P., Cheng, F., 1997. Optimality of (s, s) policies in inventory models with Markovian demand. Operations Research 45 (6), 931–939.

Smallwood, R., Sondik, E. J., 1973. Optimal control of partially observable Markov processes over a finite horizon. Operations Research 21 (5), 1071–1088.

Smith, S. A., Achabal, D. D., 1998. Clearance pricing and inventory policies for retail chains. Management Science 44 (3), 285–300.

Sondik, E. J., 1971. The optimal control of partially observable Markov processes. Thesis, Stanford University.

Song, J. S., Zipkin, P., 1993. Inventory control in a fluctuating demand environment. Operations Research 41 (2), 351–370.

Soon, W., 2011. A review of multi-product pricing models. Applied Mathematics and Computation 217 (21), 8149–8165.

Spaan, M. T. J., Vlassis, N., 2005. Perseus: Randomized point-based value iteration for POMDPs. Journal of Artificial Intelligence Research 24, 195–220.

Startz, R., 1989. The stochastic behavior of durable and nondurable consumption. The Review of Economics and Statistics 71 (2), 356–363.

Stole, L. A., 2007. Chapter 34 Price Discrimination and Competition. Vol. 3. Elsevier, pp. 2221–2299.

Su, X. M., 2007. Intertemporal pricing with strategic customer behavior. Management Science 53 (5), 726–741.

Subramanian, S., Sherali, H. D., 2010. A fractional programming approach for retail category price optimization. Journal of Global Optimization 48 (2), 263–277.

White, C., 1991. A survey of solution techniques for the partially observed Markov decision process. Annals of Operations Research 32 (1), 215–230.

White, C. C., Scherer, W. T., 1989. Solution procedures for partially observed Markov decision-processes. Operations Research 37 (5), 791–797.

Wilson, J. G., Anderson, C. K., Kim, S.-W., 2006. Optimal booking limits in the presence of strategic consumer behavior. International Transactions in Operational Research 13 (2), 99–110.

Xu, X., Hopp, W. J., 2006. A monopolistic and oligopolistic stochastic flow revenue management model. Operations Research 54 (6), 1098–1109.

Zhang, N. L., Liu, W. J., 1997. A model approximation scheme for planning in partially observable stochastic domains. Journal of Artificial Intelligence Research 7, 199–230.

Zhao, W., Zheng, Y. S., 2000. Optimal dynamic pricing for perishable assets with nonhomogeneous demand. Management Science 46 (3), 375–388.

Zhu, K., Thoneman, U. W., 2009. Coordination of pricing and inventory control across products. Naval Research Logistics 56 (2), 175–190.