Dynamic Programming and Optimal Control

Volume I

THIRD EDITION

Dimitri P. Bertsekas

Massachusetts Institute of Technology

WWW site for book information and orders

http://www.athenasc.com

Athena Scientific, Belmont, Massachusetts

Athena Scientific
Post Office Box 805
Nashua, NH 03061-0805
U.S.A.

Email: info@athenasc.com
WWW: http://www.athenasc.com

Cover Design: Ann Gallager, www.gallagerdesign.com

Publisher's Cataloging-in-Publication Data

Bertsekas, Dimitri P.
Dynamic Programming and Optimal Control
Includes Bibliography and Index
1. Mathematical Optimization. 2. Dynamic Programming. I. Title.
QA402.5 .B465 2005 519.703 00-91281

ISBN 1-886529-26-4

ABOUT THE AUTHOR

Dimitri Bertsekas studied Mechanical and Electrical Engineering at the National Technical University of Athens, Greece, and obtained his Ph.D. in system science from the Massachusetts Institute of Technology. He has held faculty positions with the Engineering-Economic Systems Dept., Stanford University, and the Electrical Engineering Dept. of the University of Illinois, Urbana. Since 1979 he has been teaching at the Electrical Engineering and Computer Science Department of the Massachusetts Institute of Technology (M.I.T.), where he is currently McAfee Professor of Engineering.

His research spans several fields, including optimization, control, large-scale computation, and data communication networks, and is closely tied to his teaching and book authoring activities. He has written numerous research papers, and thirteen books, several of which are used as textbooks in MIT classes. He consults regularly with private industry and has held editorial positions in several journals.

Professor Bertsekas was awarded the INFORMS 1997 Prize for Research Excellence in the Interface Between Operations Research and Computer Science for his book "Neuro-Dynamic Programming" (co-authored with John Tsitsiklis), the 2000 Greek National Award for Operations Research, and the 2001 ACC John R. Ragazzini Education Award. In 2001, he was elected to the United States National Academy of Engineering.

ATHENA SCIENTIFIC

OPTIMIZATION AND COMPUTATION SERIES

1. Convex Analysis and Optimization, by Dimitri P. Bertsekas, with Angelia Nedić and Asuman E. Ozdaglar, 2003, ISBN 1-886529-45-0, 560 pages
2. Introduction to Probability, by Dimitri P. Bertsekas and John N. Tsitsiklis, 2002, ISBN 1-886529-40-X, 430 pages
3. Dynamic Programming and Optimal Control, Two-Volume Set, by Dimitri P. Bertsekas, 2005, ISBN 1-886529-08-6, 840 pages
4. Nonlinear Programming, 2nd Edition, by Dimitri P. Bertsekas, 1999, ISBN 1-886529-00-0, 791 pages
5. Network Optimization: Continuous and Discrete Models, by Dimitri P. Bertsekas, 1998, ISBN 1-886529-02-7, 608 pages
6. Network Flows and Monotropic Optimization, by R. Tyrrell Rockafellar, 1998, ISBN 1-886529-06-X, 634 pages
7. Introduction to Linear Optimization, by Dimitris Bertsimas and John N. Tsitsiklis, 1997, ISBN 1-886529-19-1, 608 pages
8. Parallel and Distributed Computation: Numerical Methods, by Dimitri P. Bertsekas and John N. Tsitsiklis, 1997, ISBN 1-886529-01-9, 718 pages
9. Neuro-Dynamic Programming, by Dimitri P. Bertsekas and John N. Tsitsiklis, 1996, ISBN 1-886529-10-8, 512 pages
10. Constrained Optimization and Lagrange Multiplier Methods, by Dimitri P. Bertsekas, 1996, ISBN 1-886529-04-3, 410 pages
11. Stochastic Optimal Control: The Discrete-Time Case, by Dimitri P. Bertsekas and Steven E. Shreve, 1996, ISBN 1-886529-03-5, 330 pages

Contents

CONTENTS OF VOLUME II

Preface

This two-volume book is based on a first-year graduate course on dynamic programming and optimal control that I have taught for over twenty years at Stanford University, the University of Illinois, and the Massachusetts Institute of Technology. The course has been typically attended by students from engineering, operations research, economics, and applied mathematics. Accordingly, a principal objective of the book has been to provide a unified treatment of the subject, suitable for a broad audience. In particular, problems with a continuous character, such as stochastic control problems, popular in modern control theory, are simultaneously treated with problems with a discrete character, such as Markovian decision problems, popular in operations research. Furthermore, many applications and examples, drawn from a broad variety of fields, are discussed.

The book may be viewed as a greatly expanded and pedagogically improved version of my 1987 book "Dynamic Programming: Deterministic and Stochastic Models," published by Prentice-Hall. I have included much new material on deterministic and stochastic shortest path problems, as well as a new chapter on continuous-time optimal control problems and the Pontryagin Minimum Principle, developed from a dynamic programming viewpoint. I have also added a fairly extensive exposition of simulation-based approximation techniques for dynamic programming. These techniques, which are often referred to as "neuro-dynamic programming" or "reinforcement learning," represent a breakthrough in the practical application of dynamic programming to complex problems that involve the dual curse of large dimension and lack of an accurate mathematical model. Other material was also augmented, substantially modified, and updated.

With the new material, however, the book grew so much in size that it became necessary to divide it into two volumes: one on finite horizon, and the other on infinite horizon problems. This division was not only natural in terms of size, but also in terms of style and orientation. The first volume is more oriented towards modeling, and the second is more oriented towards mathematical analysis and computation. I have included in the first volume a final chapter that provides an introductory treatment of infinite horizon problems. The purpose is to make the first volume self-

contained for instructors who wish to cover a modest amount of infinite horizon material in a course that is primarily oriented towards modeling, conceptualization, and finite horizon problems,

Many topics in the book are relatively independent of the others. For example Chapter 2 of Vol. I on shortest path problems can be skipped without loss of continuity, and the same is true for Chapter 3 of Vol. I, which deals with continuous-time optimal control. As a result, the book can be used to teach several different types of courses.

(a) A two-semester course that covers both volumes.

(b) A one-semester course primarily focused on finite horizon problems that covers most of the first volume.

(c) A one-semester course focused on stochastic optimal control that covers Chapters 1, 4, 5, and 6 of Vol. I, and Chapters 1, 2, and 4 of Vol. II.

(d) A one-semester course that covers Chapter 1, about 50% of Chapters 2 through 6 of Vol. I, and about 70% of Chapters 1, 2, and 4 of Vol. II. This is the course I usually teach at MIT.

(e) A one-quarter engineering course that covers the first three chapters and parts of Chapters 4 through 6 of Vol. I.

(f) A one-quarter mathematically oriented course focused on infinite horizon problems that covers Vol. II.

The mathematical prerequisite for the text is knowledge of advanced calculus, introductory probability theory, and matrix-vector algebra. A summary of this material is provided in the appendixes. Naturally, prior exposure to dynamic system theory, control, optimization, or operations research will be helpful to the reader, but based on my experience, the material given here is reasonably self-contained.

The book contains a large number of exercises, and the serious reader will benefit greatly by going through them. Solutions to all exercises are compiled in a manual that is available to instructors from the author. Many thanks are due to the several people who spent long hours contributing to this manual, particularly Steven Shreve, Eric Loiederman, Lakis Polymenakos, and Cynara Wu.

Dynamic programming is a conceptually simple technique that can be adequately explained using elementary analysis. Yet a mathematically rigorous treatment of general dynamic programming requires the complicated machinery of measure-theoretic probability. My choice has been to bypass the complicated mathematics by developing the subject in generality, while claiming rigor only when the underlying probability spaces are countable. A mathematically rigorous treatment of the subject is carried out in my monograph "Stochastic Optimal Control: The Discrete Time

Case," Academic Press, 1978,† coauthored by Steven Shreve. This monograph complements the present text and provides a solid foundation for the subjects developed somewhat informally here.

Finally, I am thankful to a number of individuals and institutions for their contributions to the book. My understanding of the subject was sharpened while I worked with Steven Shreve on our 1978 monograph. My interaction and collaboration with John Tsitsiklis on stochastic shortest paths and approximate dynamic programming have been most valuable. Michael Caramanis, Emmanuel Fernandez-Gaucherand, Pierre Humblet, Lennart Ljung, and John Tsitsiklis taught from versions of the book, and contributed several substantive comments and homework problems. A number of colleagues offered valuable insights and information, particularly David Castanon, Eugene Feinberg, and Krishna Pattipati. NSF provided research support. Prentice-Hall graciously allowed the use of material from my 1987 book. Teaching and interacting with the students at MIT have kept up my interest and excitement for the subject.

Dimitri P. Bertsekas
Spring, 1995

† Note added in the 3rd edition: This monograph was republished by Athena Scientific in 1996, and can also be freely downloaded from the author's www site: http://web.mit.edu/dimitrib/www/home.html.

Preface to the Second Edition

This second edition has expanded by nearly 30% the coverage of the original. Most of the new material is concentrated in four areas:

(a) In Chapter 4, a section was added on estimation and control of systems with a non-probabilistic (set membership) description of uncertainty. This subject, a personal favorite of the author since it was the subject of his 1971 Ph.D. thesis, has become popular, as minimax and H_∞ control methods have gained increased prominence.

(b) Chapter 6 was doubled in size, to reflect the popularity of suboptimal control and neuro-dynamic programming methods. In particular, the coverage of certainty equivalent, and limited lookahead methods has been substantially expanded. Furthermore, a new section was added on neuro-dynamic programming and rollout algorithms, and their applications in combinatorial optimization and stochastic optimal control.

(c) In Chapter 7, an introduction to continuous-time, semi-Markov decision problems was added in a separate last section.

(d) A new appendix was included, which deals with various formulations of problems of decision under uncertainty. The foundations of the minimax and expected utility approaches are framed within a broader context, and some of the aspects of utility theory are discussed.

There are also miscellaneous additions and improvements scattered throughout the text, and a more detailed coverage of deterministic problems is given in Chapter 1. Finally, a new internet-based feature was added to the book, which extends its scope and coverage. Many of the theoretical exercises have been solved in detail and their solutions have been posted in the book's www page

http://www.athenasc.com/dpbook.html

These exercises have been marked with the symbol WWW

I would like to express my thanks to the many colleagues who contributed suggestions for improvement of the second edition.

Dimitri P. Bertsekas
Fall, 2000

Preface to the Third Edition

The third edition contains a substantial amount of new material, particularly on approximate dynamic programming, which has now become one of the principal focal points of the book. In particular:

(a) The subject of minimax control was developed in greater detail, including a new section in Chapter 1, which connects with new material in Chapter 6.

(b) The section on auction algorithms for shortest paths in Chapter 2 was eliminated. These methods are not currently used in dynamic programming, and a detailed discussion has been provided in a chapter from the author's Network Optimization book. This chapter can be freely downloaded from

http://web.mit.edu/dimitrib/www/net.html

(c) A section was added in Chapter 2 on dynamic programming and shortest path algorithms for constrained and multiobjective problems.

(d) The material on sufficient statistics and partially observable Markov decision problems in Section 5.4 was restructured and expanded.

(e) Considerable new material was added in Chapter 6:

(1) An expanded discussion of one-step lookahead policies and associated performance bounds in Section 6.3.1.

(2) A discussion of aggregation methods and discretization of continuous-state problems (see Subsection 6.3.4).

(3) A discussion of model predictive control and its relation to other suboptimal control methods (see Subsection 6.5.2).

(4) An expanded treatment of open-loop feedback control and related methods based on a restricted structure (see Subsection 6.5.3).

I have also added a few exercises, and revised a few sections while preserving their essential content. Thanks are due to Haixia Lin, who worked out several exercises, and to Janey Yu, who reviewed some of the new sections and gave me valuable feedback.

Dimitri P. Bertsekas
http://web.mit.edu/dimitrib/www/home.html
Summer 2005

1

The Dynamic Programming Algorithm

Contents

Life can only be understood going backwards,
but it must be lived going forwards.
Kierkegaard

1.1 INTRODUCTION

This book deals with situations where decisions are made in stages. The outcome of each decision may not be fully predictable but can be anticipated to some extent before the next decision is made. The objective is to minimize a certain cost – a mathematical expression of what is considered an undesirable outcome.

A key aspect of such situations is that decisions cannot be viewed in isolation since one must balance the desire for low present cost with the undesirability of high future costs. The dynamic programming technique captures this tradeoff. At each stage, it ranks decisions based on the sum of the present cost and the expected future cost, assuming optimal decision making for subsequent stages.

There is a very broad variety of practical problems that can be treated by dynamic programming. In this book, we try to keep the main ideas uncluttered by irrelevant assumptions on problem structure. To this end, we formulate in this section a broadly applicable model of optimal control of a dynamic system over a finite number of stages (a finite horizon). This model will occupy us for the first six chapters; its infinite horizon version will be the subject of the last chapter as well as Vol. II.

Our basic model has two principal features: (1) an underlying *discrete-time dynamic system*, and (2) a *cost function that is additive over time*. The dynamic system expresses the evolution of some variables, the system's "state", under the influence of decisions made at discrete instances of time. The system has the form

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

where

k indexes discrete time,

x_k is the state of the system and summarizes past information that is relevant for future optimization,

u_k is the control or decision variable to be selected at time k,

w_k is a random parameter (also called disturbance or noise depending on the context),

N is the horizon or number of times control is applied,

and f_k is a function that describes the system and in particular the mechanism by which the state is updated.

The cost function is additive in the sense that the cost incurred at time k, denoted by $g_k(x_k, u_k, w_k)$, accumulates over time. The total cost is

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

where $g_N(x_N)$ is a terminal cost incurred at the end of the process. However, because of the presence of w_k, the cost is generally a random variable and cannot be meaningfully optimized. We therefore formulate the problem as an optimization of the *expected cost*

$$E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\},$$

where the expectation is with respect to the joint distribution of the random variables involved. The optimization is over the controls $u_0, u_1, \ldots, u_{N-1}$, but some qualification is needed here; each control u_k is selected with some knowledge of the current state x_k (either its exact value or some other related information).

A more precise definition of the terminology just used will be given shortly. We first provide some orientation by means of examples.

Example 1.1.1 (Inventory Control)

Consider a problem of ordering a quantity of a certain item at each of N periods so as to (roughly) meet a stochastic demand, while minimizing the incurred expected cost. Let us denote

x_k stock available at the beginning of the kth period,

u_k stock ordered (and immediately delivered) at the beginning of the kth period,

w_k demand during the kth period with given probability distribution.

We assume that $w_0, w_1, \ldots, w_{N-1}$ are independent random variables, and that excess demand is backlogged and filled as soon as additional inventory becomes available. Thus, stock evolves according to the discrete-time equation

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

where negative stock corresponds to backlogged demand (see Fig. 1.1.1).

The cost incurred in period k consists of two components:

(a) A cost $r(x_k)$ representing a penalty for either positive stock x_k (holding cost for excess inventory) or negative stock x_k (shortage cost for unfilled demand).

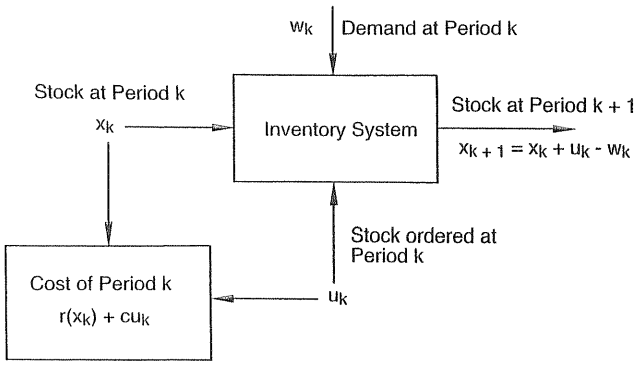


Figure 1.1.1 Inventory control example. At period k, the current stock (state) x_k, the stock ordered (control) u_k, and the demand (random disturbance) w_k determine the cost $r(x_k)+cu_k$ and the stock $x_{k+1} = x_k + u_k - w_k$ at the next period.

(b) The purchasing cost cu_k, where c is cost per unit ordered.

There is also a terminal cost $R(x_N)$ for being left with inventory x_N at the end of N periods. Thus, the total cost over N periods is

$$E\left\{R(x_N) + \sum_{k=0}^{N-1}\big(r(x_k) + cu_k\big)\right\}.$$

We want to minimize this cost by proper choice of the orders $u_0, \ldots, u_{N-1}$, subject to the natural constraint $u_k \geq 0$ for all k.

At this point we need to distinguish between *closed-loop* and *open-loop* minimization of the cost. In open-loop minimization we select all orders $u_0, \ldots, u_{N-1}$ at once at time 0, without waiting to see the subsequent demand levels. In closed-loop minimization we postpone placing the order u_k until the last possible moment (time k) when the current stock x_k will be known. The idea is that since there is no penalty for delaying the order u_k up to time k, we can take advantage of information that becomes available between times 0 and k (the demand and stock level in past periods).

Closed-loop optimization is of central importance in dynamic programming and is the type of optimization that we will consider almost exclusively in this book. Thus, in our basic formulation, decisions are made in stages while gathering information between stages that will be used to enhance the quality of the decisions. The effect of this on the structure of the resulting optimization problem is quite profound. In particular, in closed-loop inventory optimization we are not interested in finding optimal numerical values of the orders but rather we want to find an *optimal rule for selecting at each period k an order u_k for each possible value of stock x_k that can conceivably occur*. This is an "action versus strategy" distinction.

Mathematically, in closed-loop inventory optimization, we want to find a sequence of functions μ_k, $k = 0, \ldots, N-1$, mapping stock x_k into order u_k

so as to minimize the expected cost. The meaning of μ_k is that, for each k and each possible value of x_k,

$\mu_k(x_k) =$ amount that should be ordered at time k if the stock is x_k.

The sequence $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ will be referred to as a *policy* or *control law*. For each π, the corresponding cost for a fixed initial stock x_0 is

$$J_\pi(x_0) = E\left\{R(x_N) + \sum_{k=0}^{N-1}\big(r(x_k) + c\mu_k(x_k)\big)\right\},$$

and we want to minimize $J_\pi(x_0)$ for a given x_0 over all π that satisfy the constraints of the problem. This is a typical dynamic programming problem. We will analyze this problem in various forms in subsequent sections. For example, we will show in Section 4.2 that for a reasonable choice of the cost function, the optimal ordering policy is of the form

$$\mu_k(x_k) = \begin{cases} S_k - x_k & \text{if } x_k < S_k, \\ 0 & \text{otherwise,} \end{cases}$$

where S_k is a suitable threshold level determined by the data of the problem. In other words, when stock falls below the threshold S_k, order just enough to bring stock up to S_k.

The preceding example illustrates the main ingredients of the basic problem formulation:

(a) A *discrete-time system* of the form

$$x_{k+1} = f_k(x_k, u_k, w_k),$$

where f_k is some function; for example in the inventory case, we have $f_k(x_k, u_k, w_k) = x_k + u_k - w_k$.

(b) *Independent random parameters* w_k. This will be generalized by allowing the probability distribution of w_k to depend on x_k and u_k; in the context of the inventory example, we can think of a situation where the level of demand w_k is influenced by the current stock level x_k.

(c) A *control constraint*; in the example, we have $u_k \geq 0$. In general, the constraint set will depend on x_k and the time index k, that is, $u_k \in U_k(x_k)$. To see how constraints dependent on x_k can arise in the inventory context, think of a situation where there is an upper bound B on the level of stock that can be accommodated, so $u_k \leq B - x_k$.

(d) An *additive cost* of the form

$$E\left\{g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)\right\},$$

where g_k are some functions; in the inventory example, we have

$$g_N(x_N) = R(x_N), \qquad g_k(x_k, u_k, w_k) = r(x_k) + cu_k.$$

(e) *Optimization over (closed-loop) policies*, that is, rules for choosing u_k for each k and each possible value of x_k.

Discrete-State and Finite-State Problems

In the preceding example, the state x_k was a continuous real variable, and it is easy to think of multidimensional generalizations where the state is an n-dimensional vector of real variables. It is also possible, however, that the state takes values from a discrete set, such as the integers.

A version of the inventory problem where a discrete viewpoint is more natural arises when stock is measured in whole units (such as cars), each of which is a significant fraction of x_k, u_k, or w_k. It is more appropriate then to take as state space the set of all integers rather than the set of real numbers. The form of the system equation and the cost per period will, of course, stay the same.

Generally, there are many situations where the state is naturally discrete and there is no continuous counterpart of the problem. Such situations are often conveniently specified in terms of the probabilities of transition between the states. What we need to know is $p_{ij}(u, k)$, which is the probability at time k that the next state will be j, given that the current state is i, and the control selected is u, i.e.,

$$p_{ij}(u, k) = P\{x_{k+1} = j \mid x_k = i, u_k = u\}.$$

This type of state transition can alternatively be described in terms of the discrete-time system equation

$$x_{k+1} = w_k,$$

where the probability distribution of the random parameter w_k is

$$P\{w_k = j \mid x_k = i, u_k = u\} = p_{ij}(u, k).$$

Conversely, given a discrete-state system in the form

$$x_{k+1} = f_k(x_k, u_k, w_k),$$

together with the probability distribution $P_k(w_k \mid x_k, u_k)$ of w_k, we can provide an equivalent transition probability description. The corresponding transition probabilities are given by

$$p_{ij}(u, k) = P_k\{W_k(i, u, j) \mid x_k = i, u_k = u\},$$

where $W(i, u, j)$ is the set

$$W_k(i, u, j) = \{w \mid j = f_k(i, u, w)\}.$$

Thus a discrete-state system can equivalently be described in terms of a difference equation or in terms of transition probabilities. Depending on the given problem, it may be notationally or mathematically more convenient to use one description over the other.

The following examples illustrate discrete-state problems. The first example involves a *deterministic* problem, that is, a problem where there is no stochastic uncertainty. In such a problem, when a control is chosen at a given state, the next state is fully determined; that is, for any state i, control u, and time k, the transition probability $p_{ij}(u, k)$ is equal to 1 for a single state j, and it is 0 for all other candidate next states. The other three examples involve stochastic problems, where the next state resulting from a given choice of control at a given state cannot be determined a priori.

Example 1.1.2 (A Deterministic Scheduling Problem)

Suppose that to produce a certain product, four operations must be performed on a certain machine. The operations are denoted by A, B, C, and D. We assume that operation B can be performed only after operation A has been performed, and operation D can be performed only after operation B has been performed. (Thus the sequence CDAB is allowable but the sequence CDBA is not.) The setup cost C_{mn} for passing from any operation m to any other operation n is given. There is also an initial startup cost S_A or S_C for starting with operation A or C, respectively. The cost of a sequence is the sum of the setup costs associated with it; for example, the operation sequence ACDB has cost

$$S_A + C_{AC} + C_{CD} + C_{DB}.$$

We can view this problem as a sequence of three decisions, namely the choice of the first three operations to be performed (the last operation is determined from the preceding three). It is appropriate to consider as state the set of operations already performed, the initial state being an artificial state corresponding to the beginning of the decision process. The possible state transitions corresponding to the possible states and decisions for this problem is shown in Fig. 1.1.2. Here the problem is deterministic, i.e., at a given state, each choice of control leads to a uniquely determined state. For example, at state AC the decision to perform operation D leads to state ACD with certainty, and has cost C_{CD}. Deterministic problems with a finite number of states can be conveniently represented in terms of transition graphs such as the one of Fig. 1.1.2. The optimal solution corresponds to the path that starts at the initial state and ends at some state at the terminal time and has minimum sum of arc costs plus the terminal cost. We will study systematically problems of this type in Chapter 2.

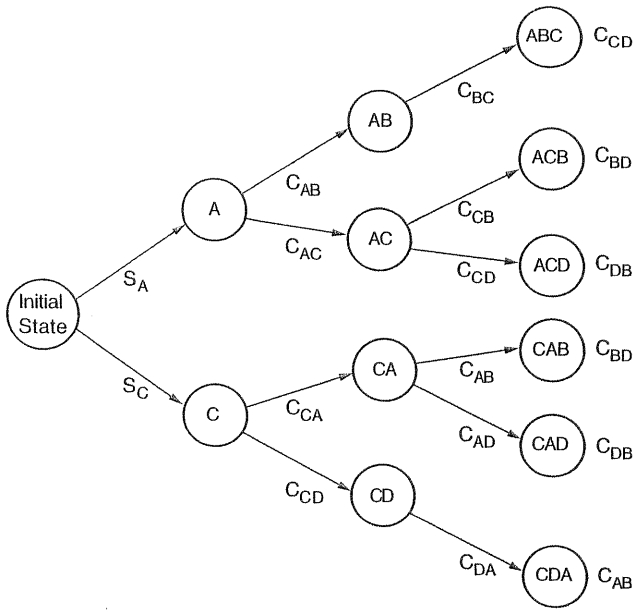


Figure 1.1.2 The transition graph of the deterministic scheduling problem of Example 1.1.2. Each arc of the graph corresponds to a decision leading from some state (the start node of the arc) to some other state (the end node of the arc). The corresponding cost is shown next to the arc. The cost of the last operation is shown as a terminal cost next to the terminal nodes of the graph.

Example 1.1.3 (Machine Replacement)

Consider a problem of operating efficiently over N time periods a machine that can be in any one of n states, denoted $1, 2, \ldots, n$. We denote by $g(i)$ the operating cost per period when the machine is in state i, and we assume that

$$g(1) \leq g(2) \leq \cdots \leq g(n).$$

The implication here is that state i is better than state $i+1$, and state 1 corresponds to a machine in best condition.

During a period of operation, the state of the machine can become worse or it may stay unchanged. We thus assume that the transition probabilities

$$p_{ij} = P\{\text{next state will be } j \mid \text{current state is } i\}$$

satisfy

$$p_{ij} = 0 \qquad \text{if } j < i.$$

We assume that at the start of each period we know the state of the machine and we must choose one of the following two options:

(a) Let the machine operate one more period in the state it currently is.

(b) Repair the machine and bring it to the best state 1 at a cost R.

We assume that the machine, once repaired, is guaranteed to stay in state 1 for one period. In subsequent periods, it may deteriorate to states $j > 1$ according to the transition probabilities p_{1j}.

Thus the objective here is to decide on the level of deterioration (state) at which it is worth paying the cost of machine repair, thereby obtaining the benefit of smaller future operating costs. Note that the decision should also be affected by the period we are in. For example, we would be less inclined to repair the machine when there are few periods left.

The system evolution for this problem can be described by the graphs of Fig. 1.1.3. These graphs depict the transition probabilities between various pairs of states for each value of the control and are known as *transition probability graphs* or simply *transition graphs*. Note that there is a different graph for each control; in the present case there are two controls (repair or not repair).

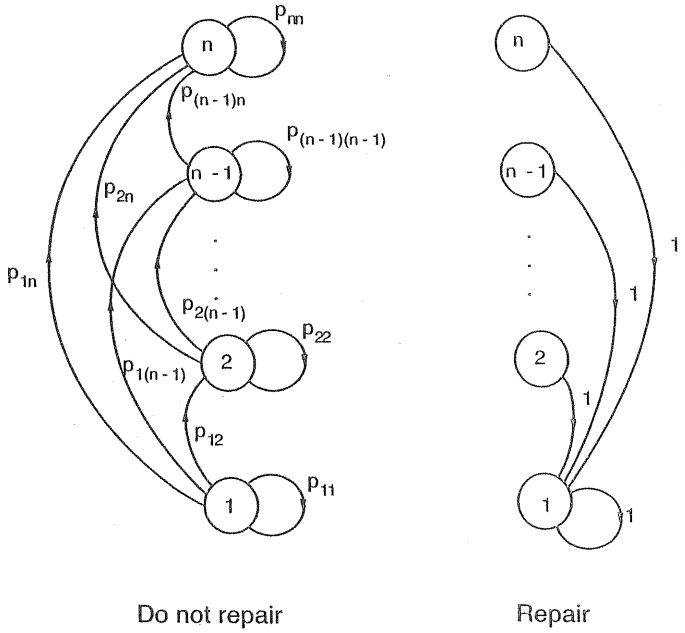


Figure 1.1.3 Machine replacement example. Transition probability graphs for each of the two possible controls (repair or not repair). At each stage and state i, the cost of repairing is $R+g(1)$, and the cost of not repairing is $g(i)$. The terminal cost is 0.

Example 1.1.4 (Control of a Queue)

Consider a queueing system with room for n customers operating over N time periods. We assume that service of a customer can start (end) only at the beginning (end) of the period and that the system can serve only one customer at a time. The probability p_m of m customer arrivals during a period is given, and the numbers of arrivals in two different periods are independent. Customers finding the system full depart without attempting to enter later. The system offers two kinds of service, *fast* and *slow*, with cost per period c_f and c_s, respectively. Service can be switched between fast and slow at the beginning of each period. With fast (slow) service, a customer in service at the beginning of a period will terminate service at the end of the period with probability q_f (respectively, q_s) independently of the number of periods the customer has been in service and the number of customers in the system ($q_f > q_s$). There is a cost $r(i)$ for each period for which there are i customers in the system. There is also a terminal cost $R(i)$ for i customers left in the system at the end of the last period.

The problem is to choose, at each period, the type of service as a function of the number of customers in the system so as to minimize the expected total cost over N periods. One expects that when there is a large number of customers i in queue, it is better to use the fast service, and the question is to find the values of i for which this is true.

Here it is appropriate to take as state the number i of customers in the system at the start of a period and as control the type of service provided. Then, the cost per period is $r(i)$ plus c_f or c_s depending on whether fast or slow service is provided. We derive the transition probabilities of the system.

When the system is empty at the start of the period, the probability that the next state is j is independent of the type of service provided. It equals the given probability of j customer arrivals when $j < n$,

$$p_{0j}(u_f) = p_{0j}(u_s) = p_j, \qquad j = 0, 1, \ldots, n-1,$$

and it equals the probability of n or more customer arrivals when $j = n$,

$$p_{0n}(u_f) = p_{0n}(u_s) = \sum_{m=n}^{\infty} p_m.$$

When there is at least one customer in the system ($i > 0$), we have

$$p_{ij}(u_f) = 0, \qquad \text{if } j < i-1,$$

$$p_{ij}(u_f) = q_f p_0, \qquad \text{if } j = i-1,$$

$$\begin{aligned} p_{ij}(u_f) &= P\{j-i+1 \text{ arrivals, service completed}\} \\ &\quad + P\{j-i \text{ arrivals, service not completed}\} \\ &= q_f p_{j-i+1} + (1-q_f)p_{j-i}, \qquad \text{if } i-1 < j < n-1, \end{aligned}$$

$$p_{i(n-1)}(u_f) = q_f \sum_{m=n-i}^{\infty} p_m + (1-q_f)p_{n-1-i},$$

$$p_{in}(u_f) = (1-q_f) \sum_{m=n-i}^{\infty} p_m.$$

The transition probabilities when slow service is provided are also given by these formulas with u_f and q_f replaced by u_s and q_s, respectively.

Example 1.1.5 (Optimizing a Chess Match Strategy)

A player is about to play a two-game chess match with an opponent, and wants to maximize his winning chances. Each game can have one of two outcomes:

(a) A win by one of the players (1 point for the winner and 0 for the loser).

(b) A draw (1/2 point for each of the two players).

If the score is tied at 1-1 at the end of the two games, the match goes into sudden-death mode, whereby the players continue to play until the first time one of them wins a game (and the match). The player has two playing styles and he can choose one of the two at will in each game, independently of the style he chose in previous games.

(1) *Timid play* with which he draws with probability $p_d > 0$, and he loses with probability $(1 - p_d)$.

(2) *Bold play* with which he wins with probability p_w, and he loses with probability $(1 - p_w)$.

Thus, in a given game, timid play never wins, while bold play never draws. The player wants to find a style selection strategy that maximizes his probability of winning the match. Note that once the match gets into sudden death, the player should play bold, since with timid play he can at best prolong the sudden death play, while running the risk of losing. Therefore, there are only two decisions for the player to make, the selection of the playing strategy in the first two games. Thus, we can model the problem as one with two stages, and with states the possible scores at the start of each of the first two stages (games), as shown in Fig. 1.1.4. The initial state is the initial score 0-0. The transition probabilities for each of the two different controls (playing styles) are also shown in Fig. 1.1.4. There is a cost at the terminal states: a cost of -1 at the winning scores 2-0 and 1.5-0.5, a cost of 0 at the losing scores 0-2 and 0.5-1.5, and a cost of $-p_w$ at the tied score 1-1 (since the probability of winning in sudden death is p_w). Note that to maximize the probability P of winning the match, we must minimize $-P$.

This problem has an interesting feature. One would think that if $p_w < 1/2$, the player would have a less than 50-50 chance of winning the match, even with optimal play, since his probability of losing is greater than his probability of winning any one game, regardless of his playing style. This is not so, however, because the player can adapt his playing style to the current score, but his opponent does not have that option. In other words, the player can use a closed-loop strategy, and it will be seen later that with optimal play, as determined by the dynamic programming algorithm, he has a better than

50-50 chance of winning the match provided p_w is higher than a threshold value $\overline{p}$, which, depending on the value of p_d, may satisfy $\overline{p} < 1/2$.

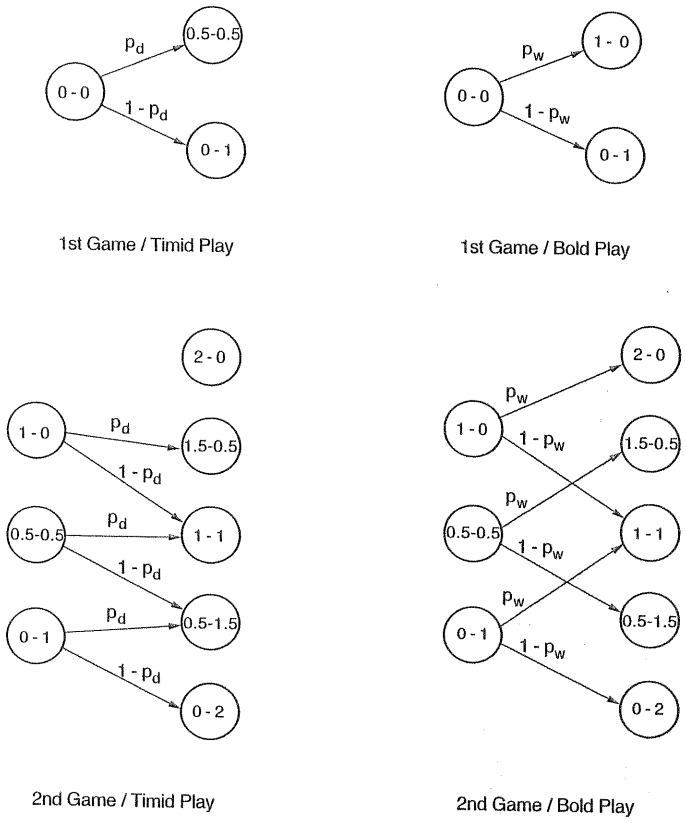


Figure 1.1.4 Chess match example. Transition probability graphs for each of the two possible controls (timid or bold play). Note here that the state space is not the same at each stage. The terminal cost is -1 at the winning final scores 2-0 and 1.5-0.5, 0 at the losing final scores 0-2 and 0.5-1.5, and $-p_w$ at the tied score 1-1.

1.2 THE BASIC PROBLEM

We now formulate a general problem of decision under stochastic uncertainty over a finite number of stages. This problem, which we call *basic*, is central in this book. We will discuss solution methods for this problem

based on dynamic programming in the first six chapters, and we will extend our analysis to versions of this problem involving an infinite number of stages in the last chapter and in Vol. II of this work.

The basic problem is very general. In particular, we will not require that the state, control, or random parameter take a finite number of values or belong to a space of n-dimensional vectors. A surprising aspect of dynamic programming is that its applicability depends very little on the nature of the state, control, and random parameter spaces. For this reason it is convenient to proceed without any assumptions on the structure of these spaces; indeed such assumptions would become a serious impediment later.

Basic Problem

We are given a discrete-time dynamic system

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

where the state x_k is an element of a space S_k, the control u_k is an element of a space C_k, and the random "disturbance" w_k is an element of a space D_k.

The control u_k is constrained to take values in a given nonempty subset $U(x_k) \subset C_k$, which depends on the current state x_k; that is, $u_k \in U_k(x_k)$ for all $x_k \in S_k$ and k.

The random disturbance w_k is characterized by a probability distribution $P_k(\cdot \mid x_k, u_k)$ that may depend explicitly on x_k and u_k but not on values of prior disturbances $w_{k-1}, \ldots, w_0$.

We consider the class of policies (also called control laws) that consist of a sequence of functions

$$\pi = \{\mu_0, \ldots, \mu_{N-1}\},$$

where μ_k maps states x_k into controls $u_k = \mu_k(x_k)$ and is such that $\mu_k(x_k) \in U_k(x_k)$ for all $x_k \in S_k$. Such policies will be called *admissible*.

Given an initial state x_0 and an admissible policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, the states x_k and disturbances w_k are random variables with distributions defined through the system equation

$$x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big), \qquad k = 0, 1, \ldots, N-1. \tag{1.1}$$

Thus, for given functions g_k, $k = 0, 1, \ldots, N$, the expected cost of π starting at x_0 is

$$J_\pi(x_0) = E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big) \right\}$$

where the expectation is taken over the random variables w_k and x_k. An optimal policy π^* is one that minimizes this cost; that is,

$$J_{\pi^*}(x_0) = \min_{\pi \in \Pi} J_\pi(x_0),$$

where Π is the set of all admissible policies.

Note that the optimal policy π^* is associated with a fixed initial state x_0. However, an interesting aspect of the basic problem and of dynamic programming is that it is typically possible to find a policy π^* that is simultaneously optimal for all initial states.

The optimal cost depends on x_0 and is denoted by $J^*(x_0)$; that is,

$$J^*(x_0) = \min_{\pi \in \Pi} J_\pi(x_0).$$

It is useful to view J^* as a function that assigns to each initial state x_0 the optimal cost $J^*(x_0)$ and call it the *optimal cost function* or *optimal value function*.†

The Role and Value of Information

We noted earlier the distinction between open-loop minimization, where we select all controls $u_0, \ldots, u_{N-1}$ at once at time 0, and closed-loop minimization, where we select a policy $\{\mu_0, \ldots, \mu_{N-1}\}$ that applies the control $\mu_k(x_k)$ at time k with knowledge of the current state x_k (see Fig. 1.2.1). With closed-loop policies, it is possible to achieve lower cost, essentially by taking advantage of the extra information (the value of the current state). The reduction in cost may be called the *value of the information* and can be significant indeed. If the information is not available, the controller cannot adapt appropriately to unexpected values of the state, and as a result the cost can be adversely affected. For example, in the inventory control example of the preceding section, the information that becomes available at the beginning of each period k is the inventory stock x_k. Clearly, this information is very important to the inventory manager, who will want to adjust the amount u_k to be purchased depending on whether the current stock x_k is running high or low.

† For the benefit of the mathematically oriented reader we note that in the preceding equation, "min" denotes the greatest lower bound (or infimum) of the set of numbers $\{J_\pi(x_0) \mid \pi \in \Pi\}$. A notation more in line with normal mathematical usage would be to write $J^*(x_0) = \inf_{\pi \in \Pi} J_\pi(x_0)$. However (as discussed in Appendix B), we find it convenient to use "min" in place of "inf" even when the infimum is not attained. It is less distracting, and it will not lead to any confusion.

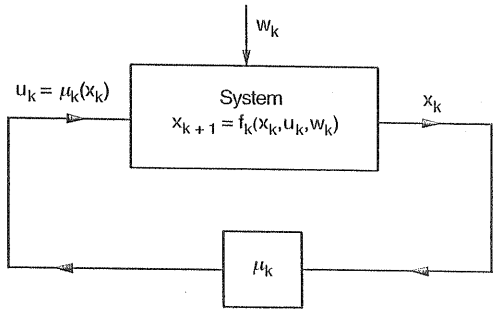


Figure 1.2.1 Information gathering in the basic problem. At each time k the controller observes the current state x_k and applies a control $u_k = \mu_k(x_k)$ that depends on that state.

Example 1.2.1

To illustrate the benefits of the proper use of information, let us consider the chess match example of the preceding section. There, a player can select timid play (probabilities p_d and $1 - p_d$ for a draw and a loss, respectively) or bold play (probabilities p_w and $1 - p_w$ for a win and a loss, respectively) in each of the two games of the match. Suppose the player chooses a policy of playing timid if and only if he is ahead in the score, as illustrated in Fig. 1.2.2; we will see in the next section that this policy is optimal, assuming $p_d > p_w$. Then after the first game (in which he plays bold), the score is 1-0 with probability p_w and 0-1 with probability $1 - p_w$. In the second game, he plays timid in the former case and bold in the latter case. Thus after two games, the probability of a match win is $p_w p_d$, the probability of a match loss is $(1 - p_w)^2$, and the probability of a tied score is $p_w(1 - p_d) + (1 - p_w)p_w$, in which case he has a probability p_w of winning the subsequent sudden-death game. Thus the probability of winning the match with the given strategy is

$$p_w p_d + p_w\big(p_w(1 - p_d) + (1 - p_w)p_w\big),$$

which, with some rearrangement, gives

$$\text{Probability of a match win} = p_w^2(2 - p_w) + p_w(1 - p_w)p_d. \qquad (1.2)$$

Suppose now that $p_w < 1/2$. Then the player has a greater probability of losing than winning any one game, regardless of the type of play he uses. From this we can infer that no open-loop strategy can give the player a greater than 50-50 chance of winning the match. Yet from Eq. (1.2) it can be seen that with the closed-loop strategy of playing timid if and only if the player is ahead in the score, the chance of a match win can be greater than 50-50, provided that p_w is close enough to $1/2$ and p_d is close enough to 1. As an example, for $p_w = 0.45$ and $p_d = 0.9$, Eq. (1.2) gives a match win probability of roughly 0.53.

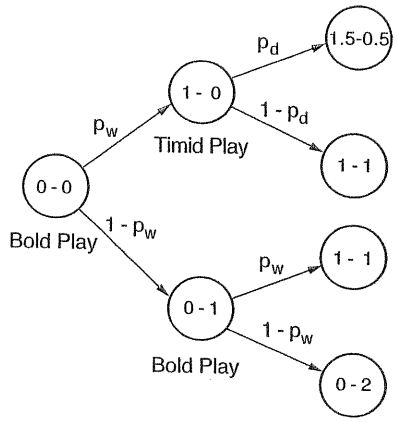


Figure 1.2.2 Illustration of the policy used in Example 1.2.1 to obtain a greater than 50-50 chance of winning the chess match and associated transition probabilities. The player chooses a policy of playing timid if and only if he is ahead in the score.

To calculate the value of information, let us consider the four open-loop policies, whereby we decide on the type of play to be used without waiting to see the result of the first game. These are:

(1) Play timid in both games; this has a probability $p_d^2 p_w$ of winning the match.

(2) Play bold in both games; this has a probability $p_w^2 + 2p_w^2(1 - p_w) = p_w^2(3 - 2p_w)$ of winning the match.

(3) Play bold in the first game and timid in the second game; this has a probability $p_w p_d + p_w^2(1 - p_d)$ of winning the match.

(4) Play timid in the first game and bold in the second game; this also has a probability $p_w p_d + p_w^2(1 - p_d)$ of winning the match.

The first policy is always dominated by the others, and the optimal open-loop probability of winning the match is

$$\begin{aligned} \text{Open-loop probability of win} &= \max\bigl(p_w^2(3 - 2p_w),\ p_w p_d + p_w^2(1 - p_d)\bigr) \\ &= p_w^2 + p_w(1 - p_w)\max(2p_w, p_d). \end{aligned} \tag{1.3}$$

Thus if $p_d > 2p_w$, we see that the optimal open-loop policy is to play timid in one of the two games and play bold in the other, and otherwise it is optimal to play bold in both games. For $p_w = 0.45$ and $p_d = 0.9$, Eq. (1.3) gives an optimal open-loop match win probability of roughly 0.425. Thus, the value of the information (the outcome of the first game) is the difference of the optimal closed-loop and open-loop values, which is approximately $0.53 - 0.425 = 0.105$.

More generally, by subtracting Eqs. (1.2) and (1.3), we see that

$$\begin{aligned} \text{Value of information} &= p_w^2(2 - p_w) + p_w(1 - p_w)p_d \\ &\quad - p_w^2 - p_w(1 - p_w)\max(2p_w, p_d) \\ &= p_w(1 - p_w)\min(p_w, p_d - p_w). \end{aligned}$$

It should be noted, however, that whereas availability of the state information cannot hurt, it may not result in an advantage either. For instance, in deterministic problems, where no random disturbances are present, one can predict the future states given the initial state and the sequence of controls. Thus, optimization over all sequences $\{u_0, u_1, \ldots, u_{N-1}\}$ of controls leads to the same optimal cost as optimization over all admissible policies. The same can be true even in some stochastic problems (see for example Exercise 1.13). This brings up a related issue. Assuming no information is forgotten, the controller actually knows the prior states and controls $x_0, u_0, \ldots, x_{k-1}, u_{k-1}$ as well as the current state x_k. Therefore, the question arises whether policies that use the entire system history can be superior to policies that use just the current state. The answer turns out to be negative although the proof is technically complicated (see [BeS78]). The intuitive reason is that, for a given time k and state x_k, all future expected costs depend explicitly just on x_k and not on prior history.

Encoding Risk in the Cost Function

As mentioned above, an important characteristic of stochastic problems is the possibility of using information with advantage. Another distinguishing characteristic is the need to take into account *risk* in the problem formulation. For example, in a typical investment problem one is not only interested in the expected profit of the investment decision, but also in its variance: given a choice between two investments with nearly equal expected profit and markedly different variance, most investors would prefer the investment with smaller variance. This indicates that expected value of cost or reward need not be the most appropriate yardstick for expressing a decision maker's preference between decisions.

As a more dramatic example of the need to take risk into account when formulating optimization problems under uncertainty, consider the so-called St. Petersburg paradox. Here, a person is offered the opportunity of paying x dollars in exchange for participation in the following game: a fair coin is flipped sequentially and the person is paid 2^k dollars, where k is the number of times heads have come up before tails come up for the first time. The decision that the person must make is whether to accept or reject participation in the game. Now if he accepts, his expected profit

from the game is

$$\sum_{k=0}^{\infty} \frac{1}{2^{k+1}} \cdot 2^k - x = \infty,$$

so if his acceptance criterion is based on maximization of expected profit, he is willing to pay any amount x to enter the game. This, however, is in strong disagreement with observed behavior, due to the risk element involved in entering the game, and shows that a different formulation of the problem is needed. The formulation of problems of decision under uncertainty so that risk is properly taken into account is a deep subject with an interesting theory. An introduction to this theory is given in Appendix G. It is shown in particular that minimization of expected cost is appropriate under reasonable assumptions, provided the cost function is suitably chosen so that it properly encodes the risk preferences of the decision maker.

1.3 THE DYNAMIC PROGRAMMING ALGORITHM

The dynamic programming (DP) technique rests on a very simple idea, the *principle of optimality*. The name is due to Bellman, who contributed a great deal to the popularization of DP and to its transformation into a systematic tool. Roughly, the principle of optimality states the following rather obvious fact.

Principle of Optimality

Let $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ be an optimal policy for the basic problem, and assume that when using π^*, a given state x_i occurs at time i with positive probability. Consider the subproblem whereby we are at x_i at time i and wish to minimize the "cost-to-go" from time i to time N

$$E\left\{ g_N(x_N) + \sum_{k=i}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big) \right\}.$$

Then the truncated policy $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ is optimal for this subproblem.

The intuitive justification of the principle of optimality is very simple. If the truncated policy $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ were not optimal as stated, we would be able to reduce the cost further by switching to an optimal policy for the subproblem once we reach x_i. For an auto travel analogy, suppose that the fastest route from Los Angeles to Boston passes through Chicago. The principle of optimality translates to the obvious fact that the Chicago to Boston portion of the route is also the fastest route for a trip that starts from Chicago and ends in Boston.

The principle of optimality suggests that an optimal policy can be constructed in piecemeal fashion, first constructing an optimal policy for the "tail subproblem" involving the last stage, then extending the optimal policy to the "tail subproblem" involving the last two stages, and continuing in this manner until an optimal policy for the entire problem is constructed. The DP algorithm is based on this idea: it proceeds sequentially, by solving all the tail subproblems of a given time length, using the solution of the tail subproblems of shorter time length. We introduce the algorithm with two examples, one deterministic and one stochastic.

The DP Algorithm for a Deterministic Scheduling Example

Let us consider the scheduling example of the preceding section, and let us apply the principle of optimality to calculate the optimal schedule. We have to schedule optimally the four operations A, B, C, and D. The transition and setup costs are shown in Fig. 1.3.1 next to the corresponding arcs.

According to the principle of optimality, the "tail" portion of an optimal schedule must be optimal. For example, suppose that the optimal schedule is CABD. Then, having scheduled first C and then A, it must be optimal to complete the schedule with BD rather than with DB. With this in mind, we solve all possible tail subproblems of length two, then all tail subproblems of length three, and finally the original problem that has length four (the subproblems of length one are of course trivial because there is only one operation that is as yet unscheduled). As we will see shortly, the tail subproblems of length $k+1$ are easily solved once we have solved the tail subproblems of length k, and this is the essence of the DP technique.

Tail Subproblems of Length 2: These subproblems are the ones that involve two unscheduled operations and correspond to the states AB, AC, CA, and CD (see Fig. 1.3.1)

State AB: Here it is only possible to schedule operation C as the next operation, so the optimal cost of this subproblem is 9 (the cost of scheduling C after B, which is 3, plus the cost of scheduling D after C, which is 6).

State AC: Here the possibilities are to (a) schedule operation B and then D, which has cost 5, or (b) schedule operation D and then B, which has cost 9. The first possibility is optimal, and the corresponding cost of the tail subproblem is 5, as shown next to node AC in Fig. 1.3.1.

State CA: Here the possibilities are to (a) schedule operation B and then D, which has cost 3, or (b) schedule operation D and then B, which has cost 7. The first possibility is optimal, and the correspond-

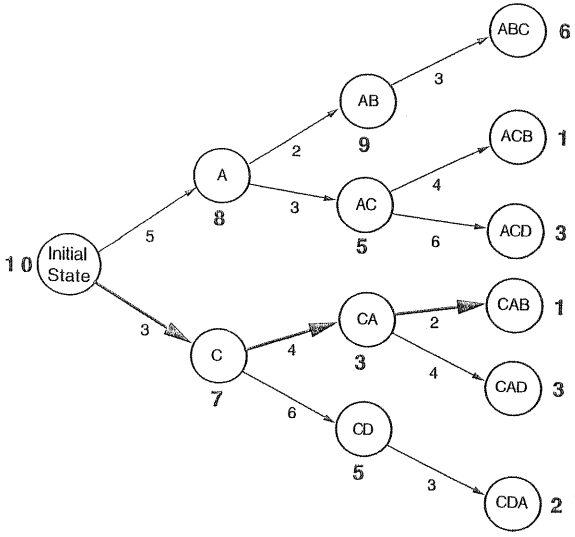


Figure 1.3.1 Transition graph of the deterministic scheduling problem, with the cost of each decision shown next to the corresponding arc. Next to each node/state we show the cost to optimally complete the schedule starting from that state. This is the optimal cost of the corresponding tail subproblem (cf. the principle of optimality). The optimal cost for the original problem is equal to 10, as shown next to the initial state. The optimal schedule corresponds to the thick-line arcs.

ing cost of the tail subproblem is 3, as shown next to node CA in Fig. 1.3.1.

State CD: Here it is only possible to schedule operation A as the next operation, so the optimal cost of this subproblem is 5.

Tail Subproblems of Length 3: These subproblems can now be solved using the optimal costs of the subproblems of length 2.

State A: Here the possibilities are to (a) schedule next operation B (cost 2) and then solve optimally the corresponding subproblem of length 2 (cost 9, as computed earlier), a total cost of 11, or (b) schedule next operation C (cost 3) and then solve optimally the corresponding subproblem of length 2 (cost 5, as computed earlier), a total cost of 8. The second possibility is optimal, and the corresponding cost of the tail subproblem is 8, as shown next to node A in Fig. 1.3.1.

State C: Here the possibilities are to (a) schedule next operation A (cost 4) and then solve optimally the corresponding subproblem of length 2 (cost 3, as computed earlier), a total cost of 7, or (b) schedule next operation D (cost 6) and then solve optimally the corresponding subproblem of length 2 (cost 5, as computed earlier), a total cost of 11. The first possibility is optimal, and the corresponding cost of the tail subproblem is 7, as shown next to node A in Fig. 1.3.1.

Original Problem of Length 4: The possibilities here are (a) start with operation A (cost 5) and then solve optimally the corresponding subproblem of length 3 (cost 8, as computed earlier), a total cost of 13, or (b) start with operation C (cost 3) and then solve optimally the corresponding subproblem of length 3 (cost 7, as computed earlier), a total cost of 10. The second possibility is optimal, and the corresponding optimal cost is 10, as shown next to the initial state node in Fig. 1.3.1.

Note that having computed the optimal cost of the original problem through the solution of all the tail subproblems, we can construct the optimal schedule by starting at the initial node and proceeding forward, each time choosing the operation that starts the optimal schedule for the corresponding tail subproblem. In this way, by inspection of the graph and the computational results of Fig. 1.3.1, we determine that CABD is the optimal schedule.

The DP Algorithm for the Inventory Control Example

Consider the inventory control example of the previous section. Similar to the solution of the preceding deterministic scheduling problem, we calculate sequentially the optimal costs of all the tail subproblems, going from shorter to longer problems. The only difference is that the optimal costs are computed as expected values, since the problem here is stochastic.

Tail Subproblems of Length 1: Assume that at the beginning of period $N-1$ the stock is x_{N-1}. Clearly, no matter what happened in the past, the inventory manager should order the amount of inventory that minimizes over $u_{N-1} \geq 0$ the sum of the ordering cost and the expected terminal holding/shortage cost. Thus, he should minimize over u_{N-1} the sum $cu_{N-1} + E\{R(x_N)\}$, which can be written as

$$cu_{N-1} + \mathop{E}_{w_{N-1}} \{R(x_{N-1} + u_{N-1} - w_{N-1})\}.$$

Adding the holding/shortage cost of period $N-1$, we see that the optimal cost for the last period (plus the terminal cost) is given by

$$J_{N-1}(x_{N-1}) = r(x_{N-1}) + \min_{u_{N-1} \geq 0} \left[cu_{N-1} + \mathop{E}_{w_{N-1}} \{R(x_{N-1} + u_{N-1} - w_{N-1})\} \right].$$

Naturally, J_{N-1} is a function of the stock x_{N-1}. It is calculated either analytically or numerically (in which case a table is used for computer

storage of the function J_{N-1}). In the process of calculating J_{N-1}, we obtain the optimal inventory policy $\mu^*_{N-1}(x_{N-1})$ for the last period: $\mu^*_{N-1}(x_{N-1})$ is the value of u_{N-1} that minimizes the right-hand side of the preceding equation for a given value of x_{N-1}.

Tail Subproblems of Length 2: Assume that at the beginning of period $N-2$ the stock is x_{N-2}. It is clear that the inventory manager should order the amount of inventory that minimizes not just the expected cost of period $N-2$ but rather the

(expected cost of period $N-2$) + (expected cost of period $N-1$, given that an optimal policy will be used at period $N-1$),

which is equal to

$$r(x_{N-2}) + cu_{N-2} + E\{J_{N-1}(x_{N-1})\}.$$

Using the system equation $x_{N-1} = x_{N-2} + u_{N-2} - w_{N-2}$, the last term is also written as $J_{N-1}(x_{N-2} + u_{N-2} - w_{N-2})$.

Thus the optimal cost for the last two periods given that we are at state x_{N-2}, denoted $J_{N-2}(x_{N-2})$, is given by

$$J_{N-2}(x_{N-2}) = r(x_{N-2}) + \min_{u_{N-2}\geq 0}\left[cu_{N-2} + \underset{w_{N-2}}{E}\{J_{N-1}(x_{N-2}+u_{N-2}-w_{N-2})\}\right]$$

Again $J_{N-2}(x_{N-2})$ is calculated for every x_{N-2}. At the same time, the optimal policy $\mu^*_{N-2}(x_{N-2})$ is also computed.

Tail Subproblems of Length $N-k$: Similarly, we have that at period k, when the stock is x_k, the inventory manager should order u_k to minimize

(expected cost of period k) + (expected cost of periods $k+1, \ldots, N-1$, given that an optimal policy will be used for these periods).

By denoting by $J_k(x_k)$ the optimal cost, we have

$$J_k(x_k) = r(x_k) + \min_{u_k\geq 0}\left[cu_k + \underset{w_k}{E}\{J_{k+1}(x_k+u_k-w_k)\}\right], \qquad (1.4)$$

which is actually the dynamic programming equation for this problem.

The functions $J_k(x_k)$ denote the optimal expected cost for the tail subproblem that starts at period k with initial inventory x_k. These functions are computed recursively backward in time, starting at period $N-1$ and ending at period 0. The value $J_0(x_0)$ is the optimal expected cost when the initial stock at time 0 is x_0. During the calculations, the optimal

policy is simultaneously computed from the minimization in the right-hand side of Eq. (1.4).

The example illustrates the main advantage offered by DP. While the original inventory problem requires an optimization over the set of policies, the DP algorithm of Eq. (1.4) decomposes this problem into a sequence of minimizations carried out over the set of controls. Each of these minimizations is much simpler than the original problem.

The DP Algorithm

We now state the DP algorithm for the basic problem and show its optimality by translating into mathematical terms the heuristic argument given above for the inventory example.

Proposition 1.3.1: For every initial state x_0, the optimal cost $J^*(x_0)$ of the basic problem is equal to $J_0(x_0)$, given by the last step of the following algorithm, which proceeds backward in time from period $N-1$ to period 0:

$$J_N(x_N) = g_N(x_N), \qquad (1.5)$$

$$J_k(x_k) = \min_{u_k\in U_k(x_k)} \underset{w_k}{E}\left\{g_k(x_k,u_k,w_k) + J_{k+1}\big(f_k(x_k,u_k,w_k)\big)\right\}, \quad k = 0,1,\ldots,N-1, \qquad (1.6)$$

where the expectation is taken with respect to the probability distribution of w_k, which depends on x_k and u_k. Furthermore, if $u^*_k = \mu^*_k(x_k)$ minimizes the right side of Eq. (1.6) for each x_k and k, the policy $\pi^* = \{\mu^*_0, \ldots, \mu^*_{N-1}\}$ is optimal.

Proof: † For any admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ and each $k = 0,1,\ldots,N-1$, denote $\pi^k = \{\mu_k, \mu_{k+1}, \ldots, \mu_{N-1}\}$. For $k = 0,1,\ldots,N-1$, let $J^*_k(x_k)$ be the optimal cost for the $(N-k)$-stage problem that starts at state x_k and time k, and ends at time N,

$$J^*_k(x_k) = \min_{\pi^k} \underset{w_k,\ldots,w_{N-1}}{E}\left\{g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i,\mu_i(x_i),w_i\big)\right\}.$$

† Our proof is somewhat informal and assumes that the functions J_k are well-defined and finite. For a strictly rigorous proof, some technical mathematical issues must be addressed; see Section 1.5. These issues do not arise if the disturbance w_k takes a finite or countable number of values and the expected values of all terms in the expression of the cost function (1.1) are well-defined and finite for every admissible policy π.

For $k = N$, we define $J_N^*(x_N) = g_N(x_N)$. We will show by induction that the functions J_k^* are equal to the functions J_k generated by the DP algorithm, so that for $k = 0$, we will obtain the desired result.

Indeed, we have by definition $J_N^* = J_N = g_N$. Assume that for some k and all x_{k+1}, we have $J_{k+1}^*(x_{k+1}) = J_{k+1}(x_{k+1})$. Then, since $\pi^k = (\mu_k, \pi^{k+1})$, we have for all x_k

$$
\begin{aligned}
J_k^*(x_k) &= \min_{(\mu_k,\pi^{k+1})} \underset{w_k,\ldots,w_{N-1}}{E} \Big\{ g_k\big(x_k,\mu_k(x_k),w_k\big) \\
&\qquad + g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i,\mu_i(x_i),w_i\big) \Big\} \\
&= \min_{\mu_k} \underset{w_k}{E} \Big\{ g_k\big(x_k,\mu_k(x_k),w_k\big) \\
&\qquad + \min_{\pi^{k+1}} \Big[\underset{w_{k+1},\ldots,w_{N-1}}{E} \Big\{ g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i,\mu_i(x_i),w_i\big) \Big\} \Big] \Big\} \\
&= \min_{\mu_k} \underset{w_k}{E} \Big\{ g_k\big(x_k,\mu_k(x_k),w_k\big) + J_{k+1}^*\big(f_k\big(x_k,\mu_k(x_k),w_k\big)\big) \Big\} \\
&= \min_{\mu_k} \underset{w_k}{E} \Big\{ g_k\big(x_k,\mu_k(x_k),w_k\big) + J_{k+1}\big(f_k\big(x_k,\mu_k(x_k),w_k\big)\big) \Big\} \\
&= \min_{u_k\in U_k(x_k)} \underset{w_k}{E} \Big\{ g_k(x_k,u_k,w_k) + J_{k+1}\big(f_k(x_k,u_k,w_k)\big) \Big\} \\
&= J_k(x_k),
\end{aligned}
$$

completing the induction. In the second equation above, we moved the minimum over π^{k+1} inside the braced expression, using a principle of optimality argument: "the tail portion of an optimal policy is optimal for the tail subproblem" (a more rigorous justification of this step is given in Section 1.5). In the third equation, we used the definition of J_{k+1}^*, and in the fourth equation we used the induction hypothesis. In the fifth equation, we converted the minimization over μ_k to a minimization over u_k, using the fact that for any function F of x and u, we have

$$\min_{\mu\in M} F\big(x,\mu(x)\big) = \min_{u\in U(x)} F(x,u),$$

where M is the set of all functions $\mu(x)$ such that $\mu(x) \in U(x)$ for all x. **Q.E.D.**

The argument of the preceding proof provides an interpretation of $J_k(x_k)$ as the optimal cost for an $(N-k)$-stage problem starting at state x_k and time k, and ending at time N. We consequently call $J_k(x_k)$ the *cost-to-go* at state x_k and time k, and refer to J_k as the *cost-to-go function* at time k.

Ideally, we would like to use the DP algorithm to obtain closed-form expressions for J_k or an optimal policy. In this book, we will discuss a large number of models that admit analytical solution by DP. Even if such models rely on oversimplified assumptions, they are often very useful. They may provide valuable insights about the structure of the optimal solution of more complex models, and they may form the basis for suboptimal control schemes. Furthermore, the broad collection of analytically solvable models provides helpful guidelines for modeling: when faced with a new problem it is worth trying to pattern its model after one of the principal analytically tractable models.

Unfortunately, in many practical cases an analytical solution is not possible, and one has to resort to numerical execution of the DP algorithm. This may be quite time-consuming since the minimization in the DP Eq. (1.6) must be carried out for each value of x_k. The state space must be discretized in some way if it is not already a finite set. The computational requirements are proportional to the number of possible values of x_k, so for complex problems the computational burden may be excessive. Nonetheless, DP is the only general approach for sequential optimization under uncertainty, and even when it is computationally prohibitive, it can serve as the basis for more practical suboptimal approaches, which will be discussed in Chapter 6.

The following examples illustrate some of the analytical and computational aspects of DP.

Example 1.3.1

A certain material is passed through a sequence of two ovens (see Fig. 1.3.2). Denote

x_0: initial temperature of the material,

x_k, $k = 1, 2$: temperature of the material at the exit of oven k,

u_{k-1}, $k = 1, 2$: prevailing temperature in oven k.

We assume a model of the form

$$x_{k+1} = (1-a)x_k + au_k, \qquad k = 0, 1,$$

where a is a known scalar from the interval $(0, 1)$. The objective is to get the final temperature x_2 close to a given target T, while expending relatively little energy. This is expressed by a cost function of the form

$$r(x_2 - T)^2 + u_0^2 + u_1^2,$$

where $r > 0$ is a given scalar. We assume no constraints on u_k. (In reality, there are constraints, but if we can solve the unconstrained problem and verify that the solution satisfies the constraints, everything will be fine.) The problem is deterministic; that is, there is no stochastic uncertainty. However,

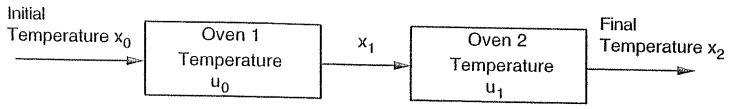


Figure 1.3.2 Problem of Example 1.3.1. The temperature of the material evolves according to $x_{k+1} = (1-a)x_k + au_k$, where a is some scalar with $0 < a < 1$.

such problems can be placed within the basic framework by introducing a fictitious disturbance taking a unique value with probability one.

We have $N = 2$ and a terminal cost $g_2(x_2) = r(x_2 - T)^2$, so the initial condition for the DP algorithm is [cf. Eq. (1.5)]

$$J_2(x_2) = r(x_2 - T)^2.$$

For the next-to-last stage, we have [cf. Eq. (1.6)]

$$\begin{aligned} J_1(x_1) &= \min_{u_1}\big[u_1^2 + J_2(x_2)\big] \\ &= \min_{u_1}\Big[u_1^2 + J_2\big((1-a)x_1 + au_1\big)\Big]. \end{aligned}$$

Substituting the previous form of J_2, we obtain

$$J_1(x_1) = \min_{u_1}\Big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2\Big]. \tag{1.7}$$

This minimization will be done by setting to zero the derivative with respect to u_1. This yields

$$0 = 2u_1 + 2ra\big((1-a)x_1 + au_1 - T\big),$$

and by collecting terms and solving for u_1, we obtain the optimal temperature for the last oven:

$$\mu_1^*(x_1) = \frac{ra\big(T - (1-a)x_1\big)}{1 + ra^2}.$$

Note that this is not a single control but rather a control function, a rule that tells us the optimal oven temperature $u_1 = \mu_1^*(x_1)$ for each possible state x_1.

By substituting the optimal u_1 in the expression (1.7) for J_1, we obtain

$$\begin{aligned} J_1(x_1) &= \frac{r^2a^2\big((1-a)x_1 - T\big)^2}{(1+ra^2)^2} + r\left((1-a)x_1 + \frac{ra^2\big(T-(1-a)x_1\big)}{1+ra^2} - T\right)^2 \\ &= \frac{r^2a^2\big((1-a)x_1 - T\big)^2}{(1+ra^2)^2} + r\left(\frac{ra^2}{1+ra^2} - 1\right)^2\big((1-a)x_1 - T\big)^2 \\ &= \frac{r\big((1-a)x_1 - T\big)^2}{1+ra^2}. \end{aligned}$$

We now go back one stage. We have [cf. Eq. (1.6)]

$$J_0(x_0) = \min_{u_0}\big[u_0^2 + J_1(x_1)\big] = \min_{u_0}\Big[u_0^2 + J_1\big((1-a)x_0 + au_0\big)\Big],$$

and by substituting the expression already obtained for J_1, we have

$$J_0(x_0) = \min_{u_0}\left[u_0^2 + \frac{r\big((1-a)^2x_0 + (1-a)au_0 - T\big)^2}{1+ra^2}\right].$$

We minimize with respect to u_0 by setting the corresponding derivative to zero. We obtain

$$0 = 2u_0 + \frac{2r(1-a)a\big((1-a)^2x_0 + (1-a)au_0 - T\big)}{1+ra^2}.$$

This yields, after some calculation, the optimal temperature of the first oven:

$$\mu_0^*(x_0) = \frac{r(1-a)a\big(T - (1-a)^2x_0\big)}{1 + ra^2\big(1 + (1-a)^2\big)}.$$

The optimal cost is obtained by substituting this expression in the formula for J_0. This leads to a straightforward but lengthy calculation, which in the end yields the rather simple formula

$$J_0(x_0) = \frac{r\big((1-a)^2x_0 - T\big)^2}{1 + ra^2\big(1 + (1-a)^2\big)}.$$

This completes the solution of the problem.

One noteworthy feature in the preceding example is the facility with which we obtained an analytical solution. A little thought while tracing the steps of the algorithm will convince the reader that what simplifies the solution is the quadratic nature of the cost and the linearity of the system equation. In Section 4.1 we will see that, generally, when the system is linear and the cost is quadratic, the optimal policy and cost-to-go function are given by closed-form expressions, regardless of the number of stages N.

Another noteworthy feature of the example is that the optimal policy remains unaffected when a zero-mean stochastic disturbance is added in the system equation. To see this, assume that the material' s temperature evolves according to

$$x_{k+1} = (1-a)x_k + au_k + w_k, \qquad k = 0, 1,$$

where w_0, w_1 are independent random variables with given distribution, zero mean

$$E\{w_0\} = E\{w_1\} = 0,$$

and finite variance. Then the equation for J_1 [cf. Eq. (1.6)] becomes

$$\begin{aligned} J_1(x_1) &= \min_{u_1} \mathop{E}_{w_1} \left\{ u_1^2 + r\big((1-a)x_1 + au_1 + w_1 - T\big)^2 \right\} \\ &= \min_{u_1} \Big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2 \\ &\qquad + 2rE\{w_1\}\big((1-a)x_1 + au_1 - T\big) + rE\{w_1^2\} \Big]. \end{aligned}$$

Since $E\{w_1\} = 0$, we obtain

$$J_1(x_1) = \min_{u_1} \Big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2 \Big] + rE\{w_1^2\}.$$

Comparing this equation with Eq. (1.7), we see that the presence of w_1 has resulted in an additional inconsequential term, $rE\{w_1^2\}$. Therefore, the optimal policy for the last stage remains unaffected by the presence of w_1, while $J_1(x_1)$ is increased by the constant term $rE\{w_1^2\}$. It can be seen that a similar situation also holds for the first stage. In particular, the optimal cost is given by the same expression as before except for an additive constant that depends on $E\{w_0^2\}$ and $E\{w_1^2\}$.

If the optimal policy is unaffected when the disturbances are replaced by their means, we say that *certainty equivalence* holds. We will derive certainty equivalence results for several types of problems involving a linear system and a quadratic cost (see Sections 4.1, 5.2, and 5.3).

Example 1.3.2

To illustrate the computational aspects of DP, consider an inventory control problem that is slightly different from the one of Sections 1.1 and 1.2. In particular, we assume that inventory u_k and the demand w_k are nonnegative integers, and that the excess demand $(w_k - x_k - u_k)$ is lost. As a result, the stock equation takes the form

$$x_{k+1} = \max(0, x_k + u_k - w_k).$$

We also assume that there is an upper bound of 2 units on the stock that can be stored, i.e. there is a constraint $x_k + u_k \leq 2$. The holding/storage cost for the kth period is given by

$$(x_k + u_k - w_k)^2,$$

implying a penalty both for excess inventory and for unmet demand at the end of the kth period. The ordering cost is 1 per unit stock ordered. Thus the cost per period is

$$g_k(x_k, u_k, w_k) = u_k + (x_k + u_k - w_k)^2.$$

The terminal cost is assumed to be 0,

$$g_N(x_N) = 0.$$

The planning horizon N is 3 periods, and the initial stock x_0 is 0. The demand w_k has the same probability distribution for all periods, given by

$$p(w_k = 0) = 0.1, \qquad p(w_k = 1) = 0.7, \qquad p(w_k = 2) = 0.2.$$

The system can also be represented in terms of the transition probabilities $p_{ij}(u)$ between the three possible states, for the different values of the control (see Fig. 1.3.3).

The starting equation for the DP algorithm is

$$J_3(x_3) = 0,$$

since the terminal state cost is 0 [cf. Eq. (1.5)]. The algorithm takes the form [cf. Eq. (1.6)]

$$J_k(x_k) = \min_{\substack{0 \leq u_k \leq 2 - x_k \\ u_k = 0,1,2}} \mathop{E}_{w_k} \Big\{ u_k + (x_k + u_k - w_k)^2 + J_{k+1}\big(\max(0, x_k + u_k - w_k)\big) \Big\},$$

where $k = 0, 1, 2$, and x_k, u_k, w_k can take the values 0, 1, and 2.

Period 2: We compute $J_2(x_2)$ for each of the three possible states. We have

$$\begin{aligned} J_2(0) &= \min_{u_2 = 0,1,2} \mathop{E}_{w_2} \big\{ u_2 + (u_2 - w_2)^2 \big\} \\ &= \min_{u_2 = 0,1,2} \big[u_2 + 0.1(u_2)^2 + 0.7(u_2 - 1)^2 + 0.2(u_2 - 2)^2 \big]. \end{aligned}$$

We calculate the expectation of the right side for each of the three possible values of u_2:

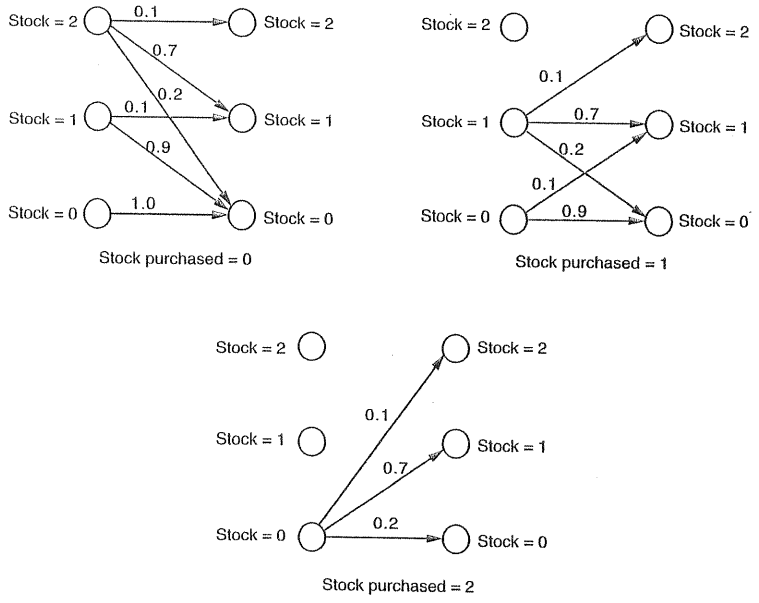
$$\begin{aligned} u_2 = 0 &: E\{\cdot\} = 0.7 \cdot 1 + 0.2 \cdot 4 = 1.5, \\ u_2 = 1 &: E\{\cdot\} = 1 + 0.1 \cdot 1 + 0.2 \cdot 1 = 1.3, \\ u_2 = 2 &: E\{\cdot\} = 2 + 0.1 \cdot 4 + 0.7 \cdot 1 = 3.1. \end{aligned}$$

Hence we have, by selecting the minimizing u_2,

$$J_2(0) = 1.3, \qquad \mu_2^*(0) = 1.$$

For $x_2 = 1$, we have

$$\begin{aligned} J_2(1) &= \min_{u_2 = 0,1} \mathop{E}_{w_2} \big\{ u_2 + (1 + u_2 - w_2)^2 \big\} \\ &= \min_{u_2 = 0,1} \big[u_2 + 0.1(1 + u_2)^2 + 0.7(u_2)^2 + 0.2(u_2 - 1)^2 \big]. \end{aligned}$$



Stock	Stage 0 Cost-to-go	Stage 0 Optimal stock to purchase	Stage 1 Cost-to-go	Stage 1 Optimal stock to purchase	Stage 2 Cost-to-go	Stage 2 Optimal stock to purchase
0	3.7	1	2.5	1	1.3	1
1	2.7	0	1.5	0	0.3	0
2	2.818	0	1.68	0	1.1	0

Figure 1.3.3 System and DP results for Example 1.3.2. The transition probability diagrams for the different values of stock purchased (control) are shown. The numbers next to the arcs are the transition probabilities. The control $u = 1$ is not available at state 2 because of the limitation $x_k + u_k \leq 2$. Similarly, the control $u = 2$ is not available at states 1 and 2. The results of the DP algorithm are given in the table.

The expected value in the right side is

$$u_2 = 0 : E\{\cdot\} = 0.1 \cdot 1 + 0.2 \cdot 1 = 0.3,$$
$$u_2 = 1 : E\{\cdot\} = 1 + 0.1 \cdot 4 + 0.7 \cdot 1 = 2.1.$$

Hence

$$J_2(1) = 0.3, \qquad \mu_2^*(1) = 0.$$

For $x_2 = 2$, the only admissible control is $u_2 = 0$, so we have

$$J_2(2) = \underset{w_2}{E}\{(2 - w_2)^2\} = 0.1 \cdot 4 + 0.7 \cdot 1 = 1.1,$$

$$J_2(2) = 1.1, \qquad \mu_2^*(2) = 0.$$

Period 1: Again we compute $J_1(x_1)$ for each of the three possible states $x_1 = 0, 1, 2$, using the values $J_2(0)$, $J_2(1)$, $J_2(2)$ obtained in the previous period. For $x_1 = 0$, we have

$$J_1(0) = \min_{u_1=0,1,2} \underset{w_1}{E}\Big\{u_1 + (u_1 - w_1)^2 + J_2\big(\max(0, u_1 - w_1)\big)\Big\},$$

$$u_1 = 0 : E\{\cdot\} = 0.1 \cdot J_2(0) + 0.7\big(1 + J_2(0)\big) + 0.2\big(4 + J_2(0)\big) = 2.8,$$
$$u_1 = 1 : E\{\cdot\} = 1 + 0.1\big(1 + J_2(1)\big) + 0.7 \cdot J_2(0) + 0.2\big(1 + J_2(0)\big) = 2.5,$$
$$u_1 = 2 : E\{\cdot\} = 2 + 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) = 3.68,$$

$$J_1(0) = 2.5, \qquad \mu_1^*(0) = 1.$$

For $x_1 = 1$, we have

$$J_1(1) = \min_{u_1=0,1} \underset{w_1}{E}\Big\{u_1 + (1 + u_1 - w_1)^2 + J_2\big(\max(0, 1 + u_1 - w_1)\big)\Big\},$$

$$u_1 = 0 : E\{\cdot\} = 0.1\big(1 + J_2(1)\big) + 0.7 \cdot J_2(0) + 0.2\big(1 + J_2(0)\big) = 1.5,$$
$$u_1 = 1 : E\{\cdot\} = 1 + 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) = 2.68,$$

$$J_1(1) = 1.5, \qquad \mu_1^*(1) = 0.$$

For $x_1 = 2$, the only admissible control is $u_1 = 0$, so we have

$$\begin{aligned} J_1(2) &= \underset{w_1}{E}\Big\{(2 - w_1)^2 + J_2\big(\max(0, 2 - w_1)\big)\Big\} \\ &= 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) \\ &= 1.68, \end{aligned}$$

$$J_1(2) = 1.68, \qquad \mu_1^*(2) = 0.$$

Period 0: Here we need to compute only $J_0(0)$ since the initial state is known to be 0. We have

$$J_0(0) = \min_{u_0=0,1,2} \underset{w_0}{E}\Big\{u_0 + (u_0 - w_0)^2 + J_1\big(\max(0, u_0 - w_0)\big)\Big\},$$

$$u_0 = 0 : E\{\cdot\} = 0.1 \cdot J_1(0) + 0.7\big(1 + J_1(0)\big) + 0.2\big(4 + J_1(0)\big) = 4.0,$$
$$u_0 = 1 : E\{\cdot\} = 1 + 0.1\big(1 + J_1(1)\big) + 0.7 \cdot J_1(0) + 0.2\big(1 + J_1(0)\big) = 3.7,$$
$$u_0 = 2 : E\{\cdot\} = 2 + 0.1\big(4 + J_1(2)\big) + 0.7\big(1 + J_1(1)\big) + 0.2 \cdot J_1(0) = 4.818,$$

$$J_0(0) = 3.7, \qquad \mu_0^*(0) = 1.$$

If the initial state were not known a priori, we would have to compute in a similar manner $J_0(1)$ and $J_0(2)$, as well as the minimizing u_0. The reader may verify (Exercise 1.2) that these calculations yield

$$J_0(1) = 2.7, \qquad \mu_0^*(1) = 0,$$

$$J_0(2) = 2.818, \qquad \mu_0^*(2) = 0.$$

Thus the optimal ordering policy for each period is to order one unit if the current stock is zero and order nothing otherwise. The results of the DP algorithm are given in tabular form in Fig. 1.3.3.

Example 1.3.3 (Optimizing a Chess Match Strategy)

Consider the chess match example of Section 1.1. There, a player can select timid play (probabilities p_d and $1 - p_d$ for a draw or loss, respectively) or bold play (probabilities p_w and $1 - p_w$ for a win or loss, respectively) in each game of the match. We want to formulate a DP algorithm for finding the policy that maximizes the player's probability of winning the match. Note that here we are dealing with a maximization problem. We can convert the problem to a minimization problem by changing the sign of the cost function, but a simpler alternative, which we will generally adopt, is to replace the minimization in the DP algorithm with maximization.

Let us consider the general case of an N-game match, and let the state be the *net score*, that is, the difference between the points of the player minus the points of the opponent (so a state of 0 corresponds to an even score). The optimal cost-to-go function at the start of the kth game is given by the dynamic programming recursion

$$J_k(x_k) = \max\big[p_d J_{k+1}(x_k) + (1 - p_d) J_{k+1}(x_k - 1), \\ p_w J_{k+1}(x_k + 1) + (1 - p_w) J_{k+1}(x_k - 1)\big]. \tag{1.8}$$

The maximum above is taken over the two possible decisions:

(a) Timid play, which keeps the score at x_k with probability p_d, and changes x_k to $x_k - 1$ with probability $1 - p_d$.

(b) Bold play, which changes x_k to $x_k + 1$ or to $x_k - 1$ with probabilities p_w or $(1 - p_w)$, respectively.

It is optimal to play bold when

$$p_w J_{k+1}(x_k + 1) + (1 - p_w) J_{k+1}(x_k - 1) \geq p_d J_{k+1}(x_k) + (1 - p_d) J_{k+1}(x_k - 1)$$

or equivalently, if

$$\frac{p_w}{p_d} \geq \frac{J_{k+1}(x_k) - J_{k+1}(x_k - 1)}{J_{k+1}(x_k + 1) - J_{k+1}(x_k - 1)}. \tag{1.9}$$

The dynamic programming recursion is started with

$$J_N(x_N) = \begin{cases} 1 & \text{if } x_N > 0, \\ p_w & \text{if } x_N = 0, \\ 0 & \text{if } x_N < 0. \end{cases} \tag{1.10}$$

In this equation, we have $J_N(0) = p_w$ because when the score is even after N games ($x_N = 0$), it is optimal to play bold in the first game of sudden death.

By executing the DP algorithm (1.8) starting with the terminal condition (1.10), and using the criterion (1.9) for optimality of bold play, we find the following, assuming that $p_d > p_w$:

$$J_{N-1}(x_{N-1}) = 1 \text{ for } x_{N-1} > 1; \quad \text{optimal play: either}$$
$$J_{N-1}(1) = \max[p_d + (1 - p_d)p_w,\ p_w + (1 - p_w)p_w]$$
$$= p_d + (1 - p_d)p_w; \quad \text{optimal play: timid}$$
$$J_{N-1}(0) = p_w; \quad \text{optimal play: bold}$$
$$J_{N-1}(-1) = p_w^2; \quad \text{optimal play: bold}$$
$$J_{N-1}(x_{N-1}) = 0 \text{ for } x_{N-1} < -1; \quad \text{optimal play: either.}$$

Also, given $J_{N-1}(x_{N-1})$, and Eqs. (1.8) and (1.9) we obtain

$$J_{N-2}(0) = \max\Big[p_d p_w + (1 - p_d)p_w^2,\ p_w\big(p_d + (1 - p_d)p_w\big) + (1 - p_w)p_w^2\Big]$$
$$= p_w\big(p_w + (p_w + p_d)(1 - p_w)\big)$$

and that if the score is even with 2 games remaining, it is optimal to play bold. Thus for a 2-game match, the optimal policy for both periods is to play timid if and only if the player is ahead in the score. The region of pairs (p_w, p_d) for which the player has a better than 50-50 chance to win a 2-game match is

$$R_2 = \Big\{(p_w, p_d) \mid J_0(0) = p_w\big(p_w + (p_w + p_d)(1 - p_w)\big) > 1/2\Big\},$$

and, as noted in the preceding section, it includes points where $p_w < 1/2$.

Example 1.3.4 (Finite-State Systems)

We mentioned earlier (cf. the examples in Section 1.1) that systems with a finite number of states can be represented either in terms of a discrete-time system equation or in terms of the probabilities of transition between the states. Let us work out the DP algorithm corresponding to the latter case. We assume for the sake of the following discussion that the problem is stationary (i.e., the transition probabilities, the cost per stage, and the control constraint sets do not change from one stage to the next). Then, if

$$p_{ij}(u) = P\{x_{k+1} = j \mid x_k = i, u_k = u\}$$

are the transition probabilities, we can alternatively represent the system by the system equation (cf. the discussion of the previous section)

$$x_{k+1} = w_k,$$

where the probability distribution of the disturbance w_k is

$$P\{w_k = j \mid x_k = i, u_k = u\} = p_{ij}(u).$$

Using this system equation and denoting by $g(i,u)$ the expected cost per stage at state i when control u is applied, the DP algorithm can be rewritten as

$$J_k(i) = \min_{u \in U(i)} \Big[g(i,u) + E\big\{ J_{k+1}(w_k) \big\} \Big]$$

or equivalently (in view of the distribution of w_k given previously)

$$J_k(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_j p_{ij}(u) J_{k+1}(j) \right].$$

As an illustration, in the machine replacement example of Section 1.1, this algorithm takes the form

$$J_N(i) = 0, \qquad i = 1, \ldots, n,$$

$$J_k(i) = \min \left[R + g(1) + J_{k+1}(1),\ g(i) + \sum_{j=i}^{n} p_{ij} J_{k+1}(j) \right].$$

The two expressions in the above minimization correspond to the two available decisions (replace or not replace the machine).

In the queueing example of Section 1.1, the DP algorithm takes the form

$$J_N(i) = R(i), \qquad i = 0, 1, \ldots, n,$$

$$J_k(i) = \min \left[r(i) + c_f + \sum_{j=0}^{n} p_{ij}(u_f) J_{k+1}(j),\ r(i) + c_s + \sum_{j=0}^{n} p_{ij}(u_s) J_{k+1}(j) \right].$$

The two expressions in the above minimization correspond to the two possible decisions (fast and slow service).

Note that if there are n states at each stage, and $U(i)$ contains as many as m controls, the minimization in the right-hand side of the DP algorithm requires, for each (i,k), as many as a constant multiple of mn operations. Since there are nN state-time pairs, the total number of operations for the DP algorithm is as large as a constant multiple of mn^2N operations. By contrast, the number of all policies is exponential in nN (it is as large as m^{nN}), so a brute force approach which enumerates all policies and compares their cost, requires an exponential number of operations in nN.

1.4 STATE AUGMENTATION AND OTHER REFORMULATIONS

We now discuss how to deal with situations where some of the assumptions of the basic problem are violated. Generally, in such cases the problem can be reformulated into the basic problem format. This process is called *state augmentation* because it typically involves the enlargement of the state space. The general guideline in state augmentation is to *include in the enlarged state at time k all the information that is known to the controller at time k and can be used with advantage in selecting* u_k. Unfortunately, state augmentation often comes at a price: the reformulated problem may have very complex state and/or control spaces. We provide some examples.

Time Lags

In many applications the system state x_{k+1} depends not only on the preceding state x_k and control u_k but also on earlier states and controls. In other words, states and controls influence future states with some time lag. Such situations can be handled by state augmentation; the state is expanded to include an appropriate number of earlier states and controls.

For simplicity, assume that there is at most a single period time lag in the state and control; that is, the system equation has the form

$$x_{k+1} = f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k), \qquad k = 1, 2, \ldots, N-1, \tag{1.11}$$

$$x_1 = f_0(x_0, u_0, w_0).$$

Time lags of more than one period can be handled similarly.

If we introduce additional state variables y_k and s_k, and we make the identifications $y_k = x_{k-1}$, $s_k = u_{k-1}$, the system equation (1.11) yields

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \\ s_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, y_k, u_k, s_k, w_k) \\ x_k \\ u_k \end{pmatrix}. \tag{1.12}$$

By defining $\tilde{x}_k = (x_k, y_k, s_k)$ as the new state, we have

$$\tilde{x}_{k+1} = \tilde{f}_k(\tilde{x}_k, u_k, w_k),$$

where the system function $\tilde{f}_k$ is defined from Eq. (1.12). By using the preceding equation as the system equation and by expressing the cost function in terms of the new state, the problem is reduced to the basic problem without time lags. Naturally, the control u_k should now depend on the new state $\tilde{x}_k$, or equivalently a policy should consist of functions μ_k of the current state x_k, as well as the preceding state x_{k-1} and the preceding control u_{k-1}.

When the DP algorithm for the reformulated problem is translated in terms of the variables of the original problem, it takes the form

$$J_N(x_N) = g_N(x_N),$$

$$J_{N-1}(x_{N-1}, x_{N-2}, u_{N-2}) = \min_{u_{N-1} \in U_{N-1}(x_{N-1})} E_{w_{N-1}} \Big\{ g_{N-1}(x_{N-1}, u_{N-1}, w_{N-1}) + J_N\big(f_{N-1}(x_{N-1}, x_{N-2}, u_{N-1}, u_{N-2}, w_{N-1})\big) \Big\},$$

$$J_k(x_k, x_{k-1}, u_{k-1}) = \min_{u_k \in U_k(x_k)} E_{w_k} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k), x_k, u_k\big) \Big\}, \quad k = 1, \ldots, N-2,$$

$$J_0(x_0) = \min_{u_0 \in U_0(x_0)} E_{w_0} \Big\{ g_0(x_0, u_0, w_0) + J_1\big(f_0(x_0, u_0, w_0), x_0, u_0\big) \Big\}.$$

Similar reformulations are possible when time lags appear in the cost; for example, in the case where the cost has the form

$$E\left\{ g_N(x_N, x_{N-1}) + g_0(x_0, u_0, w_0) + \sum_{k=1}^{N-1} g_k(x_k, x_{k-1}, u_k, w_k) \right\}.$$

The extreme case of time lags in the cost arises in the nonadditive form

$$E\big\{ g_N(x_N, x_{N-1}, \ldots, x_0, u_{N-1}, \ldots, u_0, w_{N-1}, \ldots, w_0) \big\}.$$

Then, the problem can be reduced to the basic problem format, by taking as augmented state

$$\tilde{x}_k = (x_k, x_{k-1}, \ldots, x_0, u_{k-1}, \ldots, u_0, w_{k-1}, \ldots, w_0)$$

and $E\{g_N(\tilde{x}_N)\}$ as reformulated cost. Policies consist of functions μ_k of the present and past states $x_k, \ldots, x_0$, the past controls $u_{k-1}, \ldots, u_0$, and the past disturbances $w_{k-1}, \ldots, w_0$. Naturally, we must assume that the past disturbances are known to the controller. Otherwise, we are faced with a problem where the state is imprecisely known to the controller. Such problems are known as problems with imperfect state information and will be discussed in Chapter 5.

Correlated Disturbances

We turn now to the case where the disturbances w_k are correlated over time. A common situation that can be handled efficiently by state augmentation arises when the process $w_0, \ldots, w_{N-1}$ can be represented as the output of a linear system driven by independent random variables. As an example, suppose that by using statistical methods, we determine that the evolution of w_k can be modeled by an equation of the form

$$w_k = \lambda w_{k-1} + \xi_k,$$

where λ is a given scalar and $\{\xi_k\}$ is a sequence of independent random vectors with given distribution. Then we can introduce an additional state variable

$$y_k = w_{k-1}$$

and obtain a new system equation

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k, \lambda y_k + \xi_k) \\ \lambda y_k + \xi_k \end{pmatrix},$$

where the new state is the pair $\tilde{x}_k = (x_k, y_k)$ and the new disturbance is the vector ξ_k.

More generally, suppose that w_k can be modeled by

$$w_k = C_k y_{k+1},$$

where

$$y_{k+1} = A_k y_k + \xi_k, \qquad k = 0, \ldots, N-1,$$

A_k, C_k are known matrices of appropriate dimension, and ξ_k are independent random vectors with given distribution (see Fig. 1.4.1). By viewing y_k as an additional state variable, we obtain the new system equation

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big) \\ A_k y_k + \xi_k \end{pmatrix}.$$

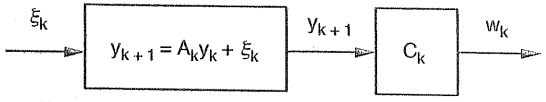


Figure 1.4.1 Representing correlated disturbances as the output of a linear system driven by independent random vectors.

Note that in order to have perfect state information, the controller must be able to observe y_k. Unfortunately, this is true only in the minority of practical cases; for example when C_k is the identity matrix and w_{k-1} is observed before u_k is applied. In the case of perfect state information, the DP algorithm takes the form

$$J_N(x_N, y_N) = g_N(x_N),$$

$$J_k(x_k, y_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{\xi_k} \Big\{ g_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big) + J_{k+1}\Big(f_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big), A_k y_k + \xi_k\Big) \Big\}.$$

Forecasts

Finally, consider the case where at time k the controller has access to a forecast y_k that results in a reassessment of the probability distribution of w_k and possibly of future disturbances. For example, y_k may be an exact prediction of w_k or an exact prediction that the probability distribution of w_k is a specific one out of a finite collection of distributions. Forecasts of interest in practice are, for example, probabilistic predictions on the state of the weather, the interest rate for money, and the demand for inventory.

Generally, forecasts can be handled by state augmentation although the reformulation into the basic problem format may be quite complex. We will treat here only a simple special case.

Assume that at the beginning of each period k, the controller receives an accurate prediction that the next disturbance w_k will be selected according to a particular probability distribution out of a given collection of distributions $\{Q_1, \ldots, Q_m\}$; that is, if the forecast is i, then w_k is selected according to Q_i. The a priori probability that the forecast will be i is denoted by p_i and is given.

For instance, suppose that in our earlier inventory example the demand w_k is determined according to one of three distributions Q_1, Q_2, and Q_3, corresponding to "small," "medium," and "large" demand. Each of the three types of demand occurs with a given probability at each time period, independently of the values of demand at previous time periods. However, the inventory manager, prior to ordering u_k, gets to know through a forecast the type of demand that will occur. (Note that it is the probability distribution of demand that becomes known through the forecast, not the demand itself.)

The forecasting process can be represented by means of the equation

$$y_{k+1} = \xi_k,$$

where y_{k+1} can take the values $1, \ldots, m$, corresponding to the m possible forecasts, and ξ_k is a random variable taking the value i with probability

p_i. The interpretation here is that when ξ_k takes the value i, then w_{k+1} will occur according to the distribution Q_i.

By combining the system equation with the forecast equation $y_{k+1} = \xi_k$, we obtain an augmented system given by

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k, w_k) \\ \xi_k \end{pmatrix}.$$

The new state is

$$\tilde{x}_k = (x_k, y_k),$$

and because the forecast y_k is known at time k, perfect state information prevails. The new disturbance is

$$\tilde{w}_k = (w_k, \xi_k),$$

and its probability distribution is determined by the distributions Q_i and the probabilities p_i, and depends explicitly on $\tilde{x}_k$ (via y_k) but not on the prior disturbances.

Thus, by suitable reformulation of the cost, the problem can be cast into the basic problem format. Note that the control applied depends on both the current state and the current forecast. The DP algorithm takes the form

$$J_N(x_N, y_N) = g_N(x_N),$$

$$J_k(x_k, y_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + \sum_{i=1}^{m} p_i J_{k+1}\big(f_k(x_k, u_k, w_k), i\big) \mid y_k \Big\}, \tag{1.13}$$

where y_k may take the values $1, \ldots, m$, and the expectation over w_k is taken with respect to the distribution Q_{y_k}.

It should be clear that the preceding formulation admits several extensions. One example is the case where forecasts can be influenced by the control action and involve several future disturbances. However, the price for these extensions is increased complexity of the corresponding DP algorithm.

Simplification for Uncontrollable State Components

When augmenting the state of a given system one often ends up with composite states, consisting of several components. It turns out that if some of these components cannot be affected by the choice of control, the DP algorithm can be simplified considerably, as we will now describe.

Let the state of the system be a composite (x_k, y_k) of two components x_k and y_k. The evolution of the main component, x_k, is affected by the control u_k according to the equation

$$x_{k+1} = f_k(x_k, y_k, u_k, w_k),$$

where the probability distribution $P_k(w_k \mid x_k, y_k, u_k)$ is given. The evolution of the other component, y_k, is governed by a given conditional distribution $P_k(y_k \mid x_k)$ and cannot be affected by the control, except indirectly through x_k. One is tempted to view y_k as a disturbance, but there is a difference: y_k is observed by the controller before applying u_k, while w_k occurs after u_k is applied, and indeed w_k may probabilistically depend on u_k.

We will formulate a DP algorithm that is executed over the controllable component of the state, with the dependence on the uncontrollable component being "averaged out." In particular, let $J_k(x_k, y_k)$ denote the optimal cost-to-go at stage k and state (x_k, y_k), and define

$$\hat{J}_k(x_k) = \mathop{E}_{y_k}\{J_k(x_k, y_k) \mid x_k\}.$$

We will derive a DP algorithm that generates $\hat{J}_k(x_k)$.

Indeed, we have

$$\begin{aligned}
\hat{J}_k(x_k) &= E_{y_k}\{J_k(x_k, y_k) \mid x_k\} \\
&= E_{y_k}\Big\{ \min_{u_k \in U_k(x_k, y_k)} E_{w_k, x_{k+1}, y_{k+1}}\{g_k(x_k, y_k, u_k, w_k) \\
&\qquad + J_{k+1}(x_{k+1}, y_{k+1}) \mid x_k, y_k, u_k\} \mid x_k\Big\} \\
&= E_{y_k}\Big\{ \min_{u_k \in U_k(x_k, y_k)} E_{w_k, x_{k+1}}\Big\{g_k(x_k, y_k, u_k, w_k) \\
&\qquad + E_{y_{k+1}}\{J_{k+1}(x_{k+1}, y_{k+1}) \mid x_{k+1}\} \mid x_k, y_k, u_k\Big\} \mid x_k\Big\},
\end{aligned}$$

and finally

$$\hat{J}_k(x_k) = \mathop{E}_{y_k}\Big\{ \min_{u_k \in U_k(x_k, y_k)} \mathop{E}_{w_k}\Big\{g_k(x_k, y_k, u_k, w_k) + \hat{J}_{k+1}\big(f_k(x_k, y_k, u_k, w_k)\big)\Big\} \Big| \, x_k\Big\}. \tag{1.14}$$

The advantage of this equivalent DP algorithm is that it is executed over a significantly reduced state space. For example, if x_k takes n possible values and y_k takes m possible values, then DP is executed over n states instead of nm states. Note, however, that the minimization in the right-hand side of the preceding equation yields an optimal control law as a function of the full state (x_k, y_k).

As an example, consider the augmented state resulting from the incorporation of forecasts, as described earlier. Then, the forecast y_k represents

an uncontrolled state component, so that the DP algorithm can be simplified as in Eq. (1.14). In particular, by defining

$$\hat{J}_k(x_k) = \sum_{i=1}^{m} p_i J_k(x_k, i), \qquad k = 0, 1, \ldots, N-1,$$

and

$$\hat{J}_N(x_N) = g_N(x_N),$$

we have, using Eq. (1.13),

$$\begin{aligned}
\hat{J}_k(x_k) = \sum_{i=1}^{m} p_i \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k}\Big\{ & g_k(x_k, u_k, w_k) \\
& + \hat{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \mid y_k = i\Big\},
\end{aligned}$$

which is executed over the space of x_k rather than x_k *and* y_k.

Uncontrolled state components often occur in arrival systems, such as queueing, where action must be taken in response to a random event (such as a customer arrival) that cannot be influenced by the choice of control. Then the state of the arrival system must be augmented to include the random event, but the DP algorithm can be executed over a smaller space, as per Eq. (1.14). Here is another example of similar type.

Example 1.4.1: (Tetris)

Tetris is a popular video game played on a two-dimensional grid. Each square in the grid can be full or empty, making up a "wall of bricks" with "holes" and a "jagged top". The squares fill up as blocks of different shapes fall from the top of the grid and are added to the top of the wall. As a given block falls, the player can move horizontally and rotate the block in all possible ways, subject to the constraints imposed by the sides of the grid and the top of the wall. The falling blocks are generated independently according to some probability distribution, defined over a finite set of standard shapes. The game starts with an empty grid and ends when a square in the top row becomes full and the top of the wall reaches the top of the grid. When a row of full squares is created, this row is removed, the bricks lying above this row move one row downward, and the player scores a point. The player's objective is to maximize the score attained (total number of rows removed) within N steps or up to termination of the game, whichever occurs first.

We can model the problem of finding an optimal tetris playing strategy as a stochastic DP problem. The control, denoted by u, is the horizontal positioning and rotation applied to the falling block. The state consists of two components:

(1) The board position, i.e., a binary description of the full/empty status of each square, denoted by x.

(2) The shape of the current falling block, denoted by y.

There is also an additional termination state which is cost-free. Once the state reaches the termination state, it stays there with no change in cost.

The shape y is generated according to a probability distribution $p(y)$, independently of the control, so it can be viewed as an uncontrollable state component. The DP algorithm (1.14) is executed over the space of x and has the intuitive form

$$\hat{J}_k(x) = \sum_y p(y) \max_u \Big[g(x,y,u) + \hat{J}_{k+1}\big(f(x,y,u)\big) \Big], \qquad \text{for all } x,$$

where $g(x,y,u)$ and $f(x,y,u)$ are the number of points scored (rows removed), and the board position (or termination state) when the state is (x,y) and control u is applied, respectively. Note, however, that despite the simplification in the DP algorithm achieved by eliminating the uncontrollable portion of the state, the number of states x is enormous, and the problem can only be addressed by suboptimal methods, which will be discussed in Chapter 6 and in Vol. II.

1.5 SOME MATHEMATICAL ISSUES

Let us now discuss some technical issues relating to the basic problem formulation and the validity of the DP algorithm. The reader who is not mathematically inclined need not be concerned about these issues and can skip this section without loss of continuity.

Once an admissible policy $\{\mu_0, \ldots, \mu_{N-1}\}$ is adopted, the following sequence of events is envisioned at the typical stage k:

1. The controller observes x_k and applies $u_k = \mu_k(x_k)$.
2. The disturbance w_k is generated according to the given distribution $P_k\big(\cdot \mid x_k, \mu_k(x_k)\big)$.
3. The cost $g_k\big(x_k, \mu_k(x_k), w_k\big)$ is incurred and added to previous costs.
4. The next state x_{k+1} is generated according to the system equation

$$x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big).$$

If this is the last stage ($k = N-1$), the terminal cost $g_N(x_N)$ is added to previous costs and the process terminates. Otherwise, k is incremented, and the same sequence of events is repeated at the next stage.

For each stage, the above process is well-defined and is couched in precise probabilistic terms. Matters are, however, complicated by the need to

view the cost as a well-defined random variable with well-defined expected value. The framework of probability theory requires that for each policy we define an underlying probability space, that is, a set Ω, a collection of events in Ω, and a probability measure on these events. In addition, the cost must be a well-defined random variable on this space in the sense of Appendix C (a measurable function from the probability space into the real line in the terminology of measure-theoretic probability theory). For this to be true, additional (measurability) assumptions on the functions f_k, g_k, and μ_k may be required, and it may be necessary to introduce additional structure on the spaces S_k, C_k, and D_k. Furthermore, these assumptions may restrict the class of admissible policies, since the functions μ_k may be constrained to satisfy additional (measurability) requirements.

Thus, unless these additional assumptions and structure are specified, the basic problem is formulated inadequately from a mathematical point of view. Unfortunately, a rigorous formulation for general state, control, and disturbance spaces is well beyond the mathematical framework of this introductory book and will not be undertaken here. Nonetheless, it turns out that these difficulties are mainly technical and do not substantially affect the basic results to be obtained. For this reason, we find it convenient to proceed with informal derivations and arguments; this is consistent with most of the literature on the subject.

We would like to stress, however, that under at least one frequently satisfied assumption, the mathematical difficulties mentioned above disappear. In particular, let us assume that the disturbance spaces D_k are all countable and the expected values of all terms in the cost are finite for every admissible policy (this is true in particular if the spaces D_k are finite sets). Then, for every admissible policy, the expected values of all the cost terms can be written as (possibly infinite) sums involving the probabilities of the elements of the spaces D_k.

Alternatively, one may write the cost as

$$J_\pi(x_0) = \mathop{E}_{x_1,\ldots,x_N} \left\{ g_N(x_N) + \sum_{k=0}^{N-1} \tilde{g}_k\big(x_k, \mu_k(x_k)\big) \right\}, \tag{1.15}$$

where

$$\tilde{g}_k\big(x_k, \mu_k(x_k)\big) = \mathop{E}_{w_k} \Big\{ g_k\big(x_k, \mu_k(x_k), w_k\big) \mid x_k, \mu_k(x_k) \Big\},$$

with the preceding expectation taken with respect to the distribution $P_k\big(\cdot \mid x_k, \mu_k(x_k)\big)$ defined on the countable set D_k. Then one may take as the basic probability space the Cartesian product of the spaces $\tilde{S}_k$, $k = 1, \ldots, N$, given for all k by

$$\tilde{S}_{k+1} = \big\{ x_{k+1} \in S_{k+1} \mid x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big),\ x_k \in \tilde{S}_k,\ w_k \in D_k \big\},$$

where $\tilde{S}_0 = \{x_0\}$. The set $\tilde{S}_k$ is the subset of all states that can be reached at time k when the policy $\{\mu_0, \ldots, \mu_{N-1}\}$ is used. Because the disturbance spaces D_k are countable, the sets $\tilde{S}_k$ are also countable (this is true since the union of any countable collection of countable sets is a countable set). The system equation $x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big)$, the probability distributions $P_k\big(\cdot \mid x_k, \mu_k(x_k)\big)$, the initial state x_0, and the policy $\{\mu_0, \ldots, \mu_{N-1}\}$ define a probability distribution on the countable set $\tilde{S}_1 \times \cdots \times \tilde{S}_N$, and the expected value in the cost expression (1.15) is defined with respect to this latter distribution.

Let us now give a more detailed proof of the validity of the DP algorithm (Prop. 1.3.1). We assume that the disturbance w_k takes a finite or countable number of values and the expected values of all terms in the expression of the cost function are finite for every admissible policy π. Furthermore, the functions $J_k(x_k)$ generated by the DP algorithm are finite for all states x_k and times k. We do not need to assume that the minimum over u_k in the definition of $J_k(x_k)$ is attained by some $u_k \in U(x_k)$.

For any admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ and each $k = 0, 1, \ldots, N-1$, denote $\pi^k = \{\mu_k, \mu_{k+1}, \ldots, \mu_{N-1}\}$. For $k = 0, 1, \ldots, N-1$, let $J_k^*(x_k)$ be the optimal cost for the $(N-k)$-stage problem that starts at state x_k and time k, and ends at time N; that is,

$$J_k^*(x_k) = \min_{\pi^k} E\left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right\}.$$

For $k = N$, we define $J_N^*(x_N) = g_N(x_N)$. We will show by induction that the functions J_k^* are equal to the functions J_k generated by the DP algorithm, so that for $k = 0$, we will obtain the desired result.

For any $\epsilon > 0$, and for all k and x_k, let $\mu_k^\epsilon(x_k)$ attain the minimum in the equation

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \underset{w_k}{E}\big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big)\big\}, \qquad k = 0, 1, \ldots, N-1, \tag{1.16}$$

within ϵ; that is, for all x_k and k, we have $\mu_k^\epsilon(x_k) \in U_k(x_k)$ and

$$\underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} \leq J_k(x_k) + \epsilon. \tag{1.17}$$

Let $J_k^\epsilon(x_k)$ be the expected cost starting at state x_k at time k, and using the policy $\{\mu_k^\epsilon, \mu_{k+1}^\epsilon, \ldots, \mu_{N-1}^\epsilon\}$. We will show that for all x_k and k, we have

$$J_k(x_k) \leq J_k^\epsilon(x_k) \leq J_k(x_k) + (N-k)\epsilon, \tag{1.18}$$

$$J_k^*(x_k) \leq J_k^\epsilon(x_k) \leq J_k^*(x_k) + (N-k)\epsilon, \tag{1.19}$$

$$J_k(x_k) = J_k^*(x_k). \tag{1.20}$$

It is seen using Eq. (1.17) that the inequalities (1.18) and (1.19) hold for $k = N-1$. By taking $\epsilon \to 0$ in Eqs. (1.18) and (1.19), it is also seen that $J_{N-1} = J_{N-1}^*$. Assume that Eqs. (1.18)-(1.20) hold for index $k+1$. We will show that they also hold for index k.

Indeed, we have

$$\begin{aligned} J_k^\epsilon(x_k) &= \underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}^\epsilon\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} \\ &\leq \underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} + (N-k-1)\epsilon \\ &\leq J_k(x_k) + \epsilon + (N-k-1)\epsilon \\ &= J_k(x_k) + (N-k)\epsilon, \end{aligned}$$

where the first equation holds by the definition of J_k^ϵ, the first inequality holds by the induction hypothesis, and the second inequality holds Eq. (1.17). We also have

$$\begin{aligned} J_k^\epsilon(x_k) &= \underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}^\epsilon\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} \\ &\geq \underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} \\ &\geq \min_{u_k \in U(x_k)} \underset{w_k}{E}\big\{ g_k\big(x_k, u_k, w_k\big) + J_{k+1}\big(f_k\big(x_k, u_k, w_k\big)\big)\big\} \\ &= J_k(x_k), \end{aligned}$$

where the first inequality holds by the induction hypothesis. Combining the preceding two relations, we see that Eq. (1.18) holds for index k.

For every policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$, we have

$$\begin{aligned} J_k^\epsilon(x_k) &= \underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}^\epsilon\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} \\ &\leq \underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big) + J_{k+1}\big(f_k\big(x_k, \mu_k^\epsilon(x_k), w_k\big)\big)\big\} + (N-k-1)\epsilon \\ &\leq \min_{u_k \in U(x_k)} \underset{w_k}{E}\big\{ g_k\big(x_k, u_k, w_k\big) + J_{k+1}\big(f_k\big(x_k, u_k, w_k\big)\big)\big\} + (N-k)\epsilon \\ &\leq \min_{u_k \in U(x_k)} \underset{w_k}{E}\big\{ g_k\big(x_k, u_k, w_k\big) + J_{k+1}\big(f_k\big(x_k, u_k, w_k\big)\big)\big\} + (N-k)\epsilon \\ &\leq \underset{w_k}{E}\big\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + J_{\pi^{k+1}}\big(f_k\big(x_k, \mu_k(x_k), w_k\big)\big)\big\} + (N-k)\epsilon \\ &= J_{\pi^k}(x_k) + (N-k)\epsilon, \end{aligned}$$

where the first inequality holds by the induction hypothesis, and the second inequality holds by Eq. (1.17). Taking the minimum over π^k in the preceding relation, we obtain for all x_k

$$J_k^\epsilon(x_k) \leq J_k^*(x_k) + (N-k)\epsilon.$$

We also have by the definition of J_k^*, for all x_k,

$$J_k^*(x_k) \leq J_k^\epsilon(x_k).$$

Combining the preceding two relations, we see that Eq. (1.19) holds for index k. Finally, Eq. (1.20) follows from Eqs. (1.18) and (1.19), by taking $\epsilon \to 0$, and the induction is complete.

Note that by using $\epsilon = 0$ in the relation

$$J_0^\epsilon(x_k) \leq J_0^*(x_k) + N\epsilon,$$

[cf. Eq. (1.19)], we see that a policy that attains the minimum for all x_k and k in Eq. (1.16) is optimal.

In conclusion, the basic problem has been formulated rigorously, and the DP algorithm has been proved rigorously only when the disturbance spaces $D_0, \ldots, D_{N-1}$ are countable sets, and the expected values of all the cost expressions associated with the problem and the DP algorithm are finite. In the absence of these assumptions, the reader should interpret subsequent results and conclusions as essentially correct but mathematically imprecise statements. In fact, when discussing infinite horizon problems (where the need for precision is greater), we will make the countability assumption explicit.

We note, however, that the advanced reader will have little difficulty in establishing most of our subsequent results concerning specific finite horizon applications, even if the countability assumption is not satisfied. This can be done by using the DP algorithm as a verification theorem. In particular, if one can find within a subset of policies $\tilde{\Pi}$ (such as those satisfying certain measurability restrictions) a policy that attains the minimum in the DP algorithm, then this policy can be readily shown to be optimal within $\tilde{\Pi}$. This result is developed in Exercise 1.12, and can be used by the mathematically oriented reader to establish rigorously many of our subsequent results concerning specific applications. For example, in linear-quadratic problems (Section 4.1) one determines from the DP algorithm a policy in closed form, which is linear in the current state. When w_k can take uncountably many values, it is necessary that admissible policies consist of Borel measurable functions μ_k. Since the linear policy obtained from the DP algorithm belongs to this class, the result of Exercise 1.12 guarantees that this policy is optimal. For a rigorous mathematical treatment of DP that resolves the associated measurability issues and supplements the present text, see the book by Bertsekas and Shreve [BeS78].

1.6 DYNAMIC PROGRAMMING AND MINIMAX CONTROL

The problem of optimal control of uncertain systems has traditionally been treated in a stochastic framework, whereby all uncertain quantities are described by probability distributions, and the expected value of the cost is

minimized. However, in many practical situations a stochastic description of the uncertainty may not be available, and one may have information with less detailed structure, such as bounds on the magnitude of the uncertain quantities. In other words, one may know a set within which the uncertain quantities are known to lie, but may not know the corresponding probability distribution. Under these circumstances one may use a minimax approach, whereby the worst possible values of the uncertain quantities within the given set are assumed to occur.

The minimax approach for decision making under uncertainty is described in Appendix G and is contrasted with the expected cost approach, which we have been following so far. In its simplest form, the corresponding decision problem is described by a triplet (Π, W, J), where Π is the set of policies under consideration, W is the set in which the uncertain quantities are known to belong, and $J : \Pi \times W \mapsto [-\infty, +\infty]$ is a given cost function. The objective is to

$$\text{minimize} \quad \max_{w \in W} J(\pi, w)$$

over all $\pi \in \Pi$.

It is possible to formulate a minimax counterpart to the basic problem with perfect state information. This problem is a special case of the abstract minimax problem above, as discussed more fully in Appendix G. Generally, it is unusual for even the simplest special cases of this problem to admit a closed-form solution. However, a computational solution using DP is possible, and our purpose in this section is to describe the corresponding algorithm.

In the framework of the basic problem, consider the case where the disturbances $w_0, w_1, \ldots, w_{N-1}$ do not have a probabilistic description but rather are known to belong to corresponding given sets $W_k(x_k, u_k) \subset D_k$, $k = 0, 1, \ldots, N-1$, which may depend on the current state x_k and control u_k. Consider the problem of finding a policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ with $\mu_k(x_k) \in U_k(x_k)$ for all x_k and k, which minimizes the cost function

$$J_\pi(x_0) = \max_{\substack{w_k \in W_k(x_k, \mu_k(x_k)) \\ k=0,1,\ldots,N-1}} \left[g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big) \right].$$

The DP algorithm for this problem takes the following form, which resembles the one corresponding to the stochastic basic problem (maximization is used in place of expectation):

$$J_N(x_N) = g_N(x_N), \tag{1.21}$$

$$J_k(x_k) = \min_{u_k \in U(x_k)} \max_{w_k \in W_k(x_k, u_k)} \Big[g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big]. \tag{1.22}$$

This algorithm can be explained by using a principle of optimality type of argument. In particular, we consider the tail subproblem whereby we are at state x_k at time k, and we wish to minimize the "cost-to-go"

$$\max_{\substack{w_i \in W_i(x_i, \mu_i(x_i)) \\ i=k,k+1,\ldots,N-1}} \left[g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right],$$

and we argue that if $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ is an optimal policy for the minimax problem, then the truncated policy $\{\mu_k^*, \mu_{k+1}^*, \ldots, \mu_{N-1}^*\}$ is optimal for the tail subproblem. The optimal cost of this subproblem is $J_k(x_k)$, as given by the DP algorithm (1.21)-(1.22). The algorithm expresses the intuitively clear fact that when at state x_k at time k, then regardless of what happened in the past, we should choose u_k that minimizes the worst/maximum value over w_k of the sum of the current stage cost plus the optimal cost of the tail subproblem that starts from the next state.

We will now give a mathematical proof that the DP algorithm (1.21)-(1.22) is valid, and that the optimal cost is equal to $J_0(x_0)$. For this it is necessary to assume that $J_k(x_k) > -\infty$ for all x_k and k. This is analogous to the assumption we made in the preceding section for the validity of the DP algorithm under stochastic disturbances, i.e., that the values $J_k(x_k)$ generated by the DP algorithm are finite for all states x_k and stages k. The following lemma provides the key argument.

Lemma 1.6.1: Let $f : W \to X$ be a function, and M be the set of all functions $\mu : X \to U$, where W, X, and U are some sets. Then for any functions $G_0 : W \to (-\infty, \infty]$ and $G_1 : X \times U \to (-\infty, \infty]$ such that

$$\min_{u \in U} G_1\big(f(w), u\big) > -\infty, \qquad \text{for all } w \in W,$$

we have

$$\min_{\mu \in M} \max_{w \in W} \Big[G_0(w) + G_1\Big(f(w), \mu\big(f(w)\big)\Big) \Big] = \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big].$$

Proof: We have for all $\mu \in M$

$$\max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] \geq \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big]$$

and by taking the minimum over $\mu \in M$, we obtain

$$\min_{\mu \in M} \max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] \geq \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big]. \tag{1.23}$$

To show the reverse inequality, for any $\epsilon > 0$, let $\mu_\epsilon \in M$ be such that

$$G_1\big(f(w), \mu_\epsilon(f(w))\big) \leq \min_{u \in U} G_1\big(f(w), u\big) + \epsilon, \qquad \text{for all } w \in W.$$

[Such a μ_ϵ exists because of the assumption $\min_{u \in U} G_1\big(f(w), u\big) > -\infty$.] Then

$$\begin{aligned} \min_{\mu \in M} \max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] &\leq \max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu_\epsilon(f(w))\big) \Big] \\ &\leq \max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big] + \epsilon. \end{aligned}$$

Since $\epsilon > 0$ can be taken arbitrarily small, we obtain the reverse to Eq. (1.23), and the desired result follows. **Q.E.D.**

To see how the conclusion of the lemma can fail without the condition $\min_{u \in U} G_1\big(f(w), u\big) > -\infty$ for all w, let u be a scalar, let $w = (w_1, w_2)$ be a two-dimensional vector, and let there be no constraints on u and w ($U = \Re$, $W = \Re \times \Re$, where $\Re$ is the real line). Let also

$$G_0(w) = w_1, \qquad f(w) = w_2, \qquad G_1\big(f(w), u\big) = f(w) + u.$$

Then, for all $\mu \in M$ we have,

$$\max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] = \max_{w_1 \in \Re,\ w_2 \in \Re} \big[w_1 + w_2 + \mu(w_2) \big] = \infty,$$

so that

$$\min_{\mu \in M} \max_{w \in W} \Big[G_0(w) + G_1\big(f(w), \mu(f(w))\big) \Big] = \infty.$$

On the other hand,

$$\max_{w \in W} \Big[G_0(w) + \min_{u \in U} G_1\big(f(w), u\big) \Big] = \max_{w_1 \in \Re,\ w_2 \in \Re} \Big[w_1 + \min_{u \in \Re} [w_2 + u] \Big] = -\infty,$$

since $\min_{u \in \Re} [w_2 + u] = -\infty$ for all w_2.

We now turn to proving the DP algorithm (1.21)-(1.22). The proof is similar to the one for the DP algorithm for stochastic problems. The optimal cost $J^*(x_0)$ of the problem is given by

$$\begin{aligned} J^*(x_0) = \min_{\mu_0} \cdots \min_{\mu_{N-1}} & \max_{w_0 \in W[x_0, \mu_0(x_0)]} \cdots \max_{w_{N-1} \in W[x_{N-1}, \mu_{N-1}(x_{N-1})]} \\ & \left[\sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big) + g_N(x_N) \right] \\ = \min_{\mu_0} \cdots \min_{\mu_{N-2}} & \left[\min_{\mu_{N-1}} \max_{w_0 \in W[x_0, \mu_0(x_0)]} \cdots \max_{w_{N-2} \in W[x_{N-2}, \mu_{N-2}(x_{N-2})]} \right. \\ & \left[\sum_{k=0}^{N-2} g_k\big(x_k, \mu_k(x_k), w_k\big) + \max_{w_{N-1} \in W[x_{N-1}, \mu_{N-1}(x_{N-1})]} \right. \\ & \left. \left. \Big[g_{N-1}\big(x_{N-1}, \mu_{N-1}(x_{N-1}), w_{N-1}\big) + J_N(x_N) \Big] \right] \right]. \end{aligned}$$

We can interchange the minimum over μ_{N-1} and the maximum over $w_0, \ldots, w_{N-2}$ by applying Lemma 1.6.1 with the identifications

$$w = (w_0, w_1, \ldots, w_{N-2}), \qquad u = u_{N-1}, \qquad f(w) = x_{N-1},$$

$$G_0(w) = \begin{cases} \sum_{k=0}^{N-2} g_k\big(x_k, \mu_k(x_k), w_k\big) & \text{if } w_k \in W_k\big(x_k, \mu_k(x_k)\big) \text{ for all } k, \\ \infty & \text{otherwise,} \end{cases}$$

$$G_1\big(f(w), u\big) = \begin{cases} \hat{G}_1\big(f(w), u\big) & \text{if } u \in U_{N-1}\big(f(w)\big), \\ \infty & \text{otherwise,} \end{cases}$$

where

$$\hat{G}_1\big(f(w), u\big) = \max_{w_{N-1} \in W_{N-1}\left(f(w), u\right)} \Big[g_{N-1}\big(f(w), u, w_{N-1}\big) + J_N\big(f_{N-1}(f(w), u, w_{N-1})\big)\Big],$$

to obtain

$$J^*(x_0) = \min_{\mu_0} \cdots \min_{\mu_{N-2}} \max_{w_0 \in W[x_0, \mu_0(x_0)]} \cdots \max_{w_{N-2} \in W[x_{N-2}, \mu_{N-2}(x_{N-2})]} \left[\sum_{k=0}^{N-2} g_k\big(x_k, \mu_k(x_k), w_k\big) + J_{N-1}(x_{N-1}) \right]. \tag{1.24}$$

The required condition $\min_{u \in U} G_1\big(f(w), u\big) > -\infty$ for all w (required for application of Lemma 1.6.1) is implied by the assumption $J_{N-1}(x_{N-1}) > -\infty$ for all x_{N-1}. Now, by working with the expression for $J^*(x_0)$ in Eq. (1.24), and by similarly continuing backwards, with $N-1$ in place of N, etc., after N steps we obtain $J^*(x_0) = J_0(x_0)$, which is the desired relation. The line of argument just given also shows that an optimal policy for the minimax problem can be constructed by minimizing in the right-hand side of the DP Eq. (1.22), similar to the case of the DP algorithm for the stochastic basic problem.

Unfortunately, as mentioned earlier, there are hardly any interesting examples of an analytical, closed-form solution of the DP algorithm (1.21)-(1.22). A computational solution, requires roughly comparable effort to the one of the stochastic DP algorithm. Instead of the expectation operation, one must carry out a maximization operation for each x_k and k.

Minimax control problems will be revisited in Chapter 4 in the context of reachability of target sets and target tubes (Section 4.6.2), and in Chapter 6 in the context of competitive games and computer chess (Section 6.3), and model predictive control (Section 6.5).

1.7 NOTES, SOURCES, AND EXERCISES

Dynamic programming is a simple mathematical technique that has been used for many years by engineers, mathematicians, and social scientists in a variety of contexts. It was Bellman, however, who realized in the early fifties that DP could be developed (in conjunction with the then appearing digital computer) into a systematic tool for optimization. In his influential books [Bel57], [BeD62], Bellman demonstrated the broad scope of DP and helped streamline its theory.

Following Bellman's works, there was much research on DP. In particular, the mathematical and algorithmic aspects of infinite horizon problems were extensively investigated, extensions to continuous-time problems were formulated and analyzed, and the mathematical issues discussed in Section 1.5, relating to the formulation of DP problems, were addressed. In addition, DP was used in a very broad variety of applications, ranging from many branches of engineering to statistics, economics, finance, and some of the social sciences. Samples of these applications will be given in subsequent chapters.

EXERCISES

1.1

Consider the system

$$x_{k+1} = x_k + u_k + w_k, \qquad k = 0, 1, 2, 3,$$

with initial state $x_0 = 5$, and the cost function

$$\sum_{k=0}^{3} (x_k^2 + u_k^2).$$

Apply the DP algorithm for the following three cases:

(a) The control constraint set $U_k(x_k)$ is $\{u \mid 0 \leq x_k + u \leq 5, u : \text{integer}\}$ for all x_k and k, and the disturbance w_k is equal to zero for all k.

(b) The control constraint and the disturbance w_k are as in part (a), but there is in addition a constraint $x_4 = 5$ on the final state. *Hint*: For this problem you need to define a state space for x_4 that consists of just the value $x_4 = 5$, and also to redefine $U_3(x_3)$. Alternatively, you may use a terminal cost $g_4(x_4)$ equal to a very large number for $x_4 \neq 5$.

(c) The control constraint is as in part (a) and the disturbance w_k takes the values -1 and 1 with equal probability 1/2 for all x_k and u_k, except if $x_k + u_k$ is equal to 0 or 5, in which case $w_k = 0$ with probability 1.

1.2

Carry out the calculations needed to verify that $J_0(1) = 2.67$ and $J_0(2) = 2.608$ in Example 1.3.2.

1.3

Suppose we have a machine that is either running or is broken down. If it runs throughout one week, it makes a gross profit of \$100. If it fails during the week, gross profit is zero. If it is running at the start of the week and we perform preventive maintenance, the probability that it will fail during the week is 0.4. If we do not perform such maintenance, the probability of failure is 0.7. However, maintenance will cost \$20. When the machine is broken down at the start of the week, it may either be repaired at a cost of \$40, in which case it will fail during the week with a probability of 0.4, or it may be replaced at a cost of \$150 by a new machine that is guaranteed to run through its first week of operation. Find the optimal repair, replacement, and maintenance policy that maximizes total profit over four weeks, assuming a new machine at the start of the first week.

1.4

A game of the blackjack variety is played by two players as follows: Both players throw a die. The first player, knowing his opponent's result, may stop or may throw the die again and add the result to the result of his previous throw. He then may stop or throw again and add the result of the new throw to the sum of his previous throws. He may repeat this process as many times as he wishes. If his sum exceeds seven (i.e., he busts), he loses the game. If he stops before exceeding seven, the second player takes over and throws the die successively until the sum of his throws is four or higher. If the sum of the second player is over seven, he loses the game. Otherwise the player with the larger sum wins, and in case of a tie the second player wins. The problem is to determine a stopping strategy for the first player that maximizes his probability of winning for each possible initial throw of the second player. Formulate the problem in terms of DP and find an optimal stopping strategy for the case where the second player's initial throw is three. *Hint*: Take $N = 6$ and a state space consisting of the following 15 states:

$$x^1 : \text{busted}$$

$$x^{1+i} : \text{already stopped at sum } i \ (1 \le i \le 7),$$

$$x^{8+i} : \text{current sum is } i \text{ but the player has not yet stopped } (1 \le i \le 7).$$

The optimal strategy is to throw until the sum is four or higher.

1.5 (Computer Assignment)

In the classical game of blackjack the player draws cards knowing only one card of the dealer. The player loses upon reaching a sum of cards exceeding 21. If the player stops before exceeding 21, the dealer draws cards until reaching 17 or higher. The dealer loses upon reaching a sum exceeding 21 or stopping at a lower sum than the player's. If player and dealer end up with an equal sum no one wins. In all other cases the dealer wins. An ace for the player may be counted as a 1 or an 11 as the player chooses. An ace for the dealer is counted as an 11 if this results in a sum from 17 to 21 and as a 1 otherwise. Jacks, queens, and kings count as 10 for both dealer and player. We assume an infinite card deck so the probability of a particular card showing up is independent of earlier cards.

(a) For every possible initial dealer card, calculate the probability that the dealer will reach a sum of 17, 18, 19, 20, 21, or over 21.

(b) Calculate the optimal choice of the player (draw or stop) for each of the possible combinations of dealer's card and player's sum of 12 to 20. Assume that the player's cards do not include an ace.

(c) Repeat part (b) for the case where the player's cards include an ace.

1.6 (Discounted Cost per Stage)

In the framework of the basic problem, consider the case where the cost is of the form

$$\underset{\substack{w_k \\ k=0,1,\ldots,N-1}}{E} \left\{ \alpha^N g_N(x_N) + \sum_{k=0}^{N-1} \alpha^k g_k(x_k, u_k, w_k) \right\},$$

where α is a discount factor with $0 < \alpha < 1$. Show that an alternate form of the DP algorithm is given by

$$V_N(x_N) = g_N(x_N),$$

$$V_k(x_k) = \min_{u_k \in U_k(x_k)} \underset{w_k}{E} \Big\{ g_k(x_k, u_k, w_k) + \alpha V_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}.$$

1.7 (Exponential Cost Function)

In the framework of the basic problem, consider the case where the cost is of the form

$$\underset{\substack{w_k \\ k=0,1,\ldots,N-1}}{E} \left\{ \exp\left(g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right) \right\}.$$

(a) Show that the optimal cost and an optimal policy can be obtained from the DP-like algorithm

$$J_N(x_N) = \exp\big(g_N(x_N)\big),$$

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \underset{w_k}{E} \Big\{ J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \exp\big(g_k(x_k, u_k, w_k)\big) \Big\}.$$

(b) Define the functions $V_k(x_k) = \ln J_k(x_k)$. Assume also that g_k is a function of x_k and u_k only (and not of w_k). Show that the above algorithm can be rewritten as

$$V_N(x_N) = g_N(x_N),$$

$$V_k(x_k) = \min_{u_k \in U_k(x_k)} \left\{ g_k(x_k, u_k) + \ln \underset{w_k}{E} \left\{ \exp\big(V_{k+1}\big(f_k(x_k, u_k, w_k)\big)\big) \right\} \right\}.$$

Note: The exponential cost function is an example of a *risk-sensitive cost function* that can be used to encode a preference for policies with a small variance of the cost $g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)$. The associated problems have a lot of interesting properties, which are discussed in several sources, e.g., Whittle [Whi90], Fernandez-Gaucherand and Markus [FeM94], James, Baras, and Elliott [JBE94], Basar and Bernhard [BaB95].

1.8 (Terminating Process)

In the framework of the basic problem, consider the case where the system evolution terminates at time i when a given value $\overline{w}_i$ of the disturbance at time i occurs, or when a termination decision u_i is made by the controller. If termination occurs at time i, the resulting cost is

$$T + \sum_{k=0}^{i} g_k(x_k, u_k, w_k),$$

where T is a termination cost. If the process has not terminated up to the final time N, the resulting cost is $g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)$. Reformulate the problem into the framework of the basic problem. *Hint*: Augment the state space with a special termination state.

1.9 (Multiplicative Cost)

In the framework of the basic problem, consider the case where the cost has the multiplicative form

$$\underset{\substack{w_k \\ k=0,1,\ldots,N-1}}{E} \Big\{ g_N(x_N) \cdot g_{N-1}(x_{N-1}, u_{N-1}, w_{N-1}) \cdots g_0(x_0, u_0, w_0) \Big\}.$$

Develop a DP-like algorithm for this problem assuming that $g_k(x_k, u_k, w_k) \geq 0$ for all x_k, u_k, w_k, and k.

1.10

Assume that we have a vessel whose maximum weight capacity is z and whose cargo is to consist of different quantities of N different items. Let v_i denote the value of the ith type of item, w_i the weight of ith type of item, and x_i the number of items of type i that are loaded in the vessel. The problem is to find the most valuable cargo, i.e., to maximize $\sum_{i=1}^{N} x_i v_i$ subject to the constraints $\sum_{i=1}^{N} x_i w_i \leq z$ and $x_i = 0, 1, 2, \ldots$ Formulate this problem in terms of DP.

1.11

Consider a device consisting of N stages connected in series, where each stage consists of a particular component. The components are subject to failure, and to increase the reliability of the device duplicate components are provided. For $j = 1, 2, \ldots, N$, let $(1 + m_j)$ be the number of components for the jth stage, let $p_j(m_j)$ be the probability of successful operation of the jth stage when $(1 + m_j)$ components are used, and let c_j denote the cost of a single component at the jth stage. Formulate in terms of DP the problem of finding the number of components at each stage that maximize the reliability of the device expressed by

$$p_1(m_1) \cdot p_2(m_2) \cdots p_N(m_N),$$

subject to the cost constraint $\sum_{j=1}^{N} c_j m_j \leq A$, where $A > 0$ is given.

1.12 (Minimization over a Subset of Policies)

This problem is primarily of theoretical interest (see the end of Section 1.5). Consider a variation of the basic problem whereby we seek

$$\min_{\pi \in \tilde{\Pi}} J_\pi(x_0),$$

where $\tilde{\Pi}$ is some given *subset* of the set of sequences $\{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ of functions $\mu_k : S_k \to C_k$ with $\mu_k(x_k) \in U_k(x_k)$ for all $x_k \in S_k$. Assume that

$$\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$$

belongs to $\tilde{\Pi}$ and attains the minimum in the DP algorithm; that is, for all $k = 0, 1, \ldots, N-1$ and $x_k \in S_k$,

$$\begin{aligned} J_k(x_k) &= \underset{w_k}{E} \Big\{ g_k\big(x_k, \mu_k^*(x_k), w_k\big) + J_{k+1}\big(f_k(x_k, \mu_k^*(x_k), w_k)\big) \Big\} \\ &= \min_{u_k \in U_k(x_k)} \underset{w_k}{E} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}, \end{aligned}$$

with $J_N(x_N) = g_N(x_N)$. Assume further that the functions J_k are real-valued and that the preceding expectations are well-defined and finite. Show that π^* is optimal within $\tilde{\Pi}$ and that the corresponding optimal cost is equal to $J_0(x_0)$.

1.13 (Semilinear Systems) WWW

Consider a problem involving the system

$$x_{k+1} = A_k x_k + f_k(u_k) + w_k,$$

where $x_k \in \Re^n$, f_k are given functions, and A_k and w_k are random $n \times n$ matrices and n-vectors, respectively, with given probability distributions that do not depend on x_k, u_k or prior values of A_k and w_k. Assume that the cost is of the form

$$\underset{\substack{A_k, w_k \\ k=0,1,\ldots,N-1}}{E} \left\{ c_N' x_N + \sum_{k=0}^{N-1} \Big(c_k' x_k + g_k\big(\mu_k(x_k)\big) \Big) \right\},$$

where c_k are given vectors and g_k are given functions. Show that if the optimal cost for this problem is finite and the control constraint sets $U_k(x_k)$ are independent of x_k, then the cost-to-go functions of the DP algorithm are affine (linear plus constant). Assuming that there is at least one optimal policy, show that there exists an optimal policy that consists of constant functions μ_k^*; that is, $\mu_k^*(x_k) =$ constant for all $x_k \in \Re^n$.

1.14

A farmer annually producing x_k units of a certain crop stores $(1 - u_k)x_k$ units of his production, where $0 \leq u_k \leq 1$, and invests the remaining $u_k x_k$ units, thus increasing the next year's production to a level x_{k+1} given by

$$x_{k+1} = x_k + w_k u_k x_k, \qquad k = 0, 1, \ldots, N-1.$$

The scalars w_k are independent random variables with identical probability distributions that do not depend either on x_k or u_k. Furthermore, $E\{w_k\} = \overline{w} > 0$. The problem is to find the optimal investment policy that maximizes the total expected product stored over N years

$$\underset{\substack{w_k \\ k=0,1,\ldots,N-1}}{E} \left\{ x_N + \sum_{k=0}^{N-1} (1 - u_k) x_k \right\}.$$

Show the optimality of the following policy that consists of constant functions:

(a) If $\overline{w} > 1$, $\mu_0^*(x_0) = \cdots = \mu_{N-1}^*(x_{N-1}) = 1$.

(b) If $0 < \overline{w} < 1/N$, $\mu_0^*(x_0) = \cdots = \mu_{N-1}^*(x_{N-1}) = 0$.

(c) If $1/N \leq \overline{w} \leq 1$,

$$\mu_0^*(x_0) = \cdots = \mu_{N-\overline{k}-1}(x_{N-\overline{k}-1}) = 1,$$

$$\mu_{N-\overline{k}}^*(x_{N-\overline{k}}) = \cdots = \mu_{N-1}^*(x_{N-1}) = 0,$$

where $\overline{k}$ is such that $1/(\overline{k}+1) < \overline{w} \leq 1/\overline{k}$.

1.15

Let x_k denote the number of educators in a certain country at time k and let y_k denote the number of research scientists at time k. New scientists (potential educators or research scientists) are produced during the kth period by educators at a rate γ_k per educator, while educators and research scientists leave the field due to death, retirement, and transfer at a rate δ_k. The scalars γ_k, $k = 0, 1, \ldots, N-1$, are independent identically distributed random variables taking values within a closed and bounded interval of positive numbers. Similarly δ_k, $k = 0, 1, \ldots, N-1$, are independent identically distributed and take values in an interval $[\delta, \delta']$ with $0 < \delta \leq \delta' < 1$. By means of incentives, a science policy maker can determine the proportion u_k of new scientists produced at time k who become educators. Thus, the number of research scientists and educators evolves according to the equations

$$x_{k+1} = (1 - \delta_k)x_k + u_k \gamma_k x_k,$$

$$y_{k+1} = (1 - \delta_k)y_k + (1 - u_k)\gamma_k x_k.$$

The initial numbers x_0, y_0 are known, and it is required to find a policy

$$\big\{\mu_0^*(x_0, y_0), \ldots, \mu_{N-1}^*(x_{N-1}, y_{N-1})\big\}$$

with

$$0 < \alpha \leq \mu_k^*(x_k, y_k) \leq \beta < 1, \qquad \text{for all } x_k, y_k, \text{ and } k,$$

which maximizes $E_{\gamma_k, \delta_k}\{y_N\}$ (i.e., the expected final number of research scientists after N periods). The scalars α and β are given.

(a) Show that the cost-to-go functions $J_k(x_k, y_k)$ are linear; that is, for some scalars ξ_k, ζ_k,

$$J_k(x_k, y_k) = \xi_k x_k + \zeta_k y_k.$$

(b) Derive an optimal policy $\{\mu_0^*, \ldots, \mu_{N-1}^*\}$ under the assumption

$$E\{\gamma_k\} > E\{\delta_k\}$$

and show that this optimal policy can consist of constant functions.

(c) Assume that the proportion of new scientists who become educators at time k is $u_k + \epsilon_k$ (rather than u_k), where ϵ_k are identically distributed independent random variables that are also independent of γ_k, δ_k and take values in the interval $[-\alpha, 1-\beta]$. Derive the form of the cost-to-go functions and the optimal policy.

1.16 WWW

Given a sequence of matrix multiplications

$$M_1 M_2 \cdots M_k M_{k+1} \cdots M_N,$$

where each M_k is a matrix of dimension $n_k \times n_{k+1}$, the order in which multiplications are carried out can make a difference. For example, if $n_1 = 1$, $n_2 = 10$, $n_3 = 1$, and $n_4 = 10$, the calculation $\big((M_1M_2)M_3\big)$ requires 20 scalar multiplications, but the calculation $\big(M_1(M_2M_3)\big)$ requires 200 scalar multiplications (multiplying an $m \times n$ matrix with an $n \times k$ matrix requires mnk scalar multiplications).

(a) Derive a DP algorithm for finding the optimal multiplication order [any order is allowed, including orders that involve multiple partial products each consisting of two or more adjacent matrices, e.g., $\big((M_1M_2)(M_3M_4)\big)$]. Solve the problem for $N = 3$, $n_1 = 2$, $n_2 = 10$, $n_3 = 5$, and $n_4 = 1$.

(b) Derive a DP algorithm for finding the optimal multiplication order within the class of orders where at each step, we maintain only one partial product that consists only of adjacent matrices, e.g., $\big(\big(M_1(M_2M_3)\big)M_4\big)$.

1.17

The paragraphing problem deals with breaking up a sequence of N words of given lengths into lines of length A. Let $w_1, \ldots, w_N$ be the words and let $L_1, \ldots, L_N$ be their lengths. In a simple version of the problem, words are separated by blanks whose ideal width is b, but blanks can stretch or shrink if necessary, so that a line $w_i, w_{i+1}, \ldots, w_{i+k}$ has length exactly A. The cost associated with the line is $(k+1)|b' - b|$, where $b' = (A - L_i - \cdots - L_{i+k})/(k+1)$ is the actual average width of the blanks, except if we have the last line ($N = i+k$), in which case the cost is zero when $b' \geq b$. Formulate a DP algorithm for finding the minimum cost separation. *Hint*: Consider the subproblems of optimally separating $w_i, \ldots, w_N$ for $i = 1, \ldots, N$.

1.18 [Shr81] (WWW)

A decision maker must continually choose between two activities over a time interval $[0, T]$. Choosing activity i at time t, where $i = 1, 2$, earns reward at a rate $g_i(t)$, and every switch between the two activities costs $c > 0$. Thus, for example, the reward for starting with activity 1, switching to 2 at time t_1, and switching back to 1 at time $t_2 > t_1$ earns total reward

$$\int_0^{t_1} g_1(t)\,dt + \int_{t_1}^{t_2} g_2(t)\,dt + \int_{t_2}^{T} g_1(t)\,dt - 2c.$$

We want to find a set of switching times that maximize the total reward. Assume that the function $g_1(t) - g_2(t)$ changes sign a finite number of times in the interval $[0, T]$. Formulate the problem as a finite horizon problem and write the corresponding DP algorithm.

1.19 (Games)

(a) Consider a smaller version of a popular puzzle game. Three square tiles numbered 1, 2, and 3 are placed in a 2×2 grid with one space left empty. The two tiles adjacent to the empty space can be moved into that space, thereby creating new configurations. Use a DP argument to answer the question whether it is possible to generate a given configuration starting from any other configuration.

(b) From a pile of eleven matchsticks, two players take turns removing one or four sticks. The player who removes the last stick wins. Use a DP argument to show that there is a winning strategy for the player who plays first.

1.20 (The Counterfeit Coin Problem)

We are given six coins, one of which is counterfeit and is known to have different weight than the rest. Construct a strategy to find the counterfeit coin using a two-pan scale in a minimum average number of tries. *Hint*: There are two initial decisions that make sense: (1) test two of the coins against two others, and (2) test one of the coins against one other.

1.21 (Regular Polygon Theorem) (WWW)

According to a famous theorem (attributed to the ancient Greek geometer Zenodorus), of all N-side polygons inscribed in a given circle, those that are regular (all sides are equal) have maximal area.

(a) Prove the theorem by applying DP to a suitable problem involving sequential placement of N points in the circle.

(b) Use DP to solve the problem of placing a given number of points on a subarc of the circle, so as to maximize the area of the polygon whose vertices are these points, the endpoints of the subarc, and the center of the circle.

1.22 (Inscribed Polygon of Maximal Perimeter)

Consider the problem of inscribing an N-side polygon in a given circle, so that the polygon has maximal perimeter.

(a) Formulate the problem as a DP problem involving sequential placement of N points in the circle.

(b) Use DP to show that the optimal polygon is regular (all sides are equal).

1.23 (Monotonicity Property of DP)

An evident, yet very important property of the DP algorithm is that if the terminal cost g_N is changed to a uniformly larger cost $\bar{g}_N$ [i.e., $g_N(x_N) \leq \bar{g}_N(x_N)$ for all x_N], then the last stage cost-to-go $J_{N-1}(x_{N-1})$ will be uniformly increased. More generally, given two functions J_{k+1} and $\bar{J}_{k+1}$ with $J_{k+1}(x_{k+1}) \leq \bar{J}_{k+1}(x_{k+1})$ for all x_{k+1}, we have, for all x_k and $u_k \in U_k(x_k)$,

$$\begin{aligned} &\mathop{E}_{w_k}\Big\{g_k(x_k,u_k,w_k) + J_{k+1}\big(f_k(x_k,u_k,w_k)\big)\Big\} \\ &\qquad \leq \mathop{E}_{w_k}\Big\{g_k(x_k,u_k,w_k) + \bar{J}_{k+1}\big(f_k(x_k,u_k,w_k)\big)\Big\}. \end{aligned}$$

Suppose now that in the basic problem the system and cost are time invariant; that is, $S_k \equiv S$, $C_k \equiv C$, $D_k \equiv D$, $f_k \equiv f$, $U_k \equiv U$, $P_k \equiv P$, and $g_k \equiv g$ for some S, C, D, f, U, P, and g. Show that if in the DP algorithm we have $J_{N-1}(x) \leq J_N(x)$ for all $x \in S$, then

$$J_k(x) \leq J_{k+1}(x), \qquad \text{for all } x \in S \text{ and } k.$$

Similarly, if we have $J_{N-1}(x) \geq J_N(x)$ for all $x \in S$, then

$$J_k(x) \geq J_{k+1}(x), \qquad \text{for all } x \in S \text{ and } k.$$

1.24 (Traveling Repairman Problem)

A repairman must service n sites, which are located along a line and are sequentially numbered $1, 2, \ldots, n$. The repairman starts at a given site s with $1 < s < n$, and is constrained to service only sites that are adjacent to the ones serviced so far, i.e., if he has already serviced sites $i, i+1, \ldots, j$, then he may service next only site $i-1$ (assuming $1 < i$) or site $j+1$ (assuming $j < n$). There is a waiting cost c_i for each time period that site i has remained unserviced and there is a travel cost t_{ij} for servicing site j immediately after servicing site i. Formulate a DP algorithm for finding a minimum cost service schedule.

1.25 (WWW)

An unscrupulous innkeeper charges a different rate for a room as the day progresses, depending on whether he has many or few vacancies. His objective is to maximize his expected total income during the day. Let x be the number of empty rooms at the start of the day, and let y be the number of customers that will ask for a room in the course of the day. We assume (somewhat unrealistically) that the innkeeper knows y with certainty, and upon arrival of a customer, quotes one of m prices r_i, $i = 1, \ldots, m$, where $0 < r_1 \leq r_2 \leq \cdots \leq r_m$. A quote of a rate r_i is accepted with probability p_i and is rejected with probability $1 - p_i$, in which case the customer departs, never to return during that day.

(a) Formulate this as a problem with y stages and show that the maximal expected income, as a function of x and y, satisfies the recursion

$$J(x,y) = \max_{i=1,\ldots,m}\Big[p_i\big(r_i + J(x-1,y-1)\big) + (1-p_i)J(x,y-1)\Big],$$

for all $x \geq 1$ and $y \geq 1$, with initial conditions

$$J(x,0) = J(0,y) = 0, \qquad \text{for all } x \text{ and } y.$$

Assuming that the product $p_i r_i$ is monotonically nondecreasing with i, and that p_i is monotonically nonincreasing with i, show that the innkeeper should always charge the highest rate r_m.

(b) Consider a variant of the problem where each arriving customer, with probability p_i, offers a price r_i for a room, which the innkeeper may accept or reject, in which case the customer departs, never to return during that day. Show that an appropriate DP algorithm is

$$J(x,y) = \sum_{i=1}^{m} p_i \max\big[r_i + J(x-1,y-1),\, J(x,y-1)\big],$$

with initial conditions

$$J(x,0) = J(0,y) = 0, \qquad \text{for all } x \text{ and } y.$$

Show also that for given x and y it is optimal to accept a customer's offer if it is larger than some threshold $\bar{r}(x,y)$. *Hint*: This part is related to DP for uncontrollable state components (cf. Section 1.4).

1.26 (Investing in a Stock) (WWW)

An investor observes at the beginning of each period k the price x_k of a stock and decides whether to buy 1 unit, sell 1 unit, or do nothing. There is a transaction cost c for buying or selling. The stock price can take one of n different values $v^1, \ldots, v^n$ and the transition probabilities $p_{ij}^k = P\{x_{k+1} = v^j \mid x_k = v^i\}$ are known. The investor wants to maximize the total worth of his stock at a fixed final period N minus his investment costs from period 0 to period $N-1$ (revenue from a sale is viewed as negative cost). We assume that the function

$$P_k(x) = E\{x_N \mid x_k = x\} - x$$

is monotonically nonincreasing as a function of x; that is, the expected profit from a purchase is a nonincreasing function of the purchase price.

(a) Assume that the investor starts with N or more units of stock and an unlimited amount of cash, so that a purchase or sale decision is possible at each period regardless of the past decisions and the current price. For every period k, let $\underline{x}_k$ be the largest value of $x \in \{v^1, \ldots, v^n\}$ such that

$P_k(x) > c$, and let $\overline{x}_k$ be the smallest value of $x \in \{v^1, \ldots, v^n\}$ such that $P_k(x) < -c$. Show that it is optimal to buy if $x_k \leq \underline{x}_k$, sell if $\overline{x}_k \leq x_k$, and do nothing otherwise. *Hint*: Formulate the problem as one of maximizing

$$E\left\{\sum_{k=0}^{N-1}\big(u_k P_k(x_k) - c\,|u_k|\big)\right\},$$

where $u_k \in \{-1, 0, 1\}$.

(b) Formulate an efficient DP algorithm for the case where the investor starts with less than N units of stock and an unlimited amount of cash. Show that it is still optimal to buy if $x_k \leq \underline{x}_k$ and it is still not optimal to sell if $x_k < \overline{x}_k$. Could it be optimal to buy at any prices x_k greater than $\underline{x}_k$?

(c) Consider the situation where the investor initially has N or more units of stock and there is a constraint that for any time k the number of purchase decisions up to k should not exceed the number of sale decisions up to k by more that a given fixed number m (this models approximately the situation where the investor has a limited initial amount of cash). Formulate an efficient DP algorithm for this case. Show that it is still optimal to sell if $\overline{x}_k \leq x_k$ and it is still not optimal to buy if $\underline{x}_k < x_k$.

(d) Consider the situation where there are restrictions on both the initial amount of stock as in part (b), and the number of purchase decisions as in part (c). Derive a DP algorithm for this problem.

(e) How would the analysis of (a)-(d) be affected if cash is invested at a given fixed interest rate?

2

Deterministic Systems and the Shortest Path Problem

Contents

In this chapter, we focus on deterministic problems, that is, problems where each disturbance w_k can take only one value. Such problems arise in many important contexts and they also arise in cases where the problem is really stochastic but, as an approximation, the disturbance is fixed at some typical value; see Chapter 6.

An important property of deterministic problems is that, in contrast with stochastic problems, *using feedback results in no advantage in terms of cost reduction*. In other words, minimizing the cost over admissible policies $\{\mu_0, \ldots, \mu_{N-1}\}$ results in the same optimal cost as minimizing over *sequences of control vectors* $\{u_0, \ldots, u_{N-1}\}$. This is true because given a policy $\{\mu_0, \ldots, \mu_{N-1}\}$ and the initial state x_0, the future states are perfectly predictable through the equation

$$x_{k+1} = f_k\big(x_k, \mu_k(x_k)\big), \qquad k = 0, 1, \ldots, N-1,$$

and the corresponding controls are perfectly predictable through the equation

$$u_k = \mu_k(x_k), \qquad k = 0, 1, \ldots, N-1.$$

Thus, the cost achieved by an admissible policy $\{\mu_0, \ldots, \mu_{N-1}\}$ for a deterministic problem is also achieved by the control sequence $\{u_0, \ldots, u_{N-1}\}$ defined above. As a result, we may restrict attention to sequences of controls without loss of optimality.

The difference just discussed between deterministic and stochastic problems often has important computational implications. In particular, in a deterministic problem with a "continuous space" character (states and controls are Euclidean vectors), optimal control sequences may be found by deterministic variational techniques to be discussed in Chapter 3, and by widely used iterative optimal control algorithms such as steepest descent, conjugate gradient, and Newton's method (see e.g., nonlinear programming texts such as Bertsekas [Ber99] or Luenberger [Lue84]). These algorithms, when applicable, are usually more efficient than DP. On the other hand, DP *has a wider scope of applicability since it can handle difficult constraint sets such as integer or discrete sets*. Furthermore, DP *leads to a globally optimal solution* as opposed to variational techniques, for which this cannot be guaranteed in general.

In this chapter, we consider deterministic problems with a discrete character for which variational optimal control techniques are inapplicable, so that specialized forms of DP are the principal solution methods.

2.1 FINITE-STATE SYSTEMS AND SHORTEST PATHS

Consider a deterministic problem where the state space S_k is a finite set for each k. Then at any state x_k, a control u_k can be associated with a

transition from the state x_k to the state $f_k(x_k, u_k)$, at a cost $g_k(x_k, u_k)$. Thus a finite-state deterministic problem can be equivalently represented by a graph such as the one of Fig. 2.1.1, where the arcs correspond to transitions between states at successive stages and each arc has an associated cost. To handle the final stage, an artificial terminal node t has been added. Each state x_N at stage N is connected to the terminal node t with an arc having cost $g_N(x_N)$. Control sequences correspond to paths originating at the initial state (node s at stage 0) and terminating at one of the nodes corresponding to the final stage N. If we view the cost of an arc as its length, we see that *a deterministic finite-state problem is equivalent to finding a minimum-length (or shortest) path from the initial node s of the graph to the terminal node t*. Here, by a path we mean a sequence of arcs of the form $(j_1, j_2), (j_2, j_3), \ldots, (j_{k-1}, j_k)$, and by the length of a path we mean the sum of the lengths of its arcs.

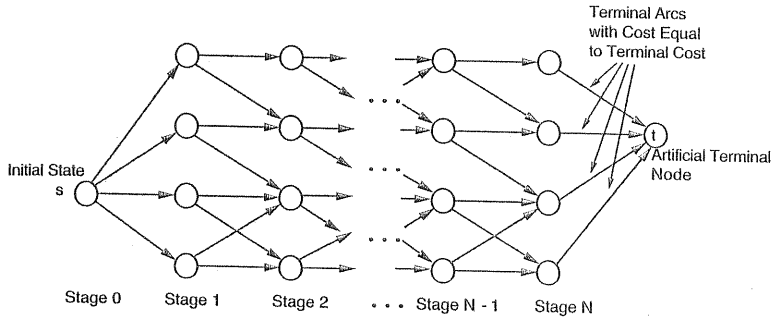


Figure 2.1.1 Transition graph for a deterministic finite-state system. Nodes correspond to states. An arc with start and end nodes x_k and x_{k+1}, respectively, corresponds to a transition of the form $x_{k+1} = f_k(x_k, u_k)$. We view the cost $g_k(x_k, u_k)$ of the transition as the length of this arc. The problem is equivalent to finding a shortest path from the initial node s to the terminal node t.

Let us denote

$$a_{ij}^k = \text{ Cost of transition at stage } k \text{ from state } i \in S_k \text{ to state } j \in S_{k+1},$$

$$a_{it}^N = \text{ Terminal cost of state } i \in S_N \text{ [which is } g_N(i)\text{]},$$

where we adopt the convention $a_{ij}^k = \infty$ if there is no control that moves the state from i to j at stage k. The DP algorithm takes the form

$$J_N(i) = a_{it}^N, \qquad i \in S_N, \tag{2.1}$$

$$J_k(i) = \min_{j \in S_{k+1}} \big[a_{ij}^k + J_{k+1}(j)\big], \qquad i \in S_k, \qquad k = 0, 1, \ldots, N-1. \tag{2.2}$$

The optimal cost is $J_0(s)$ and is equal to the length of the shortest path from s to t.

A Forward DP Algorithm

The preceding algorithm proceeds *backward* in time. It is possible to derive an equivalent algorithm that proceeds *forward* in time by means of the following simple observation. An optimal path from s to t is also an optimal path from t to s in a "reverse" shortest path problem where the direction of each arc is reversed and its length is left unchanged. The DP algorithm corresponding to this "reverse" problem starts from the states $x_1 \in S_1$ of stage 1, proceeds to states $x_2 \in S_2$ of stage 2, and continues all the way to states $x_N \in S_N$ of stage N. It is given by

$$\tilde{J}_N(j) = a^0_{sj}, \qquad j \in S_1, \tag{2.3}$$

$$\tilde{J}_k(j) = \min_{i \in S_{N-k}} \left[a^{N-k}_{ij} + \tilde{J}_{k+1}(i) \right], \qquad j \in S_{N-k+1}, \quad k = 1, 2, \ldots, N-1. \tag{2.4}$$

The optimal cost is

$$\tilde{J}_0(t) = \min_{i \in S_N} \left[a^N_{it} + \tilde{J}_1(i) \right].$$

The backward algorithm (2.1)-(2.2) and the forward algorithm (2.3)-(2.4) yield the same result in the sense that

$$J_0(s) = \tilde{J}_0(t),$$

and an optimal control sequence (or shortest path) obtained from any one of the two is optimal for the original problem. We may view $\tilde{J}_k(j)$ in Eq. (2.4) as an *optimal cost-to-arrive* to state j from the initial state s. This should be contrasted with $J_k(i)$ in Eq. (2.2), which represents the optimal cost-to-go from state i to the terminal state t.

An important use of the forward DP algorithm arises in real-time applications where the stage k problem data are unknown prior to stage k, and are revealed to the controller just before stage k begins. An example will be given in connection with the state estimation of Hidden Markov Models in Section 2.2.2. Note that to derive the forward DP algorithm, we used the shortest path formulation, which is available only for deterministic problems. Indeed, for stochastic problems, there is no analog of the forward DP algorithm.

In conclusion, *a deterministic finite-state problem is equivalent to a special type of shortest path problem and can be solved by either the ordinary (backward) DP algorithm or by an alternative forward DP algorithm.* It is also interesting to note that *any shortest path problem can be posed as a deterministic finite-state DP problem*, as we now show.

Converting a Shortest Path Problem to a Deterministic Finite-State Problem

Let $\{1, 2, \ldots, N, t\}$ be the set of nodes of a graph, and let a_{ij} be the cost of moving from node i to node j (also referred to as the *length* of the arc joining i and j). Node t is a special node, which we call the *destination*. We allow the possibility $a_{ij} = \infty$ to account for the case where there is no arc joining nodes i and j. We want to find a shortest path from each node i to node t, i.e., a sequence of moves that minimizes total cost to get to t from each of the nodes $1, 2, \ldots, N$.

For the problem to have a solution, it is necessary to make an assumption relating to cycles, i.e., paths of the form $(i, j_1), (j_1, j_2), \ldots, (j_k, i)$ that start and end at the same node. We must exclude the possibility that a cycle has negative total length. Otherwise, it would be possible to decrease the length of some paths to arbitrarily small values simply by adding more and more negative-length cycles. We thus assume that *all cycles have nonnegative length.* With this assumption, it is clear that an optimal path need not take more than N moves, so we may limit the number of moves to N. We formulate the problem as one where *we require exactly N moves but allow degenerate moves from a node i to itself with cost $a_{ii} = 0$.* We denote for $i = 1, \ldots, N$, $k = 0, \ldots, N-1$,

$$J_k(i) = \text{optimal cost of getting from } i \text{ to } t \text{ in } N-k \text{ moves.}$$

Then the cost of the optimal path from i to t is $J_0(i)$.

It is possible to formulate this problem within the framework of the basic problem and subsequently apply the DP algorithm. For simplicity, however, we write directly the DP equation, which takes the intuitively clear form

$$\text{optimal cost from } i \text{ to } t \text{ in } N-k \text{ moves}$$
$$= \min_{j=1,\ldots,N} \left[a_{ij} + (\text{optimal cost from } j \text{ to } t \text{ in } N-k-1 \text{ moves}) \right],$$

or

$$J_k(i) = \min_{j=1,\ldots,N} \left[a_{ij} + J_{k+1}(j) \right], \qquad k = 0, 1, \ldots, N-2,$$

with

$$J_{N-1}(i) = a_{it}, \qquad i = 1, 2, \ldots, N.$$

The optimal policy when at node i after k moves is to move to a node j^* that minimizes $a_{ij} + J_{k+1}(j)$ over all $j = 1, \ldots, N$. If the optimal path obtained from the algorithm contains degenerate moves from a node to itself, this simply means that the path involves in reality less than N moves.

Note that if for some $k > 0$, we have $J_k(i) = J_{k+1}(i)$ for all i, then subsequent DP iterations will not change the values of the cost-to-go

$[J_{k-m}(i) = J_k(i)$ for all $m > 0$ and $i]$, so the algorithm can be terminated with $J_k(i)$ being the shortest distance from i to t, for all i.

To demonstrate the algorithm, consider the problem shown in Fig. 2.1.2(a) where the costs a_{ij} with $i \neq j$ are shown along the connecting line segments (we assume $a_{ij} = a_{ji}$). Figure 2.1.2(b) shows the cost-to-go $J_k(i)$ at each i and k together with the optimal paths.

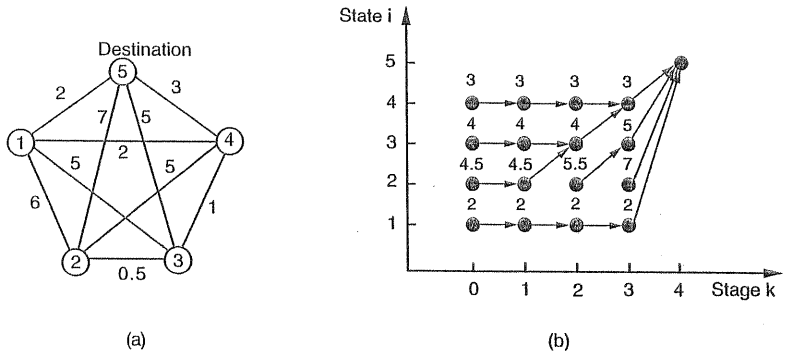


Figure 2.1.2 (a) Shortest path problem data. The destination is node 5. Arc lengths are equal in both directions and are shown along the line segments connecting nodes. (b) Costs-to-go generated by the DP algorithm. The number along stage k and state i is $J_k(i)$. Arrows indicate the optimal moves at each stage and node. The optimal paths are

$$1 \to 5, \quad 2 \to 3 \to 4 \to 5, \quad 3 \to 4 \to 5, \quad 4 \to 5.$$

2.2 SOME SHORTEST PATH APPLICATIONS

The shortest path problem appears in many diverse contexts. We provide some examples.

2.2.1 Critical Path Analysis

Consider the planning of a project involving several activities, some of which must be completed before others can begin. The duration of each activity is known in advance. We want to find the time required to complete the project, as well as the *critical* activities, those that even if slightly delayed will result in a corresponding delay of completion of the overall project.

The problem can be represented by a graph with nodes $1, \ldots, N$ such as the one shown in Fig. 2.2.1. Here nodes represent completion of some

phase of the project. An arc (i, j) represents an activity that starts once phase i is completed and has known duration $t_{ij} > 0$. A phase (node) j is completed when all activities or arcs (i, j) that are incoming to j are completed. The special nodes 1 and N represent the start and end of the project. Node 1 has no incoming arcs, while node N has no outgoing arcs. Furthermore, there is at least one path from node 1 to every other node.

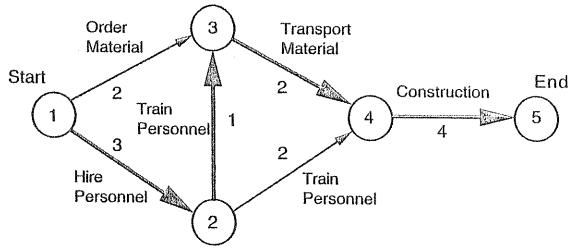


Figure 2.2.1 Graph of an activity network. Arcs represent activities and are labeled by the corresponding duration. Nodes represent completion of some phase of the project. A phase is completed if all activities associated with incoming arcs at the corresponding node are completed. The project is completed when all phases are completed. The project duration time is the length of the longest path from node 1 to node 5, which is shown with thick line.

An important characteristic of an activity network is that it is *acyclic*; that is, it has no cycles. This is inherent in the problem formulation and the interpretation of nodes as phase completions.

For any path $p = \{(1, j_1), (j_1, j_2,), \ldots, (j_k, i)\}$ from node 1 to a node i, let D_p be the duration of the path defined as the sum of durations of its activities; that is,

$$D_p = t_{1j_1} + t_{j_1 j_2} + \cdots + t_{j_k i}.$$

Then the time T_i required to complete phase i is

$$T_i = \max_{\substack{\text{paths } p \\ \text{from 1 to } i}} D_p.$$

Thus to find T_i, we should find the *longest* path from 1 to i. This problem may also be viewed as a shortest path problem with the length of each arc (i, j) being $-t_{ij}$. In particular, finding the duration of the project is equivalent to finding the shortest path from 1 to N. This path is also called a *critical* path. It can be seen that a delay by a given amount in the completion of one of the activities on the critical path will delay the completion of the overall project by the same amount. Note that because the network is acyclic, there can be only a finite number of paths from 1 to any i, so that at least one of these paths corresponds to the maximal path duration T_i.

Let us denote by S_1 the set of phases that do not depend on completion of any other phase, and more generally, for $k = 1, 2, \ldots$, let S_k be the set

$$S_k = \{i \mid \text{all paths from 1 to } i \text{ have } k \text{ arcs or less}\},$$

with $S_0 = \{1\}$. The sets S_k can be viewed as the state spaces for the equivalent DP problem. Using maximization in place of minimization while changing the sign of the arc lengths, the DP algorithm can be written as

$$T_i = \max_{\substack{(j,i) \text{ such that} \\ j \in S_{k-1}}} \big[t_{ji} + T_j\big], \qquad \text{for all } i \in S_k \text{ with } i \notin S_{k-1}.$$

Note that this is a forward algorithm; that is, it starts at the origin 1 and proceeds towards the destination N. An alternative backward algorithm, which starts at N and proceeds towards 1, is also possible, as discussed in the preceding section.

As an example, for the activity network of Fig. 2.2.1, we have

$$S_0 = \{1\}, \quad S_1 = \{1, 2\}, \quad S_2 = \{1, 2, 3\},$$

$$S_3 = \{1, 2, 3, 4\}, \quad S_4 = \{1, 2, 3, 4, 5\}.$$

A calculation using the preceding formula yields

$$T_1 = 0,\ T_2 = 3,\ T_3 = 4,\ T_4 = 6,\ T_5 = 10.$$

The critical path is $1 \to 2 \to 3 \to 4 \to 5$.

2.2.2 Hidden Markov Models and the Viterbi Algorithm

Consider a Markov chain with a finite number of states and given state transition probabilities p_{ij}. Suppose that when a transition occurs, the states corresponding to the transition are unknown (or "hidden") to us, but instead we obtain an observation that relates to that transition. Given a sequence of observations, we want to estimate in some optimal sense the sequence of corresponding transitions. We are given the probability $r(z; i, j)$ of an observation taking value z when the state transition is from i to j. We assume independent observations; that is, an observation depends only on its corresponding transition and not on other transitions. We are also given the probability π_i that the initial state takes value i. The probabilities p_{ij} and $r(z; i, j)$ are assumed to be independent of time for notational convenience. The methodology to be described admits a straightforward extension to the case of time-varying system and observation probabilities.

Markov chains whose state transitions are imperfectly observed according to the probabilistic mechanism just described are called *Hidden Markov Models (HMMs for short)* or *partially observable Markov chains*.

In Chapter 5 we will discuss the control of such Markov chains in the context of stochastic optimal control problems with imperfect state information. In the present section, we will focus on the problem of estimating the state sequence given a corresponding observation sequence. This is an important problem that arises in a broad variety of practical contexts.

We use a "most likely state" estimation criterion, whereby given the observation sequence $Z_N = \{z_1, z_2, \ldots, z_N\}$, we adopt as our estimate the state transition sequence $\hat{X}_N = \{\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_N\}$ that maximizes over all $X_N = \{x_0, x_1, \ldots, x_N\}$ the conditional probability $p(X_N \mid Z_N)$. We will show that $\hat{X}_N$ can be found by solving a special type of shortest path problem that involves an acyclic graph.

We have

$$p(X_N \mid Z_N) = \frac{p(X_N, Z_N)}{p(Z_N)},$$

where $p(X_N, Z_N)$ and $p(Z_N)$ are the unconditional probabilities of occurrence of (X_N, Z_N) and Z_N, respectively. Since $p(Z_N)$ is a positive constant once Z_N is known, we can maximize $p(X_N, Z_N)$ in place of $p(X_N \mid Z_N)$. The probability $p(X_N, Z_N)$ can be written as

$$\begin{aligned} p(X_N, Z_N) &= p(x_0, x_1, \ldots, x_N, z_1, z_2, \ldots, z_N) \\ &= \pi_{x_0} p(x_1, \ldots, x_N, z_1, z_2, \ldots, z_N \mid x_0) \\ &= \pi_{x_0} p(x_1, z_1 \mid x_0) p(x_2, \ldots, x_N, z_2, \ldots, z_N \mid x_0, x_1, z_1) \\ &= \pi_{x_0} p_{x_0 x_1} r(z_1; x_0, x_1) p(x_2, \ldots, x_N, z_2, \ldots, z_N \mid x_0, x_1, z_1). \end{aligned}$$

This calculation can be continued by writing

$$\begin{aligned} &p(x_2, \ldots, x_N, z_2, \ldots, z_N \mid x_0, x_1, z_1) \\ &\quad = p(x_2, z_2 \mid x_0, x_1, z_1) p(x_3, \ldots, x_N, z_3, \ldots, z_N \mid x_0, x_1, z_1, x_2, z_2) \\ &\quad = p_{x_1 x_2} r(z_2; x_1, x_2) p(x_3, \ldots, x_N, z_3, \ldots, z_N \mid x_0, x_1, z_1, x_2, z_2), \end{aligned}$$

where for the last equation, we used the independence of the observations, i.e., $p(z_2 \mid x_0, x_1, x_2, z_1) = r(z_2; x_1, x_2)$. Combining the above two relations,

$$\begin{aligned} p(X_N, Z_N) = \pi_{x_0} p_{x_0 x_1} r(z_1; x_0, x_1) p_{x_1 x_2} r(z_2; x_1, x_2) \\ \cdot p(x_3, \ldots, x_N, z_3, \ldots, z_N \mid x_0, x_1, z_1, x_2, z_2), \end{aligned}$$

and continuing in the same manner, we obtain

$$p(X_N, Z_N) = \pi_{x_0} \prod_{k=1}^{N} p_{x_{k-1} x_k} r(z_k; x_{k-1}, x_k). \tag{2.5}$$

We now show how the maximization of the above expression can be viewed as a shortest path problem. In particular, we construct a graph of state-time pairs, called the *trellis diagram*, by concatenating $N + 1$ copies

of the state space, and by preceding and following them with dummy nodes s and t, respectively, as shown in Fig. 2.2.2. The nodes of the kth copy correspond to the states x_{k-1} at time $k-1$. An arc connects a node x_{k-1} of the kth copy with a node x_k of the $(k+1)$st copy if the corresponding transition probability $p_{x_{k-1}x_k}$ is positive. Since maximizing a positive cost function is equivalent to maximizing its logarithm, we see from Eq. (2.5) that, given the observation sequence $Z_N = \{z_1, z_2, \ldots, z_N\}$, the problem of maximizing $p(X_N, Z_N)$ is equivalent to the problem

$$\text{minimize} \; -\ln(\pi_{x_0}) - \sum_{k=1}^{N} \ln\bigl(p_{x_{k-1}x_k} r(z_k; x_{k-1}, x_k)\bigr)$$

over all possible sequences $\{x_0, x_1, \ldots, x_N\}$.

By assigning to an arc (s, x_0) the length $-\ln(\pi_{x_0})$, to an arc (x_N, t) the length 0, and to an arc (x_{k-1}, x_k) the length $-\ln\bigl(p_{x_{k-1}x_k} r(z_k; x_{k-1}, x_k)\bigr)$, we see that the above minimization problem is equivalent to the problem of finding the shortest path from s to t in the trellis diagram. This shortest path defines the estimated state sequence $\{\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_N\}$.

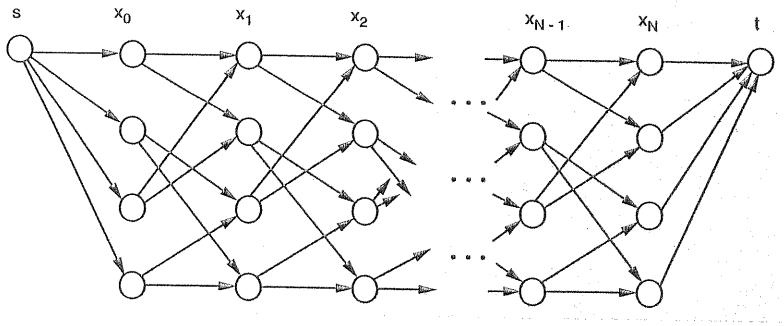


Figure 2.2.2 State estimation of an HMM viewed as a problem of finding a shortest path from s to t. Length of arcs from s to states x_0 is $-\ln(\pi_{x_0})$, and length of arcs from states x_N to t is zero. Length of an arc from a state x_{k-1} to x_k is $-\ln\bigl(p_{x_{k-1}x_k} r(z_k; x_{k-1}, x_k)\bigr)$, where z_k is the kth observation.

In practice, the shortest path is most conveniently constructed sequentially by forward DP, that is, by first calculating the shortest distance from s to each node x_1, then using these distances to calculate the shortest distance from s to each node x_2, etc. In particular, suppose that we have computed the shortest distances $D_k(x_k)$ from s to all states x_k on the basis of the observation sequence Z_k, and suppose that the new observation z_{k+1} is obtained. Then the shortest distances $D_{k+1}(x_{k+1})$ from s to any state

x_{k+1} can be computed by the DP recursion

$$D_{k+1}(x_{k+1}) = \min_{\substack{\text{all } x_k \text{ such that} \\ p_{x_k x_{k+1}} > 0}} \Bigl[D_k(x_k) - \ln\bigl(p_{x_k x_{k+1}} r(z_{k+1}; x_k, x_{k+1})\bigr)\Bigr].$$

The initial condition is $D_0(x_0) = -\ln(\pi_{x_0})$. The final estimated state sequence $\hat{X}_N$ corresponds to the shortest path from s to the final state $\hat{x}_N$ that minimizes $D_N(x_N)$ over the finite set of possible states x_N. An advantage of this procedure is that it can be executed in real time, as soon as each new observation is obtained.

There are a number of practical schemes for estimating a portion of the state sequence without waiting to receive the entire observation sequence Z_N, and this is useful if Z_N is a long sequence. For example, one can check fairly easily whether for some k, all shortest paths from s to states x_k pass through a single node in the subgraph of states $x_0, \ldots, x_{k-1}$. If so, it can be seen from Fig. 2.2.3 that the shortest path from s to that node will not be affected by reception of additional observations, and therefore the subsequence of state estimates up to that node can be determined without waiting for the remaining observations.

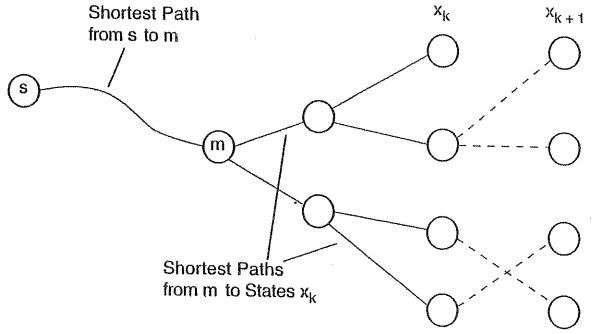


Figure 2.2.3 Estimating a portion of the state sequence prior to receiving the entire observation sequence. Suppose that the shortest paths from s to all states x_k pass through a single node m. If an additional observation is received, the shortest paths from s to all states x_{k+1} will continue to pass through m. Therefore, the portion of the state sequence up to node m can be safely estimated because additional observations will not change the initial portion of the shortest paths from s up to m.

The shortest path-based estimation procedure just described is known as the *Viterbi algorithm*, and finds numerous applications in a variety of contexts. An example is *speech recognition*, where the basic goal is to transcribe a spoken word sequence in terms of elementary speech units called *phonemes*. One possibility is to associate the states of the HHM with

phonemes, and given a sequence of recorded phonemes $Z_N = \{z_1, \ldots, z_N\}$, to find a phonemic sequence $\hat{X}_N = \{\hat{x}_1, \ldots, \hat{x}_N\}$ that maximizes over all $X_N = \{x_1, \ldots, x_N\}$ the conditional probability $p(X_N \mid Z_N)$. The probabilities $r(z_k; x_{k-1}, x_k)$ and $p_{x_{k-1}x_k}$ can be experimentally obtained, if necessary by specialized "training" for each speaker that uses the speech recognition system. The Viterbi algorithm can then be used to find the most likely phonemic sequence. There are also other HMMs used for word and sentence recognition, where only phonemic sequences that constitute words from a given dictionary are considered. We refer the reader to Rabiner [Rab89] and Picone [Pic90] for a general review of HMMs applied to speech recognition and for further references to related work. It is also possible to use similar models for computerized recognition of handwriting.

The Viterbi algorithm was originally developed as a scheme for decoding data after transmission over a noisy communication channel. The following example describes this context in some detail.

Example 2.2.1 (Convolutional Coding and Decoding)

When binary data are transmitted over a noisy communication channel, it is often essential to use coding as a means of enhancing reliability of communication. A common type of coding method, called *convolutional coding*, converts a source-generated binary data sequence

$$\{w_1, w_2, \ldots\}, \qquad w_k \in \{0, 1\}, \qquad k = 1, 2, \ldots,$$

into a coded sequence $\{y_1, y_2, \ldots\}$, where each y_k is an n-dimensional vector with binary coordinates, called *codeword*,

$$y_k = \begin{pmatrix} y_k^1 \\ \vdots \\ y_k^n \end{pmatrix}, \qquad y_k^i \in \{0, 1\}, \qquad i = 1, \ldots, n, \ k = 1, 2, \ldots.$$

The sequence $\{y_1, y_2, \ldots\}$ is then transmitted over a noisy channel and gets transformed into a sequence $\{z_1, z_2, \ldots, \}$, which is then decoded to yield the decoded data sequence $\{\hat{w}_1, \hat{w}_2, \ldots\}$; see Fig. 2.2.4. The objective is to design the encoder/decoder scheme so that the decoded sequence is as close to the original as possible.

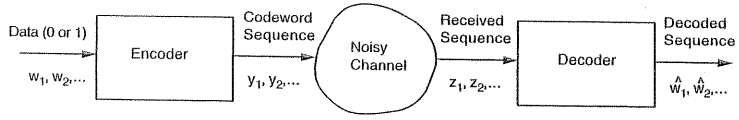


Figure 2.2.4 Encoder/decoder scheme.

The problem just discussed is central in information theory and can be approached in several different ways. In a particularly popular and effective technique called *convolutional coding*, the vectors y_k are related to w_k via equations of the form

$$y_k = Cx_{k-1} + dw_k, \qquad k = 1, 2, \ldots, \tag{2.6}$$

$$x_k = Ax_{k-1} + bw_k, \qquad k = 1, 2, \ldots, \qquad x_0 : \text{ given}, \tag{2.7}$$

where x_k is an m-dimensional vector with binary coordinates, which we view as state, and C, d, A, and b are $n \times m$, $n \times 1$, $m \times m$, and $m \times 1$ matrices, respectively, with binary coordinates. The products and the sums involved in the expressions $Cx_{k-1} + dw_k$ and $Ax_{k-1} + bw_k$ are calculated using modulo 2 arithmetic.

As an example, let $m = 2$, $n = 3$, and

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}, \qquad A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}, \qquad d = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Then the evolution of the system (2.6)-(2.7) can be represented by the diagram shown in Fig. 2.2.5. Given the initial x_0, this diagram can be used to generate the codeword sequence $\{y_1, y_2, \ldots\}$ corresponding to a data sequence $\{w_1, w_2, \ldots\}$. For example, when the initial state is $x_0 = 00$, the data sequence

$$\{w_1, w_2, w_3, w_4\} = \{1, 0, 0, 1\}$$

generates the state sequence

$$\{x_0, x_1, x_2, x_3, x_4\} = \{00, 01, 11, 10, 00\},$$

and the codeword sequence

$$\{y_1, y_2, y_3, y_4\} = \{111, 011, 111, 011\}.$$

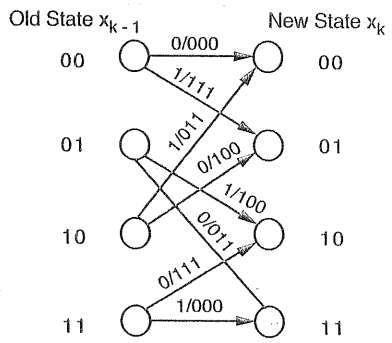


Figure 2.2.5 State transition diagram for convolutional coding. The binary number pair on each arc is the data/codeword pair w_k/y_k for the corresponding transition. So for example, when $x_{k-1} = 01$, a zero data bit ($w_k = 0$) effects a transition to $x_k = 11$ and generates the codeword 001.

Assume now that the characteristics of the noisy transmission channel are such that a codeword y is actually received as z with known probability $p(z \mid y)$, where z is any n-bit binary number. We assume independent errors so that

$$p(Z_N \mid Y_N) = \prod_{k=1}^{N} p(z_k \mid y_k), \tag{2.8}$$

where $Z_N = \{z_1, \ldots, z_N\}$ is the received sequence and $Y_N = \{y_1, \ldots, y_N\}$ is the transmitted sequence. By associating the codewords y with state transitions, we formulate a maximum likelihood estimation problem, whereby we want to find a sequence $\hat{Y}_N = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_N\}$ such that

$$p(Z_N \mid \hat{Y}_N) = \max_{Y_N} p(Z_N \mid Y_N).$$

The constraint on Y_N is that it must be a feasible codeword sequence (i.e., it must correspond to some initial state and data sequence, or equivalently, to a sequence of arcs of the trellis diagram).

Let us construct a trellis diagram by concatenating N state transition diagrams and appending dummy nodes s and t on its left and right sides, which are connected with zero-length arcs to the states x_0 and x_N, respectively. By using Eq. (2.8), we see that, given the received sequence $Z_N = \{z_1, z_2, \ldots, z_N\}$, the problem of maximizing $p(Z_N \mid Y_N)$ is equivalent to the problem

$$\text{minimize} \sum_{k=1}^{N} -\ln\big(p(z_k \mid y_k)\big)$$

over all binary sequences $\{y_1, y_2, \ldots, y_N\}$.

This is equivalent to a problem of finding a shortest path in the trellis diagram from s to t, where the length of the arc associated with the codeword y_k is $-\ln\big(p(z_k \mid y_k)\big)$, and the lengths of each arc incident to a dummy node is zero. From the shortest path and the trellis diagram, we can then obtain the corresponding data sequence $\{\hat{w}_1, \ldots, \hat{w}_N\}$, which is accepted as the decoded data.

The maximum likelihood estimate $\hat{Y}_N$ can be found by solving the corresponding shortest path problem using the Viterbi algorithm. In particular, the shortest distances $D_{k+1}(x_{k+1})$ from s to any state x_{k+1} are computed by the DP recursion

$$D_{k+1}(x_{k+1}) = \min_{\substack{\text{all } x_k \text{ such that} \\ (x_k, x_{k+1}) \text{ is an arc}}} \Big[D_k(x_k) - \ln\big(p(z_{k+1} \mid y_{k+1})\big)\Big],$$

where y_{k+1} is the codeword corresponding to the arc (x_k, x_{k+1}). The final state $\hat{x}_N$ on the shortest path is the one that minimizes $D_N(x_N)$ over x_N.

2.3 SHORTEST PATH ALGORITHMS

We have seen that shortest path problems and deterministic finite-state optimal control problems are equivalent. The computational implications of this are twofold.

(a) One can use DP to solve general shortest path problems. Note that there are several other shortest path methods, some of which have superior theoretical worst-case performance to DP. However, DP is often preferred in practice, particularly for problems with an acyclic graph structure and also when a parallel computer is available.

(b) One can use general shortest path methods (other than DP) for deterministic finite-state optimal control problems. In most cases, DP is preferable to other shortest path methods, because it is tailored to the sequential nature of optimal control problems. However, there are important cases where other shortest path methods are preferable.

In this section we discuss several alternative shortest path methods. We motivate these methods by focusing on shortest path problems with a very large number of nodes. Suppose that there is only one origin and only one destination, as in shortest path problems arising from deterministic optimal control (cf. Fig. 2.1.1). Then it is often true that most of the nodes are not relevant to the shortest path problem in the sense that they are unlikely candidates for inclusion in a shortest path between the given origin and destination. Unfortunately, however, in the DP algorithm every node and arc will participate in the computation, so there arises the possibility of other more efficient methods.

A similar situation arises in some search problems that are common in artificial intelligence and combinatorial optimization. Generally, these problems involve decisions that can be broken down into stages. With proper reformulation, the decision stages can be made to correspond to arc selections in a shortest path problem, or to the stages of a DP algorithm. We provide some examples.

Example 2.3.1 (The Four Queens Problem)

Four queens must be placed on a 4×4 portion of a chessboard so that no queen can attack another. In other words, the placement must be such that every row, column, or diagonal of the 4×4 board contains at most one queen. Equivalently, we can view the problem as a sequence of problems; first, placing a queen in one of the first two squares in the top row, then placing another queen in the second row so that it is not attacked by the first, and similarly placing the third and fourth queens. (It is sufficient to consider only the first two squares of the top row, since the other two squares lead to symmetric positions.) We can associate positions with nodes of an acyclic graph where the root node s corresponds to the position with no queens and the terminal nodes correspond to the positions where no additional queens can be placed

without some queen attacking another. Let us connect each terminal position with an artificial node t by means of an arc. Let us also assign to all arcs length zero except for the artificial arcs connecting terminal positions with less than four queens with the artificial node t. These latter arcs are assigned the length ∞ (see Fig. 2.3.1) to express the fact that they correspond to dead-end positions that cannot lead to a solution. Then, the four queens problem reduces to finding a shortest path from node s to node t.

Note that once the nodes of the graph are enumerated the problem is essentially solved. In this 4×4 problem the number of nodes is small. However, we can think of similar problems with much larger memory requirements. For example, there is an eight queens problem where the board is 8×8 instead of 4×4.

Example 2.3.2 (The Traveling Salesman Problem)

An important model for scheduling a sequence of operations is the classical traveling salesman problem. Here we are given N cities and the mileage between each pair of cities. We wish to find a minimum-mileage trip that visits each of the cities exactly once and returns to the origin node. To convert this problem to a shortest path problem, we associate a node with every sequence of n distinct cities, where $n \leq N$. The construction and arc lengths of the corresponding graph are explained by means of an example in Fig. 2.3.2. The origin node s consists of city A, taken as the start. A sequence of n cities ($n < N$) yields a sequence of $(n+1)$ cities by adding a new city. Two such sequences are connected by an arc with length equal to the mileage between the last two of the $n+1$ cities. Each sequence of N cities is connected to an artificial terminal node t with an arc having length equal to the distance from the last city of the sequence to the starting city A. Note that the number of nodes grows exponentially with the number of cities.

In the shortest path problem that we will consider in this section, there is a special node s, called the *origin*, and a special node t, called the *destination*. We will assume a single destination, but the methods to be discussed admit extensions to the case of multiple destinations (see Exercise 2.6). A node j is called a *child* of node i if there is an arc (i, j) connecting i with j. The length of arc (i, j) is denoted by a_{ij} and we assume that *all arcs have nonnegative length*. Exercise 2.7 deals with the case where all cycle lengths (rather than arc lengths) are assumed nonnegative. We wish to find a shortest path from origin to destination.

2.3.1 Label Correcting Methods

We now discuss a general type of shortest path algorithm. The idea is to progressively discover shorter paths from the origin to every other node i, and to maintain the length of the shortest path found so far in a variable d_i called the *label of* i. Each time d_i is reduced following the discovery of a

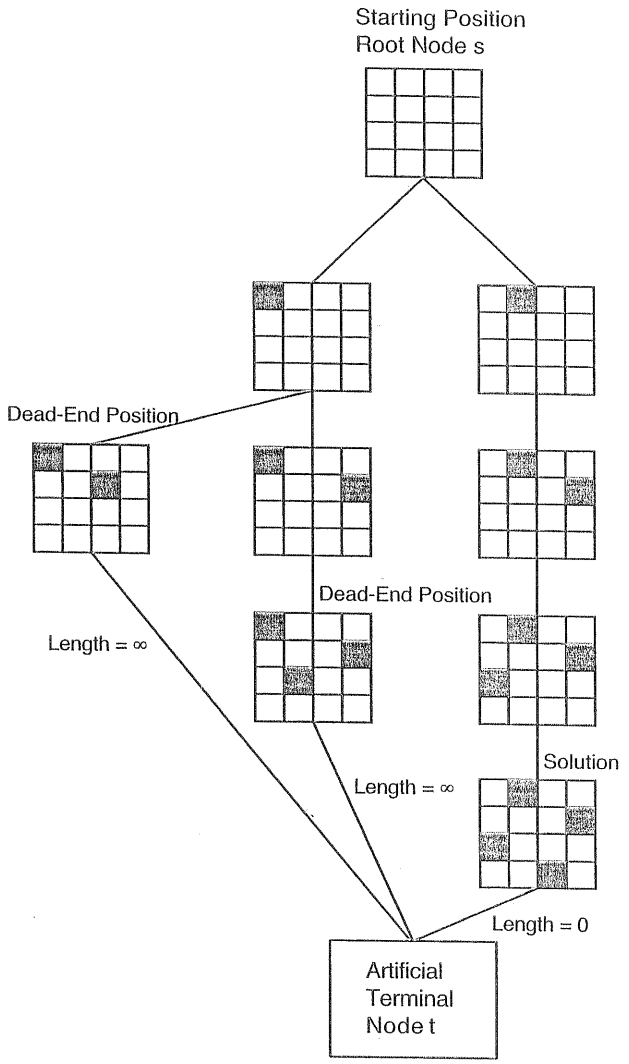
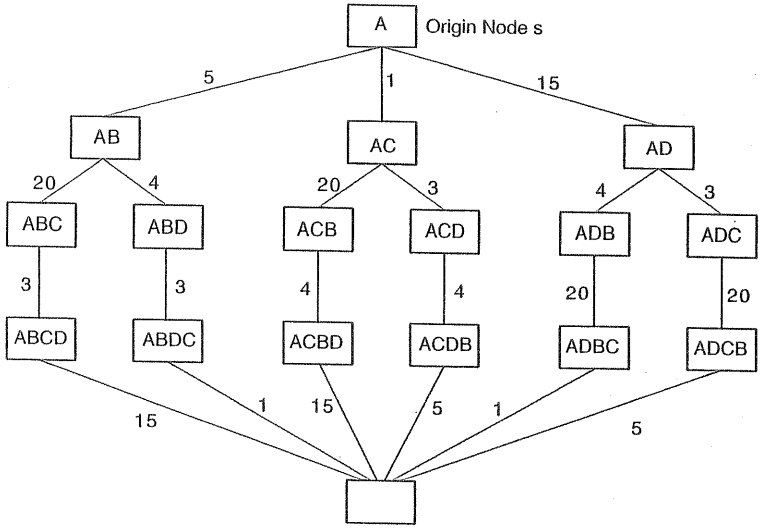


Figure 2.3.1 Shortest path formulation of the four queens problem. Symmetric positions resulting from placing a queen in one of the rightmost squares in the top row have been ignored. Squares containing a queen have been darkened. All arcs have length zero except for those connecting dead-end positions to the artificial terminal node.

shorter path to i, the algorithm checks to see if the labels d_j of the children j of i can be "corrected," that is, they can be reduced by setting them to



	5	1	15
5		20	4
1	20		3
15	4	3	

Figure 2.3.2 Example of a shortest path formulation of the traveling salesman problem. The distance between the four cities A, B, C, and D are shown in the table. The arc lengths are shown next to the arcs.

$d_i + a_{ij}$ [the length of the shortest path to i found thus far followed by arc (i, j)]. The label d_t of the destination is maintained in a variable called UPPER, which plays a special role in the algorithm. The label d_s of the origin is initialized at 0 and remains at 0 throughout the algorithm. The labels of all other nodes are initialized at ∞, i.e., $d_i = \infty$ for all $i \neq s$.

The algorithm also makes use of a list of nodes called OPEN (another name frequently used is *candidate list*). The list OPEN contains nodes that are currently active in the sense that they are candidates for further examination by the algorithm and possible inclusion in the shortest path. Initially, OPEN contains just the origin node s. Each node that has entered OPEN at least once, except s, is assigned a "parent," which is some other node. The parent nodes are not necessary for the computation of the shortest distance; they are needed for tracing the shortest path to the origin after the algorithm terminates. The steps of the algorithm are as follows (see also Fig. 2.3.3):

Label Correcting Algorithm

Step 1: Remove a node i from OPEN and for each child j of i, execute step 2.

Step 2: If $d_i + a_{ij} < \min\{d_j, \text{UPPER}\}$, set $d_j = d_i + a_{ij}$ and set i to be the parent of j. In addition, if $j \neq t$, place j in OPEN if it is not already in OPEN, while if $j = t$, set UPPER to the new value $d_i + a_{it}$ of d_t.

Step 3: If OPEN is empty, terminate; else go to step 1.

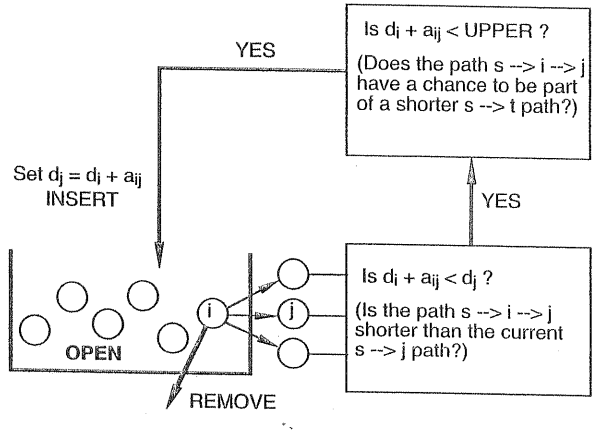


Figure 2.3.3 Diagrammatic illustration of the label correcting algorithm, with an interpretation of the tests for insertion of a node into the OPEN list.

It can be seen by induction that, throughout the algorithm, d_j is either ∞ (if node j has not yet entered the OPEN list), or else it is the length of some path from s to j consisting of nodes that have entered the OPEN list at least once. In the latter case, the path can be constructed by tracing backward the parent nodes starting with the parent of node j. Furthermore, UPPER is either ∞, or else it is the length of some path from s to t, and consequently it is an upper bound of the shortest distance from s to t. The idea in the algorithm is that when a path from s to j is discovered, which is shorter than those considered earlier ($d_i + a_{ij} < d_j$ in step 2), the value of d_j is accordingly reduced, and node j enters the OPEN list so that paths passing through j and reaching the children of j can be taken into account. It makes sense to do so, however, only when the path considered has a chance of leading to a path from s to t with length smaller than the upper bound UPPER of the shortest distance from s to t.

In view of the nonnegativity of the arc lengths, this is possible only if the path length $d_i + a_{ij}$ is smaller than UPPER. This provides the rationale for entering j into OPEN in step 2 only if $d_i + a_{ij} <$ UPPER (see Fig. 2.3.3).

Tracing the steps of the algorithm, we see that it will first remove node s from OPEN and sequentially examine its children. If t is not a child of s, the algorithm will place all children j of s in OPEN after setting $d_j = a_{sj}$. If t is a child of s, then the algorithm will place all children j of s examined before t in OPEN and will set their labels to a_{sj}; then it will examine t and set UPPER to a_{st}; finally, it will place each of the remaining children j of s in OPEN only if a_{sj} is less than the current value of UPPER, which is a_{st}. The algorithm will subsequently take a child $i \neq t$ of s from OPEN, and sequentially place in OPEN those of its children $j \neq t$ that satisfy the criterion of step 2, etc. Note that the origin s can never reenter OPEN because d_s cannot be reduced from its initial value of zero. Also, by the rules of the algorithm, the destination can never enter OPEN. When the algorithm terminates, we will show shortly that a shortest path can be obtained by tracing backwards the parent nodes starting from t and going towards s. Figure 2.3.4 illustrates the use of the algorithm to solve the traveling salesman problem of Fig. 2.3.2.

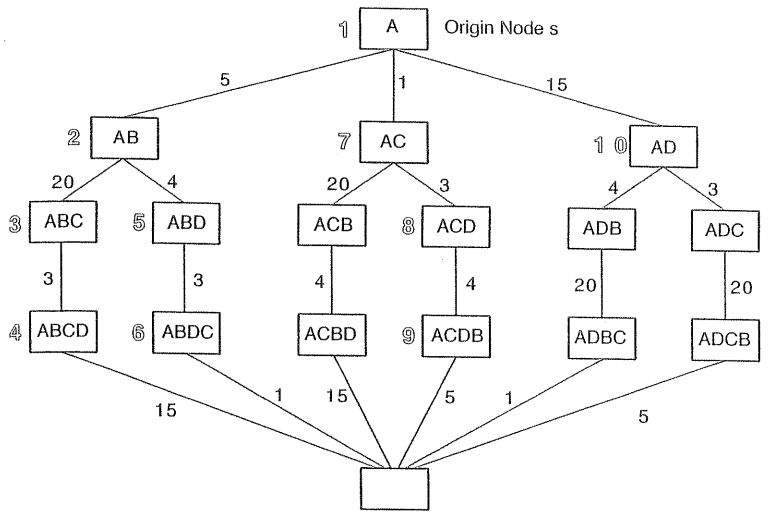
The following proposition establishes the validity of the algorithm.

Proposition 2.3.1: If there exists at least one path from the origin to the destination, the label correcting algorithm terminates with UPPER equal to the shortest distance from the origin to the destination. Otherwise the algorithm terminates with UPPER $= \infty$.

Proof: We first show that the algorithm will terminate. Indeed, each time a node j enters the OPEN list, its label is decreased and becomes equal to the length of some path from s to j. On the other hand, the number of distinct lengths of paths from s to j that are smaller than any given number is finite. The reason is that each path can be decomposed into a path with no repeated nodes (there is a finite number of distinct such paths), plus a (possibly empty) set of cycles, each having a nonnegative length. Therefore, there can be only a finite number of label reductions, implying that the algorithm will terminate.

Suppose that there is no path from s to t. Then a node i such that (i, t) is an arc cannot enter the OPEN list, because as argued earlier, this would establish that there is a path from s to i, and therefore also a path from s to t. Thus, based on the rules of the algorithm, UPPER can never be reduced from its initial value of ∞.

Suppose now that there is a path from s to t. Then, since there is a finite number of distinct lengths of paths from s to t that are smaller than any given number, there is also a shortest path. Let $(s, j_1, j_2, \ldots, j_k, t)$



Iter. No.	Node Exiting OPEN	OPEN at the End of Iteration	UPPER
0	-	1	∞
1	1	2, 7,10	∞
2	2	3, 5, 7, 10	∞
3	3	4, 5, 7, 10	∞
4	4	5, 7, 10	43
5	5	6, 7, 10	43
6	6	7, 10	13
7	7	8, 10	13
8	8	9, 10	13
9	9	10	13
10	10	Empty	13

Figure 2.3.4 The algorithm applied to the traveling salesman problem of Fig. 2.3.2. The optimal solution ABDC is found after examining nodes 1 through 10 in the figure in that order. The table shows the successive contents of the OPEN list.

be a shortest path and let d^* be the corresponding shortest distance. We will show that the value of UPPER upon termination must be equal to d^*. Indeed, each subpath $(s, j_1, \ldots, j_m)$, $m = 1, \ldots, k$, of the shortest path $(s, j_1, j_2, \ldots, j_k, t)$ must be a shortest path from s to j_m. If the value of

UPPER is larger than d^* at termination, the same must be true throughout the algorithm, and therefore UPPER will also be larger than the length of all the paths $(s, j_1, \ldots, j_m)$, $m = 1, \ldots, k$, throughout the algorithm, in view of the nonnegative arc length assumption. It follows that node j_k will never enter the OPEN list with d_{j_k} equal to the shortest distance from s to j_k, since in this case UPPER would be set to d^* in step 2 immediately following the next time node j_k is examined by the algorithm in step 2. Similarly, and using also the nonnegative length assumption, this means that node j_{k-1} will never enter the OPEN list with $d_{j_{k-1}}$ equal to the shortest distance from s to j_{k-1}. Proceeding backward, we conclude that j_1 never enters the OPEN list with d_{j_1} equal to the shortest distance from s to j_1 [which is equal to the length of the arc (s, j_1)]. This happens, however, at the first iteration of the algorithm, obtaining a contradiction. It follows that at termination, UPPER will be equal to the shortest distance from s to t. **Q.E.D.**

From the preceding proof, it can also be seen that, upon termination of the algorithm, the path constructed by tracing the parent nodes backward from t to s has length equal to UPPER, so it is a shortest path from s to t. Thus the algorithm yields not just the shortest distance but also a shortest path, provided that we keep track of the parent of each node that enters OPEN.

An important property of the algorithm is that nodes j for which $d_i + a_{ij} \geq$ UPPER in step 2 will not enter OPEN in the current iteration, and may possibly not enter in any subsequent iteration. As a result the number of nodes that enter OPEN may be much smaller than the total number of nodes. Furthermore, if a good lower bound to the shortest distance from s to t (or the shortest distance itself) is known, the computation can be terminated once UPPER reaches that bound within an acceptable tolerance. This is useful, for example, in the four queens problem, where the shortest distance is known to be zero or infinity. Then the algorithm will terminate once UPPER= 0, when a solution is found for the first time.

Specific Label Correcting Methods

There is considerable freedom in selecting the node to be removed from OPEN at each iteration. This gives rise to several different methods. The following are some of the most important (the author's textbooks on network optimization [Ber91a], [Ber98a] contain a fuller account of label correcting methods and their analysis; [Ber91a] contains several computer code implementations).

(a) *Breadth-first search*, also known as the *Bellman-Ford* method, which adopts a first-in/first-out policy; that is, the node is always removed from the top of OPEN and each node entering OPEN is placed at the bottom of OPEN. [Here and in the following methods except (c), we assume that OPEN is structured as a queue.]

(b) *Depth-first search*, which adopts a last-in/first-out policy; that is, the node is always removed from the top of OPEN and each node entering OPEN is placed at the top of OPEN. One motivation for this method is that it often requires relatively little memory. For example, suppose that the graph has a tree-like structure whereby there is a unique path from the origin node to every node other than the destination as shown in Fig. 2.3.5. Then the nodes will enter OPEN only once and in the order shown in Fig. 2.3.5. At any one time, it is only necessary to store a small portion of the graph as shown in Fig. 2.3.6.

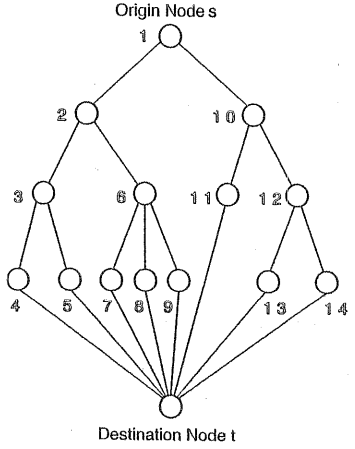


Figure 2.3.5 Searching a tree in depth-first fashion. The numbers next to the nodes indicate the order in which nodes exit the OPEN list.

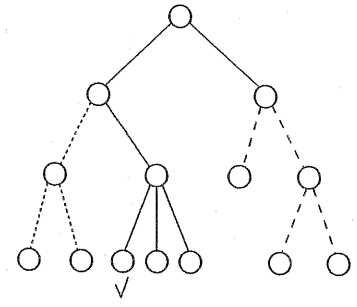


Figure 2.3.6 Memory requirements of depth-first search for the graph of Fig. 2.3.5. At the time the node marked by the checkmark exits the OPEN list, only the solid-line portion of the tree is needed in memory. The dotted-line portion has been generated and purged from memory based on the rule that for a graph where there is only one path from the origin to every node other than the destination, it is unnecessary to store a node once all of its successors are out of the OPEN list. The broken-line portion of the tree has not yet been generated.

(c) *Best-first search*, which at each iteration removes from OPEN a node with minimum value of label, i.e., a node i with

$$d_i = \min_{j \text{ in OPEN}} d_j.$$

This method, also known as *Dijkstra's method* or *label setting method*, has a particularly interesting property. It can be shown that in this method, a node will enter the OPEN list at most once (see Exercise 2.4). The drawback of this method is the overhead required to find at each iteration the node of OPEN that has minimum label. Several sophisticated methods have been developed to carry out this operation efficiently (see e.g., Bertsekas [Ber98a]).

(d) *D'Esopo-Pape method*, which at each iteration removes the node that is at the top of OPEN, but inserts a node at the top of OPEN if it has already been in OPEN earlier, and inserts the node at the bottom of OPEN otherwise.

(e) *Small-Label-First (SLF) method*, which at each iteration removes the node that is at the top of OPEN, but inserts a node i at the top of OPEN if its label d_i is less than or equal to the label d_j of the top node j of OPEN; otherwise it inserts i at the bottom of OPEN. This is a low-overhead approximation to the best-first search strategy. As a complement to the SLF strategy, one may also try to avoid removing nodes with relatively large labels from OPEN using the following device, known as the *Large-Label-Last (LLL) strategy*: at the start of an iteration, the top node of OPEN is compared with the average of the labels of the nodes in OPEN and if it is larger, the top node is placed at the bottom of OPEN and the new top node of OPEN is similarly tested against the average of the labels. In this way the removal of nodes with relatively large labels is postponed in favor of nodes with relatively small labels. The extra overhead required in this method is small: maintain the sum of the labels and the number of nodes in OPEN. When starting a new iteration, the ratio of these numbers gives the desired average. There are also several other methods, which are based on the idea of examining nodes with small labels first (see Bertsekas [Ber93], [Ber98a], and Bertsekas, Guerriero, and Musmanno [BGM96] for detailed descriptions and computational studies).

Generally, it appears that for nonnegative arc lengths, the number of iterations is reduced as the method is more successful in removing from OPEN nodes with a relatively small label. For a supporting heuristic argument, note that for a node j to reenter OPEN, some node i such that $d_i + a_{ij} < d_j$ must first exit OPEN. Thus, the smaller d_j was at the previous exit of j from OPEN, the less likely it is that $d_i + a_{ij}$ will subsequently become less than d_j for some node i in OPEN and arc (i, j). In particular,

if $d_j \leq \min_{i \text{ in OPEN}} d_i$, it is impossible that subsequent to the exit of j from OPEN we will have $d_i + a_{ij} < d_j$ for some i in OPEN, since the arc lengths a_{ij} are nonnegative. The SLF and other related but more sophisticated methods, often require a number of iterations, which is close to the minimum (the one required by the best-first method). However, they can be much faster than the best-first method, because they require much less overhead for determining the node to be removed from OPEN.

2.3.2 Label Correcting Variations - A^* Algorithm

The generic label correcting algorithm need not be started with the initial conditions $d_s = 0$ and $d_i = \infty$ for $i \neq s$ in order to work correctly. It can be shown, similar to Prop. 2.3.1, that one can use any set of initial labels d_i such that for each node i, d_i is either ∞ or else it is the length of some path from s to i. The scalar UPPER may be taken to be equal to d_t and the initial OPEN list may be taken to be the set $\{i \mid d_i < \infty\}$.

This kind of initialization is very useful if, by using heuristics or a known solution of a similar shortest path problem, we can construct a "good" path

$$P = (s, i_1, \ldots, i_k, t)$$

from s to t. Then we can initialize the algorithm with

$$d_i = \begin{cases} \text{Length of portion of path } P \text{ from } s \text{ to } i & \text{if } i \in P, \\ \infty & \text{if } i \notin P, \end{cases}$$

with UPPER equal to d_t, and with the OPEN list equal to $\{s, i_1, \ldots, i_k\}$. If P is a near-optimal path and consequently the initial value of UPPER is near its final value, the test for future admissibility into the candidate list will be relatively tight from the start of the algorithm and many unnecessary entrances of nodes into OPEN may be saved. In particular, it can be seen that all nodes whose shortest distances from the origin are greater or equal to the length of P will never enter the candidate list.

Another possibility, known as the A^* *algorithm*, is to strengthen the test $d_i + a_{ij} < \text{UPPER}$ that node j must pass before it can be placed in the OPEN list in step 2. This can be done if a *positive underestimate* h_j of the shortest distance of node j to the destination is available. Such an estimate can be obtained from special knowledge about the problem at hand. We may then speed up the computation substantially by placing a node j in OPEN in step 2 only when

$$d_i + a_{ij} + h_j < \text{UPPER}$$

(instead of $d_i + a_{ij} < \text{UPPER}$). In this way, fewer nodes will potentially be placed in OPEN before termination. Using the fact that h_j is an underestimate of the true shortest distance from j to the destination, it can be seen

that nodes j such that $d_i + a_{ij} + h_j \geq$ UPPER need not enter OPEN, and the argument given in the proof of Prop. 2.3.1 shows that the algorithm using the preceding test will terminate with a shortest path.

The A^* algorithm is just one way to sharpen the test $d_i + a_{ij} <$ UPPER for admission of node j into the OPEN list. An alternative idea is to try to reduce the value of UPPER by obtaining for the node j in step 2 an *upper bound* m_j of the shortest distance from j to the destination t (for example the length of some path from j to t). Then if $d_j + m_j <$ UPPER after step 2, we can reduce UPPER to $d_j + m_j$, thereby making the test for future admissibility into OPEN more stringent. This idea is used in some versions of the branch-and-bound algorithm, one of which we now describe.

2.3.3 Branch-and-Bound

Consider a problem of minimizing a cost function $f(x)$ over a *finite* set of feasible solutions X. We have in mind problems where the number of feasible solutions is very large, so an enumeration and comparison of these solutions is impractical, e.g., the traveling salesman problem of Example 2.3.2. The idea of the branch-and-bound method is to avoid a complete enumeration by discarding solutions that, based on certain tests, have no chance of being optimal. This is similar to label correcting methods, where based on various tests that use the value of UPPER and other data, the insertion in the OPEN list of some nodes is avoided. In fact, we will see that the branch-and-bound method can be viewed as a form of label correcting method.

The key idea of the branch-and-bound method is to partition the feasible set into smaller subsets, and then use certain bounds on the attainable cost within some of the subsets to eliminate from further consideration other subsets. The rationale for this is captured in the following simple observation.

Bounding Principle

Given the problem of minimizing $f(x)$ over $x \in X$, and two subsets $Y_1 \subset X$ and $Y_2 \subset X$, suppose that we have bounds

$$\underline{f}_1 \leq \min_{x \in Y_1} f(x), \qquad \overline{f}_2 \geq \min_{x \in Y_2} f(x).$$

Then, if $\overline{f}_2 \leq \underline{f}_1$, the solutions in Y_1 may be disregarded since their cost cannot be smaller than the cost of the best solution in Y_2.

The branch-and-bound method calculates suitable upper and lower bounds, and uses the bounding principle to eliminate from consideration substantial portions of the feasible set. To describe the method, we use

an acyclic graph with nodes that correspond on a one-to-one basis with a collection $\mathcal{X}$ of subsets of the feasible set X. We require the following:

1. $X \in \mathcal{X}$ (i.e., the set of all solutions is a node).
2. For each solution x, we have $\{x\} \in \mathcal{X}$ (i.e., each solution viewed as a singleton set is a node).
3. Each set $Y \in \mathcal{X}$ that contains more than one solution $x \in X$ is partitioned into sets $Y_1, \ldots, Y_n \in \mathcal{X}$ such that $Y_i \neq Y$ for all i:
$$\bigcup_{i=1}^{n} Y_i = Y.$$
The set Y is called the *parent* of $Y_1, \ldots, Y_n$, and the sets $Y_1, \ldots, Y_n$ are called the *children* of Y.
4. Each set in $\mathcal{X}$ other than X has at least one parent.

The collection of sets $\mathcal{X}$ defines an acyclic graph with root node the set of all feasible solutions X and terminal nodes the singleton solutions $\{x\}$, $x \in X$ (see Fig. 2.3.7). The arcs of the graph are those that connect parents Y and their children Y_i. Suppose that for every nonterminal node Y there is an algorithm that calculates upper and lower bounds $\underline{f}_Y$ and $\overline{f}_Y$ for the minimum cost over Y:

$$\underline{f}_Y \leq \min_{x \in Y} f(x) \leq \overline{f}_Y.$$

Assume further that the upper and lower bounds are exact for each singleton solution node $\{x\}$:

$$\underline{f}_{\{x\}} = f(x) = \overline{f}_{\{x\}}, \qquad \text{for all } x \in X.$$

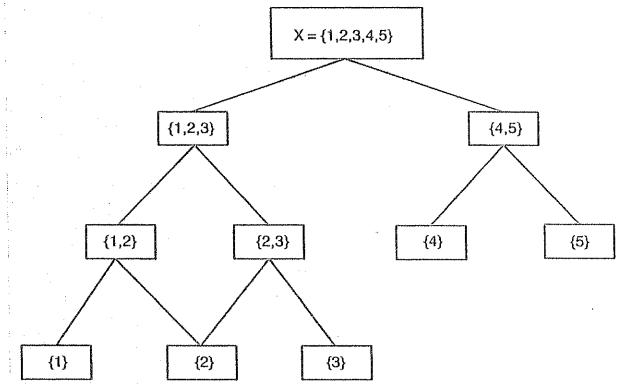


Figure 2.3.7 An acyclic graph corresponding to a branch-and-bound algorithm. Each node (subset) except those consisting of a single solution is partitioned into several other nodes (subsets).

Define now the length of an arc involving a parent Y and a child Y_i, to be the lower bound difference

$$\underline{f}_{Y_i} - \underline{f}_Y.$$

Then the length of any path from the origin node X to any node Y is $\underline{f}_Y$. Since $\underline{f}_{\{x\}} = f(x)$ for all feasible solutions $x \in X$, it is clear that minimizing $f(x)$ over $x \in X$ is equivalent to finding a shortest path from the origin node to one of the singleton nodes $\{x\}$.

Consider now a variation of the label correcting method, where in addition we use our knowledge of the upper bounds $\overline{f}_Y$ to reduce the value of UPPER. Initially, OPEN contains just X, and UPPER equals $\overline{f}_X$.

Branch-and-Bound Algorithm

Step 1: Remove a node Y from OPEN. For each child Y_j of Y, do the following: If $\underline{f}_{Y_j} <$ UPPER, then place Y_j in OPEN. If in addition $\overline{f}_{Y_j} <$ UPPER, then set UPPER $= \overline{f}_{Y_j}$, and if Y_j consists of a single solution, mark that solution as being the best solution found so far.

Step 2: (Termination Test) If OPEN is nonempty, go to step 1. Otherwise, terminate; the best solution found so far is optimal.

An alternative termination step 2 for the preceding algorithm is to set a tolerance $\epsilon > 0$, and check whether UPPER and the minimum lower bound $\underline{f}_Y$ over all sets Y in the OPEN list differ by less than ϵ. If so, the algorithm is terminated, and some set in OPEN must contain a solution that is within ϵ of being optimal. There are a number of other variations of the algorithm. For example, if the upper bound $\overline{f}_Y$ at a node is actually the cost $f(x)$ of some element $x \in Y$, then this element can be taken as the best solution found so far whenever $\overline{f}_Y <$ UPPER in step 2. Other variations relate to the method of selecting a node from OPEN in step 1. For example, two strategies of the best-first type are to select the node with minimal lower or upper bound. Note that *it is neither practical nor necessary to generate a priori the acyclic graph of the branch-and-bound method.* Instead, one may adaptively decide on the order and the manner in which the parent sets are partitioned into children sets based on the progress of the algorithm.

We finally note that to apply branch-and-bound effectively, it is important to have good algorithms for generating upper and lower bounds at each node. These bounds should be as sharp as practically possible. Then, fewer nodes will be admitted into OPEN, with attendant computational savings. Typically, continuous optimization problems (usually linear or network optimization problems) are used to obtain lower bounds to the optimal costs of the restricted problems $\min_{x \in Y} f(x)$, while various heuristics are used to construct corresponding feasible solutions whose costs can

be used as upper bounds. We refer to textbooks such as Nemhauser and Wolsey [NeW88], Bertsimas and Tsitsiklis [BeT97], Bertsekas [Ber98a], and Wolsey [Wol98] for fuller accounts.

2.3.4 Constrained and Multiobjective Problems

In some shortest path contexts, there may be constraints on the resources required to traverse the optimal path, such as limits on time, fuel, etc. For example, there could be a restriction that the total time to travel through the optimal path P should not exceed a given threshold T, i.e.,

$$\sum_{(i,j) \in P} \tau_{ij} \leq T,$$

where τ_{ij} is the time required to traverse arc (i,j). Similarly, there could be a safety constraint, whereby the probability of being able to traverse the path P safely should be no less than a given threshold. Here, we assume that traversal of an arc (i,j) will be safe with a given probability p_{ij}. Assuming probabilistic independence of the safety of arc traversals, the probability that traversal of a path P will be safe is the product $\Pi_{(i,j) \in P} p_{ij}$. The requirement that this probability is no less than a given threshold β translates to a path length constraint of the form

$$\sum_{(i,j) \in P} \ln(p_{ij}) \geq \ln(\beta).$$

We represent path constraints in the generic form

$$\sum_{(i,j) \in P} c_{ij}^m \leq b^m, \qquad m = 1, \ldots, M, \tag{2.9}$$

where c_{ij}^m is the amount of mth resource required to traverse arc (i,j), and b^m is the total amount of mth resource available. Thus the problem is to find a path P that starts at the origin s, ends at the destination t, satisfies the constraints (2.9), and minimizes

$$\sum_{(i,j) \in P} a_{ij}.$$

We refer to this problem as the *constrained shortest path problem*, and we note that it is closely related to the *constraint feasibility problem*, where we simply want to find a path P that satisfies the constraints (2.9). In particular, the constraint feasibility problem is the special case of the constrained shortest path problem where $a_{ij} = 0$ for all arcs (i,j).

Conversely, the constrained shortest path problem is equivalent to the constraint feasibility problem involving the constraints (2.9) *and* the additional constraint

$$\sum_{(i,j)\in P} a_{ij} \leq L^*,$$

where L^* is the optimal path length (which, however, is generally unknown).

Another, closely related problem is the *multiobjective shortest path problem*, where we want to find a path P that simultaneously makes all the lengths

$$\sum_{(i,j)\in P} c_{ij}^m, \qquad m = 1, \ldots, M,$$

"small," in a sense that we will now make precise. In particular, for any set $S \subset \Re^M$, let us call a vector $x = (x_1, \ldots, x_M) \in S$ *noninferior* if x is not dominated by any vector $y = (y_1, \ldots, y_M) \in S$, in the sense that

$$y_m \leq x_m, \qquad m = 1, \ldots, M,$$

and with strict inequality for at least one m (see Fig. 2.3.8). More generally, given a problem with multiple cost functions $f_1(x), \ldots, f_M(x)$ and a constraint set X, we say that x is a *noninferior* solution if the vector of costs of x, i.e., $\big(f_1(x), \ldots, f_M(x)\big)$, is a noninferior vector of the set of attainable costs

$$\Big\{\big(f_1(y), \ldots, f_M(y)\big) \mid y \in X\Big\}.$$

Note that given a finite set of solutions, there is at least one noninferior solution. Furthermore, one can extract the set of noninferior solutions with a simple algorithm: sequentially test all solutions and discard those that are dominated by some not yet discarded solution, until no more solutions can be discarded.

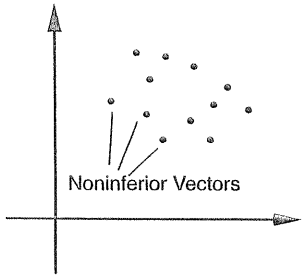


Figure 2.3.8 Illustration of the noninferior vectors of a finite set.

The multiobjective shortest path problem is to find a noninferior path P, i.e., one for which there is no other path P' satisfying

$$\sum_{(i,j)\in P'} c_{ij}^m \leq \sum_{(i,j)\in P} c_{ij}^m, \qquad m = 1, \ldots, M,$$

and with strict inequality for at least one m. Note that the constraint feasibility problem has a solution if and only if the subset of multiobjective/noninferior solutions that satisfy the constraints (2.9) is nonempty. It follows that the constraint feasibility problem can be easily solved once the set of all noninferior paths is computed. Similarly, the constrained shortest path problem can be solved by casting it as a multiobjective shortest path problem, where the multiple objectives correspond to the cost and the constraints. Given the set of all noninferior solutions, one obtains an optimal solution of the constrained shortest path problem (provided a feasible solution exists), by selecting a path from this set that satisfies the constraints and minimizes the cost. Because of the connections outlined above, the three problems, constrained shortest path, constraint feasibility, and multiobjective, fundamentally share the same mathematical structure, and can be addressed with similar methodology.

Multiobjective DP Problems

We have seen that shortest path problems and deterministic finite-state DP problems are equivalent, so it is not surprising that the methodology for multiobjective and/or constrained shortest path problems is shared by multiobjective and/or constrained deterministic finite-state DP problems. A multiobjective version of such a problem involves a single controlled deterministic finite-state system

$$x_{k+1} = f_k(x_k, u_k),$$

and multiple cost functions of the form

$$g_N^m(x_N) + \sum_{k=0}^{N-1} g_k^m(x_k, u_k), \qquad m = 1, \ldots, M.$$

Let us provide an extension of the DP algorithm that finds the set of all noninferior solutions to the multiobjective deterministic DP problem involving the above system and cost functions. This algorithm proceeds backwards from the terminal time, and calculates for each stage k and state x_k, the set of noninferior control sequences for the tail (multiobjective) subproblem that starts at state x_k. The algorithm is based on a fairly evident extension of the principle of optimality:

If $\{u_k, u_{k+1}, \ldots, u_{N-1}\}$ is a noninferior control sequence for the tail subproblem that starts at x_k, then $\{u_{k+1}, \ldots, u_{N-1}\}$ is a noninferior control sequence for the tail subproblem that starts at $f_k(x_k, u_k)$.

This allows the convenient calculation of the set of noninferior solutions of tail subproblems using the sets of noninferior solutions of shorter subproblems.

More specifically, let $\mathcal{F}_k(x_k)$ be the set of all M-tuples of costs-to-go

$$\left(g_N^1(x_N) + \sum_{i=k}^{N-1} g_i^1(x_i, u_i), \ldots, g_N^M(x_N) + \sum_{i=k}^{N-1} g_i^M(x_i, u_i) \right), \qquad (2.10)$$

which correspond to feasible control sequences $\{u_k, \ldots, u_{N-1}\}$ that start at x_k and are noninferior in the following sense: There is no other feasible control sequence $\{u_k', \ldots, u_{N-1}'\}$ with corresponding state sequence $\{x_k', \ldots, x_N'\}$ (where $x_k' = x_k$) such that

$$g_N^m(x_N') + \sum_{i=k}^{N-1} g_i^m(x_i', u_i') \leq g_N^m(x_N) + \sum_{i=k}^{N-1} g_i^m(x_i, u_i), \quad m = 1, \ldots, M,$$

and with strict inequality for at least one m. Note that $\mathcal{F}_k(x_k)$ is a finite set (since the control constraint set is finite, which implies that the set of control sequences is finite). The sets $\mathcal{F}_k(x_k)$ are generated by an algorithm that starts at the terminal time N with $\mathcal{F}_N(x_N)$ consisting of just the vector of terminal costs,

$$\mathcal{F}_N(x_N) = \left\{ \left(g_N^1(x_N), \ldots, g_N^M(x_N) \right) \right\},$$

and proceeds backwards according to the following process: Given the set $\mathcal{F}_{k+1}(x_{k+1})$ for all states x_{k+1}, the algorithm generates for each state x_k the set of vectors

$$\left(g_k^1(x_k, u_k) + c^1, \ldots, g_k^M(x_k, u_k) + c^M \right)$$

such that

$$(c^1, \ldots, c^M) \in \mathcal{F}_{k+1}\left(f_k(x_k, u_k)\right), \qquad u_k \in U_k(x_k);$$

then it obtains $\mathcal{F}_k(x_k)$ by extracting the noninferior subset, i.e., by discarding from this set the vectors that are dominated by other vectors.

After N steps, this algorithm yields $\mathcal{F}_0(x_0)$, the set of all noninferior M-tuples of costs-to-go starting at initial state x_0. Note that the algorithm is similar to the ordinary DP algorithm: it just maintains a set of noninferior M-tuples of costs-to-go at each state x_k, rather than a single cost-to-go. Note also that by a similar argument to the one of Section 2.1, it is possible to construct a forward version of this algorithm.

Constrained DP Problems

Let us now consider a related constrained DP problem with the same controlled system

$$x_{k+1} = f_k(x_k, u_k),$$

where we want to minimize the cost function

$$g_N^1(x_N) + \sum_{k=0}^{N-1} g_k^1(x_k, u_k) \qquad (2.11)$$

subject to the constraints

$$g_N^m(x_N) + \sum_{k=0}^{N-1} g_k^m(x_k, u_k) \leq b^m, \qquad m = 2, \ldots, M. \qquad (2.12)$$

We can solve this problem by finding the set of noninferior solutions/control sequences of the multiobjective DP problem involving the costs (2.10), by extracting from this set the subset of solutions/control sequences that satisfy the constraints (2.12), and by selecting from this subset a solution/control sequence that minimizes the cost (2.11).

However, we can enhance this algorithm by discarding at the earliest opportunity control sequences that cannot be part of a feasible control sequence. The rationale for this is related to the ideas underlying label correcting methods and the A^* algorithm in particular (cf. Section 2.3.2). For $m = 2, \ldots, M$, let $\tilde{J}_k^m(x_k)$ be the optimal cost to arrive to x_k from the given initial state x_0 with cost per stage equal to $g_i^m(x_i, u_i)$. This is the minimal value of

$$\sum_{i=0}^{k-1} g_k^m(x_i, u_i), \qquad m = 2, \ldots, M,$$

subject to the constraint that the state at the kth stage is x_k, and can be calculated by the forward DP algorithm of Section 2.1. Consider now a DP-like algorithm that generates for each state and stage, a subset of M-tuples. It starts at the terminal time N with the set $\mathcal{F}_N(x_N)$ that consists of just the vector of terminal costs,

$$\mathcal{F}_N(x_N) = \left\{ \left(g_N^1(x_N), \ldots, g_N^M(x_N) \right) \right\}.$$

It proceeds backwards as follows: given the set $\mathcal{F}_{k+1}(x_{k+1})$ for each state x_{k+1}, it generates for each state x_k the set of M-tuples

$$\left(g_k^1(x_k, u_k) + c^1, \ldots, g_k^M(x_k, u_k) + c^M \right) \qquad (2.13)$$

such that

$$(c^1, \ldots, c^M) \in \mathcal{F}_{k+1}\big(f_k(x_k, u_k)\big), \qquad u_k \in U_k(x_k),$$

and

$$\tilde{J}_k^m(x_k) + g_k^m(x_k, u_k) + c^m \leq b^m, \qquad m = 2, \ldots, M; \tag{2.14}$$

then it obtains $\mathcal{F}_k(x_k)$ by extracting the noninferior subset, i.e., by discarding from this set the elements that are dominated by other elements.

Note that M-tuples of the form (2.13) that violate the condition (2.14) correspond to paths that cannot be feasible, so they can be safely excluded from further consideration [in fact $\tilde{J}_k^m(x_k)$ may be replaced by any conveniently available underestimate in the condition (2.14); using $\tilde{J}_k^m(x_k)$ makes this condition as sharp as possible]. The set $\mathcal{F}_0(x_0)$ obtained by the algorithm after N steps consists of M-tuples of the cost and the constraint function values

$$g_N^m(x_N) + \sum_{k=0}^{N-1} g_k^m(x_k, u_k), \qquad m = 1, \ldots, M,$$

which correspond to all the feasible solutions that are noninferior. The first component of an M-tuple in $\mathcal{F}_0(x_0)$ corresponds to the cost (2.11). An element of $\mathcal{F}_0(x_0)$ whose first component is minimal is an optimal solution of the constrained shortest path problem. The advantage of using the criterion (2.14) is that it allows us to discard as early as possible infeasible solutions, and accordingly reduce the size of the sets $\mathcal{F}_k(x_k)$ and the attendant computation.

Note also that if an upper bound, call it UPPER, is known for the optimal path length, it can be used to introduce the additional constraint

$$g_N^1(x_N) + \sum_{k=0}^{N-1} g_k^1(x_k, u_k) \leq \text{UPPER},$$

and to make the test (2.14) more effective by augmenting it with the additional inequality

$$\tilde{J}_k^1(x_k) + g_k^1(x_k, u_k) + c^1 \leq \text{UPPER}.$$

Any M-tuple $(c^1, \ldots, c^M)$ that violates this condition corresponds to paths that cannot be optimal, so it can be safely excluded from further consideration. This idea may be further enhanced by introducing schemes to reduce UPPER as the algorithm progresses, similar to label correcting methods.

Clearly multiobjective and constrained DP algorithms require quite a bit more computation and storage than ordinary DP for the same system. For this reason, there have been many efforts to develop approximate solution methods. These methods are outside our scope and we refer to the literature on the subject.

The preceding DP algorithms for multiobjective and constrained problems are easily adapted to shortest path problems. One possibility is to use the transformation described in Section 2.1 to reformulate the shortest path problem as a (multiobjective or constrained) deterministic finite-state problem. The latter problem can in turn be solved using the DP-like algorithms just described. It is also possible to use versions of label correcting algorithms, including the A^* variant, for multiobjective and constrained shortest path problems. A label of a node now is not just a single number, but rather it is an M-dimensional vector with components that correspond to the M cost functions; see the end-of-chapter references.

2.4 NOTES, SOURCES, AND EXERCISES

Work on the shortest path problem is very extensive. Literature surveys are given by Dreyfus [Dre69], Deo and Pang [DeP84], and Gallo and Pallottino [GaP88]. For a detailed textbook treatment of shortest paths, see Bertsekas [Ber98a] (the chapter on shortest paths of this book is www-accessible), and also [Ber91a], which contains a variety of associated computer codes.

For a treatment of critical path analysis, see Elmaghraby [Elm78]. A tutorial survey of Hidden Markov Models is given by Rabiner [Rab89]. The Viterbi algorithm, first proposed in [Vit67], is also discussed by Forney [For73]. For applications in communication systems, see Proakis and Salehi [PrS94], and Sklar [Skl88]. For applications in speech recognition, see Rabiner [Rab89] and Picone [Pic90]. For applications in data network routing, see Bertsekas and Gallager [BeG92].

Label correcting methods draw their origin from the works of Bellman [Bel57] and Ford [For56]. The D'Esopo-Pape algorithm appeared in [Pap74] and is based on an earlier suggestion of D'Esopo. For a discussion of various implementations of Dijkstra's algorithm, see Bertsekas [Ber98a]. The SLF method and some variations were proposed by the author in [Ber93]; see also Bertsekas, Guerriero, and Musmanno [BGM96], where the LLL strategy as well as implementations on a parallel computer of various label correcting methods are discussed. The A^* method was proposed by Hart, Nilsson, and Raphael [HNR68] (with corrections in [HNR72]). See also the texts by Nilsson [Nil71], [Nil80], and Pearl [Pea84], which provide a broader discussion of the application of shortest path methods in artificial intelligence.

The Dijkstra algorithm has been extended to continuous space shortest path problems by Tsitsiklis [Tsi75]. The SLF/LLL methods have also

been similarly extended by Bertsekas, Guerriero, and Musmanno [BGM95], and by Polymenakos, Bertsekas, and Tsitsiklis [PBT98].

There is extensive literature on exact and approximate solution methods for constrained and multiobjective shortest path and DP problems. Analogs of label correcting and Dijkstra-like methods were proposed by Vincke [Vin74] and Hansen [Han80], respectively; see also Jaffe [Jaf84] and Martins [Mar84]. Recent work includes Guerriero and Musmanno [GuM01], who investigate analogs of the SLF/LLL methods, and give many references and computational results. For a multiobjective version of the A^* method, see Stewart and White [StW91], who also survey earlier work.

EXERCISES

2.1

Find a shortest path from each node to node 6 for the graph of Fig. 2.4.1 by using the DP algorithm.

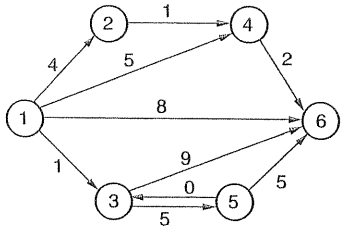


Figure 2.4.1 Graph for Exercise 2.1. The arc lengths are shown next to the arcs.

2.2

Find a shortest path from node 1 to node 5 for the graph of Fig. 2.4.2 by using the label correcting method of Section 2.3.1.

2.3

Air transportation is available between n cities, in some cases directly and in others through intermediate stops and change of carrier. The airfare between cities

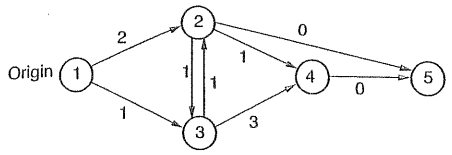


Figure 2.4.1 Graph for Exercise 2.2. The arc lengths are shown next to the arcs.

i and j is denoted by a_{ij}. We assume that $a_{ij} = a_{ji}$, and for notational convenience, we write $a_{ij} = \infty$ if there is no direct flight between i and j. The problem is to find the cheapest airfare for going between two cities perhaps through intermediate stops. Let $n = 6$ and $a_{12} = 30$, $a_{13} = 60$, $a_{14} = 25$, $a_{15} = a_{16} = \infty$, $a_{23} = a_{24} = a_{25} = \infty$, $a_{26} = 50$, $a_{34} = 35$, $a_{35} = a_{36} = \infty$, $a_{45} = 15$, $a_{46} = \infty$, $a_{56} = 15$. Find the cheapest airfare from every city to every other city by using the DP algorithm.

2.4 (Dijkstra's Algorithm for Shortest Paths) WWW

Consider the best-first version of the label correcting algorithm of Section 2.3.1. Here at each iteration we remove from OPEN a node that has minimum label over all nodes in OPEN.

(a) Show that each node j will enter OPEN at most once, and show that at the time it exits OPEN, its label d_j is equal to the shortest distance from s to j. *Hint*: Use the nonnegative arc length assumption to argue that in the label correcting algorithm, in order for the node i that exits OPEN to reenter, there must exist another node k in OPEN with $d_k + a_{ki} < d_i$.

(b) Show that the number of arithmetic operations required for termination is bounded by cN^2 where N is the number of nodes and c is some constant.

2.5 (Label Correcting for Acyclic Graphs)

Consider a shortest path problem involving an acyclic graph. Let S_k be the set of nodes i such that all paths from the origin to i have k arcs or less and at least one such path has k arcs. Consider a label correcting algorithm that removes from OPEN a node of S_k only if there are no nodes of $S_1, \ldots, S_{k-1}$ in OPEN. Show that each node will enter OPEN at most once. How does this result relate to the type of shortest path problem arising from deterministic DP (cf. Fig. 2.1.1)?

2.6 (Label Correcting with Multiple Destinations) WWW

Consider the problem of finding a shortest path from node s to each node in a subset T, assuming that all arc lengths are nonnegative. Show that the following modified version of the label correcting algorithm of Section 2.3.1 solves the problem. Initially, UPPER $= \infty$, $d_s = 0$, and $d_i = \infty$ for all $i \neq s$.

Step 1: Remove a node i from OPEN and for each child j of i, execute step 2.

Step 2: If $d_i + a_{ij} < \min\{d_j, \text{UPPER}\}$, set $d_j = d_i + a_{ij}$, set i to be the parent of j, and place j in OPEN if it is not already in OPEN. In addition, if $j \in T$, set $\text{UPPER} = \max_{t \in T} d_t$.

Step 3: If OPEN is empty, terminate; else go to step 1.

Prove a termination property such as the one of Prop. 2.3.1 for this algorithm.

2.7 (Label Correcting with Negative Arc Lengths)

Consider the problem of finding a shortest path from node s to node t, and assume that all cycle lengths are nonnegative (instead of all arc lengths being nonnegative). Suppose that a scalar u_j is known for each node j, which is an underestimate of the shortest distance from j to t (u_j can be taken $-\infty$ if no underestimate is known). Consider a modified version of the typical iteration of the label correcting algorithm of Section 2.3.1, where step 2 is replaced by the following:

Modified Step 2: If $d_i + a_{ij} < \min\{d_j, \text{UPPER} - u_j\}$, set $d_j = d_i + a_{ij}$ and set i to be the parent of j. In addition, if $j \neq t$, place j in OPEN if it is not already in OPEN, while if $j = t$, set UPPER to the new value $d_i + a_{it}$ of d_t.

(a) Show that the algorithm terminates with a shortest path, assuming there is at least one path from s to t (cf. Prop. 2.3.1).

(b) Why is the algorithm of Section 2.3.1 a special case of the one of this exercise?

2.8

We have a set of N objects, denoted $1, 2, \ldots, N$, which we want to group in clusters that consist of consecutive objects. For each cluster $i, i+1, \ldots, j$, there is an associated cost a_{ij}. We want to find a grouping of the objects in clusters such that the total cost is minimum. Formulate the problem as a shortest path problem, and write a DP algorithm for its solution. (Note: An example of this problem arises in typesetting programs, such as TEX/LATEX, that break down a paragraph into lines in a way that optimizes the paragraph's appearance.)

2.9 (Shortest Path Tour Problem [BeC04])

Consider a problem of finding a shortest path from a given origin node s to a given destination node t in a graph with nonnegative arc lengths. However, there is the constraint that the path should successively pass through at least one node from given node subsets $T_1, T_2, \ldots, T_N$ (i.e., for all k, pass through some node from the subset T_k after passing through at least one node of the subsets $T_1, \ldots, T_{k-1}$).

(a) Formulate this as a dynamic programming problem.

(b) Show that a solution can be obtained by solving a sequence of ordinary shortest path problems, each involving a single origin and multiple destinations.

2.10 (Two-Sided Dijkstra Algorithm [Nic66])

Consider a problem of finding a shortest path from a given origin node s to a given destination node t in a graph with nonnegative arc lengths. Consider an algorithm that maintains two subsets of nodes, W and V, with the following properties:

(1) $s \in W$ and $t \in V$.

(2) If $i \in W$ and $j \notin W$, then the shortest distance from s to i is less than or equal to the shortest distance from s to j.

(3) If $i \in V$ and $j \notin V$, then the shortest distance from i to t is less than or equal to the shortest distance from j to t.

At each iteration the algorithm adds a new node to W and a new node to V (the Dijkstra algorithm can be used for this purpose), and terminates when W and V have a node in common. Let d_i^s be the shortest distance from s to i using paths all the nodes of which, with the possible exception of i, lie in W ($d_i^s = \infty$ if no such path exists), and let d_i^t be the shortest distance from i to t using paths all the nodes of which, with the possible exception of i, lie in V ($d_i^t = \infty$ if no such path exists).

(a) Show that upon termination, the shortest distance D_{st} from s to t is given by

$$D_{st} = \min_{i \in W} \left\{ d_i^s + d_i^t \right\} = \min_{i \in W \cup V} \left\{ d_i^s + d_i^t \right\} = \min_{i \in V} \left\{ d_i^s + d_i^t \right\}.$$

(b) Show that the conclusion of part (a) holds if the algorithm is terminated once the condition

$$\min_{i \in W} \left\{ d_i^s + d_i^t \right\} \leq \max_{i \in W} d_i^s + \max_{i \in V} d_i^t$$

holds, even if the sets W and V have no node in common.

2.11 (DP on Two Parallel Processors [Las85])

Formulate a DP algorithm to solve the deterministic problem of Section 2.1 on a parallel computer with two processors. One processor should execute a forward algorithm and the other a backward algorithm.

2.12 (Doubling Algorithms)

Consider a deterministic finite-state problem that is time-invariant in the sense that the state and control spaces, the cost per stage, and the system equation

are the same for each stage. Let $J_k(x,y)$ be the optimal cost to reach state y at time k starting from state x at time 0. Show that for all k

$$J_{2k}(x,y) = \min_z \Big\{ J_k(x,z) + J_k(z,y) \Big\}.$$

Discuss how this equation may be used with advantage to solve problems with a large number of stages.

2.13 (Distributed Shortest Path Computation [Ber82a]) (WWW)

Consider the problem of finding a shortest path from nodes $1, 2, \ldots, N$ to node t, and assume that all arc lengths are nonnegative and all cycle lengths are positive. Consider the iteration

$$d_i^{k+1} = \min_j \big[a_{ij} + d_j^k \big], \qquad i = 1, 2, \ldots, N, \tag{2.15}$$

$$d_t^{k+1} = 0.$$

(a) It was shown in Section 2.1 that if the initial condition is $d_i^0 = \infty$ for $i = 1, \ldots, N$ and $d_t^0 = 0$, then the iteration (2.15) yields the shortest distances in N steps. Show that if the initial condition is $d_i^0 = 0$, for all $i = 1, \ldots, N$, t, then the iteration (2.15) yields the shortest distances in a finite number of steps.

(b) Assume that the iteration

$$d_i := \min_j \big[a_{ij} + d_j \big] \tag{2.16}$$

is executed at node i in parallel with the corresponding iteration for d_j at every other node j. However, the times of execution of this iteration at the various nodes are not synchronized. Furthermore, each node i communicates the results of its latest computation of d_i at arbitrary times with potentially large communication delays. Therefore, there is the possibility of a node executing iteration (2.16) several times before receiving a communication from every other neighboring node. Assume that each node never stops executing iteration (2.16) and transmitting the result to the other nodes. Show that the values d_i^T available at time T at the corresponding nodes i are equal to the shortest distances for all T greater than a finite time $\overline{T}$. *Hint*: Let $\overline{d}_i^k$ and $\underline{d}_i^k$ be generated by iteration (2.16) when starting from the first and the second initial conditions in part (a), respectively. Show that for every k there exists a time T_k such that for all $T \geq T_k$ and k, we have $\underline{d}_i^k \leq d_i^T \leq \overline{d}_i^k$. *Note*: For a detailed analysis of asynchronous iterations, including algorithms for shortest paths and DP, see Bertsekas and Tsitsiklis [BeT89], Ch. 6. Distributed asynchronous shortest path algorithms find extensive application in the problem of packet routing in data communication networks. For a related discussion and analysis, see Bertsekas and Gallager [BeG92], Ch. 5.

2.14 (Shortest Paths for an Infinite Number of Nodes)

Consider the shortest path problem of Section 2.3, except that the number of nodes in the graph may be countably infinite (although the number of outgoing arcs from each node is finite). We assume that the length of each arc is a positive integer. Furthermore, there is at least one path from the origin node s to the destination node t. Consider the label correcting algorithm as stated and initialized in Section 2.3.1, except that UPPER is initially set to some integer that is an upper bound to the shortest distance from s to t. Show that the algorithm will terminate in a finite number of steps with UPPER equal to the shortest distance from s to t. *Hint*: Show that there is a finite number of nodes whose shortest distance from s does not exceed the initial value of UPPER.

2.15 (Path Bottleneck Problem)

Consider the framework of the shortest path problem. For any path P, define the *bottleneck arc* of P as an arc that has maximum length over all arcs of P. Consider the problem of finding a path whose length of bottleneck arc is minimum, among the paths connecting an origin node and a destination node. Develop and justify an analog of the label correcting method of Section 2.3.1. *Hint*: Replace $d_i + a_{ij}$ with $\max\{d_i, a_{ij}\}$.

2.16

Air transportation is available between all pairs of n cities, but because of a perverse fare structure, it may be more economical to go from one city to another through intermediate stops. A cost-minded traveler wants to find the minimum cost fare to go from an origin city s to a destination city t. The airfare between cities i and j is denoted by a_{ij}, and for the mth intermediate stop, there is a stopover cost c_m (a_{ij} and c_m are assumed positive). Thus, for example, to go from s to t directly it costs a_{st}, while to go from s to t with intermediate stops at cities i_1 and i_2, it costs $a_{si_1} + c_1 + a_{i_1 i_2} + c_2 + a_{i_2 t}$.

(a) Formulate the problem as a shortest path problem, and identify the nodes, arcs, and arc costs.

(b) Write a corresponding DP algorithm that finds an optimal solution in $n-2$ stages.

(c) Assume that c_m is the same for all m. Devise a rule for detecting that an optimal solution has been found before iteration $n-2$ of the DP algorithm.

2.17

A businessman operates out of a van that he sets up in one of two locations on each day. If he operates in location i (where $i = 1, 2$) on day k, he makes a known and predictable profit, denoted r_k^i. However, each time he moves from one location to the other, he pays a setup cost c. The businessman wants to maximize his total profit over N days.

(a) Show that the problem can be formulated as a shortest path problem, and write the corresponding DP algorithm.

(b) Suppose he is at location i on day k. Let

$$R_k^i = r_k^{\bar{i}} - r_k^i,$$

where $\bar{i}$ denotes the location that is not equal to i. Show that if $R_k^i \leq 0$ it is optimal to stay at location i, while if $R_k^i \geq 2c$, it is optimal to switch.

(c) Suppose that on each day there is a probability of rain p_i at location i independently of rain in the other location, and independently of whether it rained on other days. If he is at location i and it rains, his profit for the day is reduced by a factor β_i. Can the problem still be formulated as a shortest path problem? Write a DP algorithm.

(d) Suppose there is a possibility of rain as in part (c), but the businessman receives an accurate rain forecast just before making the decision to switch or not switch locations. Can the problem still be formulated as a shortest path problem? Write a DP algorithm.

3

Deterministic Continuous-Time Optimal Control

Contents

In this chapter, we provide an introduction to continuous-time deterministic optimal control. We derive the analog of the DP algorithm, which is the Hamilton-Jacobi-Bellman equation. Furthermore, we develop a celebrated theorem of optimal control, the Pontryagin Minimum Principle and its variations. We discuss two different derivations of this theorem, one of which is based on DP. We also illustrate the theorem by means of examples.

3.1 CONTINUOUS-TIME OPTIMAL CONTROL

We consider a continuous-time dynamic system

$$\dot{x}(t) = f\big(x(t), u(t)\big), \qquad 0 \leq t \leq T, \tag{3.1}$$

$$x(0) : \text{given},$$

where $x(t) \in \Re^n$ is the state vector at time t, $\dot{x}(t) \in \Re^n$ is the vector of first order time derivatives of the states at time t, $u(t) \in U \subset \Re^m$ is the control vector at time t, U is the control constraint set, and T is the terminal time. The components of f, x, $\dot{x}$, and u will be denoted by f_i, x_i, $\dot{x}_i$, and u_i, respectively. Thus, the system (3.1) represents the n first order differential equations

$$\frac{dx_i(t)}{dt} = f_i\big(x(t), u(t)\big), \qquad i = 1, \ldots, n.$$

We view $\dot{x}(t)$, $x(t)$, and $u(t)$ as column vectors. We assume that the system function f_i is continuously differentiable with respect to x and is continuous with respect to u. The admissible control functions, also called *control trajectories*, are the piecewise continuous functions $\big\{u(t) \mid t \in [0,T]\big\}$ with $u(t) \in U$ for all $t \in [0,T]$.

We should stress at the outset that the subject of this chapter is highly sophisticated, and it is beyond our scope to develop it according to high standards of mathematical rigor. In particular, we assume that, for any admissible control trajectory $\big\{u(t) \mid t \in [0,T]\big\}$, the system of differential equations (3.1) has a unique solution, which is denoted $\big\{x^u(t) \mid t \in [0,T]\big\}$ and is called the corresponding *state trajectory*. In a more rigorous treatment, the issue of existence and uniqueness of this solution would have to be addressed more carefully.

We want to find an admissible control trajectory $\big\{u(t) \mid t \in [0,T]\big\}$, which, together with its corresponding state trajectory $\big\{x(t) \mid t \in [0,T]\big\}$, minimizes a cost function of the form

$$h\big(x(T)\big) + \int_0^T g\big(x(t), u(t)\big)dt,$$

where the functions g and h are continuously differentiable with respect to x, and g is continuous with respect to u.

Example 3.1.1 (Motion Control)

A unit mass moves on a line under the influence of a force u. Let $x_1(t)$ and $x_2(t)$ be the position and velocity of the mass at time t, respectively. From a given $\big(x_1(0), x_2(0)\big)$ we want to bring the mass "near" a given final position-velocity pair $(\overline{x}_1, \overline{x}_2)$ at time T. In particular, we want to

$$\text{minimize } \big|x_1(T) - \overline{x}_1\big|^2 + \big|x_2(T) - \overline{x}_2\big|^2$$

subject to the control constraint

$$|u(t)| \leq 1, \qquad \text{for all } t \in [0,T].$$

The corresponding continuous-time system is

$$\dot{x}_1(t) = x_2(t), \qquad \dot{x}_2(t) = u(t),$$

and the problem fits the general framework given earlier with cost functions given by

$$h\big(x(T)\big) = \big|x_1(T) - \overline{x}_1\big|^2 + \big|x_2(T) - \overline{x}_2\big|^2,$$

$$g\big(x(t), u(t)\big) = 0, \qquad \text{for all } t \in [0,T].$$

There are many variations of the problem; for example, the final position and/or velocity may be fixed. These variations can be handled by various reformulations of the general continuous-time optimal control problem, which will be given later.

Example 3.1.2 (Resource Allocation)

A producer with production rate $x(t)$ at time t may allocate a portion $u(t)$ of his/her production rate to reinvestment and $1 - u(t)$ to production of a storable good. Thus $x(t)$ evolves according to

$$\dot{x}(t) = \gamma u(t) x(t),$$

where $\gamma > 0$ is a given constant. The producer wants to maximize the total amount of product stored

$$\int_0^T \big(1 - u(t)\big)x(t)dt$$

subject to

$$0 \leq u(t) \leq 1, \qquad \text{for all } t \in [0,T].$$

The initial production rate $x(0)$ is a given positive number.

Example 3.1.3 (Calculus of Variations Problems)

Calculus of variations problems involve finding (possibly multidimensional) curves $x(t)$ with certain optimality properties. They are among the most celebrated problems of applied mathematics and have been worked on by many of the illustrious mathematicians of the past 300 years (Euler, Lagrange, Bernoulli, Gauss, etc.). We will see that calculus of variations problems can be reformulated as optimal control problems. We illustrate this reformulation by a simple example.

Suppose that we want to find a minimum length curve that starts at a given point and ends at a given line. The answer is of course evident, but we want to derive it by using a continuous-time optimal control formulation. Without loss of generality, we let $(0, \alpha)$ be the given point, and we let the given line be the vertical line that passes through $(T, 0)$, as shown in Fig. 3.1.1. Let also $\big(t, x(t)\big)$ be the points of the curve $(0 \leq t \leq T)$. The portion of the curve joining the points $\big(t, x(t)\big)$ and $\big(t+dt, x(t+dt)\big)$ can be approximated, for small dt, by the hypotenuse of a right triangle with sides dt and $\dot{x}(t)dt$. Thus the length of this portion is

$$\sqrt{(dt)^2 + \big(\dot{x}(t)\big)^2 (dt)^2},$$

which is equal to

$$\sqrt{1 + \big(\dot{x}(t)\big)^2}\, dt.$$

The length of the entire curve is the integral over $[0, T]$ of this expression, so the problem is to

$$\text{minimize} \quad \int_0^T \sqrt{1 + \big(\dot{x}(t)\big)^2}\, dt$$
$$\text{subject to} \quad x(0) = \alpha.$$

To reformulate the problem as a continuous-time optimal control problem, we introduce a control u and the system equation

$$\dot{x}(t) = u(t), \qquad x(0) = \alpha.$$

Our problem then becomes

$$\text{minimize} \quad \int_0^T \sqrt{1 + \big(u(t)\big)^2}\, dt.$$

This is a problem that fits our continuous-time optimal control framework.

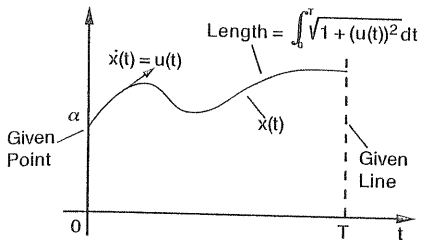


Figure 3.1.1 Problem of finding a curve of minimum length from a given point to a given line, and its formulation as a calculus of variations problem.

3.2 THE HAMILTON-JACOBI-BELLMAN EQUATION

We will now derive informally a partial differential equation, which is satisfied by the optimal cost-to-go function, under certain assumptions. This equation is the continuous-time analog of the DP algorithm, and will be motivated by applying DP to a discrete-time approximation of the continuous-time optimal control problem.

Let us divide the time horizon $[0, T]$ into N pieces using the discretization interval

$$\delta = \frac{T}{N}.$$

We denote

$$x_k = x(k\delta), \qquad k = 0, 1, \ldots, N,$$
$$u_k = u(k\delta), \qquad k = 0, 1, \ldots, N,$$

and we approximate the continuous-time system by

$$x_{k+1} = x_k + f(x_k, u_k) \cdot \delta$$

and the cost function by

$$h(x_N) + \sum_{k=0}^{N-1} g(x_k, u_k) \cdot \delta.$$

We now apply DP to the discrete-time approximation. Let

$J^*(t, x)$: Optimal cost-to-go at time t and state x for the continuous-time problem,

$\tilde{J}^*(t, x)$: Optimal cost-to-go at time t and state x for the discrete-time approximation.

The DP equations are

$$\tilde{J}^*(N\delta, x) = h(x),$$

$$\tilde{J}^*(k\delta, x) = \min_{u\in U}\Big[g(x,u)\cdot\delta + \tilde{J}^*\big((k+1)\cdot\delta, x + f(x,u)\cdot\delta\big)\Big], \quad k = 0, \ldots, N-1.$$

Assuming that $\tilde{J}^*$ has the required differentiability properties, we expand it into a first order Taylor series around $(k\delta, x)$, obtaining

$$\tilde{J}^*\big((k+1)\cdot\delta, x + f(x,u)\cdot\delta\big) = \tilde{J}^*(k\delta, x) + \nabla_t \tilde{J}^*(k\delta, x)\cdot\delta + \nabla_x \tilde{J}^*(k\delta, x)' f(x,u)\cdot\delta + o(\delta),$$

where $o(\delta)$ represents second order terms satisfying $\lim_{\delta\to 0} o(\delta)/\delta = 0$, ∇_t denotes partial derivative with respect to t, and ∇_x denotes the n-dimensional (column) vector of partial derivatives with respect to x. Substituting in the DP equation, we obtain

$$\tilde{J}^*(k\delta, x) = \min_{u\in U}\big[g(x,u)\cdot\delta + \tilde{J}^*(k\delta, x) + \nabla_t \tilde{J}^*(k\delta, x)\cdot\delta + \nabla_x \tilde{J}^*(k\delta, x)' f(x,u)\cdot\delta + o(\delta)\big].$$

Canceling $\tilde{J}^*(k\delta, x)$ from both sides, dividing by δ, and taking the limit as $\delta \to 0$, while assuming that the discrete-time cost-to-go function yields in the limit its continuous-time counterpart,

$$\lim_{k\to\infty,\ \delta\to 0,\ k\delta = t} \tilde{J}^*(k\delta, x) = J^*(t,x), \qquad \text{for all } t, x,$$

we obtain the following equation for the cost-to-go function $J^*(t,x)$:

$$0 = \min_{u\in U}\big[g(x,u) + \nabla_t J^*(t,x) + \nabla_x J^*(t,x)' f(x,u)\big], \qquad \text{for all } t, x,$$

with the boundary condition $J^*(T,x) = h(x)$.

This is the *Hamilton-Jacobi-Bellman (HJB) equation*. It is a *partial* differential equation, which should be satisfied for all time-state pairs (t,x) by the cost-to-go function $J^*(t,x)$, based on the preceding informal derivation, which assumed among other things, differentiability of $J^*(t,x)$. In fact we do not know a priori that $J^*(t,x)$ is differentiable, so we do not know if $J^*(t,x)$ solves this equation. However, it turns out that if we can solve the HJB equation analytically or computationally, then we can obtain an optimal control policy by minimizing its right-hand-side. This is shown in the following proposition, whose statement is reminiscent of a corresponding statement for discrete-time DP: if we can execute the DP algorithm, which may not be possible due to excessive computational requirements, we can find an optimal policy by minimization of the right-hand side.

Proposition 3.2.1: (Sufficiency Theorem) Suppose $V(t,x)$ is a solution to the HJB equation; that is, V is continuously differentiable in t and x, and is such that

$$0 = \min_{u\in U}\big[g(x,u) + \nabla_t V(t,x) + \nabla_x V(t,x)' f(x,u)\big], \qquad \text{for all } t, x, \tag{3.2}$$

$$V(T,x) = h(x), \qquad \text{for all } x. \tag{3.3}$$

Suppose also that $\mu^*(t,x)$ attains the minimum in Eq. (3.2) for all t and x. Let $\{x^*(t) \mid t \in [0,T]\}$ be the state trajectory obtained from the given initial condition $x(0)$ when the control trajectory $u^*(t) = \mu^*\big(t, x^*(t)\big)$, $t \in [0,T]$ is used [that is, $x^*(0) = x(0)$ and for all $t \in [0,T]$, $\dot{x}^*(t) = f\big(x^*(t), \mu^*(t, x^*(t))\big)$; we assume that this differential equation has a unique solution starting at any pair (t,x) and that the control trajectory $\{\mu^*(t, x^*(t)) \mid t \in [0,T]\}$ is piecewise continuous as a function of t]. Then V is equal to the optimal cost-to-go function, i.e.,

$$V(t,x) = J^*(t,x), \qquad \text{for all } t, x.$$

Furthermore, the control trajectory $\{u^*(t) \mid t \in [0,T]\}$ is optimal.

Proof: Let $\{\hat{u}(t) \mid t \in [0,T]\}$ be any admissible control trajectory and let $\{\hat{x}(t) \mid t \in [0,T]\}$ be the corresponding state trajectory. From Eq. (3.2) we have for all $t \in [0,T]$

$$0 \le g\big(\hat{x}(t), \hat{u}(t)\big) + \nabla_t V\big(t, \hat{x}(t)\big) + \nabla_x V\big(t, \hat{x}(t)\big)' f\big(\hat{x}(t), \hat{u}(t)\big).$$

Using the system equation $\dot{\hat{x}}(t) = f\big(\hat{x}(t), \hat{u}(t)\big)$, the right-hand side of the above inequality is equal to the expression

$$g\big(\hat{x}(t), \hat{u}(t)\big) + \frac{d}{dt}\big(V(t, \hat{x}(t))\big),$$

where $d/dt(\cdot)$ denotes total derivative with respect to t. Integrating this expression over $t \in [0,T]$, and using the preceding inequality, we obtain

$$0 \le \int_0^T g\big(\hat{x}(t), \hat{u}(t)\big)dt + V\big(T, \hat{x}(T)\big) - V\big(0, \hat{x}(0)\big).$$

Thus by using the terminal condition $V(T,x) = h(x)$ of Eq. (3.3) and the initial condition $\hat{x}(0) = x(0)$, we have

$$V\big(0, x(0)\big) \le h\big(\hat{x}(T)\big) + \int_0^T g\big(\hat{x}(t), \hat{u}(t)\big)dt.$$

If we use $u^*(t)$ and $x^*(t)$ in place of $\hat{u}(t)$ and $\hat{x}(t)$, respectively, the preceding inequalities becomes equalities, and we obtain

$$V\big(0,x(0)\big) = h\big(x^*(T)\big) + \int_0^T g\big(x^*(t),u^*(t)\big)dt.$$

Therefore the cost corresponding to $\{u^*(t) \mid t \in [0,T]\}$ is $V\big(0,x(0)\big)$ and is no larger than the cost corresponding to any other admissible control trajectory $\{\hat{u}(t) \mid t \in [0,T]\}$. It follows that $\{u^*(t) \mid t \in [0,T]\}$ is optimal and that

$$V\big(0,x(0)\big) = J^*\big(0,x(0)\big).$$

We now note that the preceding argument can be repeated with any initial time $t \in [0,T]$ and any initial state x. We thus obtain

$$V(t,x) = J^*(t,x), \qquad \text{for all } t,x.$$

Q.E.D.

Example 3.2.1

To illustrate the HJB equation, let us consider a simple example involving the scalar system

$$\dot{x}(t) = u(t),$$

with the constraint $|u(t)| \leq 1$ for all $t \in [0,T]$. The cost is

$$\frac{1}{2}\big(x(T)\big)^2.$$

The HJB equation here is

$$0 = \min_{|u|\leq 1}\big[\nabla_t V(t,x) + \nabla_x V(t,x)u\big], \qquad \text{for all } t,x, \tag{3.4}$$

with the terminal condition

$$V(T,x) = \frac{1}{2}x^2. \tag{3.5}$$

There is an evident candidate for optimality, namely moving the state towards 0 as quickly as possible, and keeping it at 0 once it is at 0. The corresponding control policy is

$$\mu^*(t,x) = -\text{sgn}(x) = \begin{cases} 1 & \text{if } x < 0 \\ 0 & \text{if } x = 0, \\ -1 & \text{if } x > 0. \end{cases} \tag{3.6}$$

For a given initial time t and initial state x, the cost associated with this policy can be calculated to be

$$J^*(t,x) = \frac{1}{2}\Big(\max\big\{0,\,|x| - (T-t)\big\}\Big)^2. \tag{3.7}$$

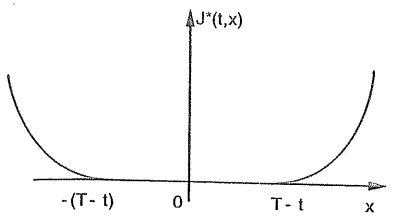


Figure 3.2.1 Optimal cost-to-go function $J^*(t,x)$ for Example 3.2.1.

This function, which is illustrated in Fig. 3.2.1, satisfies the terminal condition (3.5), since $J^*(T,x) = (1/2)x^2$. Let us verify that this function also satisfies the HJB Eq. (3.4), and that $u = -\text{sgn}(x)$ attains the minimum in the right-hand side of the equation for all t and x. Proposition 3.2.1 will then guarantee that the state and control trajectories corresponding to the policy $\mu^*(t,x)$ are optimal.

Indeed, we have

$$\nabla_t J^*(t,x) = \max\big\{0,\,|x| - (T-t)\big\},$$

$$\nabla_x J^*(t,x) = \text{sgn}(x)\cdot\max\big\{0,\,|x| - (T-t)\big\}.$$

Substituting these expressions, the HJB Eq. (3.4) becomes

$$0 = \min_{|u|\leq 1}\big[1 + \text{sgn}(x)\cdot u\big]\max\big\{0,\,|x| - (T-t)\big\}, \tag{3.8}$$

which can be seen to hold as an identity for all (t,x). Furthermore, the minimum is attained for $u = -\text{sgn}(x)$. We therefore conclude based on Prop. 3.2.1 that $J^*(t,x)$ as given by Eq. (3.7) is indeed the optimal cost-to-go function, and that the policy defined by Eq. (3.6) is optimal. Note, however, that the optimal policy is not unique. Based on Prop. 3.2.1, any policy for which the minimum is attained in Eq. (3.8) is optimal. In particular, when $|x(t)| \leq T - t$, applying any control from the range $[-1,1]$ is optimal.

The preceding derivation generalizes to the case of the cost

$$h\big(x(T)\big),$$

where h is a nonnegative differentiable convex function with $h(0) = 0$. The corresponding optimal cost-to-go function is

$$J^*(t,x) = \begin{cases} h\big(x-(T-t)\big) & \text{if } x > T-t, \\ h\big(x+(T-t)\big) & \text{if } x < -(T-t), \\ 0 & \text{if } |x| \leq T-t, \end{cases}$$

and can be similarly verified to be a solution of the HJB equation.

Example 3.2.2 (Linear-Quadratic Problems)

Consider the n-dimensional linear system

$$\dot{x}(t) = Ax(t) + Bu(t),$$

where A and B are given matrices, and the quadratic cost

$$x(T)'Q_T x(T) + \int_0^T \big(x(t)'Qx(t) + u(t)'Ru(t)\big)dt,$$

where the matrices Q_T and Q are symmetric positive semidefinite, and the matrix R is symmetric positive definite (Appendix A defines positive definite and semidefinite matrices). The HJB equation is

$$0 = \min_{u\in\Re^m}\big[x'Qx + u'Ru + \nabla_t V(t,x) + \nabla_x V(t,x)'(Ax+Bu)\big], \qquad (3.9)$$

$$V(T,x) = x'Q_T x.$$

Let us try a solution of the form

$$V(t,x) = x'K(t)x, \qquad K(t): n\times n \text{ symmetric},$$

and see if we can solve the HJB equation. We have $\nabla_x V(t,x) = 2K(t)x$ and $\nabla_t V(t,x) = x'\dot{K}(t)x$, where $\dot{K}(t)$ is the matrix with elements the first order derivatives of the elements of $K(t)$ with respect to time. By substituting these expressions in Eq. (3.9), we obtain

$$0 = \min_u\big[x'Qx + u'Ru + x'\dot{K}(t)x + 2x'K(t)Ax + 2x'K(t)Bu\big]. \qquad (3.10)$$

The minimum is attained at a u for which the gradient with respect to u is zero, that is,

$$2B'K(t)x + 2Ru = 0$$

or

$$u = -R^{-1}B'K(t)x. \qquad (3.11)$$

Substituting the minimizing value of u in Eq. (3.10), we obtain

$$0 = x'\big(\dot{K}(t) + K(t)A + A'K(t) - K(t)BR^{-1}B'K(t) + Q\big)x, \quad \text{for all } (t,x).$$

Therefore, in order for $V(t,x) = x'K(t)x$ to solve the HJB equation, $K(t)$ must satisfy the following matrix differential equation (known as the *continuous-time Riccati equation*)

$$\dot{K}(t) = -K(t)A - A'K(t) + K(t)BR^{-1}B'K(t) - Q \qquad (3.12)$$

with the terminal condition

$$K(T) = Q_T. \qquad (3.13)$$

Reversing the argument, we see that if $K(t)$ is a solution of the Riccati equation (3.12) with the boundary condition (3.13), then $V(t,x) = x'K(t)x$ is a solution of the HJB equation. Thus, by using Prop. 3.2.1, we conclude that the optimal cost-to-go function is

$$J^*(t,x) = x'K(t)x.$$

Furthermore, in view of the expression derived for the control that minimizes in the right-hand side of the HJB equation [cf. Eq. (3.11)], an optimal policy is

$$\mu^*(t,x) = -R^{-1}B'K(t)x.$$

3.3 THE PONTRYAGIN MINIMUM PRINCIPLE

In this section we discuss the continuous-time and the discrete-time versions of the Minimum Principle, starting with a DP-based informal argument.

3.3.1 An Informal Derivation Using the HJB Equation

Recall the HJB equation

$$0 = \min_{u\in U}\big[g(x,u) + \nabla_t J^*(t,x) + \nabla_x J^*(t,x)'f(x,u)\big], \quad \text{for all } t,x, \qquad (3.14)$$

$$J^*(T,x) = h(x), \qquad \text{for all } x. \qquad (3.15)$$

We argued that the optimal cost-to-go function $J^*(t,x)$ satisfies this equation under some conditions. Furthermore, the sufficiency theorem of the preceding section suggests that if for a given initial state $x(0)$, the control trajectory $\{u^*(t) \mid t\in[0,T]\}$ is optimal with corresponding state trajectory $\{x^*(t) \mid t\in[0,T]\}$, then for all $t\in[0,T]$,

$$u^*(t) = \arg\min_{u\in U}\Big[g\big(x^*(t),u\big) + \nabla_x J^*\big(t,x^*(t)\big)'f\big(x^*(t),u\big)\Big]. \qquad (3.16)$$

Note that to obtain the optimal control trajectory via this equation, we do not need to know $\nabla_x J^*$ at *all* values of x and t; it is sufficient to know $\nabla_x J^*$ at only *one* value of x for each t, that is, to know only $\nabla_x J^*\big(t,x^*(t)\big)$.

The Minimum Principle is basically the preceding Eq. (3.16). Its application is facilitated by streamlining the computation of $\nabla_x J^*\big(t,x^*(t)\big)$. It turns out that we can often calculate $\nabla_x J^*\big(t,x^*(t)\big)$ along the optimal state trajectory far more easily than we can solve the HJB equation. In particular, $\nabla_x J^*\big(t,x^*(t)\big)$ satisfies a certain differential equation, called the *adjoint equation*. We will derive this equation informally by differentiating the HJB equation (3.14). We first need the following lemma, which indicates how to differentiate functions involving minima.

Lemma 3.3.1: Let $F(t,x,u)$ be a continuously differentiable function of $t\in\Re$, $x\in\Re^n$, and $u\in\Re^m$, and let U be a convex subset of $\Re^m$. Assume that $\mu^*(t,x)$ is a continuously differentiable function such that

$$\mu^*(t,x) = \arg\min_{u\in U} F(t,x,u), \qquad \text{for all } t,x.$$

Then

$$\nabla_t \left\{ \min_{u \in U} F(t,x,u) \right\} = \nabla_t F\big(t,x,\mu^*(t,x)\big), \qquad \text{for all } t,x,$$

$$\nabla_x \left\{ \min_{u \in U} F(t,x,u) \right\} = \nabla_x F\big(t,x,\mu^*(t,x)\big), \qquad \text{for all } t,x.$$

[Note: On the left-hand side, $\nabla_t\{\cdot\}$ and $\nabla_x\{\cdot\}$ denote the gradients of the function $G(t,x) = \min_{u \in U} F(t,x,u)$ with respect to t and x, respectively. On the right-hand side, ∇_t and ∇_x denote the vectors of partial derivatives of F with respect to t and x, respectively, evaluated at $\big(t,x,\mu^*(t,x)\big)$.]

Proof: For notational simplicity, denote $y = (t,x)$, $F(y,u) = F(t,x,u)$, and $\mu^*(y) = \mu^*(t,x)$. Since $\min_{u \in U} F(y,u) = F\big(y,\mu^*(y)\big)$,

$$\nabla \left\{ \min_{u \in U} F(y,u) \right\} = \nabla_y F\big(y,\mu^*(y)\big) + \nabla\mu^*(y) \nabla_u F\big(y,\mu^*(y)\big).$$

We will prove the result by showing that the second term in the right-hand side above is zero. This is true when $U = \Re^m$, because then $\mu^*(y)$ is an unconstrained minimum of $F(y,u)$ and $\nabla_u F\big(y,\mu^*(y)\big) = 0$. More generally, for every fixed y, we have

$$\big(u - \mu^*(y)\big)' \nabla_u F\big(y,\mu^*(y)\big) \geq 0, \qquad \text{for all } u \in U,$$

[see Eq. (B.2) in Appendix B]. Now by Taylor's Theorem, we have that when y changes to $y + \Delta y$, the minimizing $\mu^*(y)$ changes from $\mu^*(y)$ to some vector $\mu^*(y + \Delta y) = \mu^*(y) + \nabla\mu^*(y)'\Delta y + o\big(\|\Delta y\|\big)$ of U, so

$$\big(\nabla\mu^*(y)'\Delta y + o(\|\Delta y\|)\big)' \nabla_u F\big(y,\mu^*(y)\big) \geq 0, \qquad \text{for all } \Delta y,$$

implying that

$$\nabla\mu^*(y) \nabla_u F\big(y,\mu^*(y)\big) = 0.$$

Q.E.D.

Consider the HJB equation (3.14), and for any (t,x), suppose that $\mu^*(t,x)$ is a control attaining the minimum in the right-hand side. We make the restrictive assumptions that U is a convex set, and that $\mu^*(t,x)$ is continuously differentiable in (t,x), so that we can use Lemma 3.3.1. (We note, however, that alternative derivations of the Minimum Principle do not require these assumptions; see Section 3.3.2.)

We differentiate both sides of the HJB equation with respect to x and with respect to t. In particular, we set to zero the gradient with respect to x and t of the function

$$g\big(x,\mu^*(t,x)\big) + \nabla_t J^*(t,x) + \nabla_x J^*(t,x)' f\big(x,\mu^*(t,x)\big),$$

and we rely on Lemma 3.3.1 to disregard the terms involving the derivatives of $\mu^*(t,x)$ with respect to t and x. We obtain for all (t,x),

$$\begin{aligned} 0 = {} & \nabla_x g\big(x,\mu^*(t,x)\big) + \nabla^2_{xt} J^*(t,x) + \nabla^2_{xx} J^*(t,x) f\big(x,\mu^*(t,x)\big) \\ & + \nabla_x f\big(x,\mu^*(t,x)\big) \nabla_x J^*(t,x), \end{aligned} \tag{3.17}$$

$$0 = \nabla^2_{tt} J^*(t,x) + \nabla^2_{xt} J^*(t,x)' f\big(x,\mu^*(t,x)\big), \tag{3.18}$$

where $\nabla_x f\big(x,\mu^*(t,x)\big)$ is the matrix

$$\nabla_x f = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_1} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_1}{\partial x_n} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$$

with the partial derivatives evaluated at the argument $\big(x,\mu^*(t,x)\big)$.

The above equations hold for all (t,x). Let us specialize them along an optimal state and control trajectory $\big\{\big(x^*(t),u^*(t)\big) \mid t \in [0,T]\big\}$, where $u^*(t) = \mu^*\big(t,x^*(t)\big)$ for all $t \in [0,T]$. We have for all t,

$$\dot{x}^*(t) = f\big(x^*(t),u^*(t)\big),$$

so that the term

$$\nabla^2_{xt} J^*\big(t,x^*(t)\big) + \nabla^2_{xx} J^*\big(t,x^*(t)\big) f\big(x^*(t),u^*(t)\big)$$

in Eq. (3.17) is equal to the following total derivative with respect to t

$$\frac{d}{dt}\Big(\nabla_x J^*\big(t,x^*(t)\big)\Big).$$

Similarly, the term

$$\nabla^2_{tt} J^*\big(t,x^*(t)\big) + \nabla^2_{xt} J^*\big(t,x^*(t)\big)' f\big(x^*(t),u^*(t)\big)$$

in Eq. (3.18) is equal to the total derivative

$$\frac{d}{dt}\Big(\nabla_t J^*\big(t,x^*(t)\big)\Big).$$

Thus, by denoting

$$p(t) = \nabla_x J^*\big(t,x^*(t)\big), \tag{3.19}$$

$$p_0(t) = \nabla_t J^*\big(t, x^*(t)\big), \tag{3.20}$$

Eq. (3.17) becomes

$$\dot{p}(t) = -\nabla_x f\big(x^*(t), u^*(t)\big)p(t) - \nabla_x g\big(x^*(t), u^*(t)\big) \tag{3.21}$$

and Eq. (3.18) becomes

$$\dot{p}_0(t) = 0$$

or equivalently,

$$p_0(t) = \text{constant}, \qquad \text{for all } t \in [0,T]. \tag{3.22}$$

Equation (3.21) is a system of n first order differential equations known as the *adjoint equation*. From the boundary condition

$$J^*(T,x) = h(x), \qquad \text{for all } x,$$

we have, by differentiation with respect to x, the relation $\nabla_x J^*(T,x) = \nabla h(x)$, and by using the definition $\nabla_x J^*\big(t, x^*(t)\big) = p(t)$, we obtain

$$p(T) = \nabla h\big(x^*(T)\big). \tag{3.23}$$

Thus, we have a terminal boundary condition for the adjoint equation (3.21).

To summarize, along optimal state and control trajectories $x^*(t)$, $u^*(t)$, $t \in [0,T]$, the adjoint equation (3.21) holds together with the boundary condition (3.23), while Eq. (3.16) and the definition of $p(t)$ imply that $u^*(t)$ satisfies

$$u^*(t) = \arg\min_{u \in U} \Big[g\big(x^*(t), u\big) + p(t)' f\big(x^*(t), u\big)\Big], \qquad \text{for all } t \in [0,T]. \tag{3.24}$$

Hamiltonian Formulation

Motivated by the condition (3.24), we introduce the Hamiltonian function mapping triplets $(x,u,p) \in \Re^n \times \Re^m \times \Re^n$ to real numbers and given by

$$H(x,u,p) = g(x,u) + p' f(x,u).$$

Note that both the system and the adjoint equations can be compactly written in terms of the Hamiltonian as

$$\dot{x}^*(t) = \nabla_p H\big(x^*(t), u^*(t), p(t)\big), \qquad \dot{p}(t) = -\nabla_x H\big(x^*(t), u^*(t), p(t)\big).$$

We state the Minimum Principle in terms of the Hamiltonian function.

Proposition 3.3.1: (Minimum Principle) Let $\{u^*(t) \mid t \in [0,T]\}$ be an optimal control trajectory and let $\{x^*(t) \mid t \in [0,T]\}$ be the corresponding state trajectory, i.e.,

$$\dot{x}^*(t) = f\big(x^*(t), u^*(t)\big), \qquad x^*(0) = x(0) : \text{given}.$$

Let also $p(t)$ be the solution of the adjoint equation

$$\dot{p}(t) = -\nabla_x H\big(x^*(t), u^*(t), p(t)\big),$$

with the boundary condition

$$p(T) = \nabla h\big(x^*(T)\big),$$

where $h(\cdot)$ is the terminal cost function. Then, for all $t \in [0,T]$,

$$u^*(t) = \arg\min_{u \in U} H\big(x^*(t), u, p(t)\big).$$

Furthermore, there is a constant C such that

$$H\big(x^*(t), u^*(t), p(t)\big) = C, \qquad \text{for all } t \in [0,T].$$

All the assertions of the Minimum Principle have been (informally) derived earlier except for the last assertion. To see why the Hamiltonian function is constant for $t \in [0,T]$ along the optimal state and control trajectories, note that by Eqs. (3.14), (3.19), and (3.20), we have for all $t \in [0,T]$

$$H\big(x^*(t), u^*(t), p(t)\big) = -\nabla_t J^*\big(t, x^*(t)\big) = -p_0(t),$$

and $p_0(t)$ is constant by Eq. (3.22). We should note here that the Hamiltonian function need not be constant along the optimal trajectory if the system and cost are not time-independent, contrary to our assumption thus far (see Section 3.4.4).

It is important to note that the Minimum Principle provides *necessary* optimality conditions, so all optimal control trajectories satisfy these conditions, but if a control trajectory satisfies these conditions, it is not necessarily optimal. Further analysis is needed to guarantee optimality. One method that often works is to prove that an optimal control trajectory exists, and to verify that there is only one control trajectory satisfying the conditions of the Minimum Principle (or that all control trajectories satisfying these conditions have equal cost). Another possibility to conclude optimality arises when the system function f is linear in (x,u), the

constraint set U is convex, and the cost functions h and g are convex. Then it can be shown that the conditions of the Minimum Principle are both necessary and sufficient for optimality.

The Minimum Principle can often be used as the basis of a numerical solution. One possibility is the *two-point boundary problem method*. In this method, we use the minimum condition

$$u^*(t) = \arg\min_{u\in U} H\big(x^*(t), u, p(t)\big),$$

to express $u^*(t)$ in terms of $x^*(t)$ and $p(t)$. We then substitute the result into the system and the adjoint equations, to obtain a set of $2n$ first order differential equations in the components of $x^*(t)$ and $p(t)$. These equations can be solved using the split boundary conditions

$$x^*(0) = x(0), \qquad p(T) = \nabla h\big(x^*(T)\big).$$

The number of boundary conditions (which is $2n$) is equal to the number of differential equations, so that we generally expect to be able to solve these differential equations numerically (although in practice this may not be simple).

Using the Minimum Principle to obtain an analytical solution is possible in many interesting problems, but typically requires considerable creativity. We give some simple examples.

Example 3.3.1 (Calculus of Variations Continued)

Consider the problem of finding the curve of minimum length from a point $(0, \alpha)$ to the line $\{(T, y) \mid y \in \Re\}$. In Section 3.1 , we formulated this problem as the problem of finding an optimal control trajectory $\{u(t) \mid t \in [0, T]\}$ that minimizes

$$\int_0^T \sqrt{1 + \big(u(t)\big)^2}\, dt$$

subject to

$$\dot{x}(t) = u(t), \qquad x(0) = \alpha.$$

Let us apply the preceding necessary conditions. The Hamiltonian is

$$H(x, u, p) = \sqrt{1 + u^2} + pu,$$

and the adjoint equation is

$$\dot{p}(t) = 0, \qquad p(T) = 0.$$

It follows that

$$p(t) = 0, \qquad \text{for all } t \in [0, T],$$

so minimization of the Hamiltonian gives

$$u^*(t) = \arg\min_{u\in\Re} \sqrt{1 + u^2} = 0, \qquad \text{for all } t \in [0, T].$$

Therefore we have $\dot{x}^*(t) = 0$ for all t, which implies that $x^*(t)$ is constant. Using the initial condition $x^*(0) = \alpha$, it follows that

$$x^*(t) = \alpha, \qquad \text{for all } t \in [0, T].$$

We thus obtain the (a priori obvious) optimal solution, which is the horizontal line passing through $(0, \alpha)$. Note that since the Minimum Principle is only a necessary condition for optimality, it does not guarantee that the horizontal line solution is optimal. For such a guarantee, we should invoke the linearity of the system function, and the convexity of the cost function. As mentioned (but not proved) earlier, under these conditions, the Minimum Principle is both necessary and sufficient for optimality.

Example 3.3.2 (Resource Allocation Continued)

Consider the optimal production problem (Example 3.1.2). We want to maximize

$$\int_0^T \big(1 - u(t)\big)x(t)dt$$

subject to

$$0 \le u(t) \le 1, \qquad \text{for all } t \in [0, T],$$

$$\dot{x}(t) = \gamma u(t)x(t), \quad x(0) > 0 : \text{given}.$$

The Hamiltonian is

$$H(x, u, p) = (1 - u)x + p\gamma ux.$$

The adjoint equation is

$$\dot{p}(t) = -\gamma u^*(t)p(t) - 1 + u^*(t),$$
$$p(T) = 0.$$

Maximization of the Hamiltonian over $u \in [0, 1]$ yields

$$u^*(t) = \begin{cases} 0 & \text{if } p(t) < \frac{1}{\gamma}, \\ 1 & \text{if } p(t) \ge \frac{1}{\gamma}. \end{cases}$$

Since $p(T) = 0$, for t close to T we will have $p(t) < 1/\gamma$ and $u^*(t) = 0$. Therefore, for t near T the adjoint equation has the form $\dot{p}(t) = -1$ and $p(t)$ has the form shown in Fig. 3.3.1.

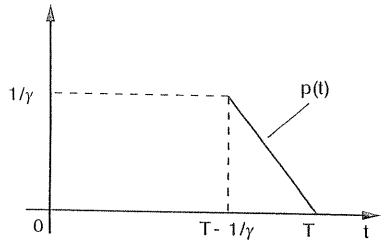


Figure 3.3.1 Form of the adjoint variable $p(t)$ for t near T in the resource allocation example.

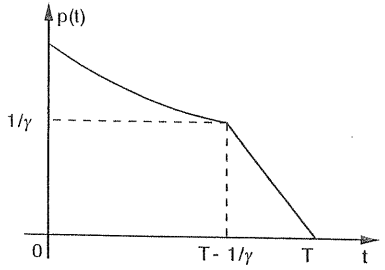
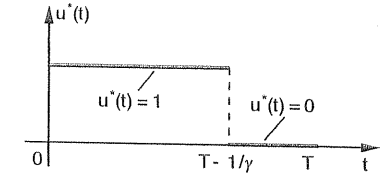


Figure 3.3.2 Form of the adjoint variable $p(t)$ and the optimal control in the resource allocation example.



Thus, near $t = T$, $p(t)$ decreases with slope -1. For $t = T - 1/\gamma$, $p(t)$ is equal to $1/\gamma$, so $u^*(t)$ changes to $u^*(t) = 1$. It follows that for $t < T - 1/\gamma$, the adjoint equation is

$$\dot{p}(t) = -\gamma p(t)$$

or

$$p(t) = e^{-\gamma t} \cdot \text{constant}.$$

Piecing together $p(t)$ for t greater and less than $T - 1/\gamma$, we obtain the form shown in Fig. 3.3.2 for $p(t)$ and $u^*(t)$. Note that if $T < 1/\gamma$, the optimal control is $u^*(t) = 0$ for all $t \in [0, T]$; that is, for a short enough horizon, it does not pay to reinvest at any time.

Example 3.3.3 (A Linear-Quadratic Problem)

Consider the one-dimensional linear system

$$\dot{x}(t) = ax(t) + bu(t),$$

where a and b are given scalars. We want to find an optimal control over a given interval $[0, T]$ that minimizes the quadratic cost

$$\frac{1}{2}q \cdot \big(x(T)\big)^2 + \frac{1}{2}\int_0^T \big(u(t)\big)^2 dt,$$

where q is a given positive scalar. There are no constraints on the control, so we have a special case of the linear-quadratic problem of Example 3.2.2. We will solve this problem via the Minimum Principle.

The Hamiltonian here is

$$H(x, u, p) = \frac{1}{2}u^2 + p(ax + bu),$$

and the adjoint equation is

$$\dot{p}(t) = -ap(t),$$

with the terminal condition

$$p(T) = qx^*(T).$$

The optimal control is obtained by minimizing the Hamiltonian with respect to u, yielding

$$u^*(t) = \arg\min_u \left[\frac{1}{2}u^2 + p(t)\big(ax^*(t) + bu\big)\right] = -bp(t). \tag{3.25}$$

We will extract the optimal solution from these conditions using two different approaches.

In the first approach, we solve the two-point boundary value problem discussed following Prop. 3.3.1. In particular, by eliminating the control from the system equation using Eq. (3.25), we obtain

$$\dot{x}^*(t) = ax^*(t) - b^2 p(t).$$

Also, from the adjoint equation, we see that

$$p(t) = e^{-at}\xi, \qquad \text{for all } t \in [0, T],$$

where $\xi = p(0)$ is an unknown parameter. The last two equations yield

$$\dot{x}^*(t) = ax^*(t) - b^2 e^{-at}\xi. \tag{3.26}$$

This differential equation, together with the given initial condition $x^*(0) = x(0)$ and the terminal condition

$$x^*(T) = \frac{e^{-aT}\xi}{q},$$

(which is the terminal condition for the adjoint equation) can be solved for the unknown variable ξ. In particular, it can be verified that the solution of the differential equation (3.26) is given by

$$x^*(t) = x(0)e^{at} + \frac{b^2\xi}{2a}\big(e^{-at} - e^{at}\big),$$

and ξ can be obtained from the last two relations. Given ξ, we obtain $p(t) = e^{-at}\xi$, and from $p(t)$, we can then determine the optimal control trajectory as $u^*(t) = -bp(t)$, $t \in [0,T]$ [cf. Eq. (3.25)].

In the second approach, we basically derive the Riccati equation encountered in Example 3.2.2. In particular, we hypothesize a linear relation between $x^*(t)$ and $p(t)$, that is,

$$K(t)x^*(t) = p(t), \qquad \text{for all } t \in [0,T],$$

and we show that $K(t)$ can be obtained by solving the Riccati equation. Indeed, from Eq. (3.25) we have

$$u^*(t) = -bK(t)x^*(t),$$

which by substitution in the system equation, yields

$$\dot{x}^*(t) = \big(a - b^2K(t)\big)x^*(t).$$

By differentiating the equation $K(t)x^*(t) = p(t)$ and by also using the adjoint equation, we obtain

$$\dot{K}(t)x^*(t) + K(t)\dot{x}^*(t) = \dot{p}(t) = -ap(t) = -aK(t)x^*(t).$$

By combining the last two relations, we have

$$\dot{K}(t)x^*(t) + K(t)\big(a - b^2K(t)\big)x^*(t) = -aK(t)x^*(t),$$

from which we see that $K(t)$ should satisfy

$$\dot{K}(t) = -2aK(t) + b^2\big(K(t)\big)^2.$$

This is the Riccati equation of Example 3.2.2, specialized to the problem of the present example. This equation can be solved using the terminal condition

$$K(T) = q,$$

which is implied by the terminal condition $p(T) = qx^*(T)$ for the adjoint equation. Once $K(t)$ is known, the optimal control is obtained in the closed-loop form $u^*(t) = -bK(t)x^*(t)$. By reversing the preceding arguments, this control can then be shown to satisfy all the conditions of the Minimum Principle.

3.3.2 A Derivation Based on Variational Ideas

In this subsection we outline an alternative and more rigorous proof of the Minimum Principle. This proof is primarily directed towards the advanced reader, and is based on making small variations in the optimal trajectory and comparing it with neighboring trajectories.

For convenience, we restrict attention to the case where the cost is

$$h\big(x(T)\big).$$

The more general cost

$$h\big(x(T)\big) + \int_0^T g\big(x(t),u(t)\big)dt \tag{3.27}$$

can be reformulated as a terminal cost by introducing a new state variable y and the additional differential equation

$$\dot{y}(t) = g\big(x(t),u(t)\big). \tag{3.28}$$

The cost then becomes

$$h\big(x(T)\big) + y(T), \tag{3.29}$$

and the Minimum Principle corresponding to this terminal cost yields the Minimum Principle for the general cost (3.27).

We introduce some assumptions:

Convexity Assumption: For every state x the set

$$D = \big\{f(x,u) \mid u \in U\big\}$$

is convex.

The convexity assumption is satisfied if U is a convex set and f is linear in u [and g is linear in u in the case where there is an integral cost of the form (3.27), which is reformulated as a terminal cost by using the additional state variable y of Eq. (3.28)]. Thus the convexity assumption is quite restrictive. However, the Minimum Principle typically holds without the convexity assumption, because even when the set $D = \{f(x,u) \mid u \in U\}$ is nonconvex, any vector in the convex hull of D can be generated by quick alternation between vectors from D (for an example, see Exercise 3.10). This involves the complicated mathematical concept of *randomized* or *relaxed* controls and will not be discussed further.

Regularity Assumption: Let $u(t)$ and $u^*(t)$, $t \in [0,T]$, be any two admissible control trajectories and let $\{x^*(t) \mid t \in [0,T]\}$ be the state trajectory corresponding to $u^*(t)$. For any $\epsilon \in [0,1]$, the solution $\{x_\epsilon(t) \mid t \in [0,T]\}$ of the system

$$\dot{x}_\epsilon(t) = (1-\epsilon)f\big(x_\epsilon(t), u^*(t)\big) + \epsilon f\big(x_\epsilon(t), u(t)\big), \tag{3.30}$$

with $x_\epsilon(0) = x^*(0)$, satisfies

$$x_\epsilon(t) = x^*(t) + \epsilon\xi(t) + o(\epsilon), \tag{3.31}$$

where $\{\xi(t) \mid t \in [0,T]\}$ is the solution of the linear differential system

$$\dot{\xi}(t) = \nabla_x f\big(x^*(t), u^*(t)\big)\xi(t) + f\big(x^*(t), u(t)\big) - f\big(x^*(t), u^*(t)\big), \tag{3.32}$$

with initial condition $\xi(0) = 0$.

The regularity assumption "typically" holds because from Eq. (3.30) we have

$$\begin{aligned}\dot{x}_\epsilon(t) - \dot{x}^*(t) = {} & f\big(x_\epsilon(t), u^*(t)\big) - f\big(x^*(t), u^*(t)\big) \\ & + \epsilon\Big(f\big(x_\epsilon(t), u(t)\big) - f\big(x_\epsilon(t), u^*(t)\big)\Big),\end{aligned}$$

so from a first order Taylor series expansion we obtain

$$\begin{aligned}\delta\dot{x}(t) = {} & \nabla f\big(x^*(t), u^*(t)\big)'\delta x(t) + o\big(\|\delta x(t)\|\big) \\ & + \epsilon\Big(f\big(x_\epsilon(t), u(t)\big) - f\big(x_\epsilon(t), u^*(t)\big)\Big),\end{aligned}$$

where

$$\delta x(t) = x_\epsilon(t) - x^*(t).$$

Dividing by ϵ and taking the limit as $\epsilon \to 0$, we see that the function

$$\xi(t) = \lim_{\epsilon \to 0} \delta x(t)/\epsilon, \qquad t \in [0,T],$$

should "typically" solve the linear system of differential equations (3.32), while satisfying Eq. (3.31).

In fact, if the system is linear of the form

$$\dot{x}(t) = Ax(t) + Bu(t),$$

where A and B are given matrices, it can be shown that the regularity assumption is satisfied. To see this, note that Eqs. (3.30) and (3.32) take the forms

$$\dot{x}_\epsilon(t) = Ax_\epsilon(t) + Bu^*(t) + \epsilon B\big(u(t) - u^*(t)\big),$$

and

$$\dot{\xi}(t) = A\xi(t) + B\big(u(t) - u^*(t)\big),$$

respectively. Thus, taking into account the initial conditions $x_\epsilon(0) = x^*(0)$ and $\xi(0) = 0$, we see that

$$x_\epsilon(t) = x^*(t) + \epsilon\xi(t), \qquad t \in [0,T],$$

so the regularity condition (3.31) is satisfied.

We now prove the Minimum Principle assuming the convexity and regularity assumptions above. Suppose that $\{u^*(t) \mid t \in [0,T]\}$ is an optimal control trajectory, and let $\{x^*(t) \mid t \in [0,T]\}$ be the corresponding state trajectory. Then for any other admissible control trajectory $\{u(t) \mid t \in [0,T]\}$ and any $\epsilon \in [0,1]$, the convexity assumption guarantees that for each t, there exists a control $\overline{u}(t) \in U$ such that

$$f\big(x_\epsilon(t), \overline{u}(t)\big) = (1-\epsilon)f\big(x_\epsilon(t), u^*(t)\big) + \epsilon f\big(x_\epsilon(t), u(t)\big).$$

Thus, the state trajectory $\{x_\epsilon(t) \mid t \in [0,T]\}$ of Eq. (3.30) corresponds to the admissible control trajectory $\{\overline{u}(t) \mid t \in [0,T]\}$. Hence, using the optimality of $\{x^*(t) \mid t \in [0,T]\}$ and the regularity assumption, we have

$$\begin{aligned}h\big(x^*(T)\big) &\leq h\big(x_\epsilon^*(T)\big) \\ &= h\big(x^*(T) + \epsilon\xi(T) + o(\epsilon)\big) \\ &= h\big(x^*(T)\big) + \epsilon\nabla h\big(x^*(T)\big)'\xi(T) + o(\epsilon),\end{aligned}$$

which implies that

$$\nabla h\big(x^*(T)\big)'\xi(T) \geq 0. \tag{3.33}$$

Using a standard result in the theory of linear differential equations (see e.g. [CoL65]), the solution of the linear differential system (3.32) can be written in closed form as

$$\xi(t) = \Phi(t,\tau)\xi(\tau) + \int_\tau^t \Phi(t,\tau)\Big(f\big(x^*(\tau), u(\tau)\big) - f\big(x^*(\tau), u^*(\tau)\big)\Big)d\tau, \tag{3.34}$$

where the square matrix Φ satisfies for all t and τ,

$$\frac{\partial\Phi(t,\tau)}{\partial\tau} = -\Phi(t,\tau)\nabla_x f\big(x^*(\tau), u^*(\tau)\big)', \tag{3.35}$$

$$\Phi(t,t) = I.$$

Since $\xi(0) = 0$, we have from Eq. (3.34),

$$\xi(T) = \int_0^T \Phi(T,t)\Big(f\big(x^*(t), u(t)\big) - f\big(x^*(t), u^*(t)\big)\Big)dt. \tag{3.36}$$

Define

$$p(T) = \nabla h\big(x^*(T)\big), \qquad p(t) = \Phi(T,t)'p(T), \quad t \in [0,T]. \tag{3.37}$$

By differentiating with respect to t, we obtain

$$\dot{p}(t) = \frac{\partial \Phi(T,t)'}{\partial t} p(T).$$

Combining this equation with Eqs. (3.35) and (3.37), we see that $p(t)$ is generated by the differential equation

$$\dot{p}(t) = -\nabla_x f\big(x^*(t), u^*(t)\big)p(t),$$

with the terminal condition

$$p(T) = \nabla h\big(x^*(T)\big).$$

This is the adjoint equation corresponding to $\big\{\big(x^*(t), u^*(t)\big) \mid t \in [0,T]\big\}$.

Now, to obtain the Minimum Principle, we note that from Eqs. (3.33), (3.36), and (3.37) we have

$$\begin{aligned} 0 &\le p(T)'\xi(T) \\ &= p(T)' \int_0^T \Phi(T,t)\Big(f\big(x^*(t), u(t)\big) - f\big(x^*(t), u^*(t)\big)\Big)dt \\ &= \int_0^T p(t)'\Big(f\big(x^*(t), u(t)\big) - f\big(x^*(t), u^*(t)\big)\Big)dt, \end{aligned} \tag{3.38}$$

from which it can be shown that for all t at which $u^*(\cdot)$ is continuous, we have

$$p(t)'f\big(x^*(t), u^*(t)\big) \le p(t)'f\big(x^*(t), u\big), \qquad \text{for all } u \in U. \tag{3.39}$$

Indeed, if for some $\hat{u} \in U$ and $t_0 \in [0,T]$, we have

$$p(t_0)'f\big(x^*(t_0), u^*(t_0)\big) > p(t_0)'f\big(x^*(t_0), \hat{u}\big),$$

while $\big\{u^*(t) \mid t \in [0,T]\big\}$ is continuous at t_0, we would also have

$$p(t)'f\big(x^*(t), u^*(t)\big) > p(t)'f\big(x^*(t), \hat{u}\big),$$

for all t in some nontrivial interval I containing t_0. By taking

$$u(t) = \begin{cases} \hat{u} & \text{for } t \in I, \\ u^*(t) & \text{for } t \notin I, \end{cases}$$

we would then obtain a contradiction of Eq. (3.38).

We have thus proved the Minimum Principle (3.39) under the convexity and regularity assumptions, and the assumption that there is only a terminal cost $h\big(x(T)\big)$. We have also seen that in the case where the constraint set U is convex and the system is linear, the convexity and regularity assumptions are satisfied. To prove the Minimum Principle for the more general integral cost function (3.27), we can apply the preceding development to the system of differential equations $\dot{x} = f(x,u)$ augmented by the additional Eq. (3.28) and the equivalent terminal cost (3.29). The corresponding convexity and regularity assumptions are automatically satisfied if the constraint set U is convex and the system function $f(x,u)$ as well as the cost function $g(x,u)$ are linear. This is necessary in order to maintain the linearity of the augmented system, thereby maintaining the validity of the regularity assumption.

3.3.3 Minimum Principle for Discrete-Time Problems

In this subsection we briefly derive a version of the Minimum Principle for discrete-time deterministic optimal control problems. Interestingly, it is essential to make some convexity assumptions in order for the Minimum Principle to hold. For continuous-time problems these convexity assumptions are typically not needed, because, as mentioned earlier, the differential system can generate any $\dot{x}(t)$ in the convex hull of the set of possible vectors $f\big(x(t), u(t)\big)$ by quick alternation between different controls (see for example Exercise 3.10).

Suppose that we want to find a control sequence $(u_0, u_1, \ldots, u_{N-1})$ and a corresponding state sequence $(x_0, x_1, \ldots, x_N)$, which minimize

$$J(u) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k),$$

subject to the discrete-time system constraints

$$x_{k+1} = f_k(x_k, u_k), \quad k = 0, \ldots, N-1, \qquad x_0 : \text{ given},$$

and the control constraints

$$u_k \in U_k \subset \Re^m, \qquad k = 0, \ldots, N-1.$$

We first develop an expression for the gradient $\nabla J(u_0, \ldots, u_{N-1})$. We have, using the chain rule,

$$\begin{aligned} \nabla_{u_{N-1}} J(u_0, \ldots, u_{N-1}) &= \nabla_{u_{N-1}} \Big(g_N\big(f_{N-1}(x_{N-1}, u_{N-1})\big) \\ &\qquad + g_{N-1}(x_{N-1}, u_{N-1})\Big) \\ &= \nabla_{u_{N-1}} f_{N-1} \cdot \nabla g_N + \nabla_{u_{N-1}} g_{N-1}, \end{aligned}$$

where all gradients are evaluated along the control trajectory $(u_0, \ldots, u_{N-1})$ and the corresponding state trajectory. Similarly, for all k,

$$\begin{aligned}\nabla_{u_k} J(u_0, \ldots, u_{N-1}) = {} & \nabla_{u_k} f_k \cdot \nabla_{x_{k+1}} f_{k+1} \cdots \nabla_{x_{N-1}} f_{N-1} \cdot \nabla g_N \\ & + \nabla_{u_k} f_k \cdot \nabla_{x_{k+1}} f_{k+1} \cdots \nabla_{x_{N-2}} f_{N-2} \cdot \nabla_{x_{N-1}} g_{N-1} \\ & \cdots \\ & + \nabla_{u_k} f_k \cdot \nabla_{x_{k+1}} g_{k+1} \\ & + \nabla_{u_k} g_k,\end{aligned} \tag{3.40}$$

which can be written in the form

$$\nabla_{u_k} J(u_0, \ldots, u_{N-1}) = \nabla_{u_k} f_k \cdot p_{k+1} + \nabla_{u_k} g_k,$$

for an appropriate vector p_{k+1}, or

$$\nabla_{u_k} J(u_0, \ldots, u_{N-1}) = \nabla_{u_k} H_k(x_k, u_k, p_{k+1}), \tag{3.41}$$

where H_k is the Hamiltonian function defined by

$$H_k(x_k, u_k, p_{k+1}) = g_k(x_k, u_k) + p'_{k+1} f_k(x_k, u_k).$$

It can be seen from Eq. (3.40) that the vectors p_{k+1} are generated backwards by the *discrete-time adjoint equation*

$$p_k = \nabla_{x_k} f_k \cdot p_{k+1} + \nabla_{x_k} g_k, \qquad k = 1, \ldots, N-1,$$

with terminal condition

$$p_N = \nabla g_N.$$

We will assume that the constraint sets U_k are convex, so that we can apply the optimality condition

$$\sum_{k=0}^{N-1} \nabla_{u_k} J(u_0^*, \ldots, u_{N-1}^*)'(u_k - u_k^*) \geq 0,$$

for all feasible $(u_0, \ldots, u_{N-1})$ (see Appendix B). This condition can be decomposed into the N conditions

$$\nabla_{u_k} J(u_0^*, \ldots, u_{N-1}^*)'(u_k - u_k^*) \geq 0, \qquad \text{for all } u_k \in U_k,\ k = 0, \ldots, N-1. \tag{3.42}$$

We thus obtain:

Proposition 3.3.2: (Discrete-Time Minimum Principle) Suppose that $(u_0^*, u_1^* \ldots, u_{N-1}^*)$ is an optimal control trajectory and that $(x_0^*, x_1^*, \ldots, x_N^*)$ is the corresponding state trajectory. Assume also that the constraint sets U_k are convex. Then for all $k = 0, \ldots, N-1$, we have

$$\nabla_{u_k} H_k\big(x_k^*, u_k^*, p_{k+1}\big)'(u_k - u_k^*) \geq 0, \qquad \text{for all } u_k \in U_k, \tag{3.43}$$

where the vectors $p_1, \ldots, p_N$ are obtained from the adjoint equation

$$p_k = \nabla_{x_k} f_k \cdot p_{k+1} + \nabla_{x_k} g_k, \qquad k = 1, \ldots, N-1,$$

with the terminal condition

$$p_N = \nabla g_N(x_N^*).$$

The partial derivatives above are evaluated along the optimal state and control trajectories. If, in addition, the Hamiltonian H_k is a convex function of u_k for any fixed x_k and p_{k+1}, we have

$$u_k^* = \arg \min_{u_k \in U_k} H_k\big(x_k^*, u_k, p_{k+1}\big), \qquad \text{for all } k = 0, \ldots, N-1. \tag{3.44}$$

Proof: Equation (3.43) is a restatement of the necessary condition (3.42) using the expression (3.41) for the gradient of J. If H_k is convex with respect to u_k, Eq. (3.42) is a sufficient condition for the minimum condition (3.44) to hold (see Appendix B). **Q.E.D.**

3.4 EXTENSIONS OF THE MINIMUM PRINCIPLE

We now consider some variations of the continuous-time optimal control problem and derive corresponding variations of the Minimum Principle.

3.4.1 Fixed Terminal State

Suppose that in addition to the initial state $x(0)$, the final state $x(T)$ is given. Then the preceding informal derivations still hold except that the terminal condition $J^*(T, x) = h(x)$ is not true anymore. In effect, here we have

$$J^*(T, x) = \begin{cases} 0 & \text{if } x = x(T), \\ \infty & \text{otherwise.} \end{cases}$$

Thus $J^*(T,x)$ cannot be differentiated with respect to x, and the terminal boundary condition $p(T) = \nabla h\big(x^*(T)\big)$ for the adjoint equation does not hold. However, as compensation, we have the extra condition

$$x(T) : \text{given},$$

thus maintaining the balance between boundary conditions and unknowns.

If only *some* of the terminal states are fixed, that is,

$$x_i(T) : \text{given}, \qquad \text{for all } i \in I,$$

where I is some index set, we have the partial boundary condition

$$p_j(T) = \frac{\partial h\big(x^*(T)\big)}{\partial x_j}, \qquad \text{for all } j \notin I,$$

for the adjoint equation.

Example 3.4.1

Consider the problem of finding the curve of minimum length connecting two points $(0,\alpha)$ and (T,β). This is a fixed endpoint variation of Example 3.3.1 in the preceding section. We have

$$\dot{x}(t) = u(t),$$

$$x(0) = \alpha, \qquad x(T) = \beta,$$

and the cost is

$$\int_0^T \sqrt{1 + \big(u(t)\big)^2}\, dt.$$

The adjoint equation is

$$\dot{p}(t) = 0,$$

implying that

$$p(t) = \text{constant}, \qquad \text{for all } t \in [0,T].$$

Minimization of the Hamiltonian,

$$\min_{u \in \Re} \left[\sqrt{1+u^2} + p(t)u \right],$$

yields

$$u^*(t) = \text{constant}, \qquad \text{for all } t \in [0,T].$$

Thus the optimal trajectory $\big\{x^*(t) \mid t \in [0,T]\big\}$ is a straight line. Since this trajectory must pass through $(0,\alpha)$ and (T,β), we obtain the (a priori obvious) optimal solution shown in Fig. 3.4.1.

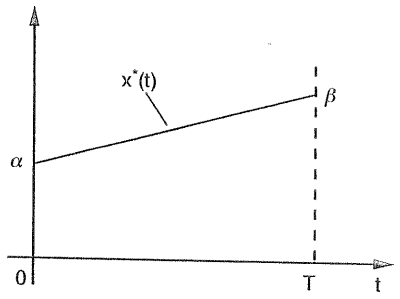


Figure 3.4.1 Optimal solution of the problem of connecting the two points $(0,\alpha)$ and (T,β) with a minimum length curve (cf. Example 3.4.1).

Example 3.4.2 (The Brachistochrone Problem)

In 1696 Johann Bernoulli challenged the mathematical world of his time with a problem that played an instrumental role in the development of the calculus of variations: Given two points A and B, find a curve connecting A and B such that a body moving along the curve under the force of gravity reaches B in minimum time (see Fig. 3.4.2). Let A be $(0,0)$ and B be $(T,-b)$ with $b > 0$. Then it can be seen that the problem is to find $\big\{x(t) \mid t \in [0,T]\big\}$ with $x(0) = 0$ and $x(T) = b$, which minimizes

$$\int_0^T \frac{\sqrt{1 + \big(\dot{x}(t)\big)^2}}{\sqrt{2\gamma x(t)}}\, dt,$$

where γ is the acceleration due to gravity. Here $\big\{\big(t,-x(t)\big) \mid t \in [0,T]\big\}$, is the desired curve, the term $\sqrt{1 + \big(\dot{x}(t)\big)^2}\,dt$ is the length of the curve from $x(t)$ to $x(t+dt)$, and the term $\sqrt{2\gamma x(t)}$ is the velocity of the body upon reaching the level $x(t)$ [if m and v denote the mass and the velocity of the body, the kinetic energy is $mv^2/2$, which at level $x(t)$ must be equal to the change in potential energy, which is $m\gamma x(t)$; this yields $v = \sqrt{2\gamma x(t)}$].

We introduce the system $\dot{x} = u$, and we obtain a fixed terminal state problem [$x(0) = 0$ and $x(T) = b$]. Letting

$$g(x,u) = \frac{\sqrt{1+u^2}}{\sqrt{2\gamma x}},$$

the Hamiltonian is

$$H(x,u,p) = g(x,u) + pu.$$

We minimize the Hamiltonian by setting to zero its derivative with respect to u:

$$p(t) = -\nabla_u g\big(x^*(t), u^*(t)\big).$$

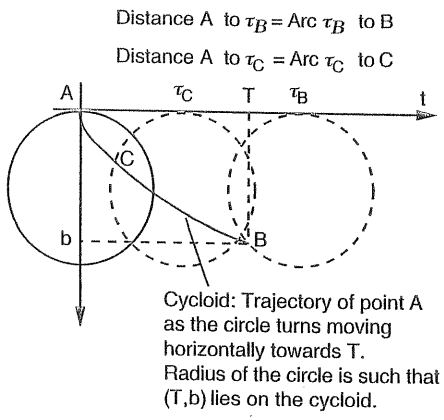


Figure 3.4.2 Formulation and optimal solution of the brachistochrone problem.

We know from the Minimum Principle that the Hamiltonian is constant along an optimal trajectory, i.e.,

$$g\big(x^*(t),u^*(t)\big) - \nabla_u g\big(x^*(t),u^*(t)\big)u^*(t) = \text{constant}, \qquad \text{for all } t \in [0,T].$$

Using the expression for g, this can be written as

$$\frac{\sqrt{1+\big(u^*(t)\big)^2}}{\sqrt{2\gamma x^*(t)}} - \frac{\big(u^*(t)\big)^2}{\sqrt{1+\big(u^*(t)\big)^2}\sqrt{2\gamma x^*(t)}} = \text{constant}, \qquad \text{for all } t \in [0,T],$$

or equivalently

$$\frac{1}{\sqrt{1+\big(u^*(t)\big)^2}\sqrt{2\gamma x^*(t)}} = \text{constant}, \qquad \text{for all } t \in [0,T].$$

Using the relation $\dot{x}^*(t) = u^*(t)$, this yields

$$x^*(t)\big(1+\dot{x}^*(t)^2\big) = C, \quad \text{for all } t \in [0,T],$$

for some constant C. Thus an optimal trajectory satisfies the differential equation

$$\dot{x}^*(t) = \sqrt{\frac{C-x^*(t)}{x^*(t)}}, \qquad \text{for all } t \in [0,T].$$

The solution of this differential equation was known at Bernoulli's time to be a *cycloid*; see Fig. 3.4.2. The unknown parameters of the cycloid are determined by the boundary conditions $x^*(0) = 0$ and $x^*(T) = b$.

3.4.2 Free Initial State

If the initial state $x(0)$ is not fixed but is subject to optimization, we have

$$J^*\big(0,x^*(0)\big) \le J^*(0,x), \qquad \text{for all } x \in \Re^n,$$

yielding

$$\nabla_x J^*\big(0,x^*(0)\big) = 0$$

and the extra boundary condition for the adjoint equation

$$p(0) = 0.$$

Also if there is a cost $\ell\big(x(0)\big)$ on the initial state, i.e., the cost is

$$\ell\big(x(0)\big) + \int_0^T g\big(x(t),u(t)\big)dt + h\big(x(T)\big),$$

the boundary condition becomes

$$p(0) = -\nabla\ell(x^*(0)).$$

This follows by setting to zero the gradient with respect to x of $\ell(x) + J(0,x)$, i.e.,

$$\nabla_x\big\{\ell(x) + J(0,x)\big\}\big|_{x=x^*(0)} = 0.$$

3.4.3 Free Terminal Time

Suppose the initial state and/or the terminal state are given, but the terminal time T is subject to optimization.

Let $\big\{\big(x^*(t),u^*(t)\big) \mid t \in [0,T]\big\}$ be an optimal state-control trajectory pair and let T^* be the optimal terminal time. Then if the terminal time were fixed at T^*, the pair $\big\{\big(u^*(t),x^*(t)\big) \mid t \in [0,T^*]\big\}$ would satisfy the conditions of the Minimum Principle. In particular,

$$u^*(t) = \arg\min_{u\in U} H\big(x^*(t),u,p(t)\big), \qquad \text{for all } t \in [0,T^*],$$

where $p(t)$ is the solution of the adjoint equation. What we lose with the terminal time being free, we gain with an extra condition derived as follows.

We argue that if the terminal time were fixed at T^* and the initial state were fixed at the given $x(0)$, but instead the initial time were subject to optimization, it would be optimal to start at $t = 0$. This means that the first order variation of the optimal cost with respect to the initial time must be zero; i.e.,

$$\nabla_t J^*\big(t,x^*(t)\big)\big|_{t=0} = 0.$$

The HJB equation can be written along the optimal trajectory as

$$\nabla_t J^*\big(t, x^*(t)\big) = -H\big(x^*(t), u^*(t), p(t)\big), \qquad \text{for all } t \in [0, T^*]$$

[cf. Eqs. (3.14) and (3.19)], so the preceding two equations yield

$$H\big(x^*(0), u^*(0), p(0)\big) = 0.$$

Since the Hamiltonian was shown earlier to be constant along the optimal trajectory, we obtain for the case of a free terminal time

$$H\big(x^*(t), u^*(t), p(t)\big) = 0, \qquad \text{for all } t \in [0, T^*].$$

Example 3.4.3 (Minimum-Time Problem)

A unit mass object moves horizontally under the influence of a force $u(t)$, so that

$$\ddot{y}(t) = u(t),$$

where $y(t)$ is the position of the object at time t. Given the object's initial position $y(0)$ and initial velocity $\dot{y}(0)$, it is required to bring the object to rest (zero velocity) at a given position, say zero, while using at most unit magnitude force,

$$-1 \le u(t) \le 1, \qquad \text{for all } t.$$

We want to accomplish this transfer in minimum time. Thus, we want to

$$\text{minimize } T = \int_0^T 1 dt.$$

Note that the integral cost, $g\big(x(t), u(t)\big) \equiv 1$, is unusual here; it does not depend on the state or the control. However, the theory does not preclude this possibility, and the problem is still meaningful because the terminal time T is free and subject to optimization.

Let the state variables be

$$x_1(t) = y(t), \qquad x_2(t) = \dot{y}(t),$$

so the system equation is

$$\dot{x}_1(t) = x_2(t), \qquad \dot{x}_2(t) = u(t).$$

The initial state $\big(x_1(0), x_2(0)\big)$ is given and the terminal state is also given

$$x_1(T) = 0, \qquad x_2(T) = 0.$$

If $\big\{u^*(t) \mid t \in [0, T]\big\}$ is an optimal control trajectory, $u^*(t)$ must minimize the Hamiltonian for each t, i.e.,

$$u^*(t) = \arg \min_{-1 \le u \le 1} \big[1 + p_1(t) x_2^*(t) + p_2(t) u\big].$$

Therefore

$$u^*(t) = \begin{cases} 1 & \text{if } p_2(t) < 0, \\ -1 & \text{if } p_2(t) \ge 0. \end{cases}$$

The adjoint equation is

$$\dot{p}_1(t) = 0, \qquad \dot{p}_2(t) = -p_1(t),$$

so

$$p_1(t) = c_1, \qquad p_2(t) = c_2 - c_1 t,$$

where c_1 and c_2 are constants. It follows that $\big\{p_2(t) \mid t \in [0, T]\big\}$ has one of the four forms shown in Fig. 3.4.3(a); that is, $\big\{p_2(t) \mid t \in [0, T]\big\}$ switches at most once in going from negative to positive or reversely. [Note that it is not possible for $p_2(t)$ to be equal to 0 for all t because this implies that $p_1(t)$ is also equal to 0 for all t, so that the Hamiltonian is equal to 1 for all t; the necessary conditions require that the Hamiltonian be 0 along the optimal trajectory.] The corresponding control trajectories are shown in Fig. 3.4.3(b). The conclusion is that, for each t, $u^*(t)$ is either $+1$ or -1, and $\big\{u^*(t) \mid t \in [0, T]\big\}$ has at most one switching point in the interval $[0, T]$.

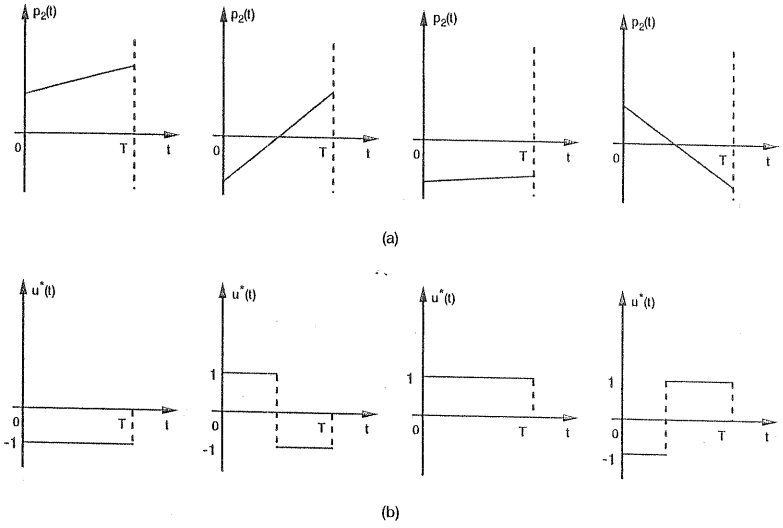


Figure 3.4.3 (a) Possible forms of the adjoint variable $p_2(t)$. (b) Corresponding forms of the optimal control trajectory.

To determine the precise form of the optimal control trajectory, we use the given initial and final states. For $u(t) \equiv \zeta$, where $\zeta = \pm 1$, the system evolves according to

$$x_1(t) = x_1(0) + x_2(0)t + \frac{\zeta}{2} t^2, \qquad x_2(t) = x_2(0) + \zeta t.$$

By eliminating the time t in these two equations, we see that for all t

$$x_1(t) - \frac{1}{2\zeta}\big(x_2(t)\big)^2 = x_1(0) - \frac{1}{2\zeta}\big(x_2(0)\big)^2.$$

Thus for intervals where $u(t) \equiv 1$, the system moves along the curves where $x_1(t) - \frac{1}{2}\big(x_2(t)\big)^2$ is constant, shown in Fig. 3.4.4(a). For intervals where $u(t) \equiv -1$, the system moves along the curves where $x_1(t) + \frac{1}{2}\big(x_2(t)\big)^2$ is constant, shown in Fig. 3.4.4(b).

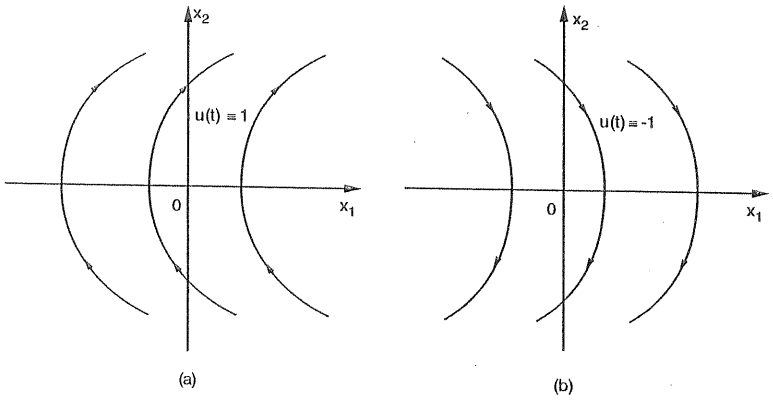


Figure 3.4.4 State trajectories when the control is $u(t) \equiv 1$ [Fig. (a)] and when the control is $u(t) \equiv -1$ [Fig. (b)].

To bring the system from the initial state $\big(x_1(0), x_2(0)\big)$ to the origin with at most one switch in the value of control, we must apply control according to the following rules involving the *switching curve* shown in Fig. 3.4.5.

(a) If the initial state lies *above* the switching curve, use $u^*(t) \equiv -1$ until the state hits the switching curve; then use $u^*(t) \equiv 1$ until reaching the origin.

(b) If the initial state lies *below* the switching curve, use $u^*(t) \equiv 1$ until the state hits the switching curve; then use $u^*(t) \equiv -1$ until reaching the origin.

(c) If the initial state lies on the top (bottom) part of the switching curve, use $u^*(t) \equiv -1$ [$u^*(t) \equiv 1$, respectively] until reaching the origin.

3.4.4 Time-Varying System and Cost

If the system equation and the integral cost depend on the time t, i.e.,

$$\dot{x}(t) = f\big(x(t), u(t), t\big),$$

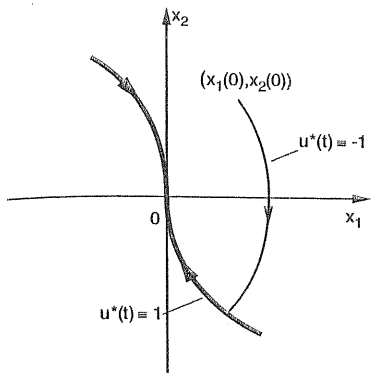


Figure 3.4.5 Switching curve (shown with a thick line) and closed-loop optimal control for the minimum time example.

$$\text{cost} = h\big(x(T)\big) + \int_0^T g\big(x(t), u(t), t\big)dt,$$

we can convert the problem to one involving a time-independent system and cost by introducing an extra state variable $y(t)$ representing time:

$$\dot{y}(t) = 1, \qquad y(0) = 0,$$

$$\dot{x}(t) = f\big(x(t), u(t), y(t)\big), \qquad x(0) : \text{given},$$

$$\text{cost} = h\big(x(T)\big) + \int_0^T g\big(x(t), u(t), y(t)\big)dt.$$

After working out the corresponding optimality conditions, we see that they are the same as when the system and cost are time-independent. The only difference is that the Hamiltonian need not be constant along the optimal trajectory.

3.4.5 Singular Problems

In some cases, the minimum condition

$$u^*(t) = \arg\min_{u \in U} H\big(x^*(t), u, p(t), t\big) \tag{3.45}$$

is insufficient to determine $u^*(t)$ for all t, because the values of $x^*(t)$ and $p(t)$ are such that $H\big(x^*(t), u, p(t), t\big)$ is independent of u over a nontrivial interval of time. Such problems are called *singular*. Their optimal trajectories consist of portions, called *regular arcs*, where $u^*(t)$ can be determined from the minimum condition (3.45), and other portions, called *singular arcs*, which can be determined from the condition that the Hamiltonian is independent of u.

Example 3.4.4 (Road Construction)

Suppose that we want to construct a road over a one-dimensional terrain whose ground elevation (altitude measured from some reference point) is known and is given by $z(t)$, $t \in [0,T]$. The elevation of the road is denoted by $x(t)$, $t \in [0,T]$, and the difference $x(t) - z(t)$ must be made up by fill-in or excavation. It is desired to minimize

$$\frac{1}{2}\int_0^T \big(x(t) - z(t)\big)^2 dt,$$

subject to the constraint that the gradient of the road $\dot{x}(t)$ lies between $-a$ and a, where a is a specified maximum allowed slope. Thus we have the constraint

$$|u(t)| \leq a, \qquad t \in [0,T],$$

where

$$\dot{x}(t) = u(t), \qquad t \in [0,T].$$

The adjoint equation here is

$$\dot{p}(t) = -x^*(t) + z(t),$$

with the terminal condition

$$p(T) = 0.$$

Minimization of the Hamiltonian

$$H\big(x^*(t), u, p(t), t\big) = \frac{1}{2}\big(x^*(t) - z(t)\big)^2 + p(t)u$$

with respect to u yields

$$u^*(t) = \arg\min_{|u|\leq a} p(t)u,$$

for all t, and shows that optimal trajectories are obtained by concatenation of three types of arcs:

(a) Regular arcs where $p(t) > 0$ and $u^*(t) = -a$ (maximum downhill slope arcs).

(b) Regular arcs where $p(t) < 0$ and $u^*(t) = a$ (maximum uphill slope arcs).

(c) Singular arcs where $p(t) = 0$ and $u^*(t)$ can take any value in $[-a,a]$ that maintains the condition $p(t) = 0$. From the adjoint equation we see that singular arcs are those along which $p(t) = 0$ and $x^*(t) = z(t)$, i.e., the road follows the ground elevation (no fill-in or excavation). Along such arcs we must have

$$\dot{z}(t) = u^*(t) \in [-a,a].$$

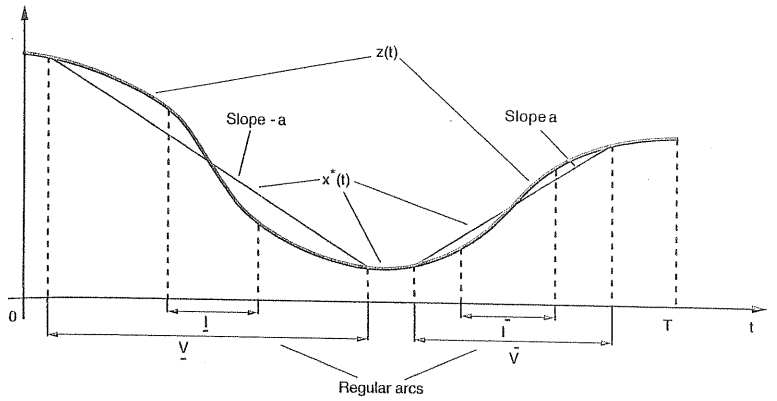


Figure 3.4.6 Graphical method for solving the road construction example. The sharply uphill (downhill) intervals $\overline{I}$ (respectively, $\underline{I}$) are first identified, and are then embedded within maximum uphill (respectively, downhill) slope regular arcs $\overline{V}$ (respectively, $\underline{V}$) within which the total fill-in is equal to the total excavation. The regular arcs are joined by singular arcs where there is no fill-in or excavation. The graphical process is started at the endpoint $t = T$.

Optimal solutions can be obtained by a graphical method using the above observations. Consider the *sharply uphill intervals* $\overline{I}$ such that $\dot{z}(t) \geq a$ for all $t \in \overline{I}$, and the *sharply downhill intervals* $\underline{I}$ such that $\dot{z}(t) \leq -a$ for all $t \in \underline{I}$. Clearly, within each sharply uphill interval $\overline{I}$ the optimal slope is $u^*(t) = a$, but the optimal slope is also equal to a within a larger maximum uphill slope interval $\overline{V} \supset \overline{I}$, which is such that $p(t) < 0$ within $\overline{V}$ and

$$p(t_1) = p(t_2) = 0$$

at the endpoints t_1 and t_2 of $\overline{V}$. In view of the form of the adjoint equation, we see that the endpoints t_1 and t_2 of $\overline{V}$ should be such that

$$\int_{t_1}^{t_2} \big(z(t) - x^*(t)\big)dt = 0;$$

that is, the *total fill-in should be equal to the total excavation within* $\overline{V}$, (see Fig. 3.4.6). Similarly, each sharply downhill interval $\underline{I}$ should be contained within a larger maximum downhill slope interval $\underline{V} \supset \underline{I}$, which is such that $p(t) > 0$ within $\underline{V}$, while the *total fill-in should be equal to the total excavation within* $\underline{V}$, (see Fig. 3.4.6). Thus the regular arcs consist of the intervals $\overline{V}$ and $\underline{V}$ described above. Between the regular arcs there can be one or more singular arcs where $x^*(t) = z(t)$. The optimal solution can be pieced together starting at the endpoint $t = T$ [where we know that $p(T) = 0$], and proceeding backwards.

3.5 NOTES, SOURCES, AND EXERCISES

The calculus of variations is a classical subject that originated with the works of the great mathematicians of the 17th and 18th centuries. Its rigorous development (by modern mathematical standards) took place in the 1930s and 1940s, with the work of a group of mathematicians that originated mostly from the University of Chicago; Bliss, McShane, and Hestenes are some of the most prominent members of this group. Curiously, this development preceded the development of nonlinear programming by many years.† The modern theory of deterministic optimal control has its roots primarily in the work of Pontryagin, Boltyanski, Gamkrelidze, and Mishchenko in the 1950s [PBG65]. A highly personal but controversial historical account of this work is given by Boltyanski in [BMS96]. The theoretical and applications literature on the subject is very extensive. We give three representative references: the book by Athans and Falb [AtF66] (a classical extensive text that includes engineering applications), the book by Hestenes [Hes66] (a rigorous mathematical treatment, containing important work that predates the work of Pontryagin et al.), and the book by Luenberger [Lue69] (which deals with optimal control within a broader infinite dimensional context). The author's nonlinear programming book [Ber99] gives a detailed treatment of optimality conditions and computational methods for discrete-time optimal control.

EXERCISES

3.1

Solve the problem of Example 3.2.1 for the case where the cost function is

$$\big(x(T)\big)^2 + \int_0^T \big(u(t)\big)^2 dt.$$

Also, calculate the cost-to-go function $J^*(t,x)$ and verify that it satisfies the HJB equation.

† In the 30s and 40s journal space was at a premium, and finite-dimensional optimization research was thought to be a simple special case of the calculus of variations, thus insufficiently challenging or novel for publication. Indeed the modern optimality conditions of finite-dimensional optimization subject to equality and inequality constraints were first developed in the 1939 Master's thesis by Karush, but first appeared in a journal quite a few years later under the names of other researchers.

3.2

A young investor has earned in the stock market a large amount of money S and plans to spend it so as to maximize his enjoyment through the rest of his life without working. He estimates that he will live exactly T more years and that his capital $x(t)$ should be reduced to zero at time T, i.e., $x(T) = 0$. Also he models the evolution of his capital by the differential equation

$$\frac{dx(t)}{dt} = \alpha x(t) - u(t),$$

where $x(0) = S$ is his initial capital, $\alpha > 0$ is a given interest rate, and $u(t) \geq 0$ is his rate of expenditure. The total enjoyment he will obtain is given by

$$\int_0^T e^{-\beta t}\sqrt{u(t)}\, dt.$$

Here β is some positive scalar, which serves to discount future enjoyment. Find the optimal $\{u(t) \mid t \in [0,T]\}$.

3.3

Consider the system of reservoirs shown in Fig. 3.5.1. The system equations are

$$\dot{x}_1(t) = -x_1(t) + u(t),$$
$$\dot{x}_2(t) = x_1(t),$$

and the control constraint is $0 \leq u(t) \leq 1$ for all t. Initially

$$x_1(0) = x_2(0) = 0.$$

We want to maximize $x_2(1)$ subject to the constraint $x_1(1) = 0.5$. Solve the problem.

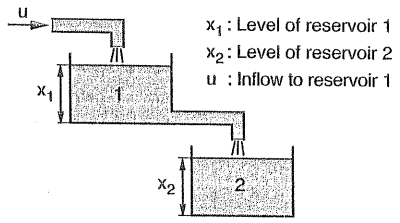


Figure 3.5.1 Reservoir system for Exercise 3.3.

3.4

Work out the minimum-time problem (Example 3.4.3) for the case where there is friction and the object's position moves according to

$$\ddot{y}(t) = -a\dot{y}(t) + u(t),$$

where $a > 0$ is given. *Hint*: The solution of the system

$$\dot{p}_1(t) = 0,$$
$$\dot{p}_2(t) = -p_1(t) + ap_2(t),$$

is

$$p_1(t) = p_1(0),$$
$$p_2(t) = \frac{1}{a}(1 - e^{at})p_1(0) + e^{at}p_2(0).$$

The trajectories of the system for $u(t) \equiv -1$ and $u(t) \equiv 1$ are sketched in Fig. 3.5.2.

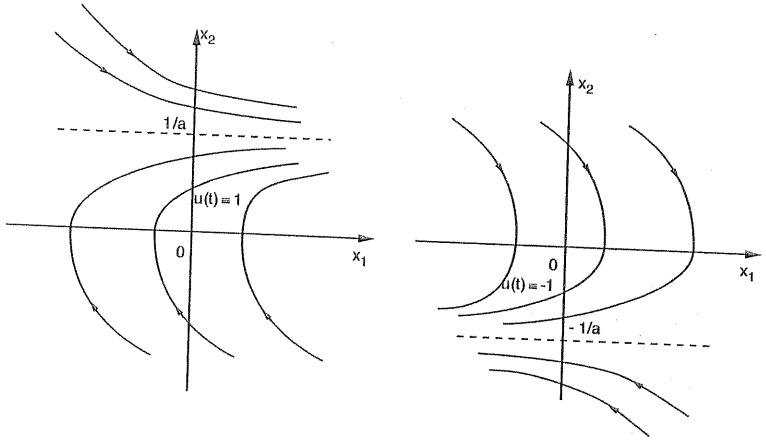


Figure 3.5.2 State trajectories of the system of Exercise 3.4 for $u(t) \equiv -1$ and $u(t) \equiv 1$.

3.5 (Isoperimetric Problem)

Analyze the problem of finding a curve $\{x(t) \mid t \in [0,T]\}$ that maximizes the area under x,

$$\int_0^T x(t)dt,$$

subject to the constraints

$$x(0) = a, \qquad x(T) = b, \qquad \int_0^T \sqrt{1 + \big(\dot{x}(t)\big)^2}\, dt = L,$$

where a, b, and L are given positive scalars. The last constraint is known as an isoperimetric constraint; it requires that the length of the curve be L. *Hint*: Introduce the system $\dot{x}_1 = u$, $\dot{x}_2 = \sqrt{1+u^2}$, and view the problem as a fixed terminal state problem. Show that the sine of the optimal $u^*(t)$ depends linearly on t. Under some assumptions on a, b, and L, the optimal curve is a circular arc.

3.6 (L'Hôpital's Problem) WWW

Let a, b, and T be positive scalars, and let $A = (0,a)$ and $B = (T,b)$ be two points in a medium within which the velocity of propagation of light is proportional to the vertical coordinate. Thus the time it takes for light to propagate from A to B along a curve $\{x(t) \mid t \in [0,T]\}$ is

$$\int_0^T \frac{\sqrt{1 + \big(\dot{x}(t)\big)^2}}{Cx(t)}\, dt,$$

where C is a given positive constant. Find the curve of minimum travel time of light from A to B, and show that it is an arc of a circle of the form

$$x(t)^2 + (t-d)^2 = D,$$

where d and D are some constants.

3.7

A boat moves with constant unit velocity in a stream moving at constant velocity s. The problem is to find the steering angle $u(t)$, $0 \le t \le T$, which minimizes the time T required for the boat to move between the point $(0,0)$ to a given point (a,b). The equations of motion are

$$\dot{x}_1(t) = s + \cos u(t), \qquad \dot{x}_2(t) = \sin u(t),$$

where $x_1(t)$ and $x_2(t)$ are the positions of the boat parallel and perpendicular to the stream velocity, respectively. Show that the optimal solution is to steer at a constant angle.

3.8

A unit mass object moves on a straight line from a given initial position $x_1(0)$ and velocity $x_2(0)$. Find the force $\{u(t) \mid t \in [0,1]\}$ that brings the object at time 1 to rest $[x_2(1) = 0]$ at position $x_1(1) = 0$, and minimizes

$$\int_0^1 \big(u(t)\big)^2 dt.$$

3.9 (WWW)

Use the Minimum Principle to solve the linear-quadratic problem of Example 3.2.2. *Hint*: Follow the lines of Example 3.3.3.

3.10 (On the Need for Convexity Assumptions)

Solve the continuous-time problem involving the system $\dot{x}(t) = u(t)$, the terminal cost $\big(x(T)\big)^2$, and the control constraint $u(t) = -1$ or 1 for all t, and show that the solution satisfies the Minimum Principle. Show that, depending on the initial state x_0, this may not be true for the discrete-time version involving the system $x_{k+1} = x_k + u_k$, the terminal cost x_N^2, and the control constraint $u_k = -1$ or 1 for all k.

3.11

Use the discrete-time Minimum Principle to solve Exercise 1.14 of Chapter 1, assuming that each w_k is fixed at a known deterministic value.

3.12

Use the discrete-time Minimum Principle to solve Exercise 1.15 of Chapter 1, assuming that γ_k and δ_k are fixed at known deterministic values.

3.13 (Lagrange Multipliers and the Minimum Principle)

Consider the discrete-time optimal control problem of Section 3.3.3, where there are no control constraints ($U = \Re^m$). Introduce a Lagrange multiplier vector p_{k+1} for each of the constraints

$$f_k(x_k, u_k) - x_{k+1} = 0, \qquad k = 0, \ldots, N-1,$$

and form the Lagrangian function

$$g_N(x_N) + \sum_{k=0}^{N-1} \Big(g_k(x_k, u_k) + p_{k+1}' \big(f_k(x_k, u_k) - x_{k+1} \big) \Big)$$

(cf. Appendix B). View both the state and the control vectors as the optimization variables of the problem, and show that by differentiation of the Lagrangian function with respect to x_k and u_k, we obtain the discrete-time Minimum Principle.

4

Problems with Perfect State Information

Contents

In this chapter we consider a number of applications of discrete-time stochastic optimal control with perfect state information. These applications are special cases of the basic problem of Section 1.2 and can be addressed via the DP algorithm. In all these applications the stochastic nature of the disturbances is significant. For this reason, in contrast with the deterministic problems of the preceding two chapters, the use of closed-loop control is essential to achieve optimal performance.

4.1 LINEAR SYSTEMS AND QUADRATIC COST

In this section we consider the special case of a linear system

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

and the quadratic cost

$$\mathop{E}_{\substack{w_k \\ k=0,1,\ldots,N-1}} \left\{ x_N' Q_N x_N + \sum_{k=0}^{N-1} (x_k' Q_k x_k + u_k' R_k u_k) \right\}.$$

In these expressions, x_k and u_k are vectors of dimension n and m, respectively, and the matrices A_k, B_k, Q_k, R_k are given and have appropriate dimension. We assume that the matrices Q_k are positive semidefinite symmetric, and the matrices R_k are positive definite symmetric. The controls u_k are unconstrained. The disturbances w_k are independent random vectors with given probability distributions that do not depend on x_k and u_k. Furthermore, each w_k has zero mean and finite second moment.

The problem described above is a popular formulation of a regulation problem whereby we want to keep the state of the system close to the origin. Such problems are common in the theory of automatic control of a motion or a process. The quadratic cost function is often reasonable because it induces a high penalty for large deviations of the state from the origin but a relatively small penalty for small deviations. Also, the quadratic cost is frequently used, even when it is not entirely justified, because it leads to a nice analytical solution. A number of variations and generalizations have similar solutions. For example, the disturbances w_k could have nonzero means and the quadratic cost could have the form

$$E\left\{ (x_N - \bar{x}_N)' Q_N (x_N - \bar{x}_N) + \sum_{k=0}^{N-1} \left((x_k - \bar{x}_k)' Q_k (x_k - \bar{x}_k) + u_k' R_k u_k \right) \right\},$$

which expresses a desire to keep the state of the system close to a given trajectory $(\bar{x}_0, \bar{x}_1, \ldots, \bar{x}_N)$ rather than close to the origin. Another generalized version of the problem arises when A_k, B_k are independent random

matrices, rather than being known. This case is considered at the end of this section.

Applying now the DP algorithm, we have

$$J_N(x_N) = x_N' Q_N x_N,$$

$$J_k(x_k) = \min_{u_k} E\{x_k' Q_k x_k + u_k' R_k u_k + J_{k+1}(A_k x_k + B_k u_k + w_k)\}. \quad (4.1)$$

It turns out that the cost-to-go functions J_k are quadratic and as a result the optimal control law is a linear function of the state. These facts can be verified by straightforward induction. We write Eq. (4.1) for $k = N-1$,

$$\begin{aligned} J_{N-1}(x_{N-1}) = \min_{u_{N-1}} E\{ & x_{N-1}' Q_{N-1} x_{N-1} + u_{N-1}' R_{N-1} u_{N-1} \\ & + (A_{N-1} x_{N-1} + B_{N-1} u_{N-1} + w_{N-1})' Q_N \\ & \cdot (A_{N-1} x_{N-1} + B_{N-1} u_{N-1} + w_{N-1}) \}, \end{aligned}$$

and we expand the last quadratic form in the right-hand side. We then use the fact $E\{w_{N-1}\} = 0$ to eliminate the term $E\{w_{N-1}' Q_N (A_{N-1} x_{N-1} + B_{N-1} u_{N-1})\}$, and we obtain

$$\begin{aligned} J_{N-1}(x_{N-1}) = x_{N-1}' Q_{N-1} x_{N-1} + \min_{u_{N-1}} [& u_{N-1}' R_{N-1} u_{N-1} \\ & + u_{N-1}' B_{N-1}' Q_N B_{N-1} u_{N-1} + 2 x_{N-1}' A_{N-1}' Q_N B_{N-1} u_{N-1}] \\ & + x_{N-1}' A_{N-1}' Q_N A_{N-1} x_{N-1} + E\{w_{N-1}' Q_N w_{N-1}\}. \end{aligned}$$

By differentiating with respect to u_{N-1} and by setting the derivative equal to zero, we obtain

$$(R_{N-1} + B_{N-1}' Q_N B_{N-1}) u_{N-1} = -B_{N-1}' Q_N A_{N-1} x_{N-1}.$$

The matrix multiplying u_{N-1} on the left is positive definite (and hence invertible), since R_{N-1} is positive definite and $B_{N-1}' Q_N B_{N-1}$ is positive semidefinite. As a result, the minimizing control vector is given by

$$u_{N-1}^* = -(R_{N-1} + B_{N-1}' Q_N B_{N-1})^{-1} B_{N-1}' Q_N A_{N-1} x_{N-1}.$$

By substitution into the expression for J_{N-1}, we have

$$J_{N-1}(x_{N-1}) = x_{N-1}' K_{N-1} x_{N-1} + E\{w_{N-1}' Q_N w_{N-1}\},$$

where by straightforward calculation, the matrix K_{N-1} is verified to be

$$\begin{aligned} K_{N-1} = {} & A_{N-1}' (Q_N - Q_N B_{N-1} (B_{N-1}' Q_N B_{N-1} + R_{N-1})^{-1} B_{N-1}' Q_N) A_{N-1} \\ & + Q_{N-1}. \end{aligned}$$

The matrix K_{N-1} is clearly symmetric. It is also positive semidefinite. To see this, note that from the preceding calculation we have for $x \in \Re^n$

$$x'K_{N-1}x = \min_u \big[x'Q_{N-1}x + u'R_{N-1}u + (A_{N-1}x + B_{N-1}u)'Q_N(A_{N-1}x + B_{N-1}u)\big].$$

Since Q_{N-1}, R_{N-1}, and Q_N are positive semidefinite, the expression within brackets is nonnegative. Minimization over u preserves nonnegativity, so it follows that $x'K_{N-1}x \geq 0$ for all $x \in \Re^n$. Hence K_{N-1} is positive semidefinite.

Since J_{N-1} is a positive semidefinite quadratic function (plus an inconsequential constant term), we may proceed similarly and obtain from the DP equation (4.1) the optimal control law for stage $N-2$. As earlier, we show that J_{N-2} is a positive semidefinite quadratic function, and by proceeding sequentially, we obtain the optimal control law for every k. It has the form

$$\mu_k^*(x_k) = L_k x_k, \tag{4.2}$$

where the gain matrices L_k are given by the equation

$$L_k = -(B_k'K_{k+1}B_k + R_k)^{-1}B_k'K_{k+1}A_k,$$

and where the symmetric positive semidefinite matrices K_k are given recursively by the algorithm

$$K_N = Q_N, \tag{4.3}$$

$$K_k = A_k'\big(K_{k+1} - K_{k+1}B_k(B_k'K_{k+1}B_k + R_k)^{-1}B_k'K_{k+1}\big)A_k + Q_k. \tag{4.4}$$

Just like DP, this algorithm starts at the terminal time N and proceeds backwards. The optimal cost is given by

$$J_0(x_0) = x_0'K_0x_0 + \sum_{k=0}^{N-1} E\{w_k'K_{k+1}w_k\}.$$

The control law (4.2) is simple and attractive for implementation in engineering applications: the current state x_k is being fed back as input through the linear feedback gain matrix L_k as shown in Fig. 4.1.1. This accounts in part for the popularity of the linear-quadratic formulation. As we will see in Chapter 5, the linearity of the control law is still maintained even for problems where the state x_k is not completely observable (imperfect state information).

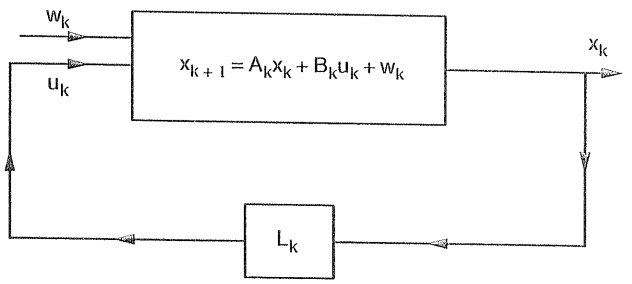


Figure 4.1.1 Linear feedback structure of the optimal controller for the linear-quadratic problem.

The Riccati Equation and Its Asymptotic Behavior

Equation (4.4) is called the *discrete-time Riccati equation*. It plays an important role in control theory. Its properties have been studied extensively and exhaustively. One interesting property of the Riccati equation is that if the matrices A_k, B_k, Q_k, R_k are constant and equal to A, B, Q, R, respectively, then the solution K_k converges as $k \to -\infty$ (under mild assumptions) to a steady-state solution K satisfying the *algebraic Riccati equation*

$$K = A'\big(K - KB(B'KB + R)^{-1}B'K\big)A + Q. \tag{4.5}$$

This property, to be proved shortly, indicates that for the system

$$x_{k+1} = Ax_k + Bu_k + w_k, \qquad k = 0, 1, \ldots, N-1,$$

and a large number of stages N, one can reasonably approximate the control law (4.2) by the control law $\{\mu^*, \mu^*, \ldots, \mu^*\}$, where

$$\mu^*(x) = Lx, \tag{4.6}$$

$$L = -(B'KB + R)^{-1}B'KA,$$

and K solves the algebraic Riccati equation (4.5). This control law is *stationary*; that is, it does not change over time.

We now turn to proving convergence of the sequence of matrices $\{K_k\}$ generated by the Riccati equation (4.4). We first introduce the notions of controllability and observability, which are very important in control theory.

Definition 4.1.1: A pair (A, B), where A is an $n \times n$ matrix and B is an $n \times m$ matrix, is said to be *controllable* if the $n \times nm$ matrix

$$[B, AB, A^2B, \ldots, A^{n-1}B]$$

has full rank (i.e., has linearly independent rows). A pair (A, C), where A is an $n \times n$ matrix and C an $m \times n$ matrix, is said to be *observable* if the pair (A', C') is controllable, where A' and C' denote the transposes of A and C, respectively.

One may show that if the pair (A, B) is controllable, then for any initial state x_0, there exists a sequence of control vectors $u_0, u_1, \ldots, u_{n-1}$ that force the state x_n of the system

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

to be equal to zero at time n. Indeed, by successively applying the above equation for $k = n-1, n-2, \ldots, 0$, we obtain

$$x_n = A^n x_0 + Bu_{n-1} + ABu_{n-2} + \cdots + A^{n-1}Bu_0$$

or equivalently

$$x_n - A^n x_0 = (B, AB, \ldots, A^{n-1}B) \begin{pmatrix} u_{n-1} \\ u_{n-2} \\ \vdots \\ u_0 \end{pmatrix}. \tag{4.7}$$

If (A, B) is controllable, the matrix $(B, AB, \ldots, A^{n-1}B)$ has full rank and as a result the right-hand side of Eq. (4.7) can be made equal to any vector in $\Re^n$ by appropriate selection of $(u_0, u_1, \ldots, u_{n-1})$. In particular, one can choose $(u_0, u_1, \ldots, u_{n-1})$ so that the right-hand side of Eq. (4.7) is equal to $-A^n x_0$, which implies $x_n = 0$. This property explains the name "controllable pair" and in fact is often used to define controllability.

The notion of observability has an analogous interpretation in the context of estimation problems; that is, given measurements $z_0, z_1, \ldots, z_{n-1}$ of the form $z_k = Cx_k$, it is possible to infer the initial state x_0 of the system $x_{k+1} = Ax_k$, in view of the relation

$$\begin{pmatrix} z_{n-1} \\ \vdots \\ z_1 \\ z_0 \end{pmatrix} = \begin{pmatrix} CA^{n-1} \\ \vdots \\ CA \\ C \end{pmatrix} x_0.$$

Alternatively, it can be seen that observability is equivalent to the property that, in the absence of control, if $Cx_k \to 0$ then $x_k \to 0$.

The notion of stability is of paramount importance in control theory. In the context of our problem it is important that the stationary control law (4.6) results in a stable closed-loop system; that is, in the absence of input disturbance, the state of the system

$$x_{k+1} = (A + BL)x_k, \qquad k = 0, 1, \ldots,$$

tends to zero as $k \to \infty$. Since $x_k = (A + BL)^k x_0$, it follows that the closed-loop system is stable if and only if $(A + BL)^k \to 0$, or equivalently (see Appendix A), if and only if the eigenvalues of the matrix $(A + BL)$ are strictly within the unit circle.

The following proposition shows that for a stationary controllable system and constant matrices Q and R, the solution of the Riccati equation (4.4) converges to a positive definite symmetric matrix K for an arbitrary positive semidefinite symmetric initial matrix. In addition, the proposition shows that the corresponding closed-loop system is stable. The proposition also requires an observability assumption, namely, that Q can be written as $C'C$, where the pair (A, C) is observable. Note that if r is the rank of Q, there exists an $r \times n$ matrix C of rank r such that $Q = C'C$ (see Appendix A). The implication of the observability assumption is that in the absence of control, if the state cost per stage $x_k'Qx_k$ tends to zero or equivalently $Cx_k \to 0$, then also $x_k \to 0$.

To simplify notation, we reverse the time indexing of the Riccati equation. Thus, P_k in the following proposition corresponds to K_{N-k} in Eq. (4.4). A graphical proof of the proposition for the case of a scalar system is given in Fig. 4.1.2.

Proposition 4.4.1: Let A be an $n \times n$ matrix, B be an $n \times m$ matrix, Q be an $n \times n$ positive semidefinite symmetric matrix, and R be an $m \times m$ positive definite symmetric matrix. Consider the discrete-time Riccati equation

$$P_{k+1} = A'\big(P_k - P_k B(B'P_k B + R)^{-1}B'P_k\big)A + Q, \qquad k = 0, 1, \ldots, \tag{4.8}$$

where the initial matrix P_0 is an arbitrary positive semidefinite symmetric matrix. Assume that the pair (A, B) is controllable. Assume also that Q may be written as $C'C$, where the pair (A, C) is observable. Then:

(a) There exists a positive definite symmetric matrix P such that for every positive semidefinite symmetric initial matrix P_0 we have

$$\lim_{k \to \infty} P_k = P.$$

Furthermore, P is the unique solution of the algebraic matrix equation

$$P = A'\big(P - PB(B'PB + R)^{-1}B'P\big)A + Q \tag{4.9}$$

within the class of positive semidefinite symmetric matrices.

(b) The corresponding closed-loop system is stable; that is, the eigenvalues of the matrix

$$D = A + BL, \tag{4.10}$$

where

$$L = -(B'PB + R)^{-1}B'PA, \tag{4.11}$$

are strictly within the unit circle.

Proof: The proof proceeds in several steps. First we show convergence of the sequence generated by Eq. (4.8) when the initial matrix P_0 is equal to zero. Next we show that the corresponding matrix D of Eq. (4.10) satisfies $D^k \to 0$. Then we show the convergence of the sequence generated by Eq. (4.8) when P_0 is any positive semidefinite symmetric matrix, and finally we show uniqueness of the solution of Eq. (4.9).

Initial Matrix $P_0 = 0$. Consider the optimal control problem of finding $u_0, u_1, \ldots, u_{k-1}$ that minimize

$$\sum_{i=0}^{k-1}(x_i'Qx_i + u_i'Ru_i)$$

subject to

$$x_{i+1} = Ax_i + Bu_i, \qquad i = 0, 1, \ldots, k-1,$$

where x_0 is given. The optimal value of this problem, according to the theory of this section, is $x_0'P_k(0)x_0$,

where $P_k(0)$ is given by the Riccati equation (4.8) with $P_0 = 0$. For any control sequence $(u_0, u_1, \ldots, u_k)$ we have

$$\sum_{i=0}^{k-1}(x_i'Qx_i + u_i'Ru_i) \le \sum_{i=0}^{k}(x_i'Qx_i + u_i'Ru_i)$$

and hence

$$\begin{aligned} x_0'P_k(0)x_0 &= \min_{\substack{u_i \\ i=0,\ldots,k-1}} \sum_{i=0}^{k-1}(x_i'Qx_i + u_i'Ru_i) \\ &\le \min_{\substack{u_i \\ i=0,\ldots,k}} \sum_{i=0}^{k}(x_i'Qx_i + u_i'Ru_i) \\ &= x_0'P_{k+1}(0)x_0, \end{aligned}$$

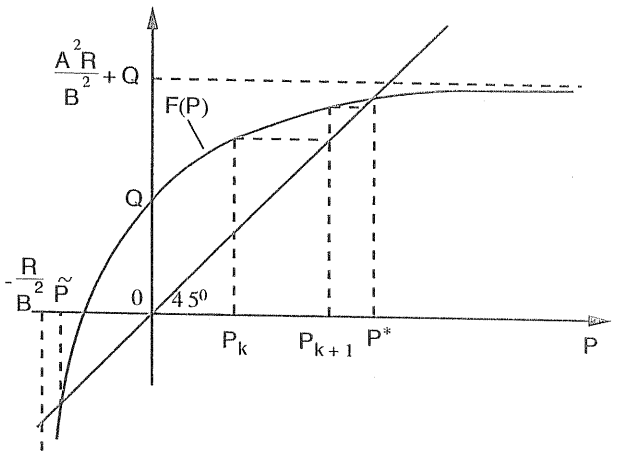


Figure 4.1.2 Graphical proof of Prop. 4.4.1 for the case of a scalar stationary system (one-dimensional state and control), assuming that $A \neq 0$, $B \neq 0$, $Q > 0$, and $R > 0$. The Riccati equation (4.8) is given by

$$P_{k+1} = A^2\left(P_k - \frac{B^2P_k^2}{B^2P_k + R}\right) + Q,$$

which can be equivalently written as

$$P_{k+1} = F(P_k),$$

where the function F is given by

$$F(P) = \frac{A^2RP}{B^2P + R} + Q.$$

Because F is concave and monotonically increasing in the interval $(-R/B^2, \infty)$, as shown in the figure, the equation $P = F(P)$ has one positive solution P^* and one negative solution $\tilde{P}$. The Riccati iteration $P_{k+1} = F(P_k)$ converges to P^* starting anywhere in the interval $(\tilde{P}, \infty)$ as shown in the figure.

where both minimizations are subject to the system equation constraint $x_{i+1} = Ax_i + Bu_i$. Furthermore, for a fixed x_0 and for every k, $x_0'P_k(0)x_0$ is bounded from above by the cost corresponding to a control sequence that forces x_0 to the origin in n steps and applies zero control after that. Such a sequence exists by the controllability assumption. Thus the sequence $\{x_0'P_k(0)x_0\}$ is nondecreasing with respect to k and bounded from above, and therefore converges to some real number for every $x_0 \in \Re^n$. It follows that the sequence $\{P_k(0)\}$ converges to some matrix P in the sense that each of the sequences of the elements of $P_k(0)$ converges to the correspond-

ing elements of P. To see this, take $x_0 = (1, 0, \ldots, 0)$. Then $x_0' P_k(0) x_0$ is equal to the first diagonal element of $P_k(0)$, so it follows that the sequence of first diagonal elements of $P_k(0)$ converges; the limit of this sequence is the first diagonal element of P. Similarly, by taking $x_0 = (0, \ldots, 0, 1, 0, \ldots, 0)$ with the 1 in the ith coordinate, for $i = 2, \ldots, n$, it follows that all the diagonal elements of $P_k(0)$ converge to the corresponding diagonal elements of P. Next take $x_0 = (1, 1, 0, \ldots, 0)$ to show that the second elements of the first row converge. Continuing similarly, we obtain

$$\lim_{k \to \infty} P_k(0) = P,$$

where $P_k(0)$ are generated by Eq. (4.8) with $P_0 = 0$. Furthermore, since $P_k(0)$ is positive semidefinite and symmetric, so is the limit matrix P. Now by taking the limit in Eq. (4.8) it follows that P satisfies

$$P = A'\big(P - PB(B'PB + R)^{-1}B'P\big)A + Q.$$

In addition, by direct calculation we can verify the following useful equality

$$P = D'PD + Q + L'RL, \tag{4.12}$$

where D and L are given by Eqs. (4.10) and (4.11). An alternative way to derive this equality is to observe that from the DP algorithm corresponding to a finite horizon N we have for all states x_{N-k}

$$x_{N-k}' P_{k+1}(0) x_{N-k} = x_{N-k}' Q x_{N-k} + \mu_{N-k}^*(x_{N-k})' R \mu_{N-k}^*(x_{N-k}) + x_{N-k+1}' P_k(0) x_{N-k+1}.$$

By using the optimal controller expression $\mu_{N-k}^*(x_{N-k}) = L_{N-k} x_{N-k}$ and the closed-loop system equation $x_{N-k+1} = (A + BL_{N-k})x_{N-k}$, we thus obtain

$$P_{k+1}(0) = Q + L_{N-k}' R L_{N-k} + (A + BL_{N-k})' P_k(0)(A + BL_{N-k}). \tag{4.13}$$

Equation (4.12) then follows by taking the limit as $k \to \infty$ in Eq. (4.13).

Stability of the Closed-Loop System. Consider the system

$$x_{k+1} = (A + BL)x_k = Dx_k \tag{4.14}$$

for an arbitrary initial state x_0. We will show that $x_k \to 0$ as $k \to \infty$. We have for all k, by using Eq. (4.12),

$$x_{k+1}' P x_{k+1} - x_k' P x_k = x_k'(D'PD - P)x_k = -x_k'(Q + L'RL)x_k.$$

Hence

$$x_{k+1}' P x_{k+1} = x_0' P x_0 - \sum_{i=0}^{k} x_i'(Q + L'RL)x_i. \tag{4.15}$$

The left-hand side of this equation is bounded below by zero, so it follows that

$$\lim_{k \to \infty} x_k'(Q + L'RL)x_k = 0.$$

Since R is positive definite and Q may be written as $C'C$, we obtain

$$\lim_{k \to \infty} Cx_k = 0, \qquad \lim_{k \to \infty} Lx_k = \lim_{k \to \infty} \mu^*(x_k) = 0. \tag{4.16}$$

The preceding relations imply that as the control asymptotically becomes negligible, we have $\lim_{k \to \infty} Cx_k = 0$, and in view of the observability assumption, this implies that $x_k \to 0$. To express this argument more precisely, let us use the relation $x_{k+1} = (A + BL)x_k$ [cf. Eq. (4.14)], to write

$$\begin{pmatrix} C\left(x_{k+n-1} - \sum_{i=1}^{n-1} A^{i-1} B L x_{k+n-i-1}\right) \\ C\left(x_{k+n-2} - \sum_{i=1}^{n-2} A^{i-1} B L x_{k+n-i-2}\right) \\ \vdots \\ C(x_{k+1} - BLx_k) \\ Cx_k \end{pmatrix} = \begin{pmatrix} CA^{n-1} \\ CA^{n-2} \\ \vdots \\ CA \\ C \end{pmatrix} x_k. \tag{4.17}$$

Since $Lx_k \to 0$ by Eq. (4.16), the left-hand side tends to zero and hence the right-hand side tends to zero also. By the observability assumption, however, the matrix multiplying x_k on the right side of (4.17) has full rank. It follows that $x_k \to 0$.

Positive Definiteness of P. Assume the contrary, i.e., there exists some $x_0 \neq 0$ such that $x_0' P x_0 = 0$. Since P is positive semidefinite, from Eq. (4.15) we obtain

$$x_k'(Q + L'RL)x_k = 0, \qquad k = 0, 1, \ldots$$

Since $x_k \to 0$, we obtain $x_k' Q x_k = x_k' C' C x_k = 0$ and $x_k' L'RL x_k = 0$, or

$$Cx_k = 0, \qquad Lx_k = 0, \qquad k = 0, 1, \ldots$$

Thus all the controls $\mu^*(x_k) = Lx_k$ of the closed-loop system are zero while we have $Cx_k = 0$ for all k. Based on the observability assumption, we will show that this implies $x_0 = 0$, thereby reaching a contradiction. Indeed, consider Eq. (4.17) for $k = 0$. By the preceding equalities, the left-hand side is zero and hence

$$0 = \begin{pmatrix} CA^{n-1} \\ \vdots \\ CA \\ C \end{pmatrix} x_0.$$

Since the matrix multiplying x_0 above has full rank by the observability assumption, we obtain $x_0 = 0$, which contradicts the hypothesis $x_0 \neq 0$ and proves that P is positive definite.

Arbitrary Initial Matrix P_0. Next we show that the sequence of matrices $\{P_k(P_0)\}$, defined by Eq. (4.8) when the starting matrix is an arbitrary positive semidefinite symmetric matrix P_0, converges to $P = \lim_{k\to\infty} P_k(0)$. Indeed, the optimal cost of the problem of minimizing

$$x_k' P_0 x_k + \sum_{i=0}^{k-1} (x_i' Q x_i + u_i' R u_i) \tag{4.18}$$

subject to the system equation $x_{i+1} = Ax_i + Bu_i$ is equal to $x_0' P_k(P_0) x_0$. Hence we have for every $x_0 \in \Re^n$

$$x_0' P_k(0) x_0 \leq x_0' P_k(P_0) x_0.$$

Consider now the cost (4.18) corresponding to the controller $\mu(x_k) = u_k = Lx_k$, where L is defined by Eq. (4.11). This cost is

$$x_0' \left(D^{k\prime} P_0 D^k + \sum_{i=0}^{k-1} D^{i\prime} (Q + L'RL) D^i \right) x_0$$

and is greater or equal to $x_0' P_k(P_0) x_0$, which is the optimal value of the cost (4.18). Hence we have for all k and $x \in \Re^n$

$$x' P_k(0) x \leq x' P_k(P_0) x \leq x' \left(D^{k\prime} P_0 D^k + \sum_{i=0}^{k-1} D^{i\prime} (Q + L'RL) D^i \right) x.$$

We have proved that

$$\lim_{k\to\infty} P_k(0) = P,$$

and we also have, using the fact $\lim_{k\to\infty} D^{k\prime} P_0 D^k = 0$, and the relation $Q + L'RL = P - D'PD$ [cf. Eq. (4.12)],

$$\begin{aligned} \lim_{k\to\infty} \left\{ D^{k\prime} P_0 D_k + \sum_{i=0}^{k-1} D^{i\prime} (Q + L'RL) D^i \right\} & \\ = \lim_{k\to\infty} \left\{ \sum_{i=0}^{k-1} D^{i\prime} (Q + L'RL) D^i \right\} & \\ = \lim_{k\to\infty} \left\{ \sum_{i=0}^{k-1} D^{i\prime} (P - D'PD) D^i \right\} & \\ = P. & \end{aligned} \tag{4.19}$$

Combining the preceding three equations, we obtain

$$\lim_{k\to\infty} P_k(P_0) = P,$$

for an arbitrary positive semidefinite symmetric initial matrix P_0.

Uniqueness of Solution. If $\tilde{P}$ is another positive semidefinite symmetric solution of the algebraic Riccati equation (4.9), we have $P_k(\tilde{P}) = \tilde{P}$ for all $k = 0, 1, \ldots$ From the convergence result just proved, we then obtain

$$\lim_{k\to\infty} P_k(\tilde{P}) = P,$$

implying that $\tilde{P} = P$. **Q.E.D.**

The assumptions of the preceding proposition can be relaxed somewhat. Suppose that, instead of controllability of the pair (A, B), we assume that the system is *stabilizable* in the sense that there exists an $m \times n$ feedback gain matrix G such that the closed-loop system $x_{k+1} = (A + BG)x_k$ is stable. Then the proof of convergence of $P_k(0)$ to some positive semidefinite P given previously carries through. [We use the stationary control law $\mu(x) = Gx$ for which the closed-loop system is stable to ensure that $x_0' P_k(0) x_0$ is bounded.] Suppose that, instead of observability of the pair (A, C), the system is assumed *detectable* in the sense that A is such that if $u_k \to 0$ and $Cx_k \to 0$ then it follows that $x_k \to 0$. (This essentially means that instability of the system can be detected by looking at the measurement sequence $\{z_k\}$ with $z_k = Cx_k$.) Then Eq. (4.16) implies that $x_k \to 0$ and that the system $x_{k+1} = (A + BL)x_k$ is stable. The other parts of the proof of the proposition follow similarly, with the exception of positive definiteness of P, which cannot be guaranteed anymore. (As an example, take $A = 0$, $B = 0$, $C = 0$, $R > 0$. Then both the stabilizability and the detectability assumptions are satisfied, but $P = 0$.)

To summarize, if the controllability and observability assumptions of the proposition are replaced by the preceding stabilizability and detectability assumptions, the conclusions of the proposition hold with the exception of positive definiteness of the limit matrix P, which can now only be guaranteed to be positive semidefinite.

Random System Matrices

We consider now the case where $\{A_0, B_0\}, \ldots, \{A_{N-1}, B_{N-1}\}$ are not known but rather are independent random matrices that are also independent of $w_0, w_1, \ldots, w_{N-1}$. Their probability distributions are given, and they are assumed to have finite second moments. This problem falls again within the framework of the basic problem by considering as disturbance at each time k the triplet (A_k, B_k, w_k). The DP algorithm is written as

$$J_N(x_N) = x_N' Q_N x_N,$$

$$J_k(x_k) = \min_{u_k} \underset{w_k, A_k, B_k}{E} \left\{ x_k' Q_k x_k + u_k' R_k u_k + J_{k+1}(A_k x_k + B_k u_k + w_k) \right\}.$$

Calculations very similar to those for the case where A_k, B_k are not random show that the optimal control law has the form

$$\mu_k^*(x_k) = L_k x_k,$$

where the gain matrices L_k are given by

$$L_k = -\big(R_k + E\{B_k' K_{k+1} B_k\}\big)^{-1} E\{B_k' K_{k+1} A_k\},$$

and where the matrices K_k are given by the recursive equation

$$K_N = Q_N,$$

$$K_k = E\{A_k' K_{k+1} A_k\} - E\{A_k' K_{k+1} B_k\}\big(R_k + E\{B_k' K_{k+1} B_k\}\big)^{-1} E\{B_k' K_{k+1} A_k\} + Q_k. \tag{4.20}$$

In the case of a stationary system and constant matrices Q_k and R_k it is not necessarily true that the above equation converges to a steady-state solution. This is demonstrated in Fig. 4.1.3 for a scalar system, where it is shown that if the expression

$$T = E\{A^2\}E\{B^2\} - \big(E\{A\}\big)^2\big(E\{B\}\big)^2$$

exceeds a certain threshold, the matrices K_k diverge to ∞ starting from any nonnegative initial condition. A possible interpretation is that if there is a lot of uncertainty about the system, as quantified by T, optimization over a long horizon is meaningless. This phenomenon has been called the *uncertainty threshold principle*; see Athans, Ku, and Gershwin [AGK77], and Ku and Athans [KuA77].

On Certainty Equivalence

We close this section by making an observation about the simplifications that arise when the cost is quadratic. Consider the minimization over u of

$$\underset{w}{E}\big\{(ax + bu + w)^2\big\},$$

where a and b are given scalars, x is known, and w is a random variable. The optimum is attained for

$$u^* = -\left(\frac{a}{b}\right)x - \left(\frac{1}{b}\right)E\{w\}.$$

Thus u^* depends on the probability distribution of w only through the mean $E\{w\}$. In particular, the result of the optimization is the same as

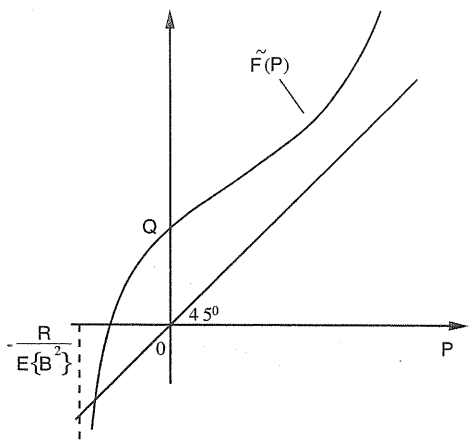


Figure 4.1.3 Graphical illustration of the asymptotic behavior of the generalized Riccati equation (4.20) in the case of a scalar stationary system (one-dimensional state and control). Using P_k in place of K_{N-k}, this equation is written as

$$P_{k+1} = \tilde{F}(P_k),$$

where the function $\tilde{F}$ is given by

$$\tilde{F}(P) = \frac{E\{A^2\}RP}{E\{B^2\}P + R} + Q + \frac{TP^2}{E\{B^2\}P + R},$$

$$T = E\{A^2\}E\{B^2\} - \big(E\{A\}\big)^2\big(E\{B\}\big)^2.$$

If $T = 0$, as in the case where A and B are not random, the Riccati equation becomes identical with the one of Fig. 4.1.2 and converges to a steady-state. Convergence also occurs when T has a small positive value. However, as illustrated in the figure, for T large enough, the graph of the function $\tilde{F}$ and the 45-degree line that passes through the origin do not intersect at a positive value of P, and the Riccati equation diverges to infinity.

for the corresponding deterministic problem where w is replaced by $E\{w\}$. This property is called the *certainty equivalence principle* and appears in various forms in many (but not all) stochastic control problems involving linear systems and quadratic cost. For the first problem of this section, where A_k, B_k are known, certainty equivalence holds because the optimal control law (4.2) is the same as the one that would be obtained from the corresponding deterministic problem where w_k is not random but rather is known and is equal to zero (its expected value). However, for the problem where A_k, B_k are random, the certainty equivalence principle does not hold, since if one replaces A_k, B_k with their expected values in Eq. (4.20) the resulting control law need not be optimal.

4.2 INVENTORY CONTROL

We consider now the inventory control problem discussed in Sections 1.1 and 1.2. We assume that excess demand at each period is backlogged and is filled when additional inventory becomes available. This is represented by negative inventory in the system equation

$$x_{k+1} = x_k + u_k - w_k, \qquad k = 0, 1, \ldots, N-1.$$

We assume that the demands w_k take values within some bounded interval and are independent. We will analyze the problem for the case of a holding/shortage cost of the form

$$r(x) = p \max(0, -x) + h \max(0, x),$$

where p and h are given nonnegative scalars. Thus the total expected cost to be minimized is

$$E\left\{\sum_{k=0}^{N-1}\big(cu_k + r(x_k + u_k - w_k)\big)\right\}.$$

We assume that the purchase cost per unit stock c is positive and that $p > c$. The last assumption is necessary for the problem to be well posed; if c, the purchase cost per unit, were greater than p, the depletion cost per unit, it would never be optimal to buy new stock at the last period and possibly in earlier periods. Much of the subsequent analysis generalizes to the case where r is a convex function that grows to infinity with asymptotic slopes p and h as its argument tends to $-\infty$ and ∞, respectively.

By applying the DP algorithm, we have

$$J_N(x_N) = 0,$$

$$J_k(x_k) = \min_{u_k \geq 0}\Big[cu_k + H(x_k + u_k) + E\big\{J_{k+1}(x_k + u_k - w_k)\big\}\Big], \qquad (4.21)$$

where the function H is defined by

$$H(y) = E\big\{r(y - w_k)\big\} = pE\big\{\max(0, w_k - y)\big\} + hE\big\{\max(0, y - w_k)\big\}.$$

Actually, H depends on k whenever the probability distribution of w_k depends on k. To simplify notation, we do not show this dependence and assume that all demands are identically distributed, but the following analysis carries through even when the demand distribution is time-varying. The function H can be seen to be convex, since $r(y - w_k)$ is convex in y for each fixed w_k, and taking expectation over w_k preserves convexity.

By introducing the variable $y_k = x_k + u_k$, we can write the DP Eq. (4.21) as

$$J_k(x_k) = \min_{y_k \geq x_k} G_k(y_k) - cx_k, \qquad (4.22)$$

where

$$G_k(y) = cy + H(y) + E\big\{J_{k+1}(y - w)\big\}. \qquad (4.23)$$

We will prove shortly that the function G_k is convex, but for the moment let us assume this convexity. Suppose that G_k has an unconstrained minimum with respect to y, denoted by S_k:

$$S_k = \arg\min_{y \in \Re} G_k(y).$$

Then, in view of the constraint $y_k \geq x_k$, it is seen that a minimizing y_k in Eq. (4.22) equals S_k if $x_k < S_k$, and equals x_k otherwise [since by convexity, $G_k(y)$ cannot decrease as y increases beyond S_k]. Using the reverse transformation $u_k = y_k - x_k$, we see that the minimum in the DP equation (4.21) is attained at $u_k = S_k - x_k$ if $x_k < S_k$, and at $u_k = 0$ otherwise. An optimal policy is determined by the sequence of scalars $\{S_0, S_1, \ldots, S_{N-1}\}$ and has the form

$$\mu_k^*(x_k) = \begin{cases} S_k - x_k & \text{if } x_k < S_k, \\ 0 & \text{if } x_k \geq S_k. \end{cases} \qquad (4.24)$$

Thus, the optimality of the policy (4.24) will be proved if we can show that the cost-to-go functions J_k [and hence also the functions G_k of Eq. (4.23)] are convex, and furthermore $\lim_{|y| \to \infty} G_k(y) = \infty$, so that the minimizing scalars S_k exist. We proceed to show these properties inductively.

We have that J_N is the zero function, so it is convex. Since $c < p$ and the derivative of $H(y)$ tends to $-p$ as $y \to -\infty$, we see that $G_{N-1}(y)$ [which is $cy + H(y)$] has a derivative that becomes negative as $y \to -\infty$ and becomes positive as $y \to \infty$ (see Fig. 4.2.1). Therefore

$$\lim_{|y| \to \infty} G_{N-1}(y) = \infty.$$

Thus, as shown above, an optimal policy at time $N-1$ is given by

$$\mu_{N-1}^*(x_{N-1}) = \begin{cases} S_{N-1} - x_{N-1} & \text{if } x_{N-1} < S_{N-1}, \\ 0 & \text{if } x_{N-1} \geq S_{N-1}. \end{cases}$$

Furthermore, from the DP equation (4.21) we have

$$J_{N-1}(x_{N-1}) = \begin{cases} c(S_{N-1} - x_{N-1}) + H(S_{N-1}) & \text{if } x_{N-1} < S_{N-1}, \\ H(x_{N-1}) & \text{if } x_{N-1} \geq S_{N-1}, \end{cases}$$

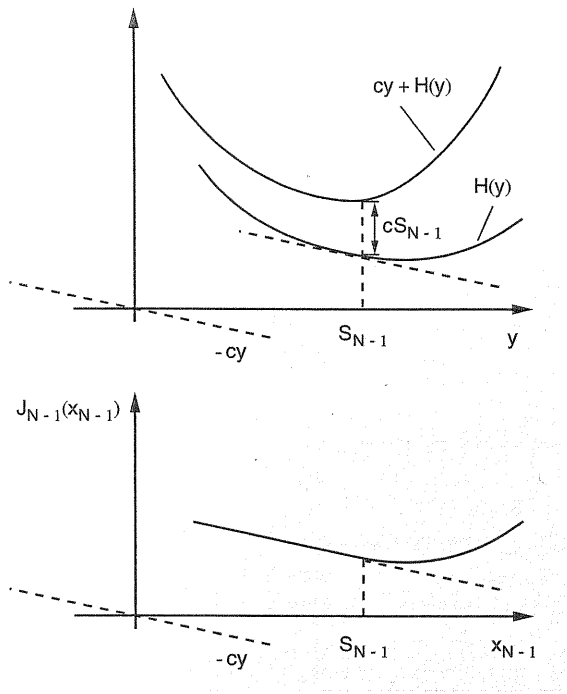


Figure 4.2.1 Structure of the cost-to-go functions when the fixed cost is zero.

which is a convex function because H is convex and S_{N-1} minimizes $cy + H(y)$ (see Fig. 4.2.1). Thus, given the convexity of J_N, we were able to prove the convexity of J_{N-1}. Furthermore,

$$\lim_{|y|\to\infty} J_{N-1}(y) = \infty.$$

This argument can be repeated to show that for all $k = N-2, \ldots, 0$, if J_{k+1} is convex, $\lim_{|y|\to\infty} J_{k+1}(y) = \infty$, and $\lim_{|y|\to\infty} G_k(y) = \infty$, then we have

$$J_k(x_k) = \begin{cases} c(S_k - x_k) + H(S_k) + E\{J_{k+1}(S_k - w_k)\} & \text{if } x_k < S_k, \\ H(x_k) + E\{J_{k+1}(x_k - w_k)\} & \text{if } x_k \geq S_k, \end{cases}$$

where S_k is an unconstrained minimum of G_k. Furthermore, J_k is convex, $\lim_{|y|\to\infty} J_k(y) = \infty$, and $\lim_{|y|\to\infty} G_{k-1}(y) = \infty$. Thus, the optimality proof of the policy (4.24) is completed.

Positive Fixed Cost and (s, S) Policies

We now turn to the more complicated case where there is a positive fixed cost K associated with a positive inventory order. Thus the cost for ordering inventory $u \geq 0$ is

$$C(u) = \begin{cases} K + cu & \text{if } u > 0, \\ 0 & \text{if } u = 0. \end{cases}$$

The DP algorithm takes the form

$$J_N(x_N) = 0,$$

$$J_k(x_k) = \min_{u_k \geq 0} \Big[C(u_k) + H(x_k + u_k) + E\big\{ J_{k+1}(x_k + u_k - w_k) \big\} \Big],$$

with H defined as earlier by

$$H(y) = E\big\{ r(y - w) \big\} = pE\big\{ \max(0, w - y) \big\} + hE\big\{ \max(0, y - w) \big\}.$$

Consider again the functions

$$G_k(y) = cy + H(y) + E\big\{ J_{k+1}(y - w) \big\}.$$

Then J_k is written as

$$J_k(x_k) = \min \Big[G_k(x_k), \min_{u_k > 0} \big[K + G_k(x_k + u_k) \big] \Big] - cx_k,$$

or equivalently, through the change of variable $y_k = x_k + u_k$,

$$J_k(x_k) = \min \Big[G_k(x_k), \min_{y_k > x_k} \big[K + G_k(y_k) \big] \Big] - cx_k.$$

If, as in the case where $K = 0$, we could prove that the functions G_k are convex, then it would not be difficult to verify [see part (d) of the following Lemma 4.2.1] that the policy

$$\mu_k^*(x_k) = \begin{cases} S_k - x_k & \text{if } x_k < s_k, \\ 0 & \text{if } x_k \geq s_k, \end{cases} \tag{4.25}$$

is optimal, where S_k is a value of y that minimizes $G_k(y)$ and s_k is the smallest value of y for which $G_k(y) = K + G_k(S_k)$. A policy of the form (4.25) is known as a *multiperiod* (s, S) *policy*.

Unfortunately, when $K > 0$ it is not necessarily true that the functions G_k are convex. This opens the possibility of G_k having the form shown in Fig. 4.2.2, where the optimal policy is to order $(S - x)$ in interval I, zero in intervals II and IV, and $(\tilde{S} - x)$ in interval III. However, we will show that even though the functions G_k may not be convex, they have the property

$$K + G_k(z+y) \geq G_k(y) + z \left(\frac{G_k(y) - G_k(y-b)}{b} \right), \qquad \text{for all } z \geq 0, b > 0, y. \tag{4.26}$$

This property is called *K-convexity* and was first used by Scarf [Sca60] to show the optimality of multiperiod (s, S) policies. Now if K-convexity holds, the situation shown in Fig. 4.2.2 is impossible; if y_0 is the local maximum in the interval III, then we must have, for sufficiently small $b > 0$,

$$\frac{G_k(y_0) - G_k(y_0 - b)}{b} \geq 0,$$

and from Eq. (4.26) it follows that

$$K + G_k(\tilde{S}) \geq G_k(y_0),$$

which contradicts the construction shown in Fig. 4.2.2. More generally, it will be shown by using part (d) of the following Lemma 4.2.1 that if the K-convexity Eq. (4.26) holds, then an optimal policy takes the (s, S) form (4.25).

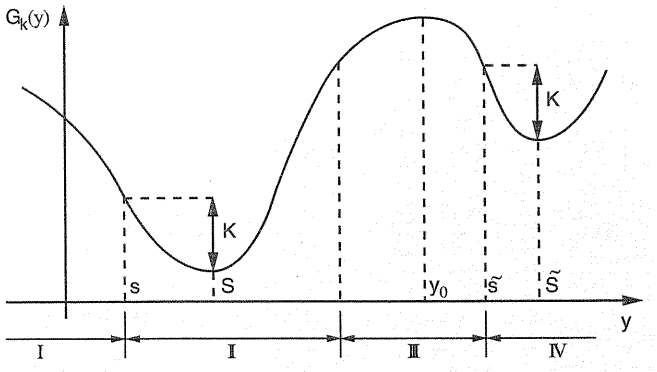


Figure 4.2.2 Potential form of the function G_k when the fixed cost is nonzero. If G_k had the form shown in the figure, the optimal policy would be to order $(S - x)$ in interval I, zero in intervals II and IV, and $(\tilde{S} - x)$ in interval III. The use of K-convexity allows us to show that the form of G_k shown in the figure is impossible.

Definition 4.2.1: We say that a real-valued function g is *K-convex*, where $K \geq 0$, if

$$K + g(z+y) \geq g(y) + z\left(\frac{g(y) - g(y-b)}{b}\right), \qquad \text{for all } z \geq 0, b > 0, y.$$

Some properties of K-convex functions are provided in the following lemma. Part (d) of the lemma essentially proves the optimality of the (s, S) policy (4.25) when the functions G_k are K-convex.

Lemma 4.2.1:

(a) A real-valued convex function g is also 0-convex and hence also K-convex for all $K \geq 0$.

(b) If $g_1(y)$ and $g_2(y)$ are K-convex and L-convex ($K \geq 0$, $L \geq 0$), respectively, then $\alpha g_1(y) + \beta g_2(y)$ is $(\alpha K + \beta L)$-convex for all $\alpha > 0$ and $\beta > 0$.

(c) If $g(y)$ is K-convex and w is a random variable, then $E_w\{g(y - w)\}$ is also K-convex, provided $E_w\{|g(y - w)|\} < \infty$ for all y.

(d) If g is a continuous K-convex function and $g(y) \to \infty$ as $|y| \to \infty$, then there exist scalars s and S with $s \leq S$ such that

(i) $g(S) \leq g(y)$, for all scalars y;

(ii) $g(S) + K = g(s) < g(y)$, for all $y < s$;

(iii) $g(y)$ is a decreasing function on $(-\infty, s)$;

(iv) $g(y) \leq g(z) + K$ for all y, z with $s \leq y \leq z$.

Proof: Part (a) follows from elementary properties of convex functions, and parts (b) and (c) follow from the definition of a K-convex function. We will thus concentrate on proving part (d).

Since g is continuous and $g(y) \to \infty$ as $|y| \to \infty$, there exists a minimizing point of g. Let S be such a point. Also let s be the smallest scalar z for which $z \leq S$ and $g(S) + K = g(z)$. For all y with $y < s$, we have from the definition of K-convexity

$$K + g(S) \geq g(s) + \frac{S - s}{s - y}\big(g(s) - g(y)\big).$$

Since $K + g(S) - g(s) = 0$, we obtain $g(s) - g(y) \leq 0$. Since $y < s$ and s is the smallest scalar for which $g(S) + K = g(s)$, we must have $g(s) < g(y)$ and (ii) is proved. To prove (iii), note that for $y_1 < y_2 < s$, we have

$$K + g(S) \geq g(y_2) + \frac{S - y_2}{y_2 - y_1}\big(g(y_2) - g(y_1)\big).$$

Also from (ii),

$$g(y_2) > g(S) + K,$$

and by adding these two inequalities we have

$$0 > \frac{S - y_2}{y_2 - y_1}\big(g(y_2) - g(y_1)\big),$$

from which we obtain $g(y_1) > g(y_2)$, thus proving (iii). To prove (iv), we note that it holds for $y = z$ as well as for either $y = S$ or $y = s$. There remain two other possibilities: $S < y < z$ and $s < y < S$. If $S < y < z$, then by K-convexity

$$K + g(z) \geq g(y) + \frac{z-y}{y-S}\big(g(y) - g(S)\big) \geq g(y),$$

and (iv) is proved. If $s < y < S$, then by K-convexity

$$g(s) = K + g(S) \geq g(y) + \frac{S-y}{y-s}\big(g(y) - g(s)\big),$$

from which

$$\left(1 + \frac{S-y}{y-s}\right) g(s) \geq \left(1 + \frac{S-y}{y-s}\right) g(y),$$

and $g(s) \geq g(y)$. Noting that

$$g(z) + K \geq g(S) + K = g(s),$$

it follows that $g(z) + K \geq g(y)$. Thus (iv) is proved for this case as well. **Q.E.D.**

Consider now the function G_{N-1}:

$$G_{N-1}(y) = cy + H(y).$$

Clearly, G_{N-1} is convex and hence by part (a) of Lemma 4.2.1, it is also K-convex. We have

$$J_{N-1}(x) = \min\Big[G_{N-1}(x), \min_{y>x}\big[K + G_{N-1}(y)\big]\Big] - cx,$$

and it can be seen that

$$J_{N-1}(x) = \begin{cases} K + G_{N-1}(S_{N-1}) - cx & \text{for } x < s_{N-1}, \\ G_{N-1}(x) - cx & \text{for } x \geq s_{N-1}, \end{cases} \tag{4.27}$$

where S_{N-1} minimizes $G_{N-1}(y)$ and s_{N-1} is the smallest value of y for which $G_{N-1}(y) = K + G_{N-1}(S_{N-1})$. Note that since $K > 0$, we have $s_{N-1} \neq S_{N-1}$ and furthermore the derivative of G_{N-1} at s_{N-1} is negative. As a result the left derivative of J_{N-1} at s_{N-1} is greater than the right derivative, as shown in Fig. 4.2.3, and J_{N-1} is not convex. However, we will show that J_{N-1} is K-convex based on the fact that G_{N-1} is K-convex. To this end we must verify that for all $z \geq 0$, $b > 0$, and y, we have

$$K + J_{N-1}(y+z) \geq J_{N-1}(y) + z\left(\frac{J_{N-1}(y) - J_{N-1}(y-b)}{b}\right). \tag{4.28}$$

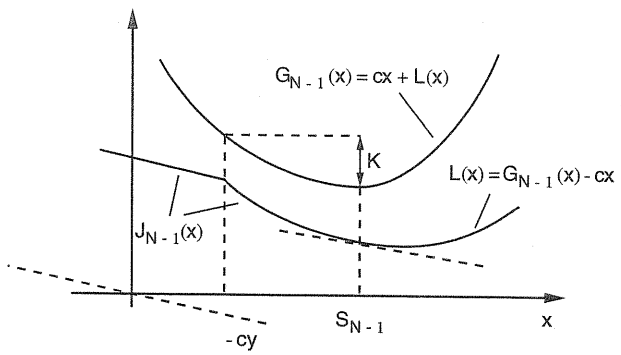


Figure 4.2.3 Structure of the cost-to-go function when fixed cost is nonzero.

We distinguish three cases:

Case 1: $y \geq s_{N-1}$. If $y - b \geq s_{N-1}$, then in this region of values of z, b, and y, the function J_{N-1}, by Eq. (4.27), is the sum of a K-convex function and a linear function. Hence by part (b) of Lemma 4.2.1, it is K-convex and Eq. (4.28) holds. If $y - b < s_{N-1}$, then in view of Eq. (4.27) we can write Eq. (4.28) as

$$\begin{aligned} K + G_{N-1}(y+z) - c(y+z) &\geq G_{N-1}(y) - cy \\ &\quad + z\left(\frac{G_{N-1}(y) - cy - G_{N-1}(s_{N-1}) + c(y-b)}{b}\right) \end{aligned}$$

or equivalently

$$K + G_{N-1}(y+z) \geq G_{N-1}(y) + z\left(\frac{G_{N-1}(y) - G_{N-1}(s_{N-1})}{b}\right). \tag{4.29}$$

Now if y is such that $G_{N-1}(y) \geq G_{N-1}(s_{N-1})$, then by K-convexity of G_{N-1} we have

$$\begin{aligned} K + G_{N-1}(y+z) &\geq G_{N-1}(y) + z\left(\frac{G_{N-1}(y) - G_{N-1}(s_{N-1})}{y - s_{N-1}}\right) \\ &\geq G_{N-1}(y) + z\left(\frac{G_{N-1}(y) - G_{N-1}(s_{N-1})}{b}\right). \end{aligned}$$

Thus Eq. (4.29) and hence also Eq. (4.28) hold. If y is such that $G_{N-1}(y) < G_{N-1}(s_{N-1})$, then we have

$$\begin{aligned} K + G_{N-1}(y+z) &\geq K + G_{N-1}(S_{N-1}) \\ &= G_{N-1}(s_{N-1}) \\ &> G_{N-1}(y) \\ &\geq G_{N-1}(y) + z\left(\frac{G_{N-1}(y) - G_{N-1}(s_{N-1})}{b}\right). \end{aligned}$$

So for this case, Eq. (4.29) holds, and hence also the desired K-convexity Eq. (4.28) holds.

Case 2: $y \le y+z \le s_{N-1}$. In this region, by Eq. (4.27), the function J_{N-1} is linear and hence the K-convexity Eq. (4.28) holds.

Case 3: $y < s_{N-1} < y+z$. For this case, in view of Eq. (4.28), we can write the K-convexity Eq. (4.28) as

$$K + G_{N-1}(y+z) - c(y+z) \ge G_{N-1}(s_{N-1}) - cy + z\left(\frac{G_{N-1}(s_{N-1}) - cy - G_{N-1}(s_{N-1}) + c(y-b)}{b}\right),$$

or equivalently

$$K + G_{N-1}(y+z) \ge G_{N-1}(s_{N-1}),$$

which holds by the definition of s_{N-1}.

We have thus proved that K-convexity and continuity of G_{N-1}, together with the fact that $G_{N-1}(y) \to \infty$ as $|y| \to \infty$, imply K-convexity of J_{N-1}. In addition, J_{N-1} can be seen to be continuous. Now using Lemma 4.2.1, it follows from Eq. (4.23) that G_{N-2} is a K-convex function. Furthermore, by using the boundedness of w_{N-2}, it follows that G_{N-2} is continuous and, in addition, $G_{N-2}(y) \to \infty$ as $|y| \to \infty$. Repeating the preceding argument, we obtain that J_{N-2} is K-convex, and proceeding similarly, we prove K-convexity and continuity of the functions G_k for all k, as well as that $G_k(y) \to \infty$ as $|y| \to \infty$. At the same time [by using part (d) of Lemma 4.2.1] we prove optimality of the multiperiod (s, S) policy of Eq. (4.25).

The optimality of policies of the (s, S) type can be proved for several other inventory problems (see Exercises 4.3 to 4.10).

4.3 DYNAMIC PORTFOLIO ANALYSIS

Portfolio theory deals with the question of how to invest a certain amount of wealth among a collection of assets, perhaps over a long time interval. One approach, to be discussed in this section, is to assume that an investor makes a decision in each of several successive time periods with the objective of maximizing final wealth. We will start with an analysis of a single-period model and then extend the results to the multiperiod case.

Let x_0 denote the initial wealth of the investor and assume that there are n risky assets, with corresponding random rates of return $e_1, \ldots, e_n$ among which the investor can allocate his wealth. The investor can also invest in a riskless asset offering a sure rate of return s. If we denote by $u_1, \ldots, u_n$ the corresponding amounts invested in the n risky assets and by

$(x_0 - u_1 - \cdots - u_n)$ the amount invested in the riskless asset, the wealth at the end of the first period is given by

$$x_1 = s(x_0 - u_1 - \cdots - u_n) + \sum_{i=1}^{n} e_i u_i,$$

or equivalently

$$x_1 = sx_0 + \sum_{i=1}^{n} (e_i - s)u_i.$$

The objective is to maximize over $u_1, \ldots, u_n$,

$$E\{U(x_1)\},$$

where U is a known function, referred to as the *utility function* for the investor (Appendix G contains a discussion of utility functions and their significance in the formulation of optimization problems under uncertainty). We assume that U is concave and twice continuously differentiable, and that the given expected value is well defined and finite for all x_0 and u_i. We will not impose constraints on $u_1, \ldots, u_n$. This is necessary in order to obtain the results in convenient form. A few additional assumptions will be made later.

Let us denote by $u_i^* = \mu^{i*}(x_0)$, $i = 1, \ldots, n$, the optimal amounts to be invested in the n risky assets when the initial wealth is x_0. We will show that when the utility function satisfies

$$-\frac{U'(x)}{U''(x)} = a + bx, \qquad \text{for all } x, \tag{4.30}$$

where U' and U'' denote the first and second derivatives of U, respectively, and a and b are some scalars, then the optimal portfolio is given by the linear policy

$$\mu^{i*}(x_0) = \alpha^i(a + bsx_0), \qquad i = 1, \ldots, n, \tag{4.31}$$

where α^i are some constant scalars. Furthermore, if $J(x_0)$ is the optimal value of the problem

$$J(x_0) = \max_{u_i} E\{U(x_1)\},$$

then we have

$$-\frac{J'(x_0)}{J''(x_0)} = \frac{a}{s} + bx_0, \qquad \text{for all } x_0. \tag{4.32}$$

It can be verified that the following utility functions $U(x)$ satisfy condition (4.30):

exponential : $e^{-x/a}$, for $b = 0$, $a > 0$,

logarithmic : $\ln(x + a)$, for $b = 1$,

power : $\big(1/(b-1)\big)(a + bx)^{1-(1/b)}$, for $b \ne 0$, $b \ne 1$.

Only concave utility functions from this class are admissible for our problem. Furthermore, if a utility function that is not defined over the whole real line is used, the problem should be formulated in a way that ensures that all possible values of the resulting final wealth are within the domain of definition of the utility function.

To show the desired relations, let us hypothesize that an optimal portfolio exists and is of the form

$$\mu^{i*}(x_0) = \alpha^i(x_0)(a + bsx_0),$$

where $\alpha^i(x_0)$, $i = 1, \ldots, n$, are some differentiable functions. We will prove that $d\alpha^i(x_0)/dx_0 = 0$ for all x_0, implying that the functions α^i must be constant, so the optimal portfolio has the linear form (4.31).

We have for every x_0 and $i = 1, \ldots, n$, by the optimality of $\mu^{i*}(x_0)$,

$$\begin{aligned}\frac{dE\{U(x_1)\}}{du_i} &= E\left\{U'\left(sx_0 + \sum_{j=1}^{n}(e_j - s)\alpha^j(x_0)(a + bsx_0)\right)(e_i - s)\right\} \\ &= 0.\end{aligned} \tag{4.33}$$

Differentiating each of the n equations above with respect to x_0 yields

$$\begin{aligned}E\left\{\begin{pmatrix}(e_1 - s)^2 \cdots (e_1 - s)(e_n - s)\\ \vdots \\ (e_n - s)(e_1 - s) \cdots (e_n - s)^2\end{pmatrix} U''(x_1)(a + bsx_0)\right\}&\begin{pmatrix}\frac{d\alpha^1(x_0)}{dx_0}\\ \vdots \\ \frac{d\alpha^n(x_0)}{dx_0}\end{pmatrix} \\ = -\begin{pmatrix}E\{U''(x_1)(e_1 - s)s\big(1 + \sum_{i=1}^{n}(e_i - s)\alpha^i(x_0)b\big)\}\\ \vdots \\ E\{U''(x_1)(e_n - s)s\big(1 + \sum_{i=1}^{n}(e_i - s)\alpha^i(x_0)b\big)\}\end{pmatrix}&.\end{aligned} \tag{4.34}$$

Using relation (4.31), we have

$$\begin{aligned}U''(x_1) &= -\frac{U'(x_1)}{a + b\big(sx_0 + \sum_{i=1}^{n}(e_i - s)\alpha^i(x_0)(a + bsx_0)\big)} \\ &= -\frac{U'(x_1)}{(a + bsx_0)\big(1 + \sum_{i=1}^{n}(e_i - s)\alpha^i(x_0)b\big)}.\end{aligned} \tag{4.35}$$

Substituting in Eq. (4.34) and using Eq. (4.33), we have that the right-hand side of Eq. (4.34) is the zero vector. The matrix on the left in Eq. (4.34), except for degenerate cases, can be shown to be nonsingular. Assuming that it is indeed nonsingular, we obtain

$$\frac{d\alpha^i(x_0)}{dx_0} = 0, \qquad i = 1, \ldots, n,$$

and $\alpha^i(x_0) = \alpha^i$, where α^i are some constants, thus proving the optimality of the linear policy (4.31).

We now prove Eq. (4.32). We have

$$\begin{aligned}J(x_0) &= E\{U(x_1)\} \\ &= E\left\{U\left(s\left(1 + \sum_{i=1}^{n}(e_i - s)\alpha^i b\right)x_0 + \sum_{i=1}^{n}(e_i - s)\alpha^i a\right)\right\}\end{aligned}$$

and hence

$$J'(x_0) = E\left\{U'(x_1)s\left(1 + \sum_{i=1}^{n}(e_i - s)\alpha^i b\right)\right\},$$

$$J''(x_0) = E\left\{U''(x_1)s^2\left(1 + \sum_{i=1}^{n}(e_i - s)\alpha^i b\right)^2\right\}. \tag{4.36}$$

The last relation after some calculation using Eq. (4.35) yields

$$J''(x_0) = -\frac{E\Big\{U'(x_1)s\big(1 + \sum_{i=1}^{n}(e_i - s)\alpha^i b\big)\Big\}s}{a + bsx_0}. \tag{4.37}$$

By combining Eqs. (4.36) and (4.37), we obtain the desired result:

$$-\frac{J'(x_0)}{J''(x_0)} = \frac{a}{s} + bx_0.$$

The Multiperiod Problem

We now extend the preceding one-period analysis to the multiperiod case. We will assume that the current wealth can be reinvested at the beginning of each of N consecutive time periods. We denote

x_k: the wealth of the investor at the start of the kth period,

u_i^k: the amount invested at the start of the kth period in the ith risky asset,

e_i^k: the rate of return of the ith risky asset during the kth period,

s_k: the rate of return of the riskless asset during the kth period.

We have the system equation

$$x_{k+1} = s_k x_k + \sum_{i=1}^{n}(e_i^k - s_k)u_i^k, \qquad k = 0, 1, \ldots, N - 1.$$

We assume that the vectors $e^k = (e_1^k, \ldots, e_n^k)$, $k = 0, \ldots, N-1$, are independent with given probability distributions that yield finite expected values throughout the following analysis. The objective is to maximize $E\{U(x_N)\}$, the expected utility of the terminal wealth x_N, where we assume that U satisfies

$$-\frac{U'(x)}{U''(x)} = a + bx, \qquad \text{for all } x.$$

Applying the DP algorithm to this problem, we have

$$J_N(x_N) = U(x_N),$$

$$J_k(x_k) = \max_{u_1^k, \ldots, u_n^k} E\left\{ J_{k+1}\left(s_k x_k + \sum_{i=1}^{n} (e_i^k - s_k) u_i^k \right) \right\}. \tag{4.38}$$

From the solution of the one-period problem, we have that the optimal policy at period $N-1$ has the form

$$\mu_{N-1}^*(x_{N-1}) = \alpha_{N-1}(a + b s_{N-1} x_{N-1}),$$

where α_{N-1} is an appropriate n-dimensional vector. Furthermore, we have

$$-\frac{J_{N-1}'(x)}{J_{N-1}''(x)} = \frac{a}{s_{N-1}} + bx.$$

Hence, applying the one-period result in the DP Eq. (4.38) for the next to the last period, we obtain the optimal policy

$$\mu_{N-2}^*(x_{N-2}) = \alpha_{N-2}\left(\frac{a}{s_{N-1}} + b s_{N-2} x_{N-2} \right),$$

where α_{N-2} is again an appropriate n-dimensional vector.

Proceeding similarly, we have for the kth period

$$\mu_k^*(x_k) = \alpha_k \left(\frac{a}{s_{N-1} \cdots s_{k+1}} + b s_k x_k \right), \tag{4.39}$$

where α_k is an n-dimensional vector that depends on the probability distributions of the rates of return e_i^k of the risky assets and are determined by maximization in the DP Eq. (4.38). The corresponding cost-to-go functions satisfy

$$-\frac{J_k'(x)}{J_k''(x)} = \frac{a}{s_{N-1} \cdots s_k} + bx, \qquad k = 0, 1, \ldots, N-1. \tag{4.40}$$

Thus it is seen that the investor, when faced with the opportunity to sequentially reinvest his wealth, uses a policy similar to that of the single-period case. Carrying the analysis one step further, it is seen that if the utility function U is such that $a = 0$, that is, U has one of the forms

$$\ln x, \qquad \text{for } b = 1,$$

$$\left(\frac{1}{b-1} \right) (bx)^{1-(1/b)}, \qquad \text{for } b \neq 0,\ b \neq 1,$$

then it follows from Eq. (4.39) that the investor acts at each stage k as if he were faced with a *single-period* investment characterized by the rates of return s_k and e_i^k, and the objective function $E\{U(x_{k+1})\}$. This policy whereby the investor can ignore the fact that he will have the opportunity to reinvest his wealth is called a *myopic policy*.

Note that a myopic policy is also optimal when $s_k = 1$ for all k, which means that wealth is discounted at the rate of return of the riskless asset. Furthermore, it has been shown by Mossin [Mos68] that when $a = 0$ a myopic policy is optimal even in the more general case where the rates of return s_k are independent random variables, and for the case where forecasts on the probability distributions of the rates of return e_i^k of the risky assets become available during the investment process (see Exercise 4.14).

It turns out that even in the more general case where $a \neq 0$ only a small amount of foresight is required on the part of the decision maker. It can be seen [compare Eqs. (4.38)-(4.40)] that the optimal policy (4.39) at period k is the one that the investor would use in a single-period problem to maximize over u_i^k, $i = 1, \ldots, n$,

$$E\{U(s_{N-1} \cdots s_{k+1} x_{k+1})\}$$

subject to

$$x_{k+1} = s_k x_k + \sum_{i=1}^{n} (e_i^k - s_k) u_i^k.$$

In other words, the investor at period k should maximize the expected utility of wealth that results if amounts u_i^k are invested in the risky assets in period k and the resulting wealth x_{k+1} is subsequently invested *exclusively* in the riskless asset during the remaining periods $k+1, \ldots, N-1$. This is known as a *partially myopic policy*. Such a policy can also be shown to be optimal when forecasts on the probability distributions of the rates of return of the risky assets become available during the investment process (see Exercise 4.14).

Another interesting aspect of the case where $a \neq 0$ is that if $s_k > 1$ for all k, then as the horizon becomes increasingly long ($N \to \infty$), the policy in the initial stages approaches a myopic policy [compare Eqs. (4.39) and (4.40)]. Thus, for $s_k > 1$, a partially myopic policy becomes asymptotically myopic as the horizon tends to infinity.

4.4 OPTIMAL STOPPING PROBLEMS

Optimal stopping problems of the type that we will consider in this and subsequent sections are characterized by the availability, at each state, of a control that stops the evolution of the system. Thus at each stage the decision maker observes the current state of the system and decides whether to continue the process (perhaps at a certain cost) or stop the process and incur a certain loss. If the decision is to continue, a control must be selected from a given set of available choices. If there is only one choice other than stopping, then each policy is characterized at each period by the *stopping set*, that is, the set of states where the policy stops the system.

Asset Selling

As a first example, consider a person having an asset (say a piece of land) for which he is offered an amount of money from period to period. We assume that the offers, denoted $w_0, w_1, \ldots, w_{N-1}$, are random and independent, and take values within some bounded interval of nonnegative numbers ($w_k = 0$ could correspond to no offer received during the period). If the person accepts an offer, he can invest the money at a fixed rate of interest $r > 0$, and if he rejects the offer, he waits until the next period to consider the next offer. Offers rejected are not renewed, and we assume that the last offer w_{N-1} must be accepted if every prior offer has been rejected. The objective is to find a policy for accepting and rejecting offers that maximizes the revenue of the person at the Nth period.

The DP algorithm for this problem can be derived by elementary reasoning. As a modeling exercise, however, we will embed the problem in the framework of the basic problem by specifying the system and cost. We define the state space to be the real line, augmented with an additional state (call it T), which is a *termination state*. By writing that the system is at state $x_k = T$ at some time $k \leq N-1$, we mean that the asset has already been sold. By writing that the system is at a state $x_k \neq T$ at some time $k \leq N-1$, we mean that the asset has not been sold as yet and the offer under consideration is equal to x_k (and also equal to the kth offer w_{k-1}). We take $x_0 = 0$ (a fictitious "null" offer). The control space consists of two elements u^1 and u^2, corresponding to the decisions "sell" and "do not sell," respectively. We view w_k as the disturbance at time k.

With these conventions, we may write a system equation of the form

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

where the function f_k is defined via the relation

$$x_{k+1} = \begin{cases} T & \text{if } x_k = T, \text{ or if } x_k \neq T \text{ and } u_k = u^1 \text{ (sell)}, \\ w_k & \text{otherwise.} \end{cases}$$

Note that a sell decision at time k ($u_k = u^1$) accepts the offer w_{k-1}, and that no explicit sell decision is required to accept the last offer w_{N-1}, as it must be accepted by assumption if the asset has not yet been sold. The corresponding reward function may be written as

$$\mathop{E}_{\substack{w_k \\ k=0,1,\ldots,N-1}} \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\}$$

where

$$g_N(x_N) = \begin{cases} x_N & \text{if } x_N \neq T, \\ 0 & \text{otherwise,} \end{cases}$$

$$g_k(x_k, u_k, w_k) = \begin{cases} (1+r)^{N-k} x_k & \text{if } x_k \neq T \text{ and } u_k = u^1 \text{ (sell)}, \\ 0 & \text{otherwise.} \end{cases}$$

Based on this formulation we can write the corresponding DP algorithm:

$$J_N(x_N) = \begin{cases} x_N & \text{if } x_N \neq T, \\ 0 & \text{if } x_N = T, \end{cases} \tag{4.41}$$

$$J_k(x_k) = \begin{cases} \max\left[(1+r)^{N-k} x_k,\, E\{J_{k+1}(w_k)\}\right] & \text{if } x_k \neq T, \\ 0 & \text{if } x_k = T. \end{cases} \tag{4.42}$$

In the above equation, $(1+r)^{N-k} x_k$ is the revenue resulting from decision u^1 (sell) when the offer is x_k, and $E\{J_{k+1}(w_k)\}$ represents the expected revenue corresponding to the decision u^2 (do not sell). Thus, the optimal policy is to accept an offer if it is greater than $E\{J_{k+1}(w_k)\}/(1+r)^{N-k}$, which represents expected revenue discounted to the present time:

$$\text{accept the offer } x_k \qquad \text{if } x_k > \alpha_k,$$

$$\text{reject the offer } x_k \qquad \text{if } x_k < \alpha_k,$$

where

$$\alpha_k = \frac{E\{J_{k+1}(w_k)\}}{(1+r)^{N-k}}.$$

When $x_k = \alpha_k$, both acceptance and rejection are optimal. Thus the optimal policy is determined by the scalar sequence $\{\alpha_k\}$ (see Fig. 4.4.1).

Properties of the Optimal Policy

We will now derive some properties of the optimal policy with some further analysis. Let us assume that the offers w_k are identically distributed, and to simplify notation, let us drop the time index k and denote by $E_w\{\cdot\}$ the expected value of the corresponding expression over w_k, for all k. We will then show that

$$\alpha_k \geq \alpha_{k+1}, \qquad \text{for all } k,$$

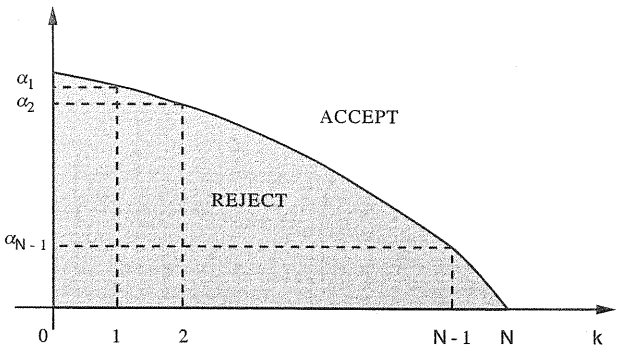


Figure 4.4.1 Optimal policy for accepting offers in the asset selling problem.

which expresses the intuitive fact that if an offer is good enough to be acceptable at time k, it should also be acceptable at time $k+1$ when there will be one less chance for improvement. We will also obtain an equation for the limit of α_k as $k \to -\infty$.

Let us introduce the functions

$$V_k(x_k) = \frac{J_k(x_k)}{(1+r)^{N-k}}, \qquad x_k \neq T.$$

It can be seen from Eqs. (4.41) and (4.42) that

$$V_N(x_N) = x_N, \tag{4.43}$$

$$V_k(x_k) = \max\left[x_k,\ (1+r)^{-1} \underset{w}{E}\{V_{k+1}(w)\}\right], \qquad k = 0, 1, \ldots, N-1, \tag{4.44}$$

and that

$$\alpha_k = \frac{E_w\{V_{k+1}(w)\}}{1+r}.$$

To prove that $\alpha_k \geq \alpha_{k+1}$, note that from Eqs. (4.43) and (4.44), we have

$$V_{N-1}(x) \geq V_N(x), \qquad \text{for all } x \geq 0.$$

Applying Eq. (4.44) for $k = N-2$ and $k = N-1$, and using the preceding inequality, we obtain for all $x \geq 0$

$$\begin{aligned} V_{N-2}(x) &= \max\left[x,\ (1+r)^{-1} \underset{w}{E}\{V_{N-1}(w)\}\right] \\ &\geq \max\left[x,\ (1+r)^{-1} \underset{w}{E}\{V_N(w)\}\right] \\ &= V_{N-1}(x). \end{aligned}$$

Continuing in the same manner, we see that

$$V_k(x) \geq V_{k+1}(x), \qquad \text{for all } x \geq 0 \text{ and } k.$$

Since $\alpha_k = E_w\{V_{k+1}(w)\}/(1+r)$, we obtain $\alpha_k \geq \alpha_{k+1}$, as desired.

Let us now see what happens when the horizon N is very large. From the algorithm (4.43) and (4.44) we have

$$V_k(x_k) = \max(x_k, \alpha_k). \tag{4.45}$$

Hence we obtain

$$\begin{aligned} \alpha_k &= \frac{1}{1+r} E_w\{V_{k+1}(w)\} \\ &= \frac{1}{1+r} \int_0^{\alpha_{k+1}} \alpha_{k+1} dP(w) + \frac{1}{1+r} \int_{\alpha_{k+1}}^{\infty} w dP(w), \end{aligned}$$

where the function P is defined for all scalars λ by

$$P(\lambda) = \text{Prob}\{w < \lambda\}.$$

The difference equation for α_k may also be written as

$$\alpha_k = \frac{P(\alpha_{k+1})}{1+r}\alpha_{k+1} + \frac{1}{1+r}\int_{\alpha_{k+1}}^{\infty} w dP(w), \qquad \text{for all } k, \tag{4.46}$$

with $\alpha_N = 0$.

Now since we have

$$0 \leq \frac{P(\alpha)}{1+r} \leq \frac{1}{1+r} < 1, \qquad \text{for all } \alpha \geq 0,$$

$$0 \leq \frac{1}{1+r}\int_{\alpha_{k+1}}^{\infty} w dP(w) \leq \frac{E\{w\}}{1+r}, \qquad \text{for all } k,$$

it can be seen, using the property $\alpha_k \geq \alpha_{k+1}$, that the sequence $\{\alpha_k\}$ generated (backward) by the difference equation (4.46) converges (as $k \to -\infty$) to a constant $\overline{\alpha}$ satisfying

$$(1+r)\overline{\alpha} = P(\overline{\alpha})\overline{\alpha} + \int_{\overline{\alpha}}^{\infty} w dP(w).$$

This equation is obtained from Eq. (4.46) by taking limits as $k \to -\infty$ and by using the fact that P is continuous from the left.

Thus, when the horizon N tends to become longer, the optimal policy for every fixed $k \geq 1$ can be approximated by the stationary policy

$$\begin{aligned} &\text{accept the offer } x_k \qquad \text{if } x_k > \overline{\alpha}, \\ &\text{reject the offer } x_k \qquad \text{if } x_k < \overline{\alpha}. \end{aligned}$$

The optimality of this policy for the corresponding infinite horizon problem will be shown in Section 7.3.

Purchasing with a Deadline

Let us consider another stopping problem that has similar nature. Assume that a certain quantity of raw material is required by a certain time. If the price of this material fluctuates, there arises the problem of deciding whether to purchase at the current price or wait a further period, during which the price may go up or down. We thus want to minimize the expected price of purchase. We assume that successive prices w_k are independent and identically distributed with distribution $P(w_k)$, and that the purchase must be made within N time periods.

This problem and the earlier asset selling problem have obvious similarities. Let us denote by

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

the price prevailing at the beginning of period $k+1$. We have similar to the earlier problem the DP algorithm

$$J_N(x_N) = x_N,$$

$$J_k(x_k) = \min\Big[x_k,\, E\big\{J_{k+1}(w_k)\big\}\Big].$$

Note that $J_k(x_k)$ is the optimal cost-to-go when the current price is x_k *and* the material has not been purchased yet. To be strictly formal, we should introduce a termination state T, to which the system moves following a purchasing decision and at which the system subsequently stays at no cost. A nonzero cost is incurred only when moving from x_k to T; this cost is equal to x_k. Thus the cost-to-go from the termination state T is 0, and for this reason it was neglected in the preceding DP equation.

The optimal policy is given by

$$\text{purchase if } x_k < \alpha_k,$$

$$\text{do not purchase if } x_k > \alpha_k,$$

where

$$\alpha_k = E\big\{J_{k+1}(w_k)\big\}.$$

Similar to the asset selling problem, the thresholds $\alpha_1, \alpha_2, \ldots, \alpha_{N-1}$ can be obtained from the discrete-time equation

$$\alpha_k = \alpha_{k+1}\big(1 - P(\alpha_{k+1})\big) + \int_0^{\alpha_{k+1}} w\,dP(w),$$

with the terminal condition

$$\alpha_{N-1} = \int_0^{\infty} w\,dP(w) = E\{w\}.$$

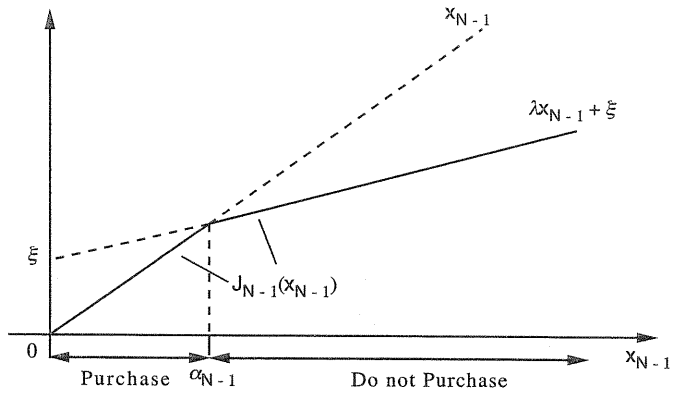


Figure 4.4.2 Structure of the cost-to-go function $J_{N-1}(x_{N-1})$ when prices are correlated.

The Case of Correlated Prices

Consider now a variation of the purchasing problem where we do not assume that the successive prices $w_0, \ldots, w_{N-1}$ are independent. Instead, we assume that they are correlated and can be represented as the state of a linear system driven by independent disturbances (cf. Section 1.4). In particular, we have

$$w_k = x_{k+1}, \qquad k = 0, 1, \ldots, N-1,$$

with

$$x_{k+1} = \lambda x_k + \xi_k, \qquad x_0 = 0,$$

where λ is a scalar with $0 \le \lambda < 1$ and $\xi_0, \xi_1, \ldots, \xi_{N-1}$ are independent identically distributed random variables taking positive values with given probability distribution. As discussed in Section 1.4, the DP algorithm under these circumstances takes the form

$$J_N(x_N) = x_N,$$

$$J_k(x_k) = \min\Big[x_k,\, E\big\{J_{k+1}(\lambda x_k + \xi_k)\big\}\Big],$$

where the cost associated with the purchasing decision is x_k and the cost associated with the waiting decision is $E\big\{J_{k+1}(\lambda x_k + \xi_k)\big\}$.

We will show that in this case the optimal policy is of the same type as the one for independent prices. Indeed, we have

$$J_{N-1}(x_{N-1}) = \min\bigl[x_{N-1},\, \lambda x_{N-1} + \overline{\xi}\,\bigr],$$

where $\overline{\xi} = E\{\xi_{N-1}\}$. As shown in Fig. 4.4.2, an optimal policy at time $N-1$ is given by

$$\text{purchase if } x_{N-1} < \alpha_{N-1},$$

$$\text{do not purchase if } x_{N-1} > \alpha_{N-1},$$

where α_{N-1} is defined from the equation $\alpha_{N-1} = \lambda\alpha_{N-1} + \overline{\xi}$:

$$\alpha_{N-1} = \frac{\overline{\xi}}{1-\lambda}.$$

Note that

$$J_{N-1}(x) \leq J_N(x), \qquad \text{for all } x,$$

and that J_{N-1} is concave and increasing in x. Using this fact in the DP algorithm, one may show (the monotonicity property of DP; Exercise 1.23 in Chapter 1) that

$$J_k(x) \leq J_{k+1}(x), \qquad \text{for all } x \text{ and } k,$$

and that J_k is concave and increasing in x for all k. Furthermore, since $\overline{\xi} = E\{\xi_k\} > 0$ for all k, one can show that

$$E\bigl\{J_{k+1}(\xi_k)\bigr\} > 0, \qquad \text{for all } k.$$

These facts imply (as illustrated in Fig. 4.4.3) that the optimal policy for every period k is of the form

$$\text{purchase if } x_k < \alpha_k,$$

$$\text{do not purchase if } x_k > \alpha_k,$$

where the scalar α_k is the unique positive solution of the equation

$$x = E\bigl\{J_{k+1}(\lambda x + \xi_k)\bigr\}.$$

Note that the relation $J_k(x) \leq J_{k+1}(x)$ for all x and k implies that

$$\alpha_{k-1} \leq \alpha_k \leq \alpha_{k+1}, \qquad \text{for all } k,$$

and hence (as one would expect) the threshold price to purchase increases as the deadline gets closer.

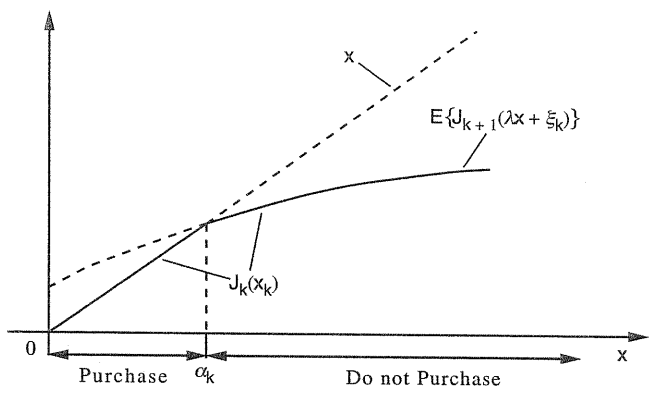


Figure 4.4.3 Determining the optimal policy when prices are correlated.

General Stopping Problems and the One-Step-Lookahead Rule

We now formulate a general type of N-stage problem where stopping is mandatory at or before stage N. Consider the stationary version of the basic problem of Chapter 1 (state, control, and disturbance spaces, disturbance distribution, control constraint set, and cost per stage are the same for all times). Assume that at each state x_k and at time k there is available, in addition to the controls $u_k \in U(x_k)$, a stopping action that forces the system to enter a termination state at a cost $t(x_k)$ and subsequently remain there at no cost. The terminal cost, assuming stopping has not occurred by the last stage, is $t(x_N)$. Thus, in effect, we assume that the termination cost will always be incurred either at the end of the horizon or earlier.

The DP algorithm is given by

$$J_N(x_N) = t(x_N), \tag{4.47}$$

$$J_k(x_k) = \min\left[t(x_k),\, \min_{u_k \in U(x_k)} E\Bigl\{g(x_k, u_k, w_k) + J_{k+1}\bigl(f(x_k, u_k, w_k)\bigr)\Bigr\}\right], \tag{4.48}$$

and it is optimal to stop at time k for states x in the set

$$T_k = \left\{x \;\middle|\; t(x) \leq \min_{u \in U(x)} E\Bigl\{g(x, u, w) + J_{k+1}\bigl(f(x, u, w)\bigr)\Bigr\}\right\}.$$

We have from Eqs. (4.47) and (4.48)

$$J_{N-1}(x) \leq J_N(x), \qquad \text{for all } x,$$

and using this fact in the DP equation (4.48), we obtain inductively

$$J_k(x) \leq J_{k+1}(x), \qquad \text{for all } x \text{ and } k.$$

[We are making use here of the stationarity of the problem and the monotonicity property of DP (Exercise 1.23 in Chapter 1).] Using this fact and the definition of T_k we see that

$$T_0 \subset \cdots \subset T_k \subset T_{k+1} \subset \cdots \subset T_{N-1}. \tag{4.49}$$

We will now consider a condition guaranteeing that all the stopping sets T_k are equal. Suppose that the set T_{N-1} is *absorbing* in the sense that if a state belongs to T_{N-1} and termination is not selected, the next state will also be in T_{N-1}:

$$f(x,u,w) \in T_{N-1}, \qquad \text{for all } x \in T_{N-1},\ u \in U(x),\ w. \tag{4.50}$$

We will show that equality holds in Eq. (4.49) and for all k we have

$$T_k = T_{N-1} = \left\{ x \in S \;\middle|\; t(x) \leq \min_{u \in U(x)} E\Big\{ g(x,u,w) + t\big(f(x,u,w)\big) \Big\} \right\}.$$

To see this, note that by the definition of T_{N-1}, we have

$$J_{N-1}(x) = t(x), \qquad \text{for all } x \in T_{N-1},$$

and using Eq. (4.50) we obtain for $x \in T_{N-1}$

$$\begin{aligned} \min_{u \in U(x)} E\Big\{ g(x,u,w) + J_{N-1}\big(f(x,u,w)\big) \Big\} &= \min_{u \in U(x)} E\Big\{ g(x,u,w) + t\big(f(x,u,w)\big) \Big\} \\ &\geq t(x). \end{aligned}$$

Therefore, stopping is optimal for all $x_{N-2} \in T_{N-1}$ or equivalently $T_{N-1} \subset T_{N-2}$. This together with Eq. (4.49) implies $T_{N-2} = T_{N-1}$. Proceeding similarly, we obtain $T_k = T_{N-1}$ for all k.

In conclusion, *if condition (4.50) holds (the one-step stopping set T_{N-1} is absorbing), then the stopping sets T_k are all equal to the set of states for which it is better to stop rather than continue for one more stage and then stop.* A policy of this type is known as a *one-step lookahead policy*. Such a policy turns out to be optimal in several types of applications. We provide next some examples. Additional examples are given in the exercises, and in Vol. II, Chapter 3.

Example 4.4.1 (Asset Selling with Past Offers Retained)

Consider the asset selling problem discussed earlier in this section with the difference that rejected offers can be accepted at a later time. Then if the asset is not sold at time k the state evolves according to

$$x_{k+1} = \max(x_k, w_k)$$

instead of $x_{k+1} = w_k$. The DP equations (4.43) and (4.44) then become

$$V_N(x_N) = x_N,$$

$$V_k(x_k) = \max\big[x_k, (1+r)^{-1} E\big\{V_{k+1}\big(\max(x_k, w_k)\big)\big\}\big].$$

The one-step stopping set is

$$T_{N-1} = \big\{ x \mid x \geq (1+r)^{-1} E\big\{\max(x,w)\big\} \big\}.$$

It is seen [compare with Eqs. (4.45) and (4.46)] that an alternative characterization is

$$T_{N-1} = \{x \mid x \geq \overline{\alpha}\}, \tag{4.51}$$

where $\overline{\alpha}$ is obtained from the equation

$$(1+r)\overline{\alpha} = P(\overline{\alpha})\overline{\alpha} + \int_{\overline{\alpha}}^{\infty} w dP(w).$$

Since past offers can be accepted at a later date, the effective offer available cannot decrease with time, and it follows that the one-step stopping set (4.51) is absorbing in the sense of Eq. (4.50). Therefore, the one-step lookahead stopping rule that accepts the first offer that equals or exceeds $\overline{\alpha}$ is optimal. Note that this policy is independent of the horizon length N.

Example 4.4.2 (The Rational Burglar [Whi82])

A burglar may at any night k choose to retire with his accumulated earnings x_k or enter a house and bring home an amount w_k. However, in the latter case he gets caught with probability p, and then he is forced to terminate his activities and forfeit his earnings thus far. The amounts w_k are independent, identically distributed with mean $\overline{w}$. The problem is to find a policy that maximizes the burglar's expected earnings over N nights.

We can formulate this problem as a stopping problem with two actions (retire or continue) and a state space consisting of the real line, the retirement state, and a special state corresponding to the burglar getting caught. The DP algorithm is given by

$$J_N(x_N) = x_N$$

$$J_k(x_k) = \max\Big[x_k,\ (1-p)E\big\{J_{k+1}(x_k + w_k)\big\}\Big].$$

The one-step stopping set is

$$T_{N-1} = \{x \mid x \geq (1-p)(x+\overline{w})\} = \left\{x \;\middle|\; x \geq \frac{(1-p)\overline{w}}{p}\right\},$$

(more accurately this set together with the special state corresponding to the burglar's arrest). Since this set is absorbing in the sense of Eq. (4.50), we see that the one-step lookahead policy by which the burglar retires when his earnings reach or exceed $(1-p)\overline{w}/p$ is optimal. The optimality of this policy for the corresponding infinite horizon problem will be demonstrated in Vol. II, Chapter 3.

4.5 SCHEDULING AND THE INTERCHANGE ARGUMENT

Suppose one has a collection of tasks to perform but the ordering of the tasks is subject to optimal choice. As examples, consider the ordering of operations in a construction project so as to minimize construction time or the scheduling of jobs in a workshop so as to minimize machine idle time. In such problems a useful technique is to start with some schedule and then to interchange two adjacent tasks and see what happens. We first provide some examples, and we then formalize mathematically the technique.

Example 4.5.1 (The Quiz Problem)

Consider a quiz contest where a person is given a list of N questions and can answer these questions in any order he chooses. Question i will be answered correctly with probability p_i, and the person will then receive a reward R_i. At the first incorrect answer, the quiz terminates and the person is allowed to keep his previous rewards. The problem is to choose the ordering of questions so as to maximize expected rewards.

Let i and j be the kth and $(k+1)$st questions in an optimally ordered list

$$L = (i_0, \ldots, i_{k-1}, i, j, i_{k+2}, \ldots, i_{N-1}).$$

Consider the list

$$L' = (i_0, \ldots, i_{k-1}, j, i, i_{k+2}, \ldots, i_{N-1})$$

obtained from L by interchanging the order of questions i and j. We compare the expected rewards of L and L'. We have

$$\begin{aligned} E\{\text{reward of } L\} = {} & E\{\text{reward of } \{i_0, \ldots, i_{k-1}\}\} \\ & + p_{i_0} \cdots p_{i_{k-1}} (p_i R_i + p_i p_j R_j) \\ & + p_{i_0} \cdots p_{i_{k-1}} p_i p_j E\{\text{reward of } \{i_{k+2}, \ldots, i_{N-1}\}\} \end{aligned}$$

$$\begin{aligned} E\{\text{reward of } L'\} = {} & E\{\text{reward of } \{i_0, \ldots, i_{k-1}\}\} \\ & + p_{i_0} \cdots p_{i_{k-1}} (p_j R_j + p_j p_i R_i) \\ & + p_{i_0} \cdots p_{i_{k-1}} p_j p_i E\{\text{reward of } \{i_{k+2}, \ldots, i_{N-1}\}\}. \end{aligned}$$

Since L is optimally ordered, we have

$$E\{\text{reward of } L\} \geq E\{\text{reward of } L'\},$$

so it follows from these equations that

$$p_i R_i + p_i p_j R_j \geq p_j R_j + p_j p_i R_i$$

or equivalently

$$\frac{p_i R_i}{1-p_i} \geq \frac{p_j R_j}{1-p_j}.$$

Therefore, to maximize expected rewards, questions should be answered in decreasing order of $p_i R_i/(1-p_i)$.

Example 4.5.2 (Job Scheduling on a Single Processor)

Suppose we have N jobs to process in sequential order with the ith job requiring a random time T_i for its execution. The times $T_1, \ldots, T_N$ are independent. If job i is completed at time t, the reward is $\alpha^t R_i$, where α is a discount factor with $0 < \alpha < 1$. The problem is to find a schedule that maximizes the total expected reward.

It can be seen that the state for this problem is just the collection of jobs yet to be processed. Indeed, because the execution times T_i are independent, and also because future costs are multiplicatively affected through discounting by the completion times of preceding jobs, the optimization of the scheduling of future jobs is unaffected by the completion times of preceding jobs. As a result these times need not be included in the state; this would not be so if either the times T_i were correlated or if the reward for completing job i at time t were not $\alpha^t R_i$ but instead had a general dependence on t. Now, given that the state is the collection of jobs yet to be processed, it is clear that an optimal policy can be mapped into an optimal job schedule $(i_0, \ldots, i_{N-1})$.

Suppose that $L = (i_0, \ldots, i_{k-1}, i, j, i_{k+2}, \ldots, i_{N-1})$ is an optimal job schedule, and consider the schedule $L' = (i_0, \ldots, i_{k-1}, j, i, i_{k+2}, \ldots, i_{N-1})$ obtained by interchanging i and j. Let t_k be the time of completion of job i_{k-1}. We compare the rewards of the schedules L and L', similar to the preceding example. Since the reward for completing the remaining jobs $i_{k+2}, \ldots, i_{N-1}$ is independent of the order in which jobs i and j are executed, we obtain

$$E\{\alpha^{t_k+T_i} R_i + \alpha^{t_k+T_i+T_j} R_j\} \geq E\{\alpha^{t_k+T_j} R_j + \alpha^{t_k+T_j+T_i} R_i\}.$$

Since t_k, T_i, and T_j are independent, this relation can be written as

$$\begin{aligned} & E\{\alpha^{t_k}\}\big(E\{\alpha^{T_i}\}R_i + E\{\alpha^{T_i}\}E\{\alpha^{T_j}\}R_j\big) \\ & \qquad \geq E\{\alpha^{t_k}\}\big(E\{\alpha^{T_j}\}R_j + E\{\alpha^{T_j}\}E\{\alpha^{T_i}\}R_i\big), \end{aligned}$$

from which we finally obtain

$$\frac{E\{\alpha^{T_i}\}R_i}{1-E\{\alpha^{T_i}\}} \geq \frac{E\{\alpha^{T_j}\}R_j}{1-E\{\alpha^{T_j}\}}.$$

It follows that scheduling jobs in order of decreasing $E\{\alpha^{T_i}\}R_i/\big(1-E\{\alpha^{T_i}\}\big)$ maximizes expected rewards. The structure of the optimal policy is identical with the one we derived for the preceding quiz contest example (identify $E\{\alpha^{T_i}\}$ with the probability p_i of answering correctly question i).

Example 4.5.3 (Job Scheduling on Two Processors in Series)

Consider the scheduling of N jobs on two processors A and B, such that B accepts the output of A as input. Job i requires known times a_i and b_i for processing in A and B, respectively. The problem is to find a schedule that minimizes the total processing time.

To formulate the problem into the form of the basic problem, we increment discrete time at the moments when processing of a job is completed at machine A and the next job is started. We take as state at time k the collection of jobs X_k that remain to be processed at A, together with the backlog of work τ_k at machine B, that is, the amount of time that the jobs currently at B need to clear B. Thus if (X_k, τ_k) is the state at stage k and job i is completed at machine A, the state changes to (X_{k+1}, τ_{k+1}) given by

$$X_{k+1} = X_k - \{i\}, \qquad \tau_{k+1} = b_i + \max(0, \tau_k - a_i).$$

The corresponding DP algorithm is

$$J_k(X_k, \tau_k) = \min_{i \in X_k} \Big[a_i + J_{k+1}\big(X_k - \{i\},\, b_i + \max(0, \tau_k - a_i)\big)\Big]$$

with the terminal condition

$$J_N(\emptyset, \tau_N) = \tau_N,$$

where $\emptyset$ is the empty set.

Since the problem is deterministic, there exists an optimal open-loop schedule

$$\{i_0, \ldots, i_{k-1}, i, j, i_{k+2}, \ldots, i_{N-1}\}.$$

By arguing that the cost of this schedule is no worse than the cost of the schedule

$$\{i_0, \ldots, i_{k-1}, j, i, i_{k+2}, \ldots, i_{N-1}\},$$

obtained by interchanging i and j, it can be verified that

$$J_{k+2}\big(X_k - \{i\} - \{j\}, \tau_{ij}\big) \leq J_{k+2}\big(X_k - \{i\} - \{j\}, \tau_{ji}\big), \tag{4.52}$$

where τ_{ij} and τ_{ji} are the backlogs at machine B at time $k+2$ when i is processed before j and j is processed before i, respectively, and the backlog at time k was τ_k. A straightforward calculation shows that

$$\tau_{ij} = b_i + b_j - a_i - a_j + \max(\tau_k, a_i, a_i + a_j - b_i), \tag{4.53}$$

$$\tau_{ji} = b_j + b_i - a_j - a_i + \max(\tau_k, a_j, a_j + a_i - b_j). \tag{4.54}$$

Clearly, J_{k+2} is monotonically increasing in τ, so from Eq. (4.52) we obtain

$$\tau_{ij} \leq \tau_{ji}.$$

In view of Eqs. (4.53) and (4.54), this relation implies two possibilities. The first is

$$\tau_k \geq \max(a_i, a_i + a_j - b_i),$$

$$\tau_k \geq \max(a_j, a_j + a_i - b_j),$$

in which case $\tau_{ij} = \tau_{ji}$ and the order of i and j makes no difference. (This is the case where the backlog at time k is so large that both jobs i and j will find B working on an earlier job.) The second possibility is that

$$\max(a_i, a_i + a_j - b_i) \leq \max(a_j, a_j + a_i - b_j),$$

which can be seen to be equivalent to

$$\min(a_i, b_j) \leq \min(a_j, b_i).$$

A schedule satisfying these necessary conditions for optimality can be constructed by the following procedure:

1. Find $\min_i \min(a_i, b_i)$.
2. If the minimizing value is an a take the corresponding job first; if it is a b, take the corresponding job last.
3. Repeat the procedure with the remaining jobs until a complete schedule is constructed.

To show that this schedule is indeed optimal, we start with an optimal schedule. We consider the job i_0 that minimizes $\min(a_i, b_i)$ and by successive interchanges we move it to the same position as in the schedule constructed previously. It is seen from the preceding analysis that the resulting schedule is still optimal. Similarly, continuing through successive interchanges and maintaining optimality throughout, we can transform the optimal schedule into the schedule constructed earlier. We leave the details to the reader.

The Interchange Argument

Let us now consider the basic problem of Chapter 1 and formalize the interchange argument used in the preceding examples. The main requirement is that the problem has structure such that *there exists an open-loop policy that is optimal*, that is, a sequence of controls that performs as well or better than any sequence of control functions. This is certainly true in deterministic problems as discussed in Chapter 1, but it is also true in some stochastic problems such as those of Examples 4.5.1 and 4.5.2.

To apply the interchange argument, we start with an optimal sequence

$$\{u_0, \ldots, u_{k-1}, \overline{u}, \tilde{u}, u_{k+2}, \ldots, u_{N-1}\}$$

and focus attention on the controls $\overline{u}$ and $\tilde{u}$ applied at times k and $k+1$, respectively. We then argue that if the order of $\overline{u}$ and $\tilde{u}$ is interchanged the expected cost cannot decrease. In particular, if X_k is the set of states that can occur with positive probability starting from the given initial state x_0 and using the control subsequence $\{u_0, \ldots, u_{k-1}\}$, we must have for all $x_k \in X_k$

$$\begin{aligned} &E\{g_k(x_k, \overline{u}, w_k) + g_{k+1}(\overline{x}_{k+1}, \tilde{u}, w_{k+1}) + J^*_{k+2}(\overline{x}_{k+2})\} \\ &\qquad \leq E\{g_k(x_k, \tilde{u}, w_k) + g_{k+1}(\tilde{x}_{k+1}, \overline{u}, w_{k+1}) + J^*_{k+2}(\tilde{x}_{k+2})\}, \end{aligned} \tag{4.55}$$

where $\overline{x}_{k+1}$ and $\overline{x}_{k+2}$ (or $\tilde{x}_{k+1}$ and $\tilde{x}_{k+2}$) are the states subsequent to x_k when $u_k = \overline{u}$ and $u_{k+1} = \tilde{u}$ (or $u_k = \tilde{u}$ and $u_{k+1} = \overline{u}$, respectively) are applied, and $J^*_{k+2}(\cdot)$ is the optimal cost-to-go function for time $k+2$.

Relation (4.55) is a *necessary* condition for optimality. It holds for every k and every optimal policy that is open-loop. There is no guarantee that this necessary condition is powerful enough to lead to an optimal solution, but it is worth considering in some specially structured problems. Generally in scheduling problems, algorithms that aim to improve a suboptimal schedule through a sequence of interchanges, may not provide an optimal solution, but are often the basis for successful heuristics.

4.6 SET-MEMBERSHIP DESCRIPTION OF UNCERTAINTY

In this section, we focus on problems where the uncertain quantities are described by their membership in given sets rather than probability distributions. This type of description is appropriate in minimax control problems, as discussed in Section 1.6. Our purpose in this section is to analyze some basic problems of estimation and control involving uncertainty with a set-membership description. These problems are conceptually important, and arise in several contexts, including the model predictive control methodology discussed in Section 6.5. However, their general solution can be computationally difficult. We will discuss some easily implementable approximations, involving linear systems and ellipsoidal descriptions.

4.6.1 Set-Membership Estimation

Suppose that we are given a linear dynamic system of the type considered in Section 4.1 but without a control ($u_k \equiv 0$):

$$x_{k+1} = A_k x_k + w_k, \qquad k = 0, 1, \ldots, N-1,$$

where $x_k \in \Re^n$ and $w_k \in \Re^n$ denote the state and disturbance vectors, respectively, and the matrices A_k are known. Suppose also that at each time k, we receive a measurement $z_k \in \Re^s$ of the form

$$z_k = C_k x_k + v_k,$$

where $v_k \in \Re^s$ is an (unknown) observation noise vector, and the matrix C_k is given.

An important and generic problem is to estimate the value of x_k, given the observations $z_1, \ldots, z_k$, accumulated up to time k. The uncertain quantities here are the initial state x_0, the system disturbances $w_0, \ldots, w_{N-1}$, and the observation noise vectors $v_1, \ldots, v_N$. When the joint probability distribution for these vectors is given, one may calculate the conditional distribution of x_k given $z_1, \ldots, z_k$, and from this, obtain estimates such as for example the conditional expectation $E\{x_k \mid z_1, \ldots, z_k\}$. This approach leads to a rich theory, centered around the Kalman filtering algorithm, which is described in detail in Appendix E.

Suppose now that, instead of a probability distribution, we have a set $\mathcal{R}$ within which the vector of unknown quantities

$$r = (x_0, w_0, \ldots, w_{N-1}, v_1, \ldots, v_N)$$

is known to belong. The state x_k can be expressed in terms of r using the system equation as

$$x_k = A_{k-1} \cdots A_0 x_0 + \sum_{i=0}^{k-1} A_{k-2} \ldots A_{i+1} w_i,$$

or more abstractly as

$$x_k = L_k r,$$

where L_k is an appropriate matrix. Thus, knowing that $r \in \mathcal{R}$ and before any measurements are received, the state x_k is known to belong to the set

$$\mathcal{X}_k = L_k \mathcal{R} = \{L_k r \mid r \in \mathcal{R}\}.$$

Each measurement z_i, when received, restricts the set of possible values of r to be such that $z_i = C_i L_i r + v_i$ or

$$z_i = E_i r$$

for an appropriate matrix E_i. Thus, with each new measurement, the set of possible vectors r is further restricted, and so is the set of possible states. In particular, given the measurements $z_1, \ldots, z_k$, the set of possible vectors r is given by

$$\mathcal{R}_k(z_1, \ldots, z_k) = \mathcal{R} \cap \{r \mid z_1 = E_1 r\} \cap \cdots \cap \{r \mid z_k = E_k r\}, \qquad (4.56)$$

and by a linear transformation, yields the set of possible states x_k as

$$\mathcal{X}_k(z_1, \ldots, z_k) = L_k \mathcal{R}_k(z_1, \ldots, z_k). \qquad (4.57)$$

The procedure just described is straightforward, and can easily be extended to systems and measurements that are nonlinear. The difficulty, however, is to specify conveniently the sets $\mathcal{R}_k(z_1, \ldots, z_k)$ and/or $\mathcal{X}_k(z_1, \ldots, z_k)$. There are only few special cases where these sets admit a simple description, e.g., one that involves a finite set of numbers. The most interesting of these are:

(a) The *polyhedral* case, where the set $\mathcal{R}$ is a polyhedron (a set specified by a finite number of linear inequalities). Then, it can be seen that the sets $\mathcal{R}_k(z_1, \ldots, z_k)$ and/or $\mathcal{X}_k(z_1, \ldots, z_k)$ are also polyhedra. The reason is that the intersection of a polyhedron with a linear manifold (a translated subspace) is a polyhedron, and a linear transformation of a polyhedron yields another polyhedron [cf. Eqs. (4.56) and (4.57)].

(b) The *ellipsoidal* case, where the set $\mathcal{R}$ is an ellipsoid (a linearly transformed sphere – a more specific description is given later). Then, it can be shown that the sets $\mathcal{R}_k(z_1, \ldots, z_k)$ and/or $\mathcal{X}_k(z_1, \ldots, z_k)$ are also ellipsoids. Similar to the polyhedral case, the reason is that the intersection of an ellipsoid with a linear manifold is an ellipsoid, and a linear transformation of this ellipsoid yields another ellipsoid.

The polyhedral case is interesting in some cases, but suffers from a quick explosion of the computational requirements to describe the associated polyhedra, as k increases. We will focus instead on the ellipsoidal case, and we will use DP methods to derive easily implementable algorithms that resemble the Kalman filtering algorithm described in Appendix E.

Energy Constraints

We first consider the most favorable case where the set of possible states $\mathcal{X}_k(z_1, \ldots, z_k)$ turns out to be an ellipsoid. Suppose that the vector of unknown quantities r is known to belong to a set of the form

$$\mathcal{R} = \left\{ r \;\middle|\; (x_0 - \hat{x}_0)' S^{-1} (x_0 - \hat{x}_0) + \sum_{i=0}^{N-1} \left(w_i' M_i^{-1} w_i + v_{i+1}' N_{i+1}^{-1} v_{i+1} \right) \leq 1 \right\},$$

where S, M_i, and N_i are positive definite symmetric matrices, and $\hat{x}_0$ is a given vector. This set is a bounded ellipsoid. Much of the analysis that follows carries through in the case of more general ellipsoids, but for simplicity, we restrict attention to bounded ellipsoids.

Let us use DP to derive the set of possible states and show that it is an ellipsoid of the form

$$\mathcal{X}_k(z_1, \ldots, z_k) = \left\{ x_k \mid (x_k - \hat{x}_k)' \Sigma_k^{-1} (x_k - \hat{x}_k) \leq 1 - \delta_k \right\},$$

where

Σ_k is a positive definite symmetric matrix that is independent of the observations $z_1, \ldots, z_k$,

$\hat{x}_k$ is a vector that depends on $z_1, \ldots, z_k$,

δ_k is a positive scalar that depends on $z_1, \ldots, z_k$.

We observe that a vector ξ belongs to $\mathcal{X}_k(z_1, \ldots, z_k)$ if and only if there exist x_0 and $w_0, \ldots, w_{k-1}$ such that

$$(x_0 - \hat{x}_0)' S^{-1} (x_0 - \hat{x}_0) + \sum_{i=0}^{k-1} w_i' M_i^{-1} w_i + \sum_{i=0}^{k-1} (z_{i+1} - C_{i+1} x_{i+1})' N_{i+1}^{-1} (z_{i+1} - C_{i+1} x_{i+1}) \leq 1$$

while ξ is equal to the vector x_k that is generated at the kth stage by the system

$$x_{i+1} = A_i x_i + w_i, \qquad i = 0, \ldots, k-1. \qquad (4.58)$$

Thus, we have $\xi \in \mathcal{X}_k(z_1, \ldots, z_k)$ if and only if $V_k(\xi) \leq 1$, where $V_k(\xi)$ is the optimal cost of the problem of minimizing the quadratic cost

$$(x_0 - \hat{x}_0)' S^{-1} (x_0 - \hat{x}_0) + \sum_{i=0}^{k-1} w_i' M_i^{-1} w_i + \sum_{i=0}^{k-1} (z_{i+1} - C_{i+1} x_{i+1})' N_{i+1}^{-1} (z_{i+1} - C_{i+1} x_{i+1})$$

subject to the system equation constraint (4.58) and the terminal condition $x_k = \xi$ (here the w_i are viewed as the controls/minimization variables). Thus

$$\mathcal{X}_k(z_1, \ldots, z_k) = \left\{ \xi \mid V_k(\xi) \leq 1 \right\}. \qquad (4.59)$$

As the analysis of Section 4.1 suggests, the function V_k is quadratic in ξ, and can be calculated by a DP recursion. This is because in the above problem the system is linear and the cost is quadratic. Thus the set of

possible states $\mathcal{X}_k(z_1,\ldots,z_k)$ of Eq. (4.59) is an ellipsoid. To calculate the matrix and center of this ellipsoid, we can use DP. Since here the terminal state x_k is specified to be equal to the given ξ, we should use a *forward* DP algorithm and view $V_k(\xi)$ as an *optimal cost to arrive at* ξ by optimal choice of x_0 and $w_0,\ldots,w_{k-1}$ in the system (4.58). Using the reasoning employed in Section 2.1, for $i=1,\ldots,k$, we have the forward recursion

$$\begin{aligned} V_i(x_i) &= \min_{\substack{w_{i-1},x_{i-1}\\ x_i=A_{i-1}x_{i-1}+w_{i-1}}} \big\{V_{i-1}(x_{i-1}) + w_{i-1}'M_{i-1}^{-1}w_{i-1} \\ &\qquad + (z_i - C_ix_i)'N_i^{-1}(z_i - C_ix_i)\big\} \\ &= \min_{x_{i-1}} \big\{V_{i-1}(x_{i-1}) + (x_i - A_{i-1}x_{i-1})'M_{i-1}^{-1}(x_i - A_{i-1}x_{i-1}) \\ &\qquad + (z_i - C_ix_i)'N_i^{-1}(z_i - C_ix_i)\big\} \end{aligned}$$

starting with the initial condition

$$V_0(x_0) = (x_0 - \hat{x}_0)'S^{-1}(x_0 - \hat{x}_0).$$

At the kth step of the recursion, we obtain the set of possible states $\mathcal{X}_k(z_1,\ldots,z_k)$ of Eq. (4.59).

Rather than provide the detailed derivation, we leave it for the reader to verify by induction the formula

$$V_k(x_k) = (x_k - \hat{x}_k)'\Sigma_k^{-1}(x_k - \hat{x}_k) + \delta_k,$$

where $\hat{x}_k$ and Σ_k are generated by the recursions

$$\hat{x}_k = A_{k-1}\hat{x}_{k-1} + \Sigma_k C_k'N_k^{-1}(z_k - C_kA_{k-1}\hat{x}_{k-1}), \tag{4.60}$$

$$\Sigma_k = \big(\hat{\Sigma}_k^{-1} + C_k'N_k^{-1}C_k\big)^{-1}, \tag{4.61}$$

$$\hat{\Sigma}_k = A_{k-1}\Sigma_{k-1}A_{k-1}' + M_{k-1}, \tag{4.62}$$

with the initial condition

$$\Sigma_0 = S,$$

and δ_k is given by

$$\delta_k = \sum_{i=1}^{k}(z_i - C_iA_{i-1}\hat{x}_{i-1})'(C_i\hat{\Sigma}_iC_i' + N_i)^{-1}(z_i - C_iA_{i-1}\hat{x}_{i-1}). \tag{4.63}$$

There are several variations of the estimation problem discussed above, for which we refer to the sources given at the end of the chapter.

Instantaneous Constraints

We now consider a different type of set description of the uncertainty. In particular, we assume that the initial state, the system disturbances, and the observation noise vectors are independently constrained to lie in ellipsoids. In other words, we know that

$$x_0'S^{-1}x_0 \le 1, \tag{4.64}$$

$$w_i'M_i^{-1}w_i \le 1, \qquad i=0,\ldots,N-1, \tag{4.65}$$

$$v_{i+1}'N_{i+1}^{-1}v_{i+1} \le 1, \qquad i=0,\ldots,N-1, \tag{4.66}$$

where S, M_i, and N_i are given symmetric positive definite matrices. Thus the vector

$$r = (x_0, w_0, \ldots, w_{N-1}, v_1, \ldots, v_N)$$

is known to belong to the set

$$\begin{aligned} \mathcal{R} = \big\{r \mid (x_0 - \hat{x}_0)'S^{-1}(x_0 - \hat{x}_0) \le 1,\ & w_i'M_i^{-1}w_i \le 1, \\ & v_{i+1}'N_{i+1}^{-1}v_{i+1} \le 1,\ i=0,\ldots,N-1\big\}. \end{aligned}$$

For this case, the set of possible states $\mathcal{X}_k(z_1,\ldots,z_k)$ is not an ellipsoid, but can be bounded by an ellipsoid, by bounding the set $\mathcal{R}$ with an ellipsoid $\overline{\mathcal{R}}$, and by bounding $\mathcal{X}_k(z_1,\ldots,z_k)$ with the ellipsoid $\overline{\mathcal{X}}_k(z_1,\ldots,z_k)$ that corresponds to $\overline{\mathcal{R}}$ as in the preceding case of energy constraints.

In particular, we observe that if $x_0, w_0, \ldots, w_{N-1}, v_1, \ldots, v_N$ satisfy the instantaneous constraints of Eqs. (4.64), (4.65), (4.66), then they also satisfy the energy constraint

$$\sigma(x_0 - \hat{x}_0)'S^{-1}(x_0 - \hat{x}_0) + \sum_{i=0}^{N-1}\big(\mu_i w_i'M_i^{-1}w_i + \nu_{i+1}v_{i+1}'N_{i+1}^{-1}v_{i+1}\big) \le 1, \tag{4.67}$$

where σ, μ_i, ν_{i+1} are any positive scalars satisfying

$$\sigma + \sum_{i=0}^{N-1}(\mu_i + \nu_{i+1}) = 1.$$

We thus replace the instantaneous constraints of Eqs. (4.64), (4.65), (4.66) with the energy constraint (4.67), and we obtain a bounding ellipsoid of the form

$$\overline{\mathcal{X}}_k(z_1,\ldots,z_k) = \big\{x_k \mid (x_k - \hat{x}_k)'\Sigma_k^{-1}(x_k - \hat{x}_k) \le 1 - \delta_k\big\},$$

where $\hat{x}_k$ and Σ_k are generated by the recursions given earlier for the energy constraint case, after we replace S with S/σ, M_i with M_i/μ_i, and N_i

with N_i/ν_i. The formulas obtained in this way are simplified if we write σ, μ_i, ν_{i+1} in the following form:

$$\sigma = (1-\beta_0)(1-\gamma_1)(1-\beta_1)(1-\gamma_2)\cdots(1-\beta_{k-1})(1-\gamma_k),$$
$$\mu_0 = \beta_0(1-\gamma_1)(1-\beta_1)(1-\gamma_2)\cdots(1-\beta_{k-1})(1-\gamma_k),$$
$$\nu_1 = \gamma_1(1-\beta_1)(1-\gamma_2)\cdots(1-\beta_{k-1})(1-\gamma_k),$$
$$\cdots$$
$$\mu_{k-1} = \beta_{k-1}(1-\gamma_k),$$
$$\nu_k = \gamma_k,$$

where β_{i-1}, γ_i, $i = 1, \ldots, k$ are any scalars with

$$0 < \beta_{i-1} < 1, \qquad 0 < \gamma_i < 1.$$

It is easy to see that for the scalars σ, μ_i, ν_{i+1} given by the above equations, we have $\sigma + \sum_{i=0}^{N-1}(\mu_i + \nu_{i+1}) = 1$.

Now, by writing the estimator equations (4.60)-(4.63), with S, M_i, and N_i replaced by S/σ, M_i/μ_i, and N_i/ν_i, respectively, we can obtain after straightforward manipulation a bounding ellipsoid of the form

$$\overline{\mathcal{X}}_k(z_1, \ldots, z_k) = \{x_k \mid (x_k - \hat{x}_k)'\Sigma_k^{-1}(x_k - \hat{x}_k) \leq 1 - \delta_k\},$$

where

$$\hat{x}_k = A_{k-1}\hat{x}_{k-1} + \gamma_k \Sigma_k C_k' N_k^{-1}(z_k - C_k A_{k-1}\hat{x}_{k-1}),$$
$$\Sigma_k = \left((1-\gamma_k)\hat{\Sigma}_k^{-1} + \gamma_k C_k' N_k^{-1} C_k\right)^{-1},$$
$$\hat{\Sigma}_k = (1-\beta_{k-1})^{-1} A_{k-1}\Sigma_{k-1}A_{k-1}' + \beta_{k-1}^{-1} M_{k-1},$$

with the initial condition

$$\Sigma_0 = S,$$

and δ_k is generated by the equation

$$\delta_k = (1-\beta_{k-1})(1-\gamma_k)\delta_{k-1} + (z_k - C_k A_{k-1}\hat{x}_{k-1})'$$
$$\left((1-\gamma_k)^{-1} C_k \hat{\Sigma}_i C_k' + \gamma_k^{-1} N_k\right)^{-1}(z_k - C_k A_{k-1}\hat{x}_{k-1}),$$

with the initial condition

$$\delta_0 = 0.$$

We omit the verification of the above equations because it is tedious, and we refer to the cited references. Note that the estimators for both cases of energy and instantaneous constraints bear close resemblance to the Kalman filtering algorithm described in Appendix E. An interesting problem variant is when the system equation has the form $x_{k+1} = x_k$. In this case, the problem is to use linear measurements to estimate the initial state x_0, which can be viewed as an unknown parameter vector. It can then be shown under mild assumptions that $\Sigma_k \to 0$ as $k \to \infty$, so that the parameter vector is identified with arbitrary accuracy as the number of measurements increases.

4.6.2 Control with Unknown-but-Bounded Disturbances

We now consider a problem of control when the uncertain quantities are described by their membership in given sets. We consider the system

$$x_{k+1} = f_k(x_k, u_k, w_k),$$

where as usual x_k is the state, u_k is the control to be selected from a set $U_k(x_k)$, and w_k is a disturbance. However, instead of probability distributions, we only know that w_k belongs to a given set $W_k(x_k, u_k)$, which may depend on the current state x_k and control u_k.

Often in control problems one is interested in keeping the state of the system close to a desired trajectory, in spite of the effects of the disturbances. We can formulate such a problem as one of finding a policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ with $\mu_k(x_k) \in U_k(x_k)$ for all x_k and k, such that for each $k = 1, 2, \ldots, N$, the state x_k of the closed-loop system

$$x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big)$$

belongs to a given set X_k, called the *target set at time k*.

We may view the set sequence $\{X_1, X_2, \ldots, X_N\}$ as a "tube" within which the state must stay, even under the worst possible choice of the disturbances w_k from within the corresponding sets $W_k\big(x_k, \mu_k(x_k)\big)$. Accordingly we refer to this problem as the problem of *reachability of a target tube*.

One may formulate this problem as a minimax control problem (cf. Section 1.6), where the cost at stage k is

$$g_k(x_k) = \begin{cases} 0 & \text{if } x_k \in X_k, \\ 1 & \text{if } x_k \notin X_k. \end{cases}$$

With this choice, the optimal cost-to-go from a given initial state x_0 is the minimum number of violations of the target tube constraints $x_k \in X_k$ that can occur when the w_k are optimally chosen, subject to the constraint $w_k \in W_k(x_k, u_k)$, by an adversary wishing to maximize the number of violations. In particular, if $J_k(x_k) = 0$ for some $x_k \in X_k$, there exists a policy such that starting from x_k, the subsequent system states x_i, $i = k+1, \ldots, N$, are guaranteed to be within the corresponding sets X_i.

It can be seen that the set

$$\overline{X}_k = \big\{x_k \mid J_k(x_k) = 0\big\}$$

is the set that we *must* reach at time k in order to be able to maintain the state within the subsequent target sets. Accordingly, we refer to $\overline{X}_k$ as the *effective target set at time k*. We can generate the sets $\overline{X}_k$ with a backwards recursion, which is derived from the DP algorithm for minimax problems

(see Section 1.6) but can also be easily justified from first principles. In particular, we start with

$$\overline{X}_N = X_N, \tag{4.68}$$

and for $k = 0, 1, \ldots, N-1$, we have

$$\overline{X}_k = \big\{x_k \in X_k \mid \text{there exists } u_k \in U_k(x_k) \text{ such that } f_k(x_k, u_k, w_k) \in \overline{X}_{k+1}, \text{ for all } w_k \in W_k(x_k, u_k)\big\}. \tag{4.69}$$

Example 4.6.1

Consider the scalar linear system

$$x_{k+1} = 2x_k + u_k + w_k,$$

and the target tube $\{X_1, X_2, \ldots, X_N\}$, where for all k,

$$X_k = [-1, 1].$$

We want to keep the state within this tube by using controls u_k that belong to the set $U_k = [-1, 1]$, and in spite of the effects of the disturbances W_k that can take any values in the set $[-1/2, 1/2]$.

Let us construct the effective target sets $\overline{X}_k$ by using the DP recursion of Eqs. (4.68) and (4.69). We have $\overline{X}_N = [-1, 1]$, and

$$\overline{X}_{N-1} = \big\{x \mid \text{for some } u \in [-1, 1] \text{ we have } -1 \le 2x + u + w \le 1 \text{ for all } w \in [-1/2, 1/2]\big\}.$$

We see that for x and u to satisfy $-1 \le 2x+u+w \le 1$ for all $w \in [-1/2, 1/2]$, it is necessary and sufficient that

$$-\frac{1}{2} \le 2x + u \le \frac{1}{2},$$

so u must be chosen (with knowledge of x) to satisfy

$$-1 \le u \le 1, \qquad -\frac{1}{2} - 2x \le u \le \frac{1}{2} - 2x.$$

For existence of such a u, the intervals $[-1, 1]$ and $[-1/2 - 2x, 1/2 - 2x]$ must have nonempty intersection, which can be seen to be true if and only if

$$-\frac{3}{4} \le x \le \frac{3}{4}.$$

Thus, to be able to reach the set X_N at time N, the state x_{N-1} must belong to the (effective target) set

$$\overline{X}_{N-1} = \left[-\frac{3}{4}, \frac{3}{4}\right].$$

We similarly proceed to construct $\overline{X}_{N-2}$. We have

$$\overline{X}_{N-2} = \big\{x \mid \text{for some } u \in [-1, 1] \text{ we have } -3/4 \le 2x + u + w \le 3/4 \text{ for all } w \in [-1/2, 1/2]\big\},$$

and we see that u must be chosen so that

$$-1 \le u \le 1, \qquad -\frac{1}{4} - 2x \le u \le \frac{1}{4} - 2x.$$

For existence of such a u, the intervals $[-1, 1]$ and $[-1/4 - 2x, 1/4 - 2x]$ must have nonempty intersection, which can be seen to be true if and only if

$$-\frac{5}{8} \le x \le \frac{5}{8}.$$

Thus, to be able to reach the effective target set $\overline{X}_{N-1}$ at time $N-1$, the state x_{N-2} must belong to the set

$$\overline{X}_{N-2} = \left[-\frac{5}{8}, \frac{5}{8}\right].$$

The above calculations illustrate the form of the algorithm that yields the effective target set $\overline{X}_k$ for every k. We have

$$\overline{X}_k = [-\alpha_k, \alpha_k],$$

where the scalars α_k satisfy the recursion

$$\alpha_k = \frac{\alpha_{k+1}}{2} + \frac{1}{4}, \qquad k = 0, 1, \ldots, N-1,$$

with the starting condition

$$\alpha_N = 1.$$

In order to guarantee reachability of the given target tube, the initial state x_0 should belong to the interval $[-\alpha_0, \alpha_0]$. Note that the scalars α_k are monotonically decreasing as $k \to -\infty$, and we have $\alpha_k \to 1/2$. Thus, if the initial state x_0 is in the interval $[-1/2, 1/2]$, then given any horizon length N, there is a policy that keeps the state of the system within the set $[-1, 1]$. In fact, it can be seen that the *linear stationary* policy $\{\mu, \mu, \ldots\}$, where

$$\mu(x) = -2x,$$

keeps the state of the system in the interval $[-1/2, 1/2]$, provides the initial state belongs to that interval. It can also be seen that if the initial state does not belong to the interval $[-1/2, 1/2]$, then there is a large enough horizon length N, such that for every admissible policy, a sequence of feasible disturbances exists that will force the state to be outside the target set $[-1, 1]$ at some time $k \le N$. These observations can be generalized for the case of linear

systems and ellipsoidal constraint sets (see the discussion and the references given below).

In general, it is not easy to characterize the effective target sets $\overline{X}_k$. However, similar to the estimation problem of the preceding subsection, a few special cases involving the linear system

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

where A_k and B_k are given matrices, are amenable to exact or approximate computational solution. One such case is when the sets X_k are ellipsoids, and the sets $U_k(x_k)$ and $W_k(x_k, u_k)$ are also ellipsoids that do not depend on x_k and (x_k, u_k), respectively. In this case, the effective target sets $\overline{X}_k$ are not ellipsoids, but can be approximated by inner ellipsoids $\tilde{X}_k \subset \overline{X}_k$ (this requires that the ellipsoids U_k have sufficiently large size, for otherwise the target tube may not be reachable and the problem may not have a solution). Furthermore, the state trajectory $\{x_1, x_2, \ldots, x_N\}$ can be maintained within the ellipsoidal tube

$$\{\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_N\}$$

by using a *linear control law* (compare with the preceding example). We outline the main algorithm in Exercise 4.31, and refer to the author's thesis work, [Ber71], [BeR71b], for a more detailed analysis.

Another case of interest is when the sets X_k are polyhedral, and the sets $U_k(x_k)$ and $W_k(x_k, u_k)$ are also polyhedral, and independent of x_k and u_k. Then the effective target sets are polyhedral and can be computed by linear programming methods.

An important special case is when the problem is stationary and f_k, X_k, U_k, and W_k do not depend on k. Then it can be shown that the effective target sets satisfy

$$\overline{X}_k \subset \overline{X}_{k+1}, \qquad \text{for all } k.$$

The intersection $\cap_{k=0}^{N} \overline{X}_k$ depends on the size of the horizon N and "decreases" to the set

$$X_\infty = \cap_{N=1}^{\infty} \cap_{k=0}^{N} \overline{X}_k$$

as N increases to ∞. We may view X_∞ as the set within which the state can be kept for an arbitrarily large (but finite) number of time periods. Paradoxically, under some unusual circumstances, there may be states in X_∞ starting from which it may be impossible to remain within X_∞ for an indefinitely long horizon, i.e., an infinite number of time periods. Conditions that preclude this possibility have been investigated by the author in [Ber72a]. References [Ber71] and [Ber72a] contain a methodology for constructing ellipsoidal inner approximations to X_∞ and an associated linear control law for the case where the system is linear, and the sets X_k, U_k, and W_k are ellipsoids.

4.7 NOTES, SOURCES, AND EXERCISES

The certainty equivalence principle for dynamic linear-quadratic problems was first discussed by Simon [Sim56]. His work was preceded by Theil [The54], who considered a single-period case, and Holt, Modigliani, and Simon [HMS55], who considered a deterministic case. Similar problems were independently considered by Kalman and Koepcke [KaK58], Joseph and Tou [JoT61], and Gunckel and Franklin [GuF63]. The linear-quadratic problem is central in control theory; see the special issue [IEE71], which contains hundreds of references.

The literature on inventory control stimulated by the pioneering paper of Arrow et al. [AHM51] is also voluminous. An important work summarizing most of the research up to 1958 is Arrow, Karlin, and Scarf [AKS58]. Veinott [Vei66] also surveys the early work on the subject. The proof of optimality of (s, S) policies in the case of nonzero fixed costs is due to Scarf [Sca60].

Most of the material in Section 4.3 is taken from Mossin [Mos68]; see also Hakansson [Hak70], [Hak71], and Samuelson [Sam69]. Many applications of DP in economics are described in Sargent [Sar87], and Stokey and Lucas [StL89].

The material of Section 4.4 is largely drawn from White [Whi69]. Example 4.5.1 is given by Ross [Ros70], Example 4.5.2 is given by Ross [Ros83], and Example 4.5.3 is due to Weiss and Pinedo [WeP80]. An extensive reference on scheduling is Pinedo [Pin95].

The problem of state estimation with a set-membership description of the uncertainty was first formulated and addressed by Witsenhausen in his Ph.D. work at MIT [Wit66], and also in the paper [Wit68]. The material given here follows closely the author's Ph.D. thesis [Ber71], where the problem of estimation for the case of an energy constraint was first formulated and solved. The estimator given in Section 4.6.1 for the case of instantaneous constraints was also first derived in the author's thesis using the method given here, and a steady-state analysis was also given. A state estimator using ellipsoidal approximations for the case of instantaneous constraints was first proposed by Schweppe [Sch68], [Sch74]. This estimator, however, has several drawbacks relative to the estimator given here. In particular, the associated matrix Σ_k depends on the observations $z_1, \ldots, z_k$, and need not converge to a steady state as $k \to \infty$.

Continuous-time versions of the estimators of Section 4.6, as well and other variants of the estimation problem (the prediction and smoothing problems) with a set-membership description of the uncertainty were first given by Bertsekas and Rhodes [BeR71a]. Kurzhanski and Valyi [KuV97] provide an account of set-membership estimation. Deller [Del89] surveys applications in signal processing. Kosut, Lau, and Boyd [KLB92] discuss applications in system identification. The state estimation problem can also be addressed as a minimax problem that involves in part a probabilis-

tic description of the uncertainty. Basar [Bas91] describes the relations between this approach and the set-membership approach.

The target tube reachability problem was first formulated by the author in his Ph.D. thesis [Ber71]; see also the papers [BeR71b] and [Ber72a], and Exercises 3.23 and 3.24 of Vol. II. The associated recursion of Eqs. (4.68)-(4.69) for the effective target sets, and methods for approximating this recursion were also given in these references (see Exercise 4.31). The target tube reachability problem arises within several contexts in control system design, including model predictive control, which is described in Section 6.5 (for a recent discussion, see the paper by Mayne [May01]). For a survey of the associated issues, including extensions to continuous-time systems and additional references, see Blanchini [Bla99].

There has been considerable recent research on minimax formulations of general optimization problems under uncertainty, such as linear programming problems. This approach, which is known as *robust optimization*, is also based on a set-membership description of the uncertainty; for some representative works, see Ben-Tal and Nemirovski [BeN98], [BeN01], and Bertsimas and Sim [BeS03].

EXERCISES

4.1 (Linear-Quadratic Problems with Forecasts)

Consider the linear-quadratic problem first examined in Section 4.1 (A_k, B_k: known) for the case where at the beginning of period k we have a forecast $y_k \in \{1, 2, \ldots, n\}$ consisting of an accurate prediction that w_k will be selected in accordance with a particular probability distribution $P_{k|y_k}$ (cf. Section 1.4). The vectors w_k need not have zero mean under the distribution $P_{k|y_k}$. Show that the optimal control law is of the form

$$\mu_k(x_k, y_k) = -(B_k' K_{k+1} B_k + R_k)^{-1} B_k' K_{k+1} \big(A_k x_k + E\{w_k \mid y_k\}\big) + \alpha_k,$$

where the matrices K_k are given by the Riccati equation (4.3) and (4.4) and α_k are appropriate vectors.

4.2

Consider a scalar linear system

$$x_{k+1} = a_k x_k + b_k u_k + w_k, \qquad k = 0, 1, \ldots, N-1,$$

where $a_k, b_k \in R$, and each w_k is a Gaussian random variable with zero mean and variance σ^2. We assume no control constraints and independent disturbances. Show that the control law $\{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ that minimizes the cost function

$$E\left\{\exp\left[x_N^2 + \sum_{k=0}^{N-1}\big(x_k^2 + r u_k^2\big)\right]\right\}, \qquad r > 0,$$

is linear in the state variable, assuming that the optimal cost is finite for every x_0. Show by example that the Gaussian assumption is essential for the result to hold. (For analyses of multidimensional versions of this exercise, see Jacobson [Jac73], Whittle [Whi82], [Whi90], and Basar [Bas00].)

4.3

Consider an inventory problem similar to the problem of Section 4.2 (zero fixed cost). The only difference is that at the beginning of each period k the decision maker, in addition to knowing the current inventory level x_k, receives an accurate forecast that the demand w_k will be selected in accordance with one out of two possible probability distributions P_ℓ, P_s (large demand, small demand). The a priori probability of a large demand forecast is known (cf. Section 1.4).

(a) Obtain the optimal ordering policy for the case of a single-period problem.

(b) Extend the result to the N-period case.

(c) Extend the result to the case of any finite number of possible distributions.

4.4

Consider the inventory problem of Section 4.2 (zero fixed cost), where the purchase costs c_k, $k = 0, 1, \ldots, N-1$, are not initially known, but instead they are independent random variables with a priori known probability distributions. The exact value of the cost c_k, however, becomes known to the decision maker at the beginning of the kth period, so that the inventory purchasing decision at time k is made with exact knowledge of the cost c_k. Characterize the optimal ordering policy assuming that p is greater than all possible values of c_k.

4.5

Consider the inventory problem of Section 4.2 for the case where the cost has the general form

$$E\left\{\sum_{k=0}^{N} r_k(x_k)\right\}.$$

The functions r_k are convex and differentiable and

$$\lim_{x \to -\infty} \frac{dr_k(x)}{dx} = -\infty, \qquad \lim_{x \to \infty} \frac{dr_k(x)}{dx} = \infty, \qquad k = 0, \ldots, N.$$

(a) Assume that the fixed cost is zero. Write the DP algorithm for this problem and show that the optimal ordering policy has the same form as the one derived in Section 4.2.

(b) Suppose there is a one-period time lag between the order and the delivery of inventory; that is, the system equation is of the form

$$x_{k+1} = x_k + u_{k-1} - w_k, \qquad k = 0, 1, \ldots, N-1,$$

where u_{-1} is given. Reformulate the problem so that it has the form of the problem of part (a). *Hint*: Make a change of variables $y_k = x_k + u_{k-1}$.

4.6 (Inventory Control for Nonzero Fixed Cost)

Consider the inventory problem of Section 4.2 (nonzero fixed cost) under the assumption that unfilled demand at each stage is not backlogged but rather is lost; that is, the system equation is $x_{k+1} = \max(0, x_k + u_k - w_k)$ instead of $x_{k+1} = x_k + u_k - w_k$. Complete the details of the following argument, which shows that a multiperiod (s, S) policy is optimal.

Abbreviated Proof: (due to S. Shreve) Let $J_N(x) = 0$ and for all k

$$G_k(y) = cy + E\Big\{h\max(0, y-w_k) + p\max(0, w_k - y) + J_{k+1}\big(\max(0, y-w_k)\big)\Big\},$$

$$J_k(x) = -cx + \min_{u\geq 0}\big[K\delta(u) + G_k(x+u)\big],$$

where $\delta(0) = 0$, $\delta(u) = 1$ for $u > 0$. The result will follow if we can show that G_k is K-convex, continuous, and $G_k(y) \to \infty$ as $|y| \to \infty$. The difficult part is to show K-convexity, since K-convexity of G_{k-1} does not imply K-convexity of $E\{J_{k+1}(\max(0, y-w))\}$. It will be sufficient to show that K-convexity of G_{k+1} implies K-convexity of

$$H(y) = p\max(0, -y) + J_{k+1}\big(\max(0, y)\big), \tag{4.70}$$

or equivalently that

$$K + H(y+z) \geq H(y) + z\left(\frac{H(y) - H(y-b)}{b}\right), \quad z \geq 0,\ b > 0,\ y \in \Re. \tag{4.71}$$

By K-convexity of G_{k+1} we have for appropriate scalars s_{k+1} and S_{k+1} such that $G_{k+1}(S_{k+1}) = \min_y G_{k+1}(y)$ and $K + G_{k+1}(S_{k+1}) = G_{k+1}(s_{k+1})$:

$$J_{k+1}(x) = \begin{cases} K + G_{k+1}(S_{k+1}) - cx & \text{if } x < s_{k+1}, \\ G_{k+1}(x) - cx & \text{if } x \geq s_{k+1}, \end{cases} \tag{4.72}$$

and J_{k+1} is K-convex by the theory of Section 4.2.

Case 1: $0 \leq y - b < y \leq y + z$. In this region, Eq. (4.71) follows from K-convexity of J_{k+1}.

Case 2: $y - b < y \leq y + z \leq 0$. In this region, H is linear and hence K-convex.

Case 3: $y - b < y \leq 0 \leq y + z$. In this region, Eq. (4.71) may be written [in view of Eq. (6.1)] as

$$K + J_{k+1}(y+z) \geq J_{k+1}(0) - p(y+z).$$

We will show that

$$K + J_{k+1}(z) \geq J_{k+1}(0) = pz, \qquad z \geq 0. \tag{4.73}$$

If $0 < s_{k+1} \leq z$, then using Eq. (4.72) and the fact $p > c$, we have

$$K + J_{k+1}(z) = K - cz + G_{k+1}(z) \geq K - pz + G_{k+1}(S_{k+1}) = J_{k+1}(0) = pz.$$

If $0 \leq z \leq s_{k+1}$, then using Eq. (4.72) and the fact $p > c$, we have

$$K + J_{k+1}(z) = 2K - cz + G_{k+1}(S_{k+1}) \geq K - pz + G_{k+1}(S_{k+1}) = J_{k+1}(0) - pz.$$

If $s_{k+1} \leq 0 \leq z$, then using Eq. (4.72), the fact $p > c$, and part (iv) of the lemma in Section 4.2, we have

$$K + J_{k+1}(z) = K - cz + G_{k+1}(z) \geq G_{k+1}(0) - pz = J_{k+1}(0) - pz.$$

Thus Eq. (4.73) is proved and Eq. (4.71) follows for the case under consideration.

Case 4: $y - b < 0 < y \leq y + z$. Then $0 < y < b$. If

$$\frac{H(y) - H(0)}{y} \geq \frac{H(y) - H(y-b)}{b}, \tag{4.74}$$

then since H agrees with J_{k+1} on $[0, \infty)$ and J_{k+1} is K-convex,

$$\begin{aligned} K + H(y+z) &\geq H(y) + z\left(\frac{H(y) - H(0)}{y}\right) \\ &\geq H(y) + z\left(\frac{H(y) - H(y-b)}{b}\right), \end{aligned}$$

where the last step follows from Eq. (4.74). If

$$\frac{H(y) - H(0)}{y} < \frac{H(y) - H(y-b)}{b},$$

then we have

$$H(y) - H(0) < \frac{y}{b}\big(H(y) - H(y-b)\big) = \frac{y}{b}\big(H(y) - H(0) + p(y-b)\big).$$

It follows that

$$\left(1 - \frac{y}{b}\right)\big(H(y) - H(0)\big) < \left(\frac{y}{b}\right)p(y-b) = -py\left(1 - \frac{y}{b}\right),$$

and since $b > y$,

$$H(y) - H(0) < -py. \tag{4.75}$$

Now we have, using the definition of H, Eqs. (4.73) and (4.75),

$$\begin{aligned} H(y) + z\frac{H(y) - H(y-b)}{b} &= H(y) + z\left(\frac{H(0) - py - H(0) + p(y-b)}{b}\right) \\ &= H(y) - pz \\ &< H(0) - p(y+z) \\ &\leq K + H(y+z). \end{aligned}$$

Hence Eq. (4.73) is proved for this case as well. **Q.E.D.**

4.7

Consider the inventory problem of Section 4.2 (zero fixed cost) with the difference that successive demands are correlated and satisfy a relation of the form

$$w_k = e_k - \gamma e_{k-1}, \qquad k = 0, 1, \ldots,$$

where γ is a given scalar, e_k are independent random variables, and e_{-1} is given.

(a) Show that this problem can be converted into an inventory problem with independent demands. *Hint*: Given $w_0, w_1, \ldots, w_{k-1}$, we can determine e_{k-1} in view of the relation

$$e_{k-1} = \gamma^k e_{-1} + \sum_{i=0}^{k-1} \gamma^i w_{k-1-i}.$$

Define $z_k = x_k + \gamma e_{k-1}$ as a new state variable.

(b) Show that the same is true when in addition there is a one-period delay in the delivery of inventory (cf. Exercise 4.5).

4.8

Consider the inventory problem of Section 4.2 (zero fixed cost), the only difference being that there is an upper bound $\overline{b}$ and a lower bound $\underline{b}$ to the allowable values of the stock x_k. This imposes the additional constraint on u_k

$$\underline{b} + d \leq u_k + x_k \leq \overline{b},$$

where $d > 0$ is the maximum value that the demand w_k can take (we assume $\underline{b} + d < \overline{b}$). Show that an optimal policy $\{\mu_0^*, \ldots, \mu_{N-1}^*\}$ is of the form

$$\mu_k^*(x_k) = \begin{cases} S_k - x_k & \text{if } x_k < S_k, \\ 0 & \text{if } x_k \geq S_k, \end{cases}$$

where $S_0, S_1, \ldots, S_{N-1}$ are some scalars.

4.9

Consider the inventory problem of Section 4.2 (nonzero fixed cost) with the difference that demand is deterministic and must be met at each time period (i.e., the shortage cost per unit is ∞). Show that it is optimal to order a positive amount at period k if and only if the stock x_k is insufficient to meet the demand w_k. Furthermore, when a positive amount is ordered, it should bring up stock to a level that will satisfy demand for an integral number of periods.

4.10 [Vei65], [Tsi84b] (www)

Consider the inventory control problem of Section 4.2 (zero fixed cost) with the only difference that the orders u_k are constrained to be nonnegative *integers*. Let J_k be the optimal cost-to-go function. Show that:

(a) J_k is continuous.

(b) $J_k(x+1) - J_k(x)$ is a nondecreasing function of x.

(c) There exists a sequence $\{S_k\}$ of numbers such that the policy given by

$$\mu_k(x_k) = \begin{cases} n & \text{if } S_k - n \leq x_k < S_k - n + 1, \quad n = 1, 2, \ldots, \\ 0 & \text{if } x_k \geq S_k \end{cases}$$

is optimal.

4.11 (Capacity Expansion Problem)

Consider a problem of expanding over N time periods the capacity of a production facility. Let us denote by x_k the production capacity at the beginning of the kth period and by $u_k \geq 0$ the addition to capacity during the kth period. Thus capacity evolves according to

$$x_{k+1} = x_k + u_k, \qquad k = 0, 1, \ldots, N-1.$$

The demand at the kth period is denoted w_k and has a known probability distribution that does not depend on either x_k or u_k. Also, successive demands are assumed to be independent and bounded. We denote:

$C_k(u_k)$: expansion cost associated with adding capacity u_k,

$P_k(x_k + u_k - w_k)$: penalty associated with capacity $x_k + u_k$ and demand w_k,

$S(x_N)$: salvage value of final capacity x_N.

Thus the cost function has the form

$$\mathop{E}_{\substack{w_k \\ k=0,1,\ldots,N-1}} \left\{ -S(x_N) + \sum_{k=0}^{N-1} \big(C_k(u_k) + P_k(x_k + u_k - w_k)\big) \right\}.$$

(a) Derive the DP algorithm for this problem.

(b) Assume that S is a concave function with $\lim_{x\to\infty} dS(x)/dx = 0$, P_k are convex functions, and the expansion cost C_k is of the form

$$C_k(u) = \begin{cases} K + c_k u & \text{if } u > 0, \\ 0 & \text{if } u = 0, \end{cases}$$

where $K \geq 0$, $c_k > 0$ for all k. Show that the optimal policy is of the (s, S) type assuming $c_k y + E\{P_k(y - w_k)\} \to \infty$ as $|y| \to \infty$.

4.12

We want to use a machine to produce a certain item in quantities that meet as closely as possible a known (nonrandom) sequence of demands d_k over N periods. The machine can be in one of two states: good (G) or bad (B). The state of the machine is perfectly observed and evolves from one period to the next according to

$$P(G \mid G) = \lambda_G, \quad P(B \mid G) = 1 - \lambda_G, \quad P(B \mid B) = \lambda_B, \quad P(G \mid B) = 1 - \lambda_B,$$

where λ_G and λ_B are given probabilities. Let x_k be the stock at the beginning of the kth period. If the machine is in good state at period k, it can produce u_k, where $u_k \in [0, \overline{u}]$, and the stock evolves according to

$$x_{k+1} = x_k + u_k - d_k;$$

otherwise the stock evolves according to

$$x_{k+1} = x_k - d_k.$$

There is a cost $g(x_k)$ for having stock x_k in period k, and the terminal cost is also $g(x_N)$. We assume that the cost per stage g is a convex function such that $g(x) \to \infty$ as $|x| \to \infty$. The objective is to find a production policy that minimizes the total expected cost.

(a) Prove inductively a convexity property of the cost-to-go functions, and show that for each k there is a target stock level S_{k+1} such that if the machine is in the good state, it is optimal to produce $u_k^* \in [0, \overline{u}]$ that will bring x_{k+1} as close as possible to S_{k+1}.

(b) Generalize part (a) for the case where each demand d_k is random and takes values in an interval $[0, \overline{d}]$ with given probability distribution. The stock and the state of the machine are still perfectly observable.

4.13 (A Gambling Problem)

A gambler enters a game whereby he may at time k stake any amount $u_k \geq 0$ that does not exceed his current fortune x_k (defined to be his initial capital plus his gain or minus his loss thus far). He wins his stake back and as much more with probability p, where $\frac{1}{2} < p < 1$, and he loses his stake with probability $(1 - p)$. Show that the gambling strategy that maximizes $E\{\ln x_N\}$, where x_N denotes his fortune after N plays, is to stake at each time k an amount $u_k = (2p - 1)x_k$. *Hint*: The problem is related to the portfolio problem of Section 4.3.

4.14

Consider the dynamic portfolio problem of Section 4.3 for the case where at each period k there is a forecast that the rates of return of the risky assets for that period will be selected in accordance with a particular probability distribution as in Section 1.4. Show that a partially myopic policy is optimal.

4.15

Consider a problem involving the linear system

$$x_{k+1} = A_k x_k + B_k u_k, \qquad k = 0, 1, \ldots, N - 1,$$

where the $n \times n$ matrices A_k are given, and the $n \times m$ matrices B_k are random and independent with given probability distributions that do not depend on x_k, u_k. The problem is to find a policy that maximizes $E\{U(c'x_N)\}$, where c is a given n-dimensional vector. We assume that U is a concave twice continuously differentiable utility function satisfying for all y

$$-\frac{U'(y)}{U''(y)} = a + by,$$

and that the control is unconstrained. Show that the optimal policy consists of linear functions of the current state. *Hint*: Reduce the problem to a one-dimensional problem and use the results of Section 4.3.

4.16

Suppose that a person wants to sell a house and an offer comes at the beginning of each day. We assume that successive offers are independent and an offer is w_j with probability p_j, $j = 1, \ldots, n$, where w_j are given nonnegative scalars. Any offer not immediately accepted is not lost but may be accepted at any later date. Also, a maintenance cost c is incurred for each day that the house remains unsold. The objective is to maximize the price at which the house is sold minus the maintenance costs. Consider the problem when there is a deadline to sell the house within N days and characterize the optimal policy.

4.17

Assume that we have x_0 items of a certain type that we want to sell over a period of N days. At each day we may sell at most one item. At the kth day, knowing the current number x_k of remaining unsold items, we can set the selling price u_k of a unit item to a nonnegative number of our choice; then, the probability $\lambda_k(u_k)$ of selling an item on the kth day depends on u_k as follows:

$$\lambda_k(u_k) = \alpha e^{-u_k},$$

where α is a given scalar with $0 < \alpha \leq 1$. The objective is to find the optimal price setting policy so as to maximize the total expected revenue over N days.

(a) Assuming that, for all k, the cost-to-go function $J_k(x_k)$ is monotonically nondecreasing as a function of x_k, prove that for $x_k > 0$, the optimal prices have the form

$$\mu_k^*(x_k) = 1 + J_{k+1}(x_k) - J_{k+1}(x_k - 1),$$

and that

$$J_k(x_k) = \alpha e^{-\mu_k^*(x_k)} + J_{k+1}(x_k).$$

(b) Prove simultaneously by induction that, for all k, the cost-to-go function $J_k(x_k)$ is indeed monotonically nondecreasing as a function of x_k, that the optimal price $\mu_k^*(x_k)$ is monotonically nonincreasing as a function of x_k, and that $J_k(x_k)$ is given in closed form by

$$J_k(x_k) = \begin{cases} (N-k)\alpha e^{-1} & \text{if } x_k \geq N-k, \\ \sum_{i=k}^{N-x_k} \alpha e^{-\mu_i^*(x_k)} + x_k \alpha e^{-1} & \text{if } 0 < x_k < N-k, \\ 0 & \text{if } x_k = 0. \end{cases}$$

4.18 (Optimal Termination of Sampling) WWW

This is a classical problem, which when appropriately paraphrased, is known as the job selection, or as the secretary selection, or as the spouse selection problem. A collection of $N \geq 2$ objects is observed randomly and sequentially one at a time. The observer may either select the current object observed, in which case the selection process is terminated, or reject the object and proceed to observe the next. The observer can rank each object relative to those already observed, and the objective is to maximize the probability of selecting the "best" object according to some criterion. It is assumed that no two objects can be judged to be equal. Let r^* be the smallest positive integer r such that

$$\frac{1}{N-1} + \frac{1}{N-2} + \cdots + \frac{1}{r} \leq 1.$$

Show that an optimal policy requires that the first r^* objects be observed. If the r^*th object has rank 1 relative to the others already observed, it should be selected; otherwise, the observation process should be continued until an object of rank 1 relative to those already observed is found. *Hint*: We assume that, if the rth object has rank 1 relative to the previous $(r-1)$ objects, then the probability that it is best is r/N. For $k \geq r^*$, let $J_k(0)$ be the maximal probability of finding the best object assuming k objects have been selected and the kth object is not best relative to the previous $(k-1)$ objects. Show that

$$J_k(0) = \frac{k}{N}\left(\frac{1}{N-1} + \cdots + \frac{1}{k}\right).$$

4.19

A driver is looking for parking on the way to his destination. Each parking place is free with probability p independently of whether other parking places are free or not. The driver cannot observe whether a parking place is free until he reaches it. If he parks k places from his destination, he incurs a cost k. If he reaches the destination without having parked the cost is C.

(a) Let F_k be the minimal expected cost if he is k parking places from his destination, where $F_0 = C$. Show that

$$F_k = p\min(k, F_{k-1}) + qF_{k-1}, \qquad k = 1, 2, \ldots,$$

where $q = 1 - p$.

(b) Show that an optimal policy is of the form: never park if $k \geq k^*$, but take the first free place if $k < k^*$, where k is the number of parking places from the destination and k^* is the smallest integer i satisfying $q^{i-1} < (pC+q)^{-1}$.

4.20 [Whi82]

A person may go hunting for a certain type of animal on a given day or stay home. When the animal population is x, the probability of capturing one animal is $p(x)$, a known increasing function, and the probability of capturing more than one is zero. A captured animal is worth one unit and a day of hunting costs c units. Assume that x does not change due to deaths or births, that the hunter knows x at all times, that the horizon is finite, and that the terminal reward is zero. Show that it is optimal to hunt only when $p(x) \geq c$.

4.21

Consider the scalar linear system $x_{k+1} = ax_k + bu_k$, where a and b are known. At each period k we have the option of using a control u_k and incurring a cost $qx_k^2 + ru_k^2$, or else stopping and incurring a stopping cost tx_k^2. If we have not stopped by period N, the terminal cost is the stopping cost tx_N^2. We assume that $q \geq 0$, $r > 0$, $t > 0$. Show that there is a threshold value for t below which immediate stopping is optimal at every initial state, and above which continuing at every state x_k and period k is optimal.

4.22

Consider a situation involving a blackmailer and his victim. In each period the blackmailer has a choice of: a) Accepting a lump sum payment of R from the victim and promising not to blackmail again. b) Demanding a payment of u, where $u \in [0, 1]$. If blackmailed, the victim will either: 1) Comply with the demand and pay u to the blackmailer. This happens with probability $1 - u$. 2) Refuse to pay and denounce the blackmailer to the police. This happens with probability u. Once known to the police, the blackmailer cannot ask for any more

money. The blackmailer wants to maximize the expected amount of money he gets over N periods by optimal choice of the payment demands u_k. (Note that there is no additional penalty for being denounced to the police.) Write a DP algorithm and find the optimal policy.

4.23 [Whi82]

The Greek mythological hero Theseus is trapped in King Minos' Labyrinth maze. He can try on each day one of N passages. If he enters passage i he will escape with probability p_i, he will be killed with probability q_i, and he will determine that the passage is a dead end with probability $(1 - p_i - q_i)$, in which case he will return to the point from which he started. Use an interchange argument to show that trying passages in order of decreasing p_i/q_i maximizes the probability of escape within N days.

4.24 (Hardy's Theorem)

Let $\{a_1, \ldots, a_n\}$ and $\{b_1, \ldots, b_n\}$ be monotonically nondecreasing sequences of numbers. Let us associate with each $i = 1, \ldots, n$ a distinct index j_i, and consider the expression $\sum_{i=1}^{n} a_i b_{j_i}$. Use an interchange argument to show that this expression is maximized when $j_i = i$ for all i, and is minimized when $j_i = n - i + 1$ for all i.

4.25

A busy professor has to complete N projects. Each project k has a deadline d_k and the time it takes the professor to complete it is t_k. The professor can work on only one project at a time and must complete it before moving on to a new project. For a given order of completion of the projects, denote by c_k the time of completion of project k, i.e.,

$$c_k = t_k + \sum_{\substack{\text{projects } i \\ \text{completed before } k}} t_i.$$

The professor wants to order the projects so as to minimize the maximum tardiness, given by $\max_{k \in \{1,\ldots,N\}} \max(0, c_k - d_k)$. Use an interchange argument to show that it is optimal to complete the projects in the order of their deadlines (do the project with the closest deadline first).

4.26

Assume that we have two gold mines, Anaconda and Bonanza, and a gold-mining machine. Let x_A and x_B be the current amounts of gold in Anaconda and Bonanza, respectively (x_A and x_B are integer). When the machine is used in Anaconda or Bonanza, there is a probability p that $\lceil r_A x_A \rceil$ (or $\lceil r_B x_B \rceil$, respectively)

of the gold will be mined without damaging the machine, and a probability $1 - p$ that the machine will be damaged beyond repair and no gold will be mined. We assume that $0 < r_A < 1$ and $0 < r_B < 1$. We want to find a policy that selects the mine in which to use the machine at each period so as to maximize the total expected amount of gold mined.

(a) Use an interchange argument to show that it is optimal to mine Anaconda if and only if $r_A x_A \geq r_B x_B$.

(b) Solve the problem for $x_A = 2$, $x_B = 4$, $r_A = 0.4$, $r_B = 0.6$, $p = 0.9$.

4.27

Consider the quiz contest problem of Example 5.1, where is an order constraint that each question i may be answered only after a given number k_i of other questions have been answered. Use an interchange argument to show that an optimal list can be constructed by ordering the questions in decreasing order of $p_i R_i/(1 - p_i)$ and by sequentially answering the top question in the list out of those that are available (have not yet been answered and satisfy the order constraints).

4.28

Consider the quiz contest problem of Example 5.1, where there is a cost $F_i \geq 0$ for failing to answer question i correctly (in addition to losing the reward R_i).

(a) Use an interchange argument to show that it is optimal to answer the questions in order of decreasing $\big(p_i R_i - (1 - p_i)F_i\big)/(1 - p_i)$.

(b) Solve the variant of the problem where there is an option to stop answering questions.

4.29

Consider the quiz contest problem of Example 5.1, where there is a maximum number of questions that can be answered, which is smaller than the number of questions that are available.

(a) Show that it is not necessarily optimal to answer the questions in order of decreasing $p_i R_i/(1 - p_i)$. *Hint*: Try the case where only one out of two available questions can be answered.

(b) Give a simple algorithm to solve the problem where the number of available questions is one more than the maximum number of questions that can be answered.

4.30 (Reachability of One-Dimensional Linear Systems)

Generalize the analysis of Example 4.6.1 for the case of the one-dimensional linear system

$$x_{k+1} = ax_k + bu_k + w_k, \qquad k = 0, \ldots, N - 1,$$

where x_k should be kept within an interval $[-\alpha, \alpha]$, using controls from an interval $[\beta, \beta]$, and in spite of the effects of the disturbances that can take values from the interval $[-\gamma, \gamma]$. Derive an algorithm to generate the effective target sets, and characterize the set of initial states from which reachability of the target tube is guaranteed. What happens to this set as $N \to \infty$?

4.31 (Reachability of Ellipsoidal Tubes [BeR71b], [Ber72a]) (WWW)

Consider the linear system

$$x_{k+1} = A_k x_k + B_k u_k + w_k,$$

where the controls u_k and the disturbances w_k must belong to the ellipsoids

$$U_k = \{x \mid u' R_k u \leq 1\}, \qquad W_k = \{x \mid w' D_k w \leq 1\},$$

where R_k and D_k are given positive definite symmetric matrices.

(a) Focus on a single period k, and consider the problem of finding an ellipsoid

$$\overline{X} = \{x \mid x' K x \leq 1\},$$

where K is a positive definite symmetric matrix, such that $\overline{X}$ is contained in the intersection of the following two sets: (1) an ellipsoid $\{x \mid x' \Xi x \leq 1\}$, where Ξ is a positive definite symmetric matrix, and (2) the set of all states x such that there exists a $u \in U_k$ with the property that for all $w \in W_k$, we have $A_k x + B_k u + w \in X$, where

$$X = \{x \mid x' \Psi x \leq 1\},$$

and Ψ is a given positive definite symmetric matrix. Show that if for some scalar $\beta \in (0, 1)$, the matrix

$$F^{-1} = (1 - \beta)(\Psi^{-1} - \beta^{-1} D_k^{-1})$$

is well-defined as a positive definite matrix, an appropriate matrix K is given by

$$K = A_k'(F^{-1} + B_k R_k^{-1} B_k')^{-1} A_k + \Xi.$$

Furthermore, the linear control law

$$\mu(x) = -(R_k + B_k' F B_k)^{-1} B_k' F A_k x$$

satisfies the constraint $\mu(x) \in U_k$ for all $x \in \overline{X}$ and achieves reachability of X if $x \in \overline{X}$, i.e., μ is such that $A_k x + B_k \mu(x) + w \in X$ for all $x \in \overline{X}$ and $w \in W_k$. *Hint*: Use the fact that the vector sum of two ellipsoids $\{x \mid x' E_1 x \leq 1\}$ and $\{x \mid x' E_2 x \leq 1\}$ (with E_1 and E_2 positive definite symmetric) is contained in the ellipsoid $\{x \mid x' E x \leq 1\}$, where

$$E^{-1} = \beta^{-1} E_1^{-1} + (1 - \beta)^{-1} E_2^{-1}$$

and β is any scalar with $0 < \beta < 1$.

(b) Consider an ellipsoidal target tube $\{\hat{X}_0, \hat{X}_1, \ldots, \hat{X}_N\}$, where

$$\hat{X}_k = \{x \mid x' \Xi_k x \leq 1\}$$

and the Ξ_k are given positive definite symmetric matrices. Let the matrix sequences $\{F_k\}$ and $\{K_k\}$ be generated by the algorithm

$$K_N = \Xi_N,$$

$$F_{k+1}^{-1} = (1 - \beta_k)(K_{k+1}^{-1} - \beta_k^{-1} D_k^{-1}), \qquad k = 0, 1, \ldots, N-1,$$

$$K_k = A_k'(F_{k+1}^{-1} + B_k R_k^{-1} B_k')^{-1} A_k + \Xi_k, \qquad k = 0, 1, \ldots, N-1,$$

where β_k are scalars satisfying $0 < \beta_k < 1$. Use the procedure of part (a) to show that a linear control law of the form

$$\mu_k(x_k) = -(R_k + B_k' F_{k+1} B_k)^{-1} B_k' F_{k+1} A_k x_k, \qquad k = 0, 1, \ldots, N-1,$$

achieves reachability of the target tube, provided the matrices F_k are well-defined as positive definite matrices and x_0 satisfies $x_0' K_0^{-1} x_0 \leq 1$.

(c) Suppose that the matrices A_k, B_k, R_k, D_k, and Ξ_k do not depend on k, and that the algebraic matrix equation

$$K = A'\big((1 - \beta)(K^{-1} - \beta^{-1} D^{-1}) + B R^{-1} B'\big)^{-1} A + \Xi$$

has a positive definite solution $\overline{K}$ for some $\beta \in (0, 1)$ for which the matrix

$$F^{-1} = (1 - \beta)(\overline{K}^{-1} - \beta^{-1} D^{-1})$$

is well defined as a positive definite matrix. Show that if the initial state belongs to the set $\overline{X} = \{x \mid x' \overline{K} x \leq 1\}$, then all subsequent states will belong to $\overline{X}$ when the stationary linear control law

$$\mu(x) = -(R + B' F B)^{-1} B' F A x$$

is used.

4.32 (Pursuit-Evasion Games and Reachability [BeR71b])

Consider the linear system

$$x_{k+1} = A_k x_k + B_k u_k + G_k v_k, \qquad k = 0, 1, \ldots, N-1$$

where the controls u_k and v_k are selected by two antagonistic players from sets U_k and V_k, respectively, with exact knowledge of x_k (but without knowledge of the other player's choice at time k). The player selecting u_k aims to bring the state of the system within some given set X at some time $k = 1, \ldots, N$, while the player selecting v_k aims to keep the state of the system outside the set X at all times $k = 1, \ldots, N$. Relate this problem to the problem of reachability of a target tube, and characterize the sets of initial conditions x_0 starting from which the two players are guaranteed to achieve their objective with suitable choice of their control laws.

4.33

A famous but somewhat vain opera singer is scheduled to sing on N successive nights. If she is satisfied with her performance on a given night k (which happens with probability p, independently of the previous history) she will sing on the following night (i.e., night $k+1$). If she is not satisfied, however, she sulks and declares that she will not sing further. In this case, the only way to placate her into performing on the following night is for the opera director to send her an expensive gift, costing G dollars, which successfully placates her with probability q (independently of the previous history). If the gift does not placate her, the missed performance costs the opera house C dollars. The opera director may send a gift on any night, regardless of the success he has had with gifts on previous nights. The objective is to find a policy for when to send a gift and when not to, that minimizes the total cost from the N nights.

(a) Write a DP algorithm for solving the problem, and characterize as best as you can the optimal policy.

(b) Repeat part (a) for the case where the probability q is not constant, but rather is a decreasing function of the current stage.

4.34

An enterprising but somewhat foolish graduate student has invested the tuition for next semester in the stock market. As a result, he currently possesses a certain amount of stock that he/she must sell by registration day, which is N days away. The stock must be sold in its entirety on a single day, and will then be deposited in a bank where it will earn interest at a daily rate r. The value of the stock on day k is denoted by x_k and it evolves according to

$$x_{k+1} = \lambda x_k + w_k, \qquad x_0 : \text{given},$$

where λ is a scalar with $0 < \lambda < 1$, and w_k is a random variable taking one of a finite number of positive values. We assume that $w_0, \ldots, w_{N-1}$ are independent and identically distributed. The student wants to maximize the expected value of the money he/she has on registration day.

(a) Write a DP algorithm for solving the problem, and characterize as best as you can the optimal policy.

(b) Assume that the student has the option of selling only a portion of his stock on a given day. What if anything would he/she do different?

5

Problems with Imperfect State Information

Contents

We have assumed so far that the controller has access to the exact value of the current state, but this assumption is often unrealistic. For example, some state variables may be inaccessible, the sensors used for measuring them may be inaccurate, or the cost of obtaining the exact value of the state may be prohibitive. We model situations of this type by assuming that at each stage the controller receives some observations about the value of the current state, which may be corrupted by stochastic uncertainty.

Problems where the controller uses observations of this type in place of the state are called problems of *imperfect state information*, and are the subject of this chapter. We will find that even though there are DP algorithms for imperfect information problems, these algorithms are far more computationally intensive than in the perfect information case. For this reason, in the absence of an analytical solution, imperfect information problems are typically solved suboptimally in practice. On the other hand, we will see that conceptually, imperfect state information problems are no different than the perfect state information problems we have been studying so far. In fact by various reformulations, we can reduce an imperfect state information problem to one with perfect state information. We will study two different reductions of this type, which will yield two different DP algorithms. The first reduction is the subject of the next section, while the second reduction will be given in Section 5.4.

5.1 REDUCTION TO THE PERFECT INFORMATION CASE

We first formulate the imperfect state information counterpart of the basic problem.

Basic Problem with Imperfect State Information

Consider the basic problem of Section 1.2 where the controller, instead of having perfect knowledge of the state, has access to observations z_k of the form

$$z_0 = h_0(x_0, v_0), \qquad z_k = h_k(x_k, u_{k-1}, v_k), \qquad k = 1, 2, ..., N-1.$$

The observation z_k belongs to a given observation space Z_k. The random observation disturbance v_k belongs to a given space V_k and is characterized by a given probability distribution

$$P_{v_k}(\cdot \mid x_k, \ldots, x_0, u_{k-1}, \ldots, u_0, w_{k-1}, \ldots, w_0, v_{k-1}, \ldots, v_0),$$

which depends on the current state and the past states, controls, and disturbances.

The initial state x_0 is also random and characterized by a given probability distribution P_{x_0}. The probability distribution $P_{w_k}(\cdot \mid x_k, u_k)$ of w_k is given, and it may depend explicitly on x_k and u_k but not on the prior disturbances $w_0, \ldots, w_{k-1}, v_0, \ldots, v_{k-1}$. The control u_k is constrained to take values from a given nonempty subset U_k of the control space C_k. It is assumed that this subset does not depend on x_k.

Let us denote by I_k the information available to the controller at time k and call it the *information vector*. We have

$$\begin{aligned} &I_k = (z_0, z_1, \ldots, z_k, u_0, u_1, \ldots, u_{k-1}), \qquad k = 1, 2, \ldots, N-1, \\ &I_0 = z_0. \end{aligned} \tag{5.1}$$

We consider the class of policies consisting of a sequence of functions $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$, where each function μ_k maps the information vector I_k into the control space C_k and

$$\mu_k(I_k) \in U_k, \qquad \text{for all } I_k, \ k = 0, 1, \ldots, N-1.$$

Such policies are called *admissible*. We want to find an admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ that minimizes the cost function

$$J_\pi = \mathop{E}_{\substack{x_0, w_k, v_k \\ k=0,\ldots,N-1}} \left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(I_k), w_k\big) \right\}$$

subject to the system equation

$$x_{k+1} = f_k\big(x_k, \mu_k(I_k), w_k\big), \qquad k = 0, 1, \ldots, N-1,$$

and the measurement equation

$$\begin{aligned} &z_0 = h_0(x_0, v_0), \\ &z_k = h_k\big(x_k, \mu_{k-1}(I_{k-1}), v_k\big), \qquad k = 1, 2, \ldots, N-1. \end{aligned}$$

Note the difference from the perfect state information case. Whereas before we were trying to find a rule that would specify the control u_k to be applied for each state x_k and time k, now we are looking for a rule that gives the control to be applied for every possible information vector I_k (or state of information), that is, for every sequence of observations received and controls employed up to time k.

Example 5.1.1 (Multiaccess Communication)

Consider a collection of transmitting stations sharing a common channel, for example, a set of ground stations communicating with a satellite at a common frequency. The stations are synchronized to transmit packets of

data at integer times. Each packet requires one time unit (also called a *slot*) for transmission. The total number a_k of packet arrivals during slot k is independent of prior arrivals and has a given probability distribution. The stations do not know the backlog x_k at the beginning of the kth slot (the number of packets waiting to be transmitted). Packet transmissions are scheduled using a strategy (known as *slotted Aloha*) whereby each packet residing in the system at the beginning of the kth slot is transmitted during the kth slot with probability u_k (common for all packets). If two or more packets are transmitted simultaneously, they collide and have to rejoin the backlog for retransmission at a later slot. However, the stations can observe the channel and determine whether in any one slot there was a collision (two or more packets), a success (one packet), or an idle (no packets). These observations provide information about the state of the system (the backlog x_k) and can be used to select appropriately the control (the transmission probability u_k). The objective is to keep the backlog small, so we assume a cost per stage $g_k(x_k)$, which is a monotonically increasing function of x_k.

The state of the system here is the backlog x_k and evolves according to the equation

$$x_{k+1} = x_k + a_k - t_k,$$

where a_k is the number of new arrivals and t_k is the number of packets successfully transmitted during slot k. Both a_k and t_k may be viewed as disturbances, and the distribution of t_k depends on the state x_k and the control u_k. It can be seen that $t_k = 1$ (a success) with probability $x_k u_k (1 - u_k)^{x_k - 1}$, and $t_k = 0$ (idle or collision) otherwise [the probability of any one of the x_k waiting packets being transmitted, while all the other packets are not transmitted, is $u_k(1 - u_k)^{x_k - 1}$].

If we had perfect state information (i.e., the backlog x_k were known at the beginning of slot k), the optimal policy would be to select the value of u_k that maximizes $x_k u_k (1 - u_k)^{x_k - 1}$, which is the success probability.† By setting the derivative of this probability to zero, we find the optimal (perfect state information) policy to be

$$\mu_k(x_k) = \frac{1}{x_k}, \qquad \text{for all } x_k \geq 1.$$

† For a more detailed derivation, note that the DP algorithm for the perfect state information problem is

$$J_k(x_k) = g_k(x_k) + \min_{0 \leq u_k \leq 1} \underset{a_k}{E} \Big\{ p(x_k, u_k) J_{k+1}(x_k + a_k - 1) + \big(1 - p(x_k, u_k)\big) J_{k+1}(x_k + a_k) \Big\},$$

where $p(x_k, u_k)$ is the success probability $x_k u_k (1 - u_k)^{x_k - 1}$. Since the cost per stage $g_k(x_k)$ is an increasing function of the backlog x_k, it is clear that each cost-to-go function $J_k(x_k)$ is an increasing function of x_k (this can also be proved by induction). Thus $J_{k+1}(x_k + a_k) \geq J_{k+1}(x_k + a_k - 1)$ for all x_k and a_k, based on which the DP algorithm implies that the optimal u_k maximizes $p(x_k, u_k)$ over $[0, 1]$.

In practice, however, x_k is not known (imperfect state information), and the optimal control must be chosen on the basis of the available observations (i.e., the entire channel history of successes, idles, and collisions). These observations relate to the backlog history (the past states) and the past transmission probabilities (the past controls), but are corrupted by stochastic uncertainty. Mathematically, we may write an equation $z_{k+1} = v_{k+1}$, where z_{k+1} is the observation obtained at the end of the kth slot, and the random variable v_{k+1} yields an idle with probability $(1 - u_k)^{x_k}$, a success with probability $x_k u_k (1 - u_k)^{x_k - 1}$, and a collision otherwise.

It can be seen that this is a problem that fits the given imperfect state information framework. Unfortunately, the optimal solution to this problem is very complicated and for all practical purposes cannot be computed. A suboptimal solution will be discussed in Section 6.1.

Reformulation as a Perfect State Information Problem

We now show how to effect the reduction from imperfect to perfect state information. As in the discussion of state augmentation in Section 1.4, it is intuitively clear that we should define a new system whose state at time k is the set of all variables the knowledge of which can be of benefit to the controller when making the kth decision. Thus a first candidate as the state of the new system is the information vector I_k. Indeed we will show that this choice is appropriate.

We have by the definition of the information vector [cf. Eq. (5.1)]

$$I_{k+1} = (I_k, z_{k+1}, u_k), \qquad k = 0, 1, \ldots, N-2, \qquad I_0 = z_0. \tag{5.2}$$

These equations can be viewed as describing the evolution of a system of the same nature as the one considered in the basic problem of Section 1.2. The state of the system is I_k, the control is u_k, and z_{k+1} can be viewed as a random disturbance. Furthermore, we have

$$P(z_{k+1} \mid I_k, u_k) = P(z_{k+1} \mid I_k, u_k, z_0, z_1, \ldots, z_k),$$

since $z_0, z_1, \ldots, z_k$ are part of the information vector I_k. Thus the probability distribution of z_{k+1} depends explicitly only on the state I_k and control u_k of the new system (5.2) and not on the prior "disturbances" $z_k, \ldots, z_0$.

By writing

$$E\big\{g_k(x_k, u_k, w_k)\big\} = E\left\{ \underset{x_k, w_k}{E} \big\{ g_k(x_k, u_k, w_k) \mid I_k, u_k \big\} \right\},$$

we can similarly reformulate the cost function in terms of the variables of the new system. The cost per stage as a function of the new state I_k and the control u_k is

$$\tilde{g}_k(I_k, u_k) = \underset{x_k, w_k}{E} \big\{ g_k(x_k, u_k, w_k) \mid I_k, u_k \big\}. \tag{5.3}$$

Thus the basic problem with imperfect state information has been reformulated as a problem with perfect state information that involves the system (5.2) and the cost per stage (5.3). By writing the DP algorithm for this latter problem and substituting the expressions (5.2) and (5.3), we obtain

$$J_{N-1}(I_{N-1}) = \min_{u_{N-1}\in U_{N-1}} \left[\mathop{E}_{x_{N-1},\, w_{N-1}} \Big\{ g_N\big(f_{N-1}(x_{N-1}, u_{N-1}, w_{N-1})\big) + g_{N-1}(x_{N-1}, u_{N-1}, w_{N-1}) \mid I_{N-1}, u_{N-1} \Big\} \right], \tag{5.4}$$

and for $k = 0, 1, \ldots, N-2$,

$$J_k(I_k) = \min_{u_k\in U_k} \left[\mathop{E}_{x_k,\, w_k,\, z_{k+1}} \Big\{ g_k(x_k, u_k, w_k) + J_{k+1}(I_k, z_{k+1}, u_k) \mid I_k, u_k \Big\} \right]. \tag{5.5}$$

These equations constitute one possible DP algorithm for the imperfect state information problem. An optimal policy $\{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ is obtained by first minimizing in the right-hand side of the DP Eq. (5.4) for every possible value of the information vector I_{N-1} to obtain $\mu_{N-1}^*(I_{N-1})$. Simultaneously, $J_{N-1}(I_{N-1})$ is computed and used in the computation of $J_{N-2}(I_{N-2})$ via the minimization in the DP Eq. (5.5), which is carried out for every possible value of I_{N-2}. Proceeding similarly, $J_{N-3}(I_{N-3})$ and μ_{N-3}^* and so on are obtained, until $J_0(I_0) = J_0(z_0)$ is computed. The optimal cost J^* is then given by

$$J^* = \mathop{E}_{z_0}\{J_0(z_0)\}.$$

Machine Repair Example

A machine can be in one of two states denoted P and $\overline{P}$. State P corresponds to a machine in proper condition (good state) and state $\overline{P}$ to a machine in improper condition (bad state). If the machine is operated for one time period, it stays in state P with probability $\frac{2}{3}$ if it started in P, and it stays in state $\overline{P}$ with probability 1 if it started in $\overline{P}$. The machine is operated for a total of three time periods and starts in state P. At the end of the first and second time periods the machine is inspected and there are two possible inspection outcomes denoted G (probably good state) and B (probably bad state). If the machine is in the good state P, the inspection outcome is G with probability $\frac{3}{4}$; if the machine is in the bad state $\overline{P}$, the inspection outcome is B with probability $\frac{3}{4}$:

$$P(G \mid x = P) = \frac{3}{4}, \qquad P(B \mid x = P) = \frac{1}{4},$$

$$P(G \mid x = \overline{P}) = \frac{1}{4}, \qquad P(B \mid x = \overline{P}) = \frac{3}{4};$$

see Fig. 5.1.1. After each inspection one of two possible actions can be taken:

C: Continue operation of the machine.

S: Stop the machine, determine its state through an accurate diagnostic test, and if it is in the bad state $\overline{P}$ bring it back to the good state P.

At each period there is a cost of 2 and 0 units for starting the period with a machine in the bad state $\overline{P}$ and the good state P, respectively. The cost for taking the stop-and-repair action S is 1 unit and the terminal cost is 0.

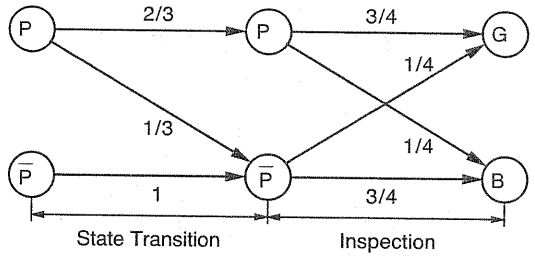


Figure 5.1.1 State transition diagram and probabilities of inspection outcomes in the machine repair example.

The problem is to determine the policy that minimizes the expected costs over the three time periods. In other words, we want to find the optimal action after the result of the first inspection is known, and after the results of the first and second inspections, as well as the action taken after the first inspection, are known.

It can be seen that this example falls within the general framework of the problem of this section. The state space consists of the two states P and $\overline{P}$,

$$\text{state space} = \{P, \overline{P}\},$$

and the control space consists of the two actions

$$\text{control space} = \{C, S\}.$$

The system evolution may be described by introducing a system equation

$$x_{k+1} = w_k, \qquad k = 0, 1,$$

where for $k = 0, 1$, the probability distribution of w_k is given by

$$P(w_k = P \mid x_k = P, u_k = C) = \frac{2}{3}, \qquad P(w_k = \overline{P} \mid x_k = P, u_k = C) = \frac{1}{3},$$

$$P(w_k = P \mid x_k = \overline{P}, u_k = C) = 0, \qquad P(w_k = \overline{P} \mid x_k = \overline{P}, u_k = C) = 1,$$

$$P(w_k = P \mid x_k = P, u_k = S) = \frac{2}{3}, \qquad P(w_k = \overline{P} \mid x_k = P, u_k = S) = \frac{1}{3},$$

$$P(w_k = P \mid x_k = \overline{P}, u_k = S) = \frac{2}{3}, \qquad P(w_k = \overline{P} \mid x_k = \overline{P}, u_k = S) = \frac{1}{3}.$$

We denote by x_0, x_1, x_2 the state of the machine at the end of the first, second, and third time period, respectively. Also we denote by u_0 the action taken after the first inspection (end of first time period) and by u_1 the action taken after the second inspection (end of second time period). The probability distribution of x_0 is

$$P(x_0 = P) = \frac{2}{3}, \qquad P(x_0 = \overline{P}) = \frac{1}{3}.$$

Note that we do not have perfect state information, since the inspections do not reveal the state of the machine with certainty. Rather the result of each inspection may be viewed as a measurement of the system state of the form

$$z_k = v_k, \qquad k = 0, 1,$$

where for $k = 0, 1$, the probability distribution of v_k is given by

$$P(v_k = G \mid x_k = P) = \frac{3}{4}, \qquad P(v_k = B \mid x_k = P) = \frac{1}{4},$$

$$P(v_k = G \mid x_k = \overline{P}) = \frac{1}{4}, \qquad P(v_k = B \mid x_k = \overline{P}) = \frac{3}{4}.$$

The cost resulting from a sequence of states x_0, x_1 and actions u_0, u_1 is

$$g(x_0, u_0) + g(x_1, u_1),$$

where

$$g(P, C) = 0, \qquad g(P, S) = 1, \qquad g(\overline{P}, C) = 2, \qquad g(\overline{P}, S) = 1.$$

The information vector at times 0 and 1 is

$$I_0 = z_0, \qquad I_1 = (z_0, z_1, u_0),$$

and we seek functions $\mu_0(I_0), \mu_1(I_1)$ that minimize

$$\mathop{E}_{\substack{x_0, w_0, w_1 \\ v_0, v_1}} \Big\{ g\big(x_0, \mu_0(I_0)\big) + g\big(x_1, \mu_1(I_1)\big) \Big\}$$

$$= \mathop{E}_{\substack{x_0, w_0, w_1 \\ v_0, v_1}} \Big\{ g\big(x_0, \mu_0(z_0)\big) + g\Big(x_1, \mu_1\big(z_0, z_1, \mu_0(z_0)\big)\Big) \Big\}.$$

We now apply the DP algorithm. It involves taking the minimum over the two possible actions, C and S, and it has the form

$$\begin{aligned} J_k(I_k) = \min \Big[& P(x_k = P \mid I_k, C) g(P, C) + P(x_k = \overline{P} \mid I_k, C) g(\overline{P}, C) \\ & + \mathop{E}_{z_{k+1}} \big\{ J_{k+1}(I_k, C, z_{k+1}) \mid I_k, C \big\}, \\ & P(x_k = P \mid I_k, S) g(P, S) + P(x_k = \overline{P} \mid I_k, S) g(\overline{P}, S) \\ & + \mathop{E}_{z_{k+1}} \big\{ J_{k+1}(I_k, S, z_{k+1}) \mid I_k, S \big\} \Big], \end{aligned}$$

where $k = 0, 1$, and the terminal condition is $J_2(I_2) = 0$.

Last Stage: We use Eq. (5.4) to compute $J_1(I_1)$ for each of the eight possible information vectors $I_1 = (z_0, z_1, u_0)$. As indicated by the above DP algorithm, for each of these vectors, we shall compute the expected cost of the possible actions, $u_1 = C$ and $u_1 = S$, and select as optimal the action with the smallest cost. We have

$$\text{cost of } C = 2 \cdot P(x_1 = \overline{P} \mid I_1), \qquad \text{cost of } S = 1,$$

and therefore

$$J_1(I_1) = \min\big[2P(x_1 = \overline{P} \mid I_1),\ 1\big].$$

The probabilities $P(x_1 = \overline{P} \mid I_1)$ can be computed by using Bayes' rule and the problem data. Some of the details will be omitted. We have:

(1) For $I_1 = (G, G, S)$

$$\begin{aligned} P(x_1 = \overline{P} \mid G, G, S) &= \frac{P(x_1 = \overline{P}, G, G \mid S)}{P(G, G \mid S)} \\ &= \frac{\frac{1}{3} \cdot \frac{1}{4} \cdot \left(\frac{2}{3} \cdot \frac{3}{4} + \frac{1}{3} \cdot \frac{1}{4}\right)}{\left(\frac{2}{3} \cdot \frac{3}{4} + \frac{1}{3} \cdot \frac{1}{4}\right)^2} = \frac{1}{7}. \end{aligned}$$

Hence

$$J_1(G, G, S) = \frac{2}{7}, \qquad \mu_1^*(G, G, S) = C.$$

(2) For $I_1 = (B, G, S)$

$$P(x_1 = \overline{P} \mid B, G, S) = P(x_1 = \overline{P} \mid G, G, S) = \frac{1}{7},$$

$$J_1(B, G, S) = \frac{2}{7}, \qquad \mu_1^*(B, G, S) = C.$$

(3) For $I_1 = (G, B, S)$

$$P(x_1 = \overline{P} \mid G, B, S) = \frac{P(x_1 = \overline{P}, G, B \mid S)}{P(G, B \mid S)} = \frac{\frac{1}{3} \cdot \frac{3}{4} \cdot \left(\frac{2}{3} \cdot \frac{3}{4} + \frac{1}{3} \cdot \frac{1}{4}\right)}{\left(\frac{2}{3} \cdot \frac{1}{4} + \frac{1}{3} \cdot \frac{3}{4}\right)\left(\frac{2}{3} \cdot \frac{3}{4} + \frac{1}{3} \cdot \frac{1}{4}\right)} = \frac{3}{5},$$

$$J_1(G, B, S) = 1, \qquad \mu_1^*(G, B, S) = S.$$

(4) For $I_1 = (B, B, S)$

$$P(x_1 = \overline{P} \mid B, B, S) = P(x_1 = \overline{P} \mid G, B, S) = \frac{3}{5},$$

$$J_1(B, B, S) = 1, \qquad \mu_1^*(B, B, S) = S.$$

(5) For $I_1 = (G, G, C)$

$$P(x_1 = \overline{P} \mid G, G, C) = \frac{P(x_1 = \overline{P}, G, G \mid C)}{P(G, G \mid C)} = \frac{1}{5},$$

$$J_1(G, G, C) = \frac{2}{5}, \qquad \mu_1^*(G, G, C) = C.$$

(6) For $I_1 = (B, G, C)$

$$P(x_1 = \overline{P} \mid B, G, C) = \frac{11}{23},$$

$$J_1(B, G, C) = \frac{22}{23}, \qquad \mu_1^*(B, G, C) = C.$$

(7) For $I_1 = (G, B, C)$

$$P(x_1 = \overline{P} \mid G, B, C) = \frac{9}{13},$$

$$J_1(G, B, C) = 1, \qquad \mu_1^*(G, B, C) = S.$$

(8) For $I_1 = (B, B, C)$

$$P(x_1 = \overline{P} \mid B, B, C) = \frac{33}{37},$$

$$J_1(B, B, C) = 1, \qquad \mu_1^*(B, B, C) = S.$$

Summarizing the results for the last stage, the optimal policy is to continue ($u_1 = C$) if the result of the last inspection was G, and to stop ($u_1 = S$) if the result of the last inspection was B.

First Stage: Here we use the DP Eq. (5.5) to compute $J_0(I_0)$ for each of the two possible information vectors $I_0 = (G)$, $I_0 = (B)$. We have

$$\begin{aligned}\text{cost of } C &= 2P(x_0 = \overline{P} \mid I_0, C) + \underset{z_1}{E}\{J_1(I_0, z_1, C) \mid I_0, C\} \\ &= 2P(x_0 = \overline{P} \mid I_0, C) + P(z_1 = G \mid I_0, C) J_1(I_0, G, C) \\ &\quad + P(z_1 = B \mid I_0, C) J_1(I_0, B, C),\end{aligned}$$

$$\begin{aligned}\text{cost of } S &= 1 + \underset{z_1}{E}\{J_1(I_0, z_1, S) \mid I_0, S\} \\ &= 1 + P(z_1 = G \mid I_0, S) J_1(I_0, G, S) + P(z_1 = B \mid I_0, S) J_1(I_0, B, S),\end{aligned}$$

and

$$\begin{aligned}J_0(I_0) = \min \Big[&2P(x_0 = \overline{P} \mid I_0, C) + \underset{z_1}{E}\{J_1(I_0, z_1, C) \mid I_0, C\}, \\ &1 + \underset{z_1}{E}\{J_1(I_0, z_1, S) \mid I_0, S\}\Big]\end{aligned}$$

(1) For $I_0 = (G)$: Direct calculation yields

$$P(z_1 = G \mid G, C) = \frac{15}{28}, \qquad P(z_1 = B \mid G, C) = \frac{13}{28},$$

$$P(z_1 = G \mid G, S) = \frac{7}{12}, \qquad P(z_1 = B \mid G, S) = \frac{5}{12},$$

$$P(x_0 = \overline{P} \mid G, C) = \frac{1}{7},$$

and hence

$$\begin{aligned}J_0(G) = \min \Big[&2 \cdot \frac{1}{7} + \frac{15}{28} J_1(G, G, C) + \frac{13}{28} J_1(G, B, C), \\ &1 + \frac{7}{12} J_1(G, G, S) + \frac{5}{12} J_1(G, B, S)\Big].\end{aligned}$$

Using the values of J_1 obtained in the previous stage

$$\begin{aligned}J_0(G) &= \min \left[2 \cdot \frac{1}{7} + \frac{15}{28} \cdot \frac{2}{5} + \frac{13}{28} \cdot 1,\ 1 + \frac{7}{12} \cdot \frac{2}{7} + \frac{5}{12} \cdot 1\right] \\ &= \min \left[\frac{27}{28}, \frac{19}{12}\right] = \frac{27}{28},\end{aligned}$$

$$J_0(G) = \frac{27}{28}, \qquad \mu_0^*(G) = C.$$

(2) For $I_0 = (B)$: Direct calculation yields

$$P(z_1 = G \mid B, C) = \frac{23}{60}, \qquad P(z_1 = B \mid B, C) = \frac{37}{60},$$

$$P(z_1 = G \mid B, S) = \frac{7}{12}, \qquad P(z_1 = B \mid B, S) = \frac{5}{12},$$

$$P(x_0 = \overline{P} \mid B, C) = \frac{3}{5},$$

and

$$J_0(B) = \min \left[2 \cdot \frac{3}{5} + \frac{23}{60} J_1(B, G, C) + \frac{37}{60} J_1(B, B, C), \right.$$
$$\left. 1 + \frac{7}{12} J_1(B, G, S) + \frac{5}{12} J_1(B, B, S) \right].$$

Using the values of J_1 obtained in the previous state

$$J_0(B) = \min \left[\frac{131}{60}, \frac{19}{12} \right] = \frac{19}{12},$$

$$J_0(B) = \frac{19}{12}, \qquad \mu_0^*(B) = S.$$

Summarizing, the optimal policy for both stages is to continue if the result of the latest inspection is G, and to stop and repair otherwise.

The optimal cost is

$$J^* = P(G) J_0(G) + P(B) J_0(B).$$

We can verify that $P(G) = \frac{7}{12}$ and $P(B) = \frac{5}{12}$, so that

$$J^* = \frac{7}{12} \cdot \frac{27}{28} + \frac{5}{12} \cdot \frac{19}{12} = \frac{176}{144}.$$

In the above example, the computation of the optimal policy and the optimal cost by means of the DP algorithm (5.4) and (5.5) was possible because the problem was very simple. It is easy to see that for a more complex problem, the computational requirements of the DP algorithm can be prohibitive, particularly if the number of possible information vectors I_k is large (or infinite). Unfortunately, even if the control and observation spaces are simple (one-dimensional or finite), the space of the information vector I_k may have large dimension. This makes the application of the algorithm very difficult or computationally impossible in many cases. However, there are some problems where an analytical solution is possible, and the next two sections deal with such problems.

5.2 LINEAR SYSTEMS AND QUADRATIC COST

We will show how the DP algorithm of the preceding section can be used to solve the imperfect state information analog of the linear system/quadratic cost problem of Section 4.1. We have the same linear system

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

and quadratic cost

$$E \left\{ x_N' Q_N x_N + \sum_{k=0}^{N-1} (x_k' Q_k x_k + u_k' R_k u_k) \right\},$$

but now the controller does not have access to the current state. Instead it receives at the beginning of each period k an observation of the form

$$z_k = C_k x_k + v_k, \qquad k = 0, 1, \ldots, N-1,$$

where $z_k \in \Re^s$, C_k is a given $s \times n$ matrix, and $v_k \in \Re^s$ is an observation noise vector with given probability distribution. Furthermore, the vectors v_k are independent, and independent from w_k and x_0 as well. We make the same assumptions as in Section 4.1 concerning the input disturbances w_k, i.e., that they are independent, zero mean, and that they have finite variance. The system matrices A_k, B_k are known; there is no analytical solution of the imperfect information counterpart of the model with random system matrices considered in Section 4.1.

From the DP Eq. (5.4) we have

$$J_{N-1}(I_{N-1}) = \min_{u_{N-1}} \Big[\underset{x_{N-1}, w_{N-1}}{E} \{ x_{N-1}' Q_{N-1} x_{N-1} + u_{N-1}' R_{N-1} u_{N-1}$$
$$+ (A_{N-1} x_{N-1} + B_{N-1} u_{N-1} + w_{N-1})'$$
$$\cdot Q_N (A_{N-1} x_{N-1} + B_{N-1} u_{N-1} + w_{N-1}) \mid I_{N-1} \} \Big]$$

Since $E\{w_{N-1} \mid I_{N-1}\} = E\{w_{N-1}\} = 0$, this expression can be written as

$$J_{N-1}(I_{N-1}) = \underset{x_{N-1}}{E} \{ x_{N-1}' (A_{N-1}' Q_N A_{N-1} + Q_{N-1}) x_{N-1} \mid I_{N-1} \}$$
$$+ \underset{w_{N-1}}{E} \{ w_{N-1}' Q_N w_{N-1} \}$$
$$+ \min_{u_{N-1}} \Big[u_{N-1}' (B_{N-1}' Q_N B_{N-1} + R_{N-1}) u_{N-1}$$
$$+ 2E\{x_{N-1} \mid I_{N-1}\}' A_{N-1}' Q_N B_{N-1} u_{N-1} \Big]. \qquad (5.6)$$

The minimization yields the optimal policy for the last stage:

$$\begin{aligned} u_{N-1}^* &= \mu_{N-1}^*(I_{N-1}) \\ &= -(B_{N-1}'Q_NB_{N-1} + R_{N-1})^{-1}B_{N-1}'Q_NA_{N-1}E\{x_{N-1} \mid I_{N-1}\}, \end{aligned} \tag{5.7}$$

and upon substitution in Eq. (5.6), we obtain

$$\begin{aligned} J_{N-1}(I_{N-1}) = &\underset{x_{N-1}}{E}\{x_{N-1}'K_{N-1}x_{N-1} \mid I_{N-1}\} \\ &+ \underset{x_{N-1}}{E}\Big\{\big(x_{N-1} - E\{x_{N-1} \mid I_{N-1}\}\big)' \\ &\qquad \cdot P_{N-1}\big(x_{N-1} - E\{x_{N-1} \mid I_{N-1}\}\big) \mid I_{N-1}\Big\} \\ &+ \underset{w_{N-1}}{E}\{w_{N-1}'Q_Nw_{N-1}\}, \end{aligned}$$

where the matrices K_{N-1} and P_{N-1} are given by

$$\begin{aligned} P_{N-1} &= A_{N-1}'Q_NB_{N-1}(R_{N-1} + B_{N-1}'Q_NB_{N-1})^{-1}B_{N-1}'Q_NA_{N-1}, \\ K_{N-1} &= A_{N-1}'Q_NA_{N-1} - P_{N-1} + Q_{N-1}. \end{aligned}$$

Note that the optimal policy (5.6) is identical to its perfect state information counterpart except that x_{N-1} is replaced by its conditional expectation $E\{x_{N-1} \mid I_{N-1}\}$. Note also that the cost-to-go $J_{N-1}(I_{N-1})$ exhibits a corresponding similarity to its perfect state information counterpart except that $J_{N-1}(I_{N-1})$ contains an additional middle term, which is in effect a penalty for estimation error.

Now the DP equation for period $N-2$ is

$$\begin{aligned} J_{N-2}(I_{N-2}) &= \min_{u_{N-2}}\Big[\underset{x_{N-2},w_{N-2},z_{N-1}}{E}\{x_{N-2}'Q_{N-2}x_{N-2} + u_{N-2}'R_{N-2}u_{N-2} \\ &\qquad + J_{N-1}(I_{N-1}) \mid I_{N-2}, u_{N-2}\}\Big] \\ &= E\{x_{N-2}'Q_{N-2}x_{N-2} \mid I_{N-2}\} \\ &\quad + \min_{u_{N-2}}\Big[u_{N-2}'R_{N-2}u_{N-2} + E\{x_{N-1}'K_{N-1}x_{N-1} \mid I_{N-2}, u_{N-2}\}\Big] \\ &\quad + E\Big\{\big(x_{N-1} - E\{x_{N-1} \mid I_{N-1}\}\big)' \\ &\qquad \cdot P_{N-1}\big(x_{N-1} - E\{x_{N-1} \mid I_{N-1}\}\big) \mid I_{N-2}, u_{N-2}\Big\} \\ &\quad + \underset{w_{N-1}}{E}\{w_{N-1}'Q_Nw_{N-1}\}. \end{aligned} \tag{5.8}$$

Note that we have excluded the next to last term from the minimization with respect to u_{N-2}. We have done so since this term turns out to be independent of u_{N-2}. To show this fact, we need the following lemma.

The lemma says essentially that the quality of estimation as expressed by the statistics of the error $x_k - E\{x_k \mid I_k\}$ cannot be influenced by the choice of control. This is due to the linearity of both the system and the measurement equation. In particular, x_k and $E\{x_k \mid I_k\}$ contain the same linear terms in $(u_0, \ldots, u_{k-1})$, which cancel each other out.

Lemma 5.2.1: For every k, there is a function M_k such that we have

$$x_k - E\{x_k \mid I_k\} = M_k(x_0, w_0, \ldots, w_{k-1}, v_0, \ldots, v_k),$$

independently of the policy being used.

Proof: Fix a policy and consider the following two systems. In the first system there is control as determined by the policy,

$$x_{k+1} = A_kx_k + B_ku_k + w_k, \qquad z_k = C_kx_k + v_k,$$

while in the second system there is no control,

$$\overline{x}_{k+1} = A_k\overline{x}_k + \overline{w}_k, \qquad \overline{z}_k = C_k\overline{x}_k + \overline{v}_k.$$

We consider the evolution of these two systems when their initial conditions are identical,

$$x_0 = \overline{x}_0,$$

and when their system disturbance and observation noise vectors are also identical,

$$w_k = \overline{w}_k, \qquad v_k = \overline{v}_k, \qquad k = 0, 1, \ldots, N-1.$$

Consider the vectors

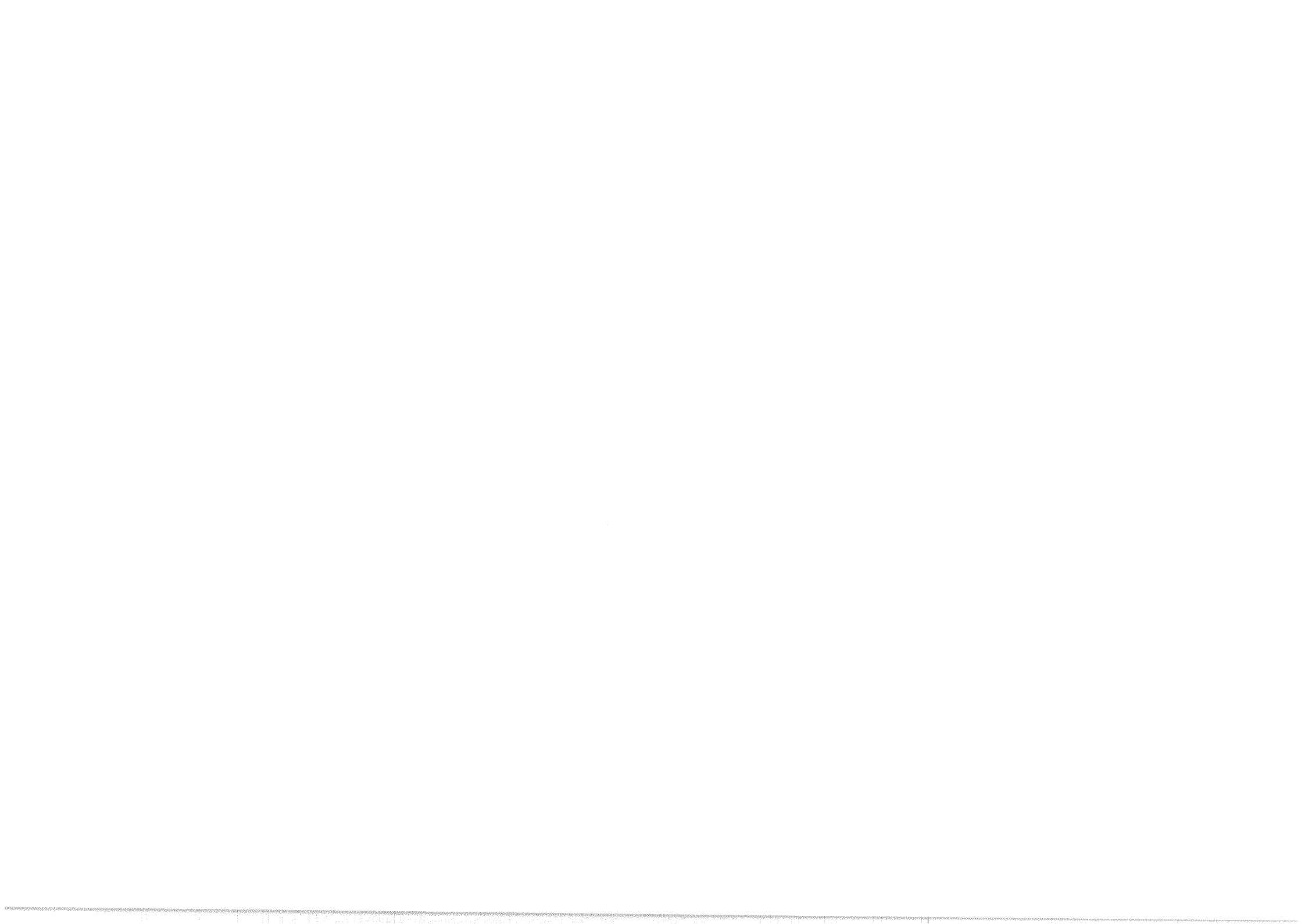
$$\begin{aligned} Z^k &= (z_0, \ldots, z_k)', & \overline{Z}^k &= (\overline{z}_0, \ldots, \overline{z}_k)', \\ W^k &= (w_0, \ldots, w_k)', & V^k &= (v_0, \ldots, v_k)', & U^k &= (u_0, \ldots, u_k)'. \end{aligned}$$

Linearity implies the existence of matrices F_k, G_k, and H_k such that

$$\begin{aligned} x_k &= F_kx_0 + G_kU^{k-1} + H_kW^{k-1}, \\ \overline{x}_k &= F_kx_0 + H_kW^{k-1}. \end{aligned}$$

Since the vector $U^{k-1} = (u_0, \ldots, u_{k-1})'$ is part of the information vector I_k, we have $U^{k-1} = E\{U^{k-1} \mid I_k\}$, so

$$\begin{aligned} E\{x_k \mid I_k\} &= F_kE\{x_0 \mid I_k\} + G_kU^{k-1} + H_kE\{W^{k-1} \mid I_k\}, \\ E\{\overline{x}_k \mid I_k\} &= F_kE\{x_0 \mid I_k\} + H_kE\{W^{k-1} \mid I_k\}. \end{aligned}$$



We thus obtain

$$x_k - E\{x_k \mid I_k\} = \overline{x}_k - E\{\overline{x}_k \mid I_k\}.$$

From the equations for z_k and $\overline{z}_k$, we see that

$$\overline{Z}^k = Z^k - R_k U^{k-1} = S_k W^{k-1} + T_k V^k,$$

where R_k, S_k, and T_k are some matrices of appropriate dimension. Thus, the information provided by $I_k = (Z^k, U^{k-1})$ regarding $\overline{x}_k$ is summarized in $\overline{Z}^k$, and we have $E\{\overline{x}_k \mid I_k\} = E\{\overline{x}_k \mid \overline{Z}^k\}$, so that

$$x_k - E\{x_k \mid I_k\} = \overline{x}_k - E\{\overline{x}_k \mid \overline{Z}^k\}.$$

The function M_k given by

$$M_k(x_0, w_0, \ldots, w_{k-1}, v_0, \ldots, v_k) = \overline{x}_k - E\{\overline{x}_k \mid \overline{Z}^k\}$$

serves the purpose stated in the lemma. **Q.E.D.**

We can now justify excluding the term

$$E\Big\{\big(x_{N-1} - E\{x_{N-1} \mid I_{N-1}\}\big)' P_{N-1}\big(x_{N-1} - E\{x_{N-1} \mid I_{N-1}\}\big) \mid I_{N-2}, u_{N-2}\Big\}$$

from the minimization in Eq. (5.8), as being independent of u_{N-2}. Indeed, by using the lemma, we see that

$$x_{N-1} - E\{x_{N-1} \mid I_{N-1}\} = \xi_{N-1},$$

where ξ_{N-1} is a function of $x_0, w_0, \ldots, w_{N-2}, v_0, \ldots, v_{N-1}$. Since ξ_{N-1} is independent of u_{N-2}, the conditional expectation of $\xi'_{N-1} P_{N-1} \xi_{N-1}$ satisfies

$$E\{\xi'_{N-1} P_{N-1} \xi_{N-1} \mid I_{N-2}, u_{N-2}\} = E\{\xi'_{N-1} P_{N-1} \xi_{N-1} \mid I_{N-2}\}.$$

Returning now to our problem, the minimization in Eq. (5.8) yields, using an argument similar to the one for the last stage,

$$\begin{aligned} u^*_{N-2} &= \mu^*_{N-2}(I_{N-2}) \\ &= -(R_{N-2} + B'_{N-2} K_{N-1} B_{N-2})^{-1} B'_{N-2} K_{N-1} A_{N-2} E\{x_{N-2} \mid I_{N-2}\}. \end{aligned}$$

We can proceed similarly to obtain the optimal policy for every stage:

$$\mu^*_k(I_k) = L_k E\{x_k \mid I_k\}, \tag{5.9}$$

where the matrix L_k is given by

$$L_k = -(R_k + B'_k K_{k+1} B_k)^{-1} B'_k K_{k+1} A_k,$$

with the matrices K_k given recursively by the Riccati equation

$$K_N = Q_N,$$

$$P_k = A'_k K_{k+1} B_k (R_k + B'_k K_{k+1} B_k)^{-1} B'_k K_{k+1} A_k,$$

$$K_k = A'_k K_{k+1} A_k - P_k + Q_k.$$

The key step in this derivation is that at stage k of the DP algorithm, the minimization over u_k that defines $J_k(I_k)$ involves the additional terms

$$E\Big\{\big(x_s - E\{x_s \mid I_s\}\big)' P_s \big(x_s - E\{x_s \mid I_s\}\big) \mid I_k, u_k\Big\},$$

where $s = k+1, \ldots, N-1$. By using the argument of the proof of the earlier lemma, it can be seen that none of these terms depends on u_k so that the presence of these terms does not affect the minimization in the DP algorithm. As a result, the optimal policy is the same as the one for the perfect information case, except that the state x_k is replaced by its conditional expectation $E\{x_k \mid I_k\}$.

It is interesting to note that the optimal controller can be decomposed into the two parts shown in Fig. 5.2.1:

(a) An *estimator*, which uses the data to generate the conditional expectation $E\{x_k \mid I_k\}$.

(b) An *actuator*, which multiplies $E\{x_k \mid I_k\}$ by the gain matrix L_k and applies the control input $u_k = L_k E\{x_k \mid I_k\}$.

Furthermore, the gain matrix L_k is independent of the statistics of the problem and is the same as the one that would be used if we were faced with the deterministic problem, where w_k and x_0 would be fixed and equal to their expected values. On the other hand, as shown in Appendix E, the estimate $\hat{x}$ of a random vector x given some information (random vector) I, which minimizes the mean squared error $E_x\{\|x - \hat{x}\|^2 \mid I\}$ is precisely the conditional expectation $E\{x \mid I\}$ (expand the quadratic form and set to zero the derivative with respect to $\hat{x}$). Thus the *estimator portion of the optimal controller is an optimal solution of the problem of estimating the state x_k assuming no control takes place, while the actuator portion is an optimal solution of the control problem assuming perfect state information prevails*. This property, which shows that the two portions of the optimal controller can be designed independently as optimal solutions of an estimation and a control problem, has been called the *separation theorem for*

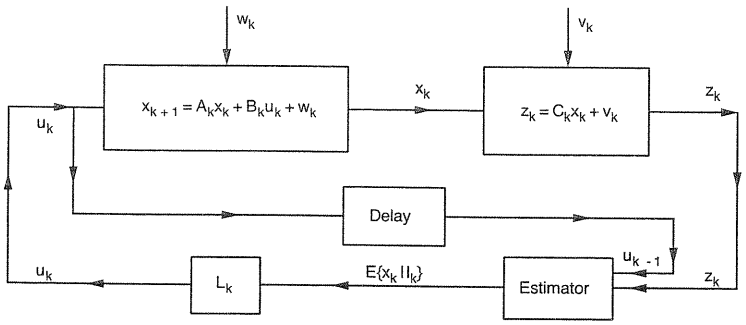


Figure 5.2.1 Structure of the optimal controller for the linear-quadratic problem. It consists of an estimator, which generates the conditional expectation $E\{x_k \mid I_k\}$, and an actuator, which multiplies $E\{x_k \mid I_k\}$ by the gain matrix L_k.

linear systems and quadratic cost and occupies a central position in modern automatic control theory.

Another interesting observation is that the optimal controller applies at each time k the control that would be applied when faced with the deterministic problem of minimizing the cost-to-go

$$x_N' Q_N x_N + \sum_{i=k}^{N-1} (x_i' Q_i x_i + u_i' R_i u_i),$$

and the input disturbances $w_k, w_{k+1}, \ldots, w_{N-1}$ and current state x_k were known and fixed at their conditional expected values, which are zero and $E\{x_k \mid I_k\}$, respectively. This is another manifestation of the *certainty equivalence principle*, which was referred to in Section 4.1. A similar result holds in the case of correlated disturbances; see Exercise 5.1.

Implementation Aspects – Steady-State Controller

As explained in the perfect information case, the linear form of the actuator portion of the optimal policy is particularly attractive for implementation. In the imperfect information case, however, we need to implement an estimator that produces the conditional expectation

$$\hat{x}_k = E\{x_k \mid I_k\},$$

and this is not easy in general. Fortunately, in the important special case, where *the disturbances w_k, v_k, and the initial state x_0 are Gaussian random vectors*, a convenient implementation of the estimator is possible by means of the Kalman filtering algorithm, which is developed in Appendix E. This algorithm is organized recursively so that to produce $\hat{x}_{k+1}$ at time $k+1$,

only the most recent measurement z_{k+1} is needed, together with $\hat{x}_k$ and u_k. In particular, we have for all $k = 0, \ldots, N-1$,

$$\hat{x}_{k+1} = A_k \hat{x}_k + B_k u_k + \Sigma_{k+1|k+1} C_{k+1}' N_{k+1}^{-1} \big(z_{k+1} - C_{k+1}(A_k \hat{x}_k + B_k u_k)\big),$$

and

$$\hat{x}_0 = E\{x_0\} + \Sigma_{0|0} C_0' N_0^{-1} \big(z_0 - C_0 E\{x_0\}\big),$$

where the matrices $\Sigma_{k|k}$ are precomputable and are given recursively by

$$\Sigma_{k+1|k+1} = \Sigma_{k+1|k} - \Sigma_{k+1|k} C_{k+1}' (C_{k+1} \Sigma_{k+1|k} C_{k+1}' + N_{k+1})^{-1} C_{k+1} \Sigma_{k+1|k},$$

$$\Sigma_{k+1|k} = A_k \Sigma_{k|k} A_k' + M_k, \qquad k = 0, 1, \ldots, N-1,$$

with

$$\Sigma_{0|0} = S - S C_0' (C_0 S C_0' + N_0)^{-1} C_0 S.$$

In these equations, M_k, N_k, and S are the covariance matrices of w_k, v_k, and x_0, respectively, and we assume that w_k and v_k have zero mean; that is

$$E\{w_k\} = E\{v_k\} = 0,$$

$$M_k = E\{w_k w_k'\}, \qquad N_k = E\{v_k v_k'\},$$

$$S = E\Big\{\big(x_0 - E\{x_0\}\big)\big(x_0 - E\{x_0\}\big)'\Big\}.$$

In addition, the matrices N_k are assumed to be positive definite.

Consider now the case where the system and measurement equations, and the disturbance statistics are stationary. We can then drop subscripts from the system matrices. Assume that *the pair (A, B) is controllable and that the matrix Q can be written as $Q = F'F$, where F is a matrix such that the pair (A, F) is observable*. By the theory of Section 4.1, if the horizon tends to infinity, the optimal controller tends to the steady-state policy

$$\mu^*(I_k) = L \hat{x}_k, \tag{5.10}$$

where

$$L = -(R + B'KB)^{-1} B'KA, \tag{5.11}$$

and K is the unique positive semidefinite symmetric solution of the algebraic Riccati equation

$$K = A'\big(K - KB(R + B'KB)^{-1} B'K\big)A + Q.$$

By a similar argument, it can be shown (see Appendix E) that $\hat{x}_k$ can be generated in the limit as $k \to \infty$ by a steady-state Kalman filtering algorithm:

$$\hat{x}_{k+1} = (A + BL)\hat{x}_k + \overline{\Sigma} C' N^{-1} \big(z_{k+1} - C(A + BL)\hat{x}_k\big), \tag{5.12}$$

where $\overline{\Sigma}$ is given by

$$\overline{\Sigma} = \Sigma - \Sigma C'(C\Sigma C' + N)^{-1}C\Sigma,$$

and Σ is the unique positive semidefinite symmetric solution of the Riccati equation

$$\Sigma = A\big(\Sigma - \Sigma C'(C\Sigma C' + N)^{-1}C\Sigma\big)A' + M.$$

The assumptions required for this are that *the pair (A, C) is observable and that the matrix M can be written as $M = DD'$, where D is a matrix such that the pair (A, D) is controllable.* The steady-state controller of Eqs. (5.10)-(5.12) is particularly attractive for practical implementation. Furthermore, as shown in Appendix E, it results in a stable closed-loop system, under the preceding controllability and observability assumptions.

5.3 MINIMUM VARIANCE CONTROL OF LINEAR SYSTEMS

We have considered so far the control of linear systems in state variable form as in the previous section. However, linear systems are often modeled by means of an input-output equation, which is more economical in the number of parameters needed to describe the system dynamics. In this section we consider single-input, single-output, linear, time-invariant systems, and a special type of quadratic cost function. The resulting optimal policy is particularly simple and has found wide application. We first introduce some of the basic facts regarding linear systems in input-output form. Detailed discussions may be found in the books by Aström and Wittenmark [AsW84], [AsW90], Goodwin and Sin [GoS84], and Whittle [Whi63].

We consider a single-input single-output discrete-time linear system, which is specified by an equation of the form

$$y_k + a_1 y_{k-1} + \cdots + a_m y_{k-m} = b_0 u_k + b_1 u_{k-1} + \cdots + b_m u_{k-m}, \quad (5.13)$$

where a_i, b_i are given scalars. The scalar sequences $\{u_k \mid k = 0, \pm 1, \pm 2, \ldots\}$, $\{y_k \mid k = 0, \pm 1, \pm 2, \ldots\}$ are viewed as the input and output of the system, respectively. Note that we allow time to extend to $-\infty$ as well as $+\infty$; this will be useful for describing generic properties of the system relating to stability. We will later revert to our usual convention of starting at time 0 and proceeding forward.

It is convenient to describe this type of system by means of the *backward shift operator*, denoted s, which when operating on a sequence $\{x_k \mid k = 0, \pm 1, \pm 2, \ldots\}$ shifts its index by one unit; that is,

$$s(x_k) = x_{k-1}, \qquad k = 0, \pm 1, \pm 2, \ldots$$

We denote by s^r the operator resulting from r successive applications of s:

$$s^r(x_k) = x_{k-r}, \qquad k = 0, \pm 1, \pm 2, \ldots \quad (5.14)$$

We also write for simplicity $s^r x_k = x_{k-r}$. The *forward shift operator*, denoted s^{-1}, is the inverse of s and is defined by

$$s^{-1}(x_k) = x_{k+1}, \qquad k = 0, \pm 1, \pm 2, \ldots$$

Thus the notation (5.14) holds for all integers r. We can form linear combinations of operators of the form s^r. Thus, for example, the operator $(s + 2s^2)$ is defined by

$$(s + 2s^2)(x_k) = x_{k-1} + 2x_{k-2}, \qquad k = 0, \pm 1, \pm 2, \ldots$$

With this notation, Eq. (5.13) can be written as

$$A(s)y_k = B(s)u_k,$$

where $A(s)$, $B(s)$ are the operators

$$A(s) = 1 + a_1 s + \cdots + a_m s^m,$$

$$B(s) = b_0 + b_1 s + \cdots + b_m s^m.$$

Sometimes it is convenient to write the equation $A(s)y_k = B(s)u_k$ as

$$y_k = \frac{B(s)}{A(s)} u_k$$

or

$$\frac{A(s)}{B(s)} y_k = u_k.$$

The meaning of both equations is that the sequences $\{y_k\}$ and $\{u_k\}$ are related via $A(s)y_k = B(s)u_k$. There is a certain ambiguity here in that, for a fixed $\{u_k\}$, the equation $A(s)y_k = B(s)u_k$ has an infinite number of solutions in $\{y_k\}$. For example, the equation

$$y_k + a y_{k-1} = u_k$$

for $u_k \equiv 0$ has as solutions all sequences of the form $y_k = \beta(-a)^k$, where β is any scalar; the solution becomes unique only after some boundary condition for the sequence $\{y_k\}$ is specified. As will be discussed shortly, however, for stable systems and for a *bounded* sequence $\{u_k\}$ there is a unique solution $\{y_k\}$ that is *bounded*. It is this solution that will be denoted by $\big(B(s)/A(s)\big)u_k$ in what follows. The reader who is familiar with linear

dynamic system theory will note that $B(s)/A(s)$ can be viewed as a *transfer function* involving z-transforms.

We now introduce some terminology. When the sequences $\{y_k\}$ and $\{u_k\}$ satisfy $A(s)y_k = B(s)u_k$, we say that y_k is *obtained by passing* u_k *through the filter* $B(s)/A(s)$. This comes from engineering terminology, where linear time-invariant systems are commonly referred to as filters. We also refer to the equation $A(s)y_k = B(s)u_k$ as the filter $B(s)/A(s)$.

A filter $B(s)/A(s)$ is said to be *stable* if the polynomial $A(s)$ has all its (complex) roots strictly outside the unit circle of the complex plane; that is, $|\rho| > 1$ for all complex ρ satisfying $A(\rho) = 0$. A stable filter $B(s)/A(s)$ has the following two properties:

(a) Every solution $\{y_k\}$ of

$$A(s)y_k = 0$$

satisfies $\lim_{k\to\infty} y_k = 0$; that is, the output y_k tends to zero if the input sequence $\{u_k\}$ is identically zero.

(b) For every bounded sequence $\{\overline{u}_k\}$, the equation

$$A(s)y_k = B(s)\overline{u}_k$$

has a *unique* solution $\{\overline{y}_k\}$ within the class of bounded sequences. Furthermore, every solution $\{y_k\}$ (possibly unbounded) of the equation satisfies

$$\lim_{k\to\infty} (y_k - \overline{y}_k) = 0.$$

For example, consider the system

$$y_k - 0.5y_{k-1} = u_k.$$

Given the bounded input sequence $\overline{u}_k = \{\ldots, 1, 1, 1, \ldots\}$, the set of all solutions is given by

$$y_k = 2 + \frac{\beta}{2^k},$$

where β is a scalar, but of these the only bounded solution is $\overline{y}_k = \{\ldots, 2, 2, 2, \ldots\}$. The solution $\{\overline{y}_k\}$ can thus be naturally associated with the input sequence $\{u_k\}$; it is also known as the *forced response* of the system due to the input $\{u_k\}$.

ARMAX Models – Reduction to State Space Form

We now consider a linear system with output y_k, which is driven by two inputs: a random noise input ϵ_k, and a control input u_k. It has the form

$$\begin{aligned} y_k + a_1 y_{k-1} + \cdots + a_m y_{k-m} = {} & b_1 u_{k-1} + \cdots + b_m u_{k-m} \\ & + \epsilon_k + c_1 \epsilon_{k-1} + \cdots + c_m \epsilon_{k-m}, \end{aligned} \tag{5.15}$$

and it is known as an ARMAX model (AutoRegressive, Moving Average, with eXogenous input). We assume throughout that the random variables ϵ_k are mutually independent. We can write the model in the shorthand form

$$A(s)y_k = B(s)u_k + C(s)\epsilon_k,$$

where the polynomials $A(s)$, $B(s)$, and $C(s)$ are given by

$$\begin{aligned} A(s) &= 1 + a_1 s + \cdots + a_m s^m, \\ B(s) &= b_1 s + \cdots + b_m s^m, \\ C(s) &= 1 + c_1 s + \cdots + c_m s^m. \end{aligned}$$

The ARMAX model is very common and its derivation is outlined in Appendix F, where it is shown that without loss of generality we can assume that $C(s)$ *has no roots strictly inside the unit circle*. For much of the analysis in subsequent sections, it will be necessary to exclude the critical case where $C(s)$ has roots *on* the unit circle and assume that $C(s)$ has all its roots strictly outside the unit circle. This assumption is usually satisfied in practice.

In several situations, analysis and algorithms relating to the ARMAX model are greatly simplified if $C(s) = 1$ so that the noise terms $C(s)\epsilon_k = \epsilon_k$ are independent. However, this is typically an unrealistic assumption. To emphasize this point and see how easily the noise can be correlated, suppose that we have a first-order system

$$x_{k+1} = ax_k + w_k,$$

where we observe

$$y_k = x_k + v_k.$$

Then

$$\begin{aligned} y_{k+1} &= x_{k+1} + v_{k+1} \\ &= ax_k + w_k + v_{k+1} \\ &= a(y_k - v_k) + w_k + v_{k+1}, \end{aligned}$$

so finally

$$y_{k+1} = ay_k + v_{k+1} - av_k + w_k.$$

However, the noise sequence $\{v_{k+1} - av_k + w_k\}$ is correlated even if $\{v_k\}$ and $\{w_k\}$ are individually and mutually independent.

The ARMAX model (5.15) can be put into state space form. The process is based on state augmentation and can perhaps be best understood in terms of an example. Consider the system

$$y_k + a_1 y_{k-1} + a_2 y_{k-2} = b_1 u_{k-1} + b_2 u_{k-2} + \epsilon_k + c_1 \epsilon_{k-1}. \tag{5.16}$$

We have

$$\begin{pmatrix} y_{k+1} \\ y_k \\ u_k \\ \epsilon_{k+1} \end{pmatrix} = \begin{pmatrix} -a_1 & -a_2 & b_2 & c_1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} y_k \\ y_{k-1} \\ u_{k-1} \\ \epsilon_k \end{pmatrix} + \begin{pmatrix} b_1 \\ 0 \\ 1 \\ 0 \end{pmatrix} u_k + \begin{pmatrix} \epsilon_{k+1} \\ 0 \\ 0 \\ \epsilon_{k+1} \end{pmatrix}. \tag{5.17}$$

By setting

$$x_k = \begin{pmatrix} y_k \\ y_{k-1} \\ u_{k-1} \\ \epsilon_k \end{pmatrix}, \qquad w_k = \begin{pmatrix} \epsilon_{k+1} \\ 0 \\ 0 \\ \epsilon_{k+1} \end{pmatrix},$$

$$A = \begin{pmatrix} -a_1 & -a_2 & b_2 & c_1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} b_1 \\ 0 \\ 1 \\ 0 \end{pmatrix},$$

we can write Eq. (5.17) as

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

where $\{w_k\}$ is a stationary independent process. We have arrived at this state space model through state augmentation. Notice that the state x_k includes ϵ_k. Thus if the controller is assumed to know at time k only the present and past outputs $y_k, y_{k-1}, \ldots$, and past controls $u_{k-1}, u_{k-2}, \ldots$ (but not $\epsilon_k, \epsilon_{k-1}, \ldots$), we are faced with a model of imperfect state information. If $c_1 = 0$ in Eq. (5.16) then the state space model can be simplified so that

$$x_k = \begin{pmatrix} y_k \\ y_{k-1} \\ u_{k-1} \end{pmatrix},$$

in which case we have perfect state information. More generally, we have perfect state information in the ARMAX model (5.15) if $b_1 \neq 0$ and $c_1 = c_2 = \cdots = c_m = 0$.

Minimum Variance Control: Perfect State Information Case

Consider the perfect state information case of the ARMAX model (5.15):

$$y_k + a_1 y_{k-1} + \cdots + a_m y_{k-m} = b_1 u_{k-1} + \cdots + b_m u_{k-m} + \epsilon_k,$$

where $b_1 \neq 0$. A problem of interest, known as the *minimum variance control problem*, is to select u_k as a function of the present and past outputs $y_k, y_{k-1}, \ldots$, as well as the past controls $u_{k-1}, u_{k-2}, \ldots$, so as to minimize the cost

$$E\left\{\sum_{k=1}^{N} (y_k)^2\right\}.$$

There are no constraints on u_k. By transforming the system to state space form, we see that this problem can be reduced to a perfect state information linear-quadratic problem where the state x_k is

$$(y_k, y_{k-1}, \ldots, y_{k-m+1}, u_{k-1}, \ldots, u_{k-m+1})'.$$

The problem is of the same nature as the linear-quadratic problem of Section 4.1 except that the corresponding matrices R_k in the quadratic cost function are zero here. Nonetheless, in Section 4.1 we used the invertibility of R_k only to ensure that various matrices in the optimal policy and the Riccati equation are invertible. If invertibility of these matrices can be guaranteed by other means, the same analysis applies even if R_k is positive semidefinite. This is indeed the case here. An analysis analogous to the one of Section 4.1 shows that the optimal control u_k^* at time k (given $y_k, y_{k-1}, \ldots, y_{k-m+1}$ and $u_{k-1}, \ldots, u_{k-m+1}$) is the same as the one that would be applied if all future disturbances $\epsilon_{k+1}, \ldots, \epsilon_N$ were set equal to zero, their expected value (certainty equivalence). It follows that

$$\mu_k^*(y_k, \ldots, y_{k-m+1}, u_{k-1}, \ldots, u_{k-m+1}) = \frac{1}{b_1}(a_1 y_k + \cdots + a_m y_{k-m+1} - b_2 u_{k-1} - \cdots - b_m u_{k-m+1}),$$

and $\{u_k^*\}$ is generated via the equation

$$b_1 u_k^* + b_2 u_{k-1}^* + \cdots + b_m u_{k-m+1}^* = a_1 y_k + a_2 y_{k-1} + \cdots + a_m y_{k-m+1}.$$

In other words, $\{u_k^*\}$ is generated by passing $\{y_k\}$ through the linear filter $\overline{A}(s)/\overline{B}(s)$, where

$$\overline{A}(s) = a_1 + a_2 s + \cdots + a_m s^{m-1} = s^{-1}\big(A(s) - 1\big),$$

$$\overline{B}(s) = b_1 + b_2 s + \cdots + b_m s^{m-1} = s^{-1} B(s),$$

as shown in Fig. 5.3.1. The resulting closed-loop system is

$$y_k = \epsilon_k \tag{5.18}$$

and the associated cost is

$$N\, E\{(\epsilon_k)^2\}.$$

Notice that the optimal policy, called *minimum variance* control law, is time invariant and does not depend on the horizon N.

Whereas the optimal closed-loop system as given by Eq. (5.18) is clearly stable, we can anticipate serious difficulties if the filter $\overline{A}(s)/\overline{B}(s)$ in the feedback loop is unstable. For if $\overline{B}(s)$ has some roots inside the unit circle, then the sequence $\{u_k\}$ will tend to be unbounded. This is illustrated by the following example.

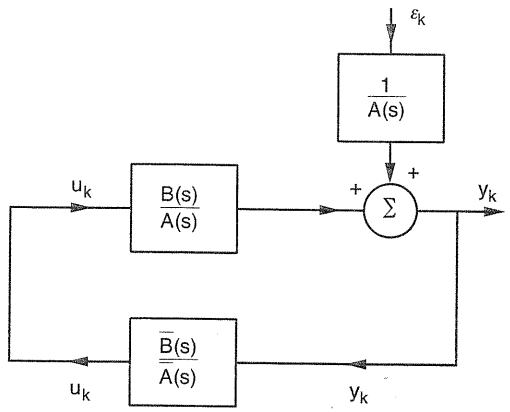


Figure 5.3.1 Minimum variance control with perfect state information. Structure of the optimal closed-loop system, where $A(s) = 1 + a_1 s + \cdots + a_m s^m$, $B(s) = b_1 s + \cdots + b_m s^m$, $\overline{A}(s) = s^{-1}\big(A(s) - 1\big)$, and $\overline{B}(s) = s^{-1}B(s)$.

Example 5.3.1 (An Optimal but Unstable Controller)

Consider the system

$$y_k + y_{k-1} = u_{k-1} - 2u_{k-2} + \epsilon_k.$$

The optimal policy is

$$u_k = 2u_{k-1} + y_k$$

and the optimal closed-loop system is

$$y_k = \epsilon_k,$$

which is a stable system. On the other hand, the last two equations yield

$$u_k = 2u_{k-1} + \epsilon_k.$$

Thus, u_k is generated by an *unstable* system, and in fact it is given by

$$u_k = \sum_{n=0}^{k} 2^n \epsilon_{k-n}.$$

Therefore, even though the output y_k stays bounded, the control u_k typically becomes unbounded.

For another view of the same difficulty, suppose that the coefficients $b_1, \ldots, b_m$ of $\overline{B}(s)$ are slightly different from the ones of the true system.

We will show that if the feedback filter $\overline{A}(s)/\overline{B}(s)$ is unstable, then the closed-loop system is also unstable in the sense that both u_k and y_k become unbounded with probability one.

Assume that the system is governed by

$$A^0(s)y_k = B^0(s)u_k + \epsilon_k, \tag{5.19}$$

while the policy is calculated under the assumption that the system model is

$$A(s)y_k = B(s)u_k + \epsilon_k,$$

where the coefficients of $A(s)$ and $B(s)$ differ slightly from those of $A^0(s)$, $B^0(s)$. Define $\overline{A}^0(s)$, $\overline{B}^0(s)$ by

$$1 + s\overline{A}^0(s) = A^0(s),$$

$$s\overline{B}^0(s) = B^0(s).$$

Note that $\overline{A}^0(s) = \overline{A}(s)$ and $\overline{B}^0(s) = \overline{B}(s)$ if $A^0(s) = A(s), B^0(s) = B(s)$. By multiplying Eq. (5.19) with $\overline{B}(s)$ and by using the relation defining the optimal policy

$$\overline{B}(s)u_k = \overline{A}(s)y_k,$$

we obtain

$$\overline{B}(s)A^0(s)y_k = B^0(s)\overline{A}(s)y_k + \overline{B}(s)\epsilon_k.$$

If the coefficients of $\overline{A}^0(s)$ and $\overline{B}^0(s)$ are close to those of $\overline{A}(s)$, $\overline{B}(s)$, then the roots of the polynomial

$$\overline{B}(s) + s\big(\overline{B}(s)\overline{A}^0(s) - \overline{B}^0(s)\overline{A}(s)\big)$$

are close to the roots of $\overline{B}(s)$. Thus *the closed-loop system is stable only if the roots of* $\overline{B}(s)$ *are outside the unit circle*, or equivalently, if and only if the filter $\overline{A}(s)/\overline{B}(s)$ is stable. If our model is exact, the closed-loop system will be stable due to what is commonly referred to as a *pole-zero cancellation*. However, the slightest modeling discrepancy will induce instability.

The conclusion from the preceding analysis is that the minimum variance control law is advisable only if it can be realized through a stable filter [$\overline{B}(s)$ has roots outside the unit circle]. Even if $\overline{B}(s)$ has its roots outside the unit circle, but some of these roots are near the unit circle, the performance of the minimum variance policy can be very sensitive to variations in the parameters of the polynomials $A(s)$ and $B(s)$. One way to overcome this sensitivity is to change the cost to

$$E\left\{\sum_{k=1}^{N}\big((y_k)^2 + R(u_{k-1})^2\big)\right\},$$

where R is some positive parameter. This requires solution via the Riccati equation as in Section 4.1. For a detailed derivation, see Aström [Ast83].

In some problems, the system equation includes an additional external input sequence $\{v_k\}$, the values of which can be measured by the controller as they occur. In particular, consider the scalar system

$$y_k + a_1 y_{k-1} + \cdots + a_m y_{k-m} = b_1 u_{k-1} + \cdots + b_m u_{k-m} + d_1 v_{k-1} + \cdots + d_m v_{k-m} + \epsilon_k,$$

where each value v_k becomes known to the controller without error at time k. The minimum variance controller then takes the form

$$\mu_k^*(y_k, \ldots, y_{k-m+1}, u_{k-1}, \ldots, u_{k-m+1}, v_k, \ldots, v_{k-m+1}) = \frac{1}{b_1}\big(a_1 y_k + \cdots + a_m y_{k-m+1} - d_1 v_k - \cdots - d_m v_{k-m+1} - b_2 u_{k-1} \cdots - b_m u_{k-m+1}\big),$$

and the optimal controls $\{u_k^*\}$ are generated by

$$\overline{B}(s) u_k^* = \overline{A}(s) y_k - \overline{D}(s) v_k,$$

where

$$\overline{A}(s) = a_1 + a_2 s + \cdots + a_m s^{m-1},$$
$$\overline{B}(s) = b_1 + b_2 s + \cdots + b_m s^{m-1},$$
$$\overline{D}(s) = d_1 + d_2 s + \cdots + d_m s^{m-1}.$$

The closed-loop system is again $y_k = \epsilon_k$, but for practical purposes it is stable only if $\overline{B}(s)$ has its roots outside the unit circle. The process whereby external inputs are measured and used for control is commonly referred to as *feedforward control*.

Imperfect State Information Case

Consider now the general ARMAX model

$$y_k + a_1 y_{k-1} + \cdots + a_m y_{k-m} = b_M u_{k-M} + \cdots + b_m u_{k-m} + \epsilon_k + c_1 \epsilon_{k-1} + \cdots + c_m \epsilon_{k-m}$$

or, equivalently,

$$A(s) y_k = B(s) u_k + C(s) \epsilon_k,$$

where

$$A(s) = 1 + a_1 s + \cdots + a_m s^m,$$
$$B(s) = b_M s^M + \cdots + b_m s^m,$$
$$C(s) = 1 + c_1 s + \cdots + c_m s^m.$$

We assume the following:

(1) $b_M \neq 0$ and $1 \leq M \leq m$.

(2) $\{\epsilon_k\}$ is a zero mean, independent, stationary process.

(3) The polynomial $C(s)$ has all its roots outside the unit circle. (As explained in Appendix F, this assumption is not overly restrictive.)

The controller knows at each time k the past inputs and outputs. Thus the information vector at time k is

$$I_k = \big(y_k, y_{k-1}, \ldots, y_{-m+1}, u_{k-1}, u_{k-2}, \ldots, u_{-m+M}\big).$$

(We include in the information vector the control inputs $u_{-1}, \ldots, u_{-m+M}$. If control starts at time 0, these inputs will be zero.) There are no constraints on u_k. The problem is to find a policy $\{\mu_0(I_0), \ldots, \mu_{N-1}(I_{N-1})\}$ that minimizes

$$E\left\{\sum_{k=1}^{N} (y_k)^2\right\}.$$

By using state augmentation, we can cast this problem into the framework of the linear-quadratic problem of Section 5.2. The corresponding linear system in state space format involves a state x_k given by

$$x_k = (y_{k+M-1}, \ldots, y_{k+M-m}, u_{k-1}, \ldots, u_{k+M-m}, \epsilon_{k+M-1}, \ldots, \epsilon_{k+M-m}).$$

Because $y_{k+M-1}, \ldots, y_{k+1}$ and $\epsilon_{k+M-1}, \ldots, \epsilon_{k+M-m}$ are unknown to the controller, we are faced with a problem of imperfect state information.

An analysis analogous to the one of Section 5.2 shows that certainty equivalence holds; that is, the *optimal control u_k^* at time k given I_k is the same as the one that would be applied in the deterministic problem where the current state*

$$x_k = (y_{k+M-1}, \ldots, y_{k+M-m}, u_{k-1}, \ldots, u_{k+M-m}, \epsilon_{k+M-1}, \ldots, \epsilon_{k+M-m})$$

is set equal to its conditional expected value given I_k, and the future disturbances $\epsilon_{k+M}, \ldots, \epsilon_N$ are set equal to zero (their expected value).

Thus the optimal control $u_k^* = \mu_k^*(I_k)$ is obtained by solving for u_k the equation

$$E\{y_{k+M} \mid u_k, I_k\} = E\{y_{k+M} \mid y_k, y_{k-1}, \ldots, y_{-m+1}, u_k, u_{k-1}, \ldots, u_{-m+M}\} = 0.$$

This leads to the problem of calculating $E\{y_{k+M} \mid I_k, u_k\}$, known as the *forecasting* or *prediction* problem, which is important in its own right. We first treat the easier case where there is no delay ($M = 1$) and then discuss the more general case where the delay can be positive.

Forecasting for ARMAX Models – No Delay ($M = 1$)

Assume that $M = 1$. We would like to generate an equation for the forecast $E\{y_{k+1} \mid I_k, u_k\}$, and then determine the optimal control $u_k^* = \mu_k^*(I_k)$ by setting this forecast to zero. Let us introduce an auxiliary sequence $\{z_k\}$ via the equation

$$z_k = y_k - \epsilon_k.$$

A key fact is that, since $\{\epsilon_k\}$ is an independent, zero-mean sequence, we have

$$E\{z_{k+1} \mid I_k, u_k\} = E\{y_{k+1} \mid I_k, u_k\}.$$

We can thus obtain the desired forecast of y_{k+1} by forecasting z_{k+1} instead. We can then obtain the optimal control u_k^* by setting $E\{z_{k+1} \mid I_k, u_k^*\} = 0$.

By using the definition $z_k = y_k - \epsilon_k$ to express y_k in terms of z_k in the ARMAX model equation for $M = 1$, we obtain

$$z_{k+1} + c_1 z_k + \cdots + c_m z_{k-m+1} = b_1 u_k + \cdots + b_m u_{k-m+1} + w_k, \quad (5.20)$$

where

$$w_k = (c_1 - a_1)y_k + \cdots + (c_m - a_m)y_{k-m+1}.$$

We note that w_k is perfectly observable by the controller; however, the scalars $z_k, \ldots, z_{k-m+1}$ are not known to the controller because the initial conditions $z_0, \ldots, z_{1-m}$ of the system (5.20) are unknown. Nonetheless, the system (5.20) is stable, since the roots of the polynomial $C(s)$ have been assumed to be outside the unit circle. As a result, the initial conditions do not matter asymptotically. In other words, if we generate a sequence $\{\hat{y}_k\}$ using the system (5.20) and zero initial conditions, i.e.,

$$\hat{y}_{k+1} + c_1 \hat{y}_k + \cdots + c_m \hat{y}_{k-m+1} = b_1 u_k + \cdots + b_m u_{k-m+1} + w_k,$$

with

$$\hat{y}_0 = 0, \qquad \hat{y}_{-1} = 0, \qquad \ldots \qquad \hat{y}_{1-m} = 0,$$

then we will have

$$\lim_{k\to\infty} (\hat{y}_k - z_k) = 0.$$

Thus, $\hat{y}_{k+1}$ is an asymptotically accurate approximation to the optimal forecast $E\{y_{k+1} \mid I_k, u_k\}$.

Minimum Variance Control: Imperfect State Information and No Delay

Based on the earlier discussion, an asymptotically accurate approximation to the minimum variance policy is obtained by setting u_k to the value that makes $\hat{y}_{k+1} = 0$; that is, by solving for u_k the equation

$$\hat{y}_{k+1} + c_1 \hat{y}_k + \cdots + c_m \hat{y}_{k-m+1} = b_1 u_k + \cdots + b_m u_{k-m+1} + w_k.$$

If this policy is followed, however, the earlier forecasts $\hat{y}_k, \ldots, \hat{y}_{k-m+1}$ will be equal to zero. Thus the (approximate) minimum variance policy is given by

$$\begin{aligned} u_k &= \frac{1}{b_1}(w_k - b_2 u_{k-1} - \cdots - b_m u_{k-m+1}) \\ &= \frac{1}{b_1}\big((a_1 - c_1)y_k + \cdots + (a_m - c_m)y_{k-m-1} \\ &\qquad - b_2 u_{k-1} - \cdots - b_m u_{k-m+1}\big). \end{aligned}$$

By substituting this policy in the ARMAX model

$$\begin{aligned} y_{k+1} + a_1 y_k + \cdots + a_m y_{k-m+1} = b_1 u_k + \cdots + b_m u_{k-m+1} \\ + \epsilon_{k+1} + c_1 \epsilon_k + \cdots + c_m \epsilon_{k-m+1}, \end{aligned}$$

we see that the closed-loop system becomes

$$y_{k+1} - \epsilon_{k+1} + c_1(y_k - \epsilon_k) + \cdots + c_m(y_{k-m+1} - \epsilon_{k-m+1}) = 0,$$

or equivalently $C(s)(y_k - \epsilon_k) = 0$. Since $C(s)$ has its roots outside the unit circle, this is a stable system, and we have

$$y_k = \epsilon_k + \gamma(k),$$

where $\gamma(k) \to 0$ as $k \to \infty$.

Forecasting: The General Case

Consider now the general case where the delay M can be greater than 1. The forecasting problem can still be nicely solved by using a certain trick to transform the ARMAX equation into a more convenient form. To this end, we first obtain polynomials $F(s)$ and $G(s)$ of the form

$$F(s) = 1 + f_1 s + \cdots + f_{M-1} s^{M-1},$$

$$G(s) = g_0 + g_1 s + \cdots + g_{m-1} s^{m-1},$$

which satisfy

$$C(s) = A(s)F(s) + s^M G(s). \quad (5.21)$$

The coefficients of $F(s)$ and $G(s)$ are uniquely determined from those of $C(s)$ and $A(s)$ by equating coefficients of both sides of the relation

$$\begin{aligned} 1 + c_1 s + \cdots + c_m s^m = (1 + a_1 s + \cdots + a_m s^m)(1 + f_1 s + \cdots + f_{M-1} s^{M-1}) \\ + s^M (g_0 + g_1 s + \cdots + g_{m-1} s^{m-1}). \end{aligned}$$

Example 5.3.2

Let $m = 3$ and $M = 2$. Then the preceding equation takes the form

$$1 + c_1 s + c_2 s^2 + c_3 s^3 = (1 + a_1 s + a_2 s^2 + a_3 s^3)(1 + f_1 s) + s^2(g_0 + g_1 s + g_2 s^2),$$

and by equating coefficients we have

$$c_1 = a_1 + f_1, \quad c_2 = a_2 + a_1 f_1 + g_0, \quad c_3 = a_3 + a_2 f_1 + g_1, \quad a_3 f_1 + g_2 = 0,$$

from which f_1, g_0, g_1, and g_2 are uniquely determined.

The ARMAX model can be written as

$$A(s)y_{k+M} = \overline{B}(s)u_k + C(s)\epsilon_{k+M}, \tag{5.22}$$

where

$$\overline{B}(s) = s^{-M}B(s) = b_M + b_{M+1}s + \cdots + b_m s^{m-M}.$$

Multiplying both sides of Eq. (5.22) with $F(s)$, we have

$$F(s)A(s)y_{k+M} = F(s)\overline{B}(s)u_k + F(s)C(s)\epsilon_{k+M},$$

and using Eq. (5.21) to express $F(s)A(s)$ as $C(s) - s^M G(s)$, we obtain

$$\big(C(s) - s^M G(s)\big)y_{k+M} = F(s)\overline{B}(s)u_k + F(s)C(s)\epsilon_{k+M},$$

or equivalently

$$C(s)\big(y_{k+M} - F(s)\epsilon_{k+M}\big) = F(s)\overline{B}(s)u_k + G(s)y_k. \tag{5.23}$$

Let us now introduce the auxiliary sequence $\{z_k\}$ via the equation

$$z_{k+M} = y_{k+M} - F(s)\epsilon_{k+M} = y_{k+M} - \epsilon_{k+M} - f_1\epsilon_{k+M-1} - \cdots - f_{M-1}\epsilon_{k+1}.$$

Note that when $M = 1$, we have $F(s) = 1$ and $z_k = y_k - \epsilon_k$, so $\{z_k\}$ is the same sequence as the one introduced earlier for the case of no delay. Again, since $\{\epsilon_k\}$ is an independent, zero-mean sequence, by taking expectations in the definition $z_{k+M} = y_{k+M} - F(s)\epsilon_{k+M}$, we obtain

$$E\{z_{k+M} \mid I_k, u_k\} = E\{y_{k+M} \mid I_k, u_k\},$$

and we can obtain the desired forecast of y_{k+M} by forecasting z_{k+M} in its place. Furthermore, by Eq. (5.23), z_{k+M} is written as

$$C(s)z_{k+M} = w_k$$

or

$$z_{k+M} + c_1 z_{k+M-1} + \cdots + c_m z_{k+M-m} = w_k, \tag{5.24}$$

where

$$w_k = F(s)\overline{B}(s)u_k + G(s)y_k. \tag{5.25}$$

Since the scalar w_k of Eq. (5.25) is available at time k (i.e., it is determined from I_k and u_k), the system (5.24) can serve as a basis for forecasting z_{k+M}. We would be able to predict exactly z_{k+M} and use it as a forecast of y_{k+M} if we knew appropriate initial conditions with which to start the equation (5.24) that generates it. We don't know such initial conditions, but because this equation represents a stable system, the choice of initial conditions does not matter asymptotically, as we proceed to explain more formally.

We consider the sequence $\hat{y}_{k+M}$ generated by

$$\hat{y}_{k+M} + c_1\hat{y}_{k+M-1} + \cdots + c_m\hat{y}_{k+M-m} = w_k$$

with initial condition

$$\hat{y}_{M-1} = \hat{y}_{M-2} = \cdots = \hat{y}_{M-m} = 0, \tag{5.26}$$

and we claim that the forecast $E\{z_{k+M} \mid I_k\}$ can be approximated by $\hat{y}_{k+M}$. To see this, note that from Eqs. (5.24) to (5.26) we have

$$z_{k+M} = \hat{y}_{k+M} + \big(\gamma_1(k)z_{M-1} + \cdots + \gamma_m(k)z_{M-m}\big)$$

and

$$E\{z_{k+M} \mid I_k, u_k\} = \hat{y}_{k+M} + \sum_{i=1}^{m}\gamma_i(k)E\{z_{M-i} \mid I_k, u_k\},$$

where $\gamma_1(k), \ldots, \gamma_m(k)$ are appropriate scalars depending on k. Since $C(s)$ has all its roots outside the unit circle, we have (compare with the discussion on stability earlier in this section)

$$\lim_{k\to\infty}\gamma_1(k) = \lim_{k\to\infty}\gamma_2(k) = \cdots = \lim_{k\to\infty}\gamma_m(k) = 0.$$

It follows that, for large values of k,

$$\hat{y}_{k+M} \simeq E\{z_{k+M} \mid I_k, u_k\} = E\{y_{k+M} \mid I_k, u_k\}.$$

(More precisely, we have $|\hat{y}_{k+M} - E\{y_{k+M} \mid I_k, u_k\}| \to 0$ as $k \to \infty$, where the convergence is in the mean-square sense.)

In conclusion, an asymptotically accurate approximation to the optimal forecast $E\{y_{k+M} \mid I_k, u_k\}$ is given by $\hat{y}_{k+M}$ and is generated by the equation

$$\hat{y}_{k+M} + c_1\hat{y}_{k+M-1} + \cdots + c_m\hat{y}_{k+M-m} = F(s)\overline{B}(s)u_k + G(s)y_k \tag{5.27}$$

with the initial condition

$$\hat{y}_{M-1} = \hat{y}_{M-2} = \cdots = \hat{y}_{M-m} = 0. \tag{5.28}$$

Minimum Variance Control: The General Case

Based on the earlier discussion, the minimum variance policy is obtained by solving for u_k the equation $E\{y_{k+M} \mid I_k, u_k\} = 0$. Thus an asymptotically accurate approximation is obtained by setting u_k to the value that makes $\hat{y}_{k+M} = 0$, that is, by solving for u_k the equation [cf. Eqs. (5.27) and (5.28)]

$$F(s)\overline{B}(s)u_k + G(s)y_k = c_1\hat{y}_{k+M-1} + \cdots + c_m\hat{y}_{k+M-m}.$$

If this policy is followed, however, the earlier forecasts $\hat{y}_{k+M-1}, \ldots, \hat{y}_{k+M-m}$ will be equal to zero. Thus the (approximate) minimum variance policy is given by

$$F(s)\overline{B}(s)u_k + G(s)y_k = 0; \tag{5.29}$$

that is, u_k^* is generated by passing y_k through the linear filter

$$-G(s)/F(s)\overline{B}(s),$$

as shown in Fig. 5.3.2.

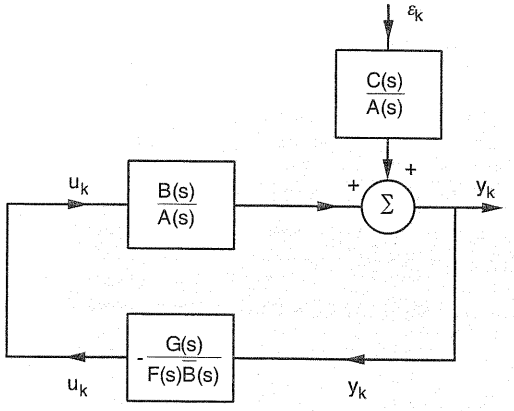


Figure 5.3.2 Minimum variance control with imperfect state information. Structure of the optimal closed-loop system.

From Eqs. (5.23) and (5.29), we obtain the equation for the closed-loop system

$$C(s)\big(y_{k+M} - F(s)\epsilon_{k+M}\big) = 0.$$

Since $C(s)$ has its roots outside the unit circle, we obtain

$$y_{k+M} = F(s)\epsilon_{k+M} + \gamma(k),$$

where $\gamma(k) \to 0$ as $k \to \infty$. So asymptotically, the closed-loop system takes the form

$$y_k = \epsilon_k + f_1\epsilon_{k-1} + \cdots + f_{M-1}\epsilon_{k-M+1}.$$

Let us consider now the stability properties of the closed-loop system when the true system parameters differ slightly from those of the assumed model. Let the true system be described by

$$A^0(s)y_k = s^M\overline{B}^0(s)u_k + C^0(s)\epsilon_k, \tag{5.30}$$

while u_k is given by the minimum variance policy

$$F(s)\overline{B}(s)u_k + G(s)y_k = 0, \tag{5.31}$$

where

$$C(s) = A(s)F(s) + s^M G(s).$$

Operating on Eq. (5.30) with $F(s)\overline{B}(s)$ and using Eq. (5.31), we obtain

$$F(s)\overline{B}(s)A^0(s)y_k = -s^M\overline{B}^0(s)G(s)y_k + F(s)\overline{B}(s)C^0(s)\epsilon_k.$$

Combining the last two equations and collecting terms, we have

$$\Big\{F(s)\overline{B}(s)A^0(s) + \big(C(s) - A(s)F(s)\big)\overline{B}^0(s)\Big\}y_k = F(s)\overline{B}(s)C^0(s)\epsilon_k$$

or

$$\Big\{\overline{B}^0(s)C(s) + F(s)\big(\overline{B}(s)A^0(s) - A(s)\overline{B}^0(s)\big)\Big\}y_k = F(s)\overline{B}(s)C^0(s)\epsilon_k.$$

If the coefficients of $A^0(s)$, $\overline{B}^0(s)$, and $C^0(s)$ are near those of $A(s)$, $\overline{B}(s)$, and $C(s)$, then the poles of the closed-loop system will be near the roots of $\overline{B}(s)C(s)$. Thus *the closed-loop system will be in effect stable only if the roots of $\overline{B}(s)$ are strictly outside the unit circle*, similar to the perfect state information case examined earlier.

5.4 SUFFICIENT STATISTICS

The main difficulty with the DP algorithm for imperfect state information problems is that it is carried out over a state space of expanding dimension. As a new measurement is added at each stage k, the dimension of the state (the information vector I_k) increases accordingly. This motivates an effort to reduce the data that are truly necessary for control purposes. In other words, it is of interest to look for quantities known as *sufficient statistics*,

which ideally would be of smaller dimension than I_k and yet summarize all the essential content of I_k as far as control is concerned.

Consider the DP algorithm (5.4) and (5.5), restated here for convenience:

$$J_{N-1}(I_{N-1}) = \min_{u_{N-1}\in U_{N-1}} \Big[\underset{x_{N-1},w_{N-1}}{E} \big\{ g_N\big(f_{N-1}(x_{N-1},u_{N-1},w_{N-1})\big) + g_{N-1}(x_{N-1},u_{N-1},w_{N-1}) \mid I_{N-1},u_{N-1} \big\} \Big], \tag{5.32}$$

$$J_k(I_k) = \min_{u_k\in U_k} \left[\underset{x_k,w_k,z_{k+1}}{E} \big\{ g_k(x_k,u_k,w_k) + J_{k+1}(I_k,z_{k+1},u_k) \mid I_k,u_k \big\} \right]. \tag{5.33}$$

Suppose that we can find a function $S_k(I_k)$ of the information vector, such that a minimizing control in Eqs. (5.32) and (5.33) depends on I_k via $S_k(I_k)$. By this we mean that the minimization in the right-hand side of the DP algorithm (5.32) and (5.33) can be written in terms of some function H_k as

$$\min_{u_k\in U_k} H_k\big(S_k(I_k),u_k\big).$$

Such a function S_k is called a *sufficient statistic*. Its salient feature is that an optimal policy obtained by the preceding minimization can be written as

$$\mu_k^*(I_k) = \overline{\mu}_k\big(S_k(I_k)\big),$$

where $\overline{\mu}_k$ is an appropriate function. Thus, if the sufficient statistic is characterized by a set of fewer numbers than the information vector I_k, it may be easier to implement the policy in the form $u_k = \overline{\mu}_k\big(S_k(I_k)\big)$ and take advantage of the resulting data reduction.

5.4.1 The Conditional State Distribution

There are many different functions that can serve as sufficient statistics. The identity function $S_k(I_k) = I_k$ is certainly one of them. In this section, we will derive another important sufficient statistic: the conditional probability distribution $P_{x_k|I_k}$ of the state x_k, given the information vector I_k. An extra assumption is necessary for this, namely that *the probability distribution of the observation disturbance v_{k+1} depends explicitly only on the immediately preceding state, control, and system disturbance x_k, u_k, w_k, and not on $x_{k-1},\ldots,x_0$, $u_{k-1},\ldots,u_0$, $w_{k-1}\ldots,w_0$, $v_{k-1},\ldots,v_0$.* Under this assumption, we will show that for all k and I_k, we have

$$J_k(I_k) = \min_{u_k\in U_k} H_k\big(P_{x_k|I_k},u_k\big) = \overline{J}_k(P_{x_k|I_k}), \tag{5.34}$$

where H_k and $\overline{J}_k$ are appropriate functions.

To this end, we will use an important fact that relates to state estimation of discrete-time stochastic systems: the conditional distribution $P_{x_k|I_k}$ can be generated recursively by an equation of the form

$$P_{x_{k+1}|I_{k+1}} = \Phi_k\big(P_{x_k|I_k},u_k,z_{k+1}\big), \tag{5.35}$$

where Φ_k is some function that can be determined from the data of the problem. Let us postpone a justification of this for the moment, and accept it for the purpose of the following discussion.

We note that to perform the minimization in Eq. (5.32), it is sufficient to know the distribution $P_{x_{N-1}|I_{N-1}}$ together with the distribution $P_{w_{N-1}|x_{N-1},u_{N-1}}$, which is part of the problem data. Thus, the minimization in the right-hand side of Eq. (5.32) is of the form

$$J_{N-1}(I_{N-1}) = \min_{u_{N-1}\in U_{N-1}} H_{N-1}\big(P_{x_{N-1}|I_{N-1}},u_{N-1}\big) = \overline{J}_{N-1}(P_{x_{N-1}|I_{N-1}})$$

for appropriate functions H_{N-1} and $\overline{J}_{N-1}$.

We now use induction, i.e., we assume that

$$J_{k+1}(I_{k+1}) = \min_{u_{k+1}\in U_{k+1}} H_{k+1}\big(P_{x_{k+1}|I_{k+1}},u_{k+1}\big) = \overline{J}_{k+1}(P_{x_{k+1}|I_{k+1}}), \tag{5.36}$$

for appropriate functions H_{k+1} and $\overline{J}_{k+1}$, and we show that

$$J_k(I_k) = \min_{u_k\in U_k} H_k\big(P_{x_k|I_k},u_k\big) = \overline{J}_k(P_{x_k|I_k}), \tag{5.37}$$

for appropriate functions H_k and $\overline{J}_k$.

Indeed, using Eqs. (5.35) and (5.36), the DP equation (5.33) is written as

$$J_k(I_k) = \min_{u_k\in U_k} E\Big\{ g_k(x_k,u_k,w_k) + \overline{J}_{k+1}\big(\Phi_k(P_{x_k|I_k},u_k,z_{k+1})\big) \mid I_k,u_k \Big\}. \tag{5.38}$$

In order to calculate the expression being minimized over u_k above, we need, in addition to $P_{x_k|I_k}$, the joint distribution

$$P(x_k,w_k,z_{k+1} \mid I_k,u_k)$$

or, equivalently,

$$P\big(x_k,w_k,h_{k+1}\big(f_k(x_k,u_k,w_k),u_k,v_{k+1}\big) \mid I_k,u_k\big).$$

This distribution can be expressed in terms of $P_{x_k|I_k}$, the given distributions

$$P(w_k \mid x_k,u_k), \qquad P\big(v_{k+1} \mid f_k(x_k,u_k,w_k),u_k,w_k\big),$$

and the system equation $x_{k+1} = f_k(x_k, u_k, w_k)$. Therefore the expression minimized over u_k in Eq. (5.38) can be written as a function of $P_{x_k|I_k}$ and u_k, and the DP equation (5.33) can be written as

$$J_k(I_k) = \min_{u_k \in U_k} H_k\big(P_{x_k|I_k}, u_k\big)$$

for a suitable function H_k. Thus the induction is complete and it follows that the distribution $P_{x_k|I_k}$ is a sufficient statistic.

Note that if the conditional distribution $P_{x_k|I_k}$ is uniquely determined by another expression $S_k(I_k)$, i.e.,

$$P_{x_k|I_k} = G_k\big(S_k(I_k)\big)$$

for an appropriate function G_k, then $S_k(I_k)$ is also a sufficient statistic. Thus, for example, if we can show that $P_{x_k|I_k}$ is a Gaussian distribution, then the mean and the covariance matrix corresponding to $P_{x_k|I_k}$ form a sufficient statistic.

Regardless of its computational value, the representation of the optimal policy as a sequence of functions of the conditional probability distribution $P_{x_k|I_k}$,

$$\mu_k(I_k) = \overline{\mu}_k(P_{x_k|I_k}), \qquad k = 0, 1, \ldots, N-1,$$

is conceptually very useful. It provides a decomposition of the optimal controller in two parts:

(a) An *estimator*, which uses at time k the measurement z_k and the control u_{k-1} to generate the probability distribution $P_{x_k|I_k}$.

(b) An *actuator*, which generates a control input to the system as a function of the probability distribution $P_{x_k|I_k}$ (Fig. 5.4.1).

This interpretation has formed the basis for various suboptimal control schemes that separate the controller a priori into an estimator and an actuator and attempt to design each part in a manner that seems "reasonable." Schemes of this type will be discussed in Chapter 6.

The Conditional State Distribution Recursion

There remains to justify the recursion

$$P_{x_{k+1}|I_{k+1}} = \Phi_k\big(P_{x_k|I_k}, u_k, z_{k+1}\big). \tag{5.39}$$

Let us first give an example.

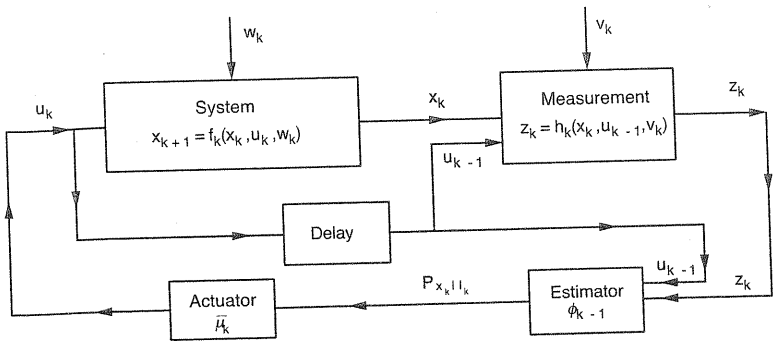


Figure 5.4.1 Conceptual separation of the optimal controller into an estimator and an actuator.

Example 5.4.1 (Search Problem)

In a classical problem of search, one has to decide at each period whether to search a site that may contain a treasure. If a treasure is present, the search reveals it with probability β, in which case the treasure is removed from the site. Here the state has two values: either a treasure is present in the site or it is not. The control u_k takes two values: search and not search. If the site is searched, the observation z_{k+1} takes two values, treasure found or not found, while if the site is not searched, the value of z_{k+1} is irrelevant.

Denote

p_k : probability a treasure is present at the beginning of period k.

This probability evolves according to the equation

$$p_{k+1} = \begin{cases} p_k & \text{if the site is not searched at time } k, \\ 0 & \text{if the site is searched and a treasure is found,} \\ \dfrac{p_k(1-\beta)}{p_k(1-\beta)+1-p_k} & \text{if the site is searched but no treasure is found.} \end{cases}$$

The second relation holds because the treasure is removed after a successful search. The third relation follows by application of Bayes' rule (p_{k+1} is equal to the kth period probability of a treasure being present *and* the search being unsuccessful, divided by the probability of an unsuccessful search). The preceding equation defines the desired recursion for the conditional distribution of the state and is a special case of Eq. (5.39).

The general form of the recursion

$$P_{x_{k+1}|I_{k+1}} = \Phi_k\big(P_{x_k|I_k}, u_k, z_{k+1}\big)$$

is developed in Exercise 5.7 for the case where the state, control, observation, and disturbance spaces are finite sets. In the case where these spaces

are the real line and all random variables involved possess probability density functions, the conditional density $p(x_{k+1} \mid I_{k+1})$ is generated from $p(x_k \mid I_k)$, u_k, and z_{k+1} by means of the equation

$$\begin{aligned} p(x_{k+1} \mid I_{k+1}) &= p(x_{k+1} \mid I_k, u_k, z_{k+1}) \\ &= \frac{p(x_{k+1}, z_{k+1} \mid I_k, u_k)}{p(z_{k+1} \mid I_k, u_k)} \\ &= \frac{p(x_{k+1} \mid I_k, u_k) p(z_{k+1} \mid I_k, u_k, x_{k+1})}{\int_{-\infty}^{\infty} p(x_{k+1} \mid I_k, u_k) p(z_{k+1} \mid I_k, u_k, x_{k+1}) dx_{k+1}}. \end{aligned}$$

In this equation all the probability densities appearing in the right-hand side may be expressed in terms of $p(x_k \mid I_k)$, u_k, and z_{k+1}. In particular, the density $p(x_{k+1} \mid I_k, u_k)$ may be expressed through $p(x_k \mid I_k)$, u_k, and the system equation $x_{k+1} = f_k(x_k, u_k, w_k)$ using the given density $p(w_k \mid x_k, u_k)$ and the relation

$$p(w_k \mid I_k, u_k) = \int_{-\infty}^{\infty} p(x_k \mid I_k) p(w_k \mid x_k, u_k) dx_k.$$

Similarly, the density $p(z_{k+1} \mid I_k, u_k, x_{k+1})$ is expressed through the measurement equation $z_{k+1} = h_{k+1}(x_{k+1}, u_k, v_{k+1})$ using the densities

$$p(x_k \mid I_k), \quad p(w_k \mid x_k, u_k), \quad p(v_{k+1} \mid x_k, u_k, w_k).$$

By substituting these expressions in the equation for $p(x_{k+1} \mid I_{k+1})$, we obtain a dynamic system equation for the conditional state distribution of the desired form. Other similar examples will be given in subsequent sections. A mathematically rigorous substantiation of the recursion $P_{x_{k+1}|I_{k+1}} = \Phi_k(P_{x_k|I_k}, u_k, z_{k+1})$ can be found in Bertsekas and Shreve [BeS78].

Alternative Perfect State Information Reduction

Finally, let us formally rewrite the DP algorithm in terms of the sufficient statistic $P_{x_k|I_k}$. Using Eqs. (5.35), (5.37), and (5.38), we have for $k < N-1$

$$\overline{J}_k(P_{x_k|I_k}) = \min_{u_k \in U_k} \left[\underset{x_k, w_k, z_{k+1}}{E} \left\{ g_k(x_k, u_k, w_k) + \overline{J}_{k+1}\big(\Phi_k(P_{x_k|I_k}, u_k, z_{k+1})\big) \mid I_k, u_k \right\} \right]. \tag{5.40}$$

In the case where $k = N-1$, we have

$$\overline{J}_{N-1}(P_{x_{N-1}|I_{N-1}}) = \min_{u_{N-1} \in U_{N-1}} \left[\underset{x_{N-1}, w_{N-1}}{E} \left\{ g_N\big(f_{N-1}(x_{N-1}, u_{N-1}, w_{N-1})\big) + g_{N-1}(x_{N-1}, u_{N-1}, w_{N-1}) \mid I_{N-1}, u_{N-1} \right\} \right]. \tag{5.41}$$

This DP algorithm yields the optimal cost as

$$J^* = \underset{z_0}{E}\big\{\overline{J}_0(P_{x_0|z_0})\big\},$$

where $\overline{J}_0$ is obtained by the last step, and the probability distribution of z_0 is obtained from the measurement equation $z_0 = h_0(x_0, v_0)$ and the distributions of x_0 and v_0.

By observing the form of Eq. (5.40), we note that it has the standard DP structure, except that $P_{x_k|I_k}$ plays the role of the "state." Indeed the role of the "system" is played by the recursive estimator of $P_{x_k|I_k}$,

$$P_{x_{k+1}|I_{k+1}} = \Phi_k\big(P_{x_k|I_k}, u_k, z_{k+1}\big),$$

and this system fits the framework of the basic problem (the role of control is played by u_k and the role of the disturbance is played by z_{k+1}). Furthermore, the controller can calculate (at least in principle) the state $P_{x_k|I_k}$ of this system at time k, so perfect state information prevails. Thus the alternate DP algorithm (5.40)-(5.41) may be viewed as the DP algorithm of the perfect state information problem that involves the above system, whose state is $P_{x_k|I_k}$, and an appropriately reformulated cost function. In the absence of perfect knowledge of the state, *the controller can be viewed as controlling the "probabilistic state" $P_{x_k|I_k}$ so as to minimize the expected cost-to-go conditioned on the information I_k available.*

Example 5.4.1 (Continued)

Let us write the DP algorithm (5.40) for the search problem of Example 5.4.1, assuming that the treasure's worth is V, that each search costs C, and that once we decide not to search at a particular time, then we cannot search at future times. The algorithm takes the form

$$\overline{J}_k(p_k) = \max\left[0, -C + p_k\beta V + (1 - p_k\beta)\overline{J}_{k+1}\left(\frac{p_k(1-\beta)}{p_k(1-\beta) + 1 - p_k}\right)\right],$$

with $\overline{J}_N(p_N) = 0$. From this algorithm, it is straightforward to show by induction that the functions $\overline{J}_k$ satisfy $\overline{J}_k(p_k) = 0$ for all $p_k \leq C/(\beta V)$, and that it is optimal to search at period k if and only if

$$C \leq p_k\beta V.$$

Thus, it is optimal to search if and only if the expected reward from the next search is greater or equal to the cost of the search.

5.4.2 Finite-State Systems

We will now consider systems that are stationary finite-state Markov chains, in which case the conditional probability distribution $P_{x_k|I_k}$ is characterized by a finite set of numbers. The states are denoted $1, 2, \ldots, n$. When a control u is applied, the system moves from state i to state j with probability $p_{ij}(u)$. The control u is chosen from a finite set U. Following a state transition, an observation is made by the controller. There is a finite number of possible observation outcomes, and the probability of each depends on the current state and the preceding control. The information available to the controller at stage k is the information vector

$$I_k = (z_1, \ldots, z_k, u_0, \ldots, u_{k-1}),$$

where for all i, z_i and u_i are the observation and control at stage i, respectively. Following the observation z_k, a control u_k is chosen by the controller, and a cost $g(x_k, u_k)$ is incurred, where x_k is the current (hidden) state. The terminal cost for being at state x at the end of the N stages is denoted $G(x)$. We wish to minimize the expected value of the sum of costs incurred over the N stages.

As discussed in Section 5.4.1, one can reformulate the problem into a problem of perfect state information: the objective is to control the column vector of conditional probabilities

$$p_k = (p_k^1, \ldots, p_k^n)',$$

where

$$p_k^i = P(x_k = i \mid I_k), \qquad i = 1, \ldots, n.$$

We refer to p_k as the *belief state*. It evolves according to an equation of the form

$$p_{k+1} = \Phi(p_k, u_k, z_{k+1}),$$

where the function Φ represents an estimator, as discussed in Section 5.4.1. The initial belief state p_0 is given.

The corresponding DP algorithm [see Eqs. (5.40) and (5.41)] has the form

$$\overline{J}_k(p_k) = \min_{u_k \in U} \left[p_k' g(u_k) + E_{z_{k+1}} \left\{ \overline{J}_{k+1}\big(\Phi(p_k, u_k, z_{k+1})\big) \mid p_k, u_k \right\} \right], \quad (5.42)$$

where $g(u_k)$ is the column vector with components $g(1, u_k), \ldots, g(n, u_k)$, and $p_k' g(u_k)$, the expected stage cost, is the inner product of the vectors p_k and $g(u_k)$. The algorithm starts at stage N, with

$$\overline{J}_N(p_N) = p_N' G,$$

where G is the column vector with components the terminal costs $G(i)$, $i = 1, \ldots, n$, and proceeds backwards. Note that in this DP algorithm, the conditional distribution of z_{k+1} given p_k and u_k can be computed using the transition probabilities $p_{ij}(u)$, and the known conditional distribution of z_{k+1} given x_{k+1} and u_k. In particular, we have for any possible observation value z,

$$P(z_{k+1} = z \mid p_k, u_k) = \sum_{i=1}^{n} p_k^i \sum_{j=1}^{n} p_{ij}(u_k) P(z_{k+1} = z \mid x_{k+1} = j, u_k).$$

It turns out that the cost-to-go functions $\overline{J}_k$ in the DP algorithm are *piecewise linear* and *concave*. The demonstration of this fact is straightforward, but tedious, and is outlined in Exercise 5.7. A consequence of the piecewise linearity property is that $\overline{J}_k$ can be characterized by a finite set of scalars. Still, however, for fixed k, the number of these scalars can increase fast with N, and there may be no computationally efficient way to solve the problem (see Papadimitriou and Tsitsiklis [PaT87]). We will not discuss here any special procedures for computing $\overline{J}_k$ (see Lovejoy [Lov91a], [Lov91b], and Smallwood and Sondik [SmS73], [Son71]). Instead we will demonstrate the DP algorithm by means of examples.

Example 5.4.2 (Machine Repair)

In the two-state machine repair example of Section 5.1, let us denote

$$p_1 = P(x_1 = \overline{P} \mid I_1), \qquad p_0 = P(x_0 = \overline{P} \mid I_0).$$

The equation relating p_1, p_0, u_0, z_1 is written as

$$p_1 = \Phi_0(p_0, u_0, z_1).$$

One may verify by straightforward calculation that Φ_0 is given by

$$p_1 = \Phi_0(p_0, u_0, z_1) = \begin{cases} \frac{1}{7} & \text{if } u_0 = S, \quad z_1 = G, \\ \frac{3}{5} & \text{if } u_0 = S, \quad z_1 = B, \\ \frac{1+2p_0}{7-4p_0} & \text{if } u_1 = C, \quad z_1 = G, \\ \frac{3+6p_0}{5+4p_0} & \text{if } u_0 = C, \quad z_1 = B. \end{cases}$$

The DP algorithm (5.42) may be written in terms of p_0, p_1, and Φ_0 above as

$$\overline{J}_1(p_1) = \min[2p_1, 1],$$

$$\begin{aligned} \overline{J}_0(p_0) = \min \Big[& 2p_0 + P(z_1 = G \mid p_0, C)\overline{J}_1\big(\Phi_0(p_0, C, G)\big) \\ & + P(z_1 = B \mid p_0, C)\overline{J}_1\big(\Phi_0(p_0, C, B)\big), \\ & 1 + P(z_1 = G \mid p_0, S)\overline{J}_1\big(\Phi_0(p_0, S, G)\big) \\ & + p(z_1 = B \mid p_0, S)\overline{J}_1\big(\Phi_0(p_0, S, B)\big) \Big]. \end{aligned}$$

The probabilities entering in the second equation may be expressed in terms of p_0 by straightforward calculation as

$$P(z_1 = G \mid p_0, C) = \frac{7 - 4p_0}{12}, \qquad P(z_1 = B \mid p_0, C) = \frac{5 + 4p_0}{12},$$
$$P(z_1 = G \mid p_0, S) = \frac{7}{12}, \qquad P(z_1 = B \mid p_0, S) = \frac{5}{12}.$$

Using these values we have

$$\overline{J}_0(p_0) = \min \left[2p_0 + \frac{7 - 4p_0}{12} \overline{J}_1 \left(\frac{1 + 2p_0}{7 - 4p_0} \right) + \frac{5 + 4p_0}{12} \overline{J}_1 \left(\frac{3 + 6p_0}{5 + 4p_0} \right), \right.$$
$$\left. 1 + \frac{7}{12} \overline{J}_1 \left(\frac{1}{7} \right) + \frac{5}{12} \overline{J}_1 \left(\frac{3}{5} \right) \right].$$

By minimization in the equation defining $\overline{J}_1(p_1)$, we obtain an optimal policy for the last stage

$$\overline{\mu}_1^*(p_1) = \begin{cases} C & \text{if } p_1 \leq \frac{1}{2}, \\ S & \text{if } p_1 > \frac{1}{2}. \end{cases}$$

Also by substitution of $\overline{J}_1(p_1)$ and by carrying out the straightforward calculation, we obtain

$$\overline{J}_0(p_0) = \begin{cases} \frac{19}{12} & \text{if } \frac{3}{8} \leq p_0 \leq 1, \\ \frac{7 + 32p_0}{12} & \text{if } 0 \leq p_0 \leq \frac{3}{8}, \end{cases}$$

and an optimal policy for the first stage:

$$\overline{\mu}_0^*(p_0) = \begin{cases} C & \text{if } p_0 \leq \frac{3}{8}, \\ S & \text{if } p_0 > \frac{3}{8}. \end{cases}$$

Note that

$$P(x_0 = \overline{P} \mid z_0 = G) = \frac{1}{7}, \qquad P(x_0 = \overline{P} \mid z_0 = B) = \frac{3}{5},$$
$$P(z_0 = G) = \frac{7}{12}, \qquad P(z_0 = B) = \frac{5}{12},$$

so that the formula

$$J^* = \mathop{E}_{z_0} \{ \overline{J}_0(P_{x_0 \mid z_0}) \} = \frac{7}{12} \overline{J}_0 \left(\frac{1}{7} \right) + \frac{5}{12} \overline{J}_0 \left(\frac{3}{5} \right) = \frac{176}{144}$$

yields the same optimal cost as the one obtained in Section 5.1 by means of the DP algorithm (5.4) and (5.5).

Example 5.4.3 (A Problem of Instruction)

Consider a problem of instruction where the objective is to teach a student a certain simple item. At the beginning of each period, the student may be in one of two possible states:

L: Item learned.

$\overline{L}$: Item not learned.

At the beginning of each period, the instructor must make one of two decisions:

T: Terminate the instruction.

$\overline{T}$: Continue the instruction for one period and then conduct a test that indicates whether the student has learned the item.

The test has two possible outcomes:

R: Student gives a correct answer.

$\overline{R}$: Student gives an incorrect answer.

The transition probabilities from one state to the next if instruction takes place are given by

$$P(x_{k+1} = L \mid x_k = L) = 1, \qquad P(x_{k+1} = \overline{L} \mid x_k = L) = 0,$$
$$P(x_{k+1} = L \mid x_k = \overline{L}) = t, \qquad P(x_{k+1} = \overline{L} \mid x_k = \overline{L}) = 1 - t,$$

where t is a given scalar with $0 < t < 1$.

The outcome of the test depends probabilistically on the state of knowledge of the student as follows:

$$P(z_k = R \mid x_k = L) = 1, \qquad P(z_k = \overline{R} \mid x_k = L) = 0,$$
$$P(z_k = R \mid x_k = \overline{L}) = r, \qquad P(z_k = \overline{R} \mid x_k = \overline{L}) = 1 - r,$$

where r is a given scalar with $0 < r < 1$. The probabilistic structure of the problem is illustrated in Fig. 5.4.2.

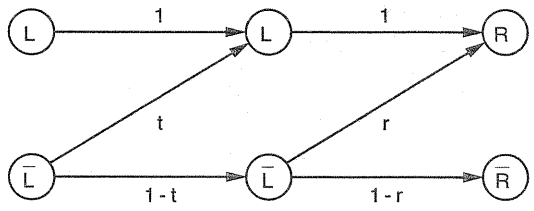


Figure 5.4.2 Probabilistic structure of the instruction problem.

The cost of instruction and testing is I per period. The cost of terminating instruction is 0 or $C > 0$ if the student has learned or has not learned the item, respectively. The objective is to find an optimal instruction-termination policy for each period k as a function of the test information accrued up to that period, assuming that there is a maximum of N periods of instruction.

It is straightforward to reformulate this problem into the framework of the basic problem with imperfect state information and to conclude that the decision whether to terminate or continue instruction at period k should depend on the conditional probability that the student has learned the item given the test results so far. This probability is denoted

$$p_k = P(x_k|I_k) = P(x_k = L \mid z_0, z_1, \ldots, z_k).$$

In addition, we can use the DP algorithm (5.40) and (5.41) defined over the space of the sufficient statistic p_k to obtain an optimal policy. However, rather than proceeding with this elaborate reformulation, we prefer to argue and obtain this DP algorithm directly.

Concerning the evolution of the conditional probability p_k (assuming instruction occurs), we have by Bayes' rule

$$p_{k+1} = P(x_{k+1} = L \mid z_0, \ldots, z_{k+1}) = \frac{P(x_{k+1} = L, z_{k+1} \mid z_0, \ldots, z_k)}{P(z_{k+1} \mid z_0, \ldots, z_k)},$$

where

$$\begin{aligned} &P(z_{k+1} \mid z_0, \ldots, z_k) \\ &\quad = P(x_{k+1} = L \mid z_0, \ldots, z_k)P(z_{k+1} \mid z_0, \ldots, z_k, x_{k+1} = L) \\ &\qquad + P(x_{k+1} = \overline{L} \mid z_0, \ldots, z_k)P(z_{k+1} \mid z_0, \ldots, z_k, x_{k+1} = \overline{L}). \end{aligned}$$

From the probabilistic descriptions given, we have

$$P(z_{k+1} \mid z_0, \ldots, z_k, x_{k+1} = L) = P(z_{k+1} \mid x_{k+1} = L) = \begin{cases} 1 & \text{if } z_{k+1} = R, \\ 0 & \text{if } z_{k+1} = \overline{R}. \end{cases}$$

$$\begin{aligned} P(z_{k+1} \mid z_0, \ldots, z_k, x_{k+1} = \overline{L}) &= P(z_{k+1} \mid x_{k+1} = \overline{L}) \\ &= \begin{cases} r & \text{if } z_{k+1} = R, \\ 1 - r & \text{if } z_{k+1} = \overline{R}. \end{cases} \\ P(x_{k+1} = L \mid z_0, \ldots, z_k) &= p_k + (1 - p_k)t, \\ P(x_{k+1} = \overline{L} \mid z_0, \ldots, z_k) &= (1 - p_k)(1 - t). \end{aligned}$$

Combining these equations, we obtain

$$p_{k+1} = \Phi(p_k, z_{k+1}) = \begin{cases} \frac{p_k + (1-p_k)t}{p_k + (1-p_k)t + (1-p_k)(1-t)r} & \text{if } z_{k+1} = R, \\ 0 & \text{if } z_{k+1} = \overline{R}, \end{cases}$$

or equivalently

$$p_{k+1} = \Phi(p_k, z_{k+1}) = \begin{cases} \frac{1-(1-t)(1-p_k)}{1-(1-t)(1-r)(1-p_k)} & \text{if } z_{k+1} = R, \\ 0 & \text{if } z_{k+1} = \overline{R}. \end{cases} \tag{5.43}$$

This equation is a special case of the general recursive update equation (5.39) for the conditional probability of the state. A cursory examination of Eq. (5.43) shows that, as expected, the conditional probability p_{k+1} that the student has learned the item increases with every correct answer and drops to zero with every incorrect answer.

We now derive the DP algorithm for the problem. At the end of the Nth period, assuming instruction has continued to that period, the expected cost is

$$\overline{J}_N(p_N) = (1 - p_N)C.$$

At the end of period $N - 1$, the instructor has calculated the conditional probability p_{N-1} that the student has learned the item and wishes to decide whether to terminate instruction and incur an expected cost $(1 - p_{N-1})C$ or continue the instruction and incur an expected cost $I + E\{\overline{J}_N(p_N)\}$. This leads to the following equation for the optimal expected cost-to-go:

$$\overline{J}_{N-1}(p_{N-1}) = \min\big[(1 - p_{N-1})C,\ I + (1 - t)(1 - p_{N-1})C\big].$$

The term $(1 - p_{N-1})C$ is the cost of terminating instruction, while the term $(1 - t)(1 - p_{N-1})$ is the probability that the student still has not learned the item following an additional period of instruction.

Similarly, the algorithm is written for every stage k by replacing N by $k + 1$:

$$\overline{J}_k(p_k) = \min\left[(1 - p_k)C,\ I + \mathop{E}_{z_{k+1}}\left\{\overline{J}_{k+1}\big(\Phi(p_k, z_{k+1})\big)\right\}\right].$$

Now using expression (5.43) for the function Φ and the probabilities

$$\begin{aligned} P(z_{k+1} = \overline{R} \mid p_k) &= (1 - t)(1 - r)(1 - p_k), \\ P(z_{k+1} = R \mid p_k) &= 1 - (1 - t)(1 - r)(1 - p_k), \end{aligned}$$

we have

$$\overline{J}_k(p_k) = \min\big[(1 - p_k)C,\ I + A_k(p_k)\big], \tag{5.44}$$

where

$$\begin{aligned} A_k(p_k) = {} & P(z_{k+1} = R \mid I_k)\overline{J}_{k+1}\big(\Phi(p_k, R)\big) \\ & + P(z_{k+1} = \overline{R} \mid I_k)\overline{J}_{k+1}\big(\Phi(p_k, \overline{R})\big) \end{aligned}$$

or, equivalently, using Eq. (5.43),

$$\begin{aligned} A_k(p_k) = {} & \big(1 - (1 - t)(1 - r)(1 - p_k)\big)\overline{J}_{k+1}\left(\frac{1 - (1 - t)(1 - p_k)}{1 - (1 - t)(1 - r)(1 - p_k)}\right) \\ & + (1 - t)(1 - r)(1 - p_k)\overline{J}_{k+1}(0). \end{aligned}$$

As shown in Fig. 5.4.3, if $I + (1 - t)C \leq C$ or, equivalently, if

$$I < tC,$$

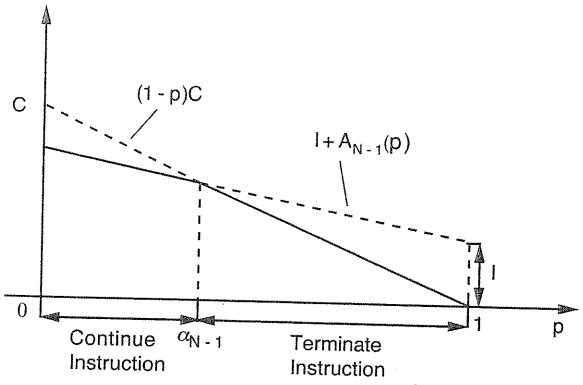


Figure 5.4.3 Determining the optimal instruction policy in the last period.

then there exists a scalar α_{N-1} with $0 < \alpha_{N-1} < 1$ that determines an optimal policy for the last period:

$$\begin{array}{ll} \text{continue instruction} & \text{if } p_{N-1} \leq \alpha_{N-1}, \\ \text{terminate instruction} & \text{if } p_{N-1} > \alpha_{N-1}. \end{array}$$

In the opposite case, where $I \geq tC$, the cost of instruction is so high relative to the cost of not learning that instructing the student is never optimal.

It may be shown by induction (Exercise 5.8) that if $I < tC$, the functions $A_k(p)$ are concave and piecewise linear for each k and satisfy, for all k,

$$A_k(1) = 0.$$

Furthermore, they satisfy for all k,

$$A_k(p) \geq A_k(p'), \qquad \text{for } 0 \leq p < p' \leq 1,$$

$$A_{k-1}(p) \leq A_k(p) \leq A_{k+1}(p), \qquad \text{for all } p \in [0,1].$$

Thus the functions $(1-p_k)C$ and $I + A_k(p_k)$ intersect at a single point, and from the DP algorithm (5.44), we obtain that the optimal policy for each period is determined by the unique scalars a_k, which are such that

$$(1-\alpha_k)C = I + A_k(\alpha_k), \qquad k = 0, 1, \ldots, N-1.$$

An optimal policy for period k is given by

$$\begin{array}{ll} \text{continue instruction} & \text{if } p_k \leq \alpha_k, \\ \text{terminate instruction} & \text{if } p_k > \alpha_k. \end{array}$$

Since the functions $A_k(p)$ are monotonically nondecreasing with respect to k, it follows from Fig. 5.4.4 that

$$\alpha_{N-1} \leq \alpha_{N-2} \leq \cdots \leq \alpha_k \leq \cdots \leq 1 - \frac{1}{C},$$

and therefore the sequence $\{\alpha_k\}$ converges to some scalar $\overline{\alpha}$ as $k \to -\infty$. Thus, as the horizon gets longer, the optimal policy (at least for the initial stages) can be approximated by the stationary policy

$$\begin{array}{ll} \text{continue instruction} & \text{if } p_k \leq \overline{\alpha}_k, \\ \text{terminate instruction} & \text{if } p_k > \overline{\alpha}_k. \end{array} \tag{5.45}$$

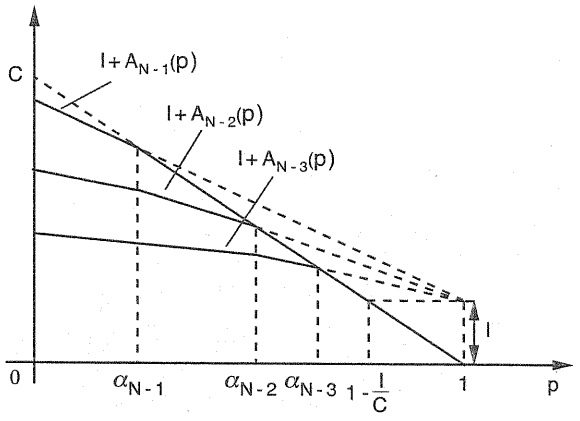


Figure 5.4.4 Demonstrating that the instruction thresholds are decreasing with time.

It turns out that this stationary policy has a convenient implementation that does not require the calculation of the conditional probability at each stage. From Eq. (5.43), we see that p_{k+1} increases over p_k if a correct answer R is given, and drops to zero if an incorrect answer $\overline{R}$ is given. For $m = 1, 2, \ldots$, define recursively the probability π_m of getting m successive correct answers following an incorrect answer:

$$\pi_1 = \Phi(0, R), \quad \pi_2 = \Phi(\pi_1, R), \quad \ldots \quad , \pi_{k+1} = \Phi(\pi_k, R), \ldots$$

Let n be the smallest integer for which $\pi_n > \overline{\alpha}$. Then the stationary policy (5.45) can be implemented by terminating instruction if and only if n successive correct answers have been received.

Example 5.4.4 (Sequential Hypothesis Testing)

Let us consider a hypothesis testing problem that is typical of statistical sequential analysis. A decision maker can make observations, at a cost C each, relating to two hypotheses. Given a new observation, he can either accept one of the hypotheses or delay the decision for one more period, pay the cost C, and obtain a new observation. At issue is trading off the cost of observation with the higher probability of accepting the wrong hypothesis.

Let $z_0, z_1, \ldots, z_{N-1}$ be the sequence of observations. We assume that they are independent, identically distributed random variables taking values on a finite set Z. Suppose we know that the probability distribution of the z_k's is either f_0 or f_1 and that we are trying to decide on one of these. Here, for any element $z \in Z$, $f_0(z)$ and $f_1(z)$ denote the probabilities of z when f_0 and f_1 are the true distributions, respectively. At time k after observing $z_0, \ldots, z_k$, we may either stop observing and accept either f_0 or f_1, or we may take an additional observation at a cost $C > 0$. If we stop observing and make a choice, then we incur zero cost if our choice is correct, and costs L_0 and L_1 if we choose incorrectly f_0 and f_1, respectively. We are given the a priori probability p that the true distribution is f_0, and we assume that at most N observations are possible.

It can be seen that we are faced with an imperfect state information problem involving the two states:

$$x^0 : \text{true distribution is } f_0,$$

$$x^1 : \text{true distribution is } f_1.$$

The alternate DP algorithm (5.40) and (5.41) is defined over the interval $[0, 1]$ of possible values of the conditional probability

$$p_k = P(x_k = x^0 \mid z_0, \ldots, z_k).$$

Similar to the previous section, we will obtain this algorithm directly.

The conditional probability p_k is generated recursively according to the following equation [assuming $f_0(z) > 0$, $f_1(z) > 0$ for all $z \in Z$]:

$$p_0 = \frac{p f_0(z_0)}{p f_0(z_0) + (1-p) f_1(z_0)}, \tag{5.46}$$

$$p_{k+1} = \frac{p_k f_0(z_{k+1})}{p_k f_0(z_{k+1}) + (1-p_k) f_1(z_{k+1})}, \qquad k = 0, 1, \ldots, N-2, \tag{5.47}$$

where p is the a priori probability that the true distribution is f_0. The optimal expected cost for the last period is

$$\overline{J}_{N-1}(p_{N-1}) = \min\big[(1-p_{N-1})L_0,\, p_{N-1}L_1\big], \tag{5.48}$$

where $(1-p_{N-1})L_0$ is the expected cost for accepting f_0 and $p_{N-1}L_1$ is the expected cost for accepting f_1. Taking into account Eqs. (5.46) and (5.47),

we obtain the optimal expected cost-to-go for the kth period as

$$\overline{J}_k(p_k) = \min\Bigg[(1-p_k)L_0,\, p_k L_1,\; C + \mathop{E}_{z_{k+1}}\left\{\overline{J}_{k+1}\left(\frac{p_k f_0(z_{k+1})}{p_k f_0(z_{k+1}) + (1-p_k) f_1(z_{k+1})}\right)\right\}\Bigg],$$

where the expectation over z_{k+1} is taken with respect to the probability distribution

$$p(z_{k+1}) = p_k f_0(z_{k+1}) + (1-p_k) f_1(z_{k+1}), \qquad z_{k+1} \in Z.$$

Equivalently, for $k = 0, 1, \ldots, N-2$,

$$\overline{J}_k(p_k) = \min\big[(1-p_k)L_0,\, p_k L_1,\, C + A_k(p_k)\big], \tag{5.49}$$

where

$$A_k(p_k) = \mathop{E}_{z_{k+1}}\left\{\overline{J}_{k+1}\left(\frac{p_k f_0(z_{k+1})}{p_k f_0(z_{k+1}) + (1-p_k) f_1(z_{k+1})}\right)\right\}. \tag{5.50}$$

An optimal policy for the last period (see Fig. 5.4.5) is obtained from the minimization indicated in Eq. (5.48):

$$\text{accept } f_0 \qquad \text{if } p_{N-1} \geq \gamma,$$

$$\text{accept } f_1 \qquad \text{if } p_{N-1} < \gamma,$$

where γ is determined from the relation $(1-\gamma)L_0 = \gamma L_1$ or equivalently

$$\gamma = \frac{L_0}{L_0 + L_1}.$$

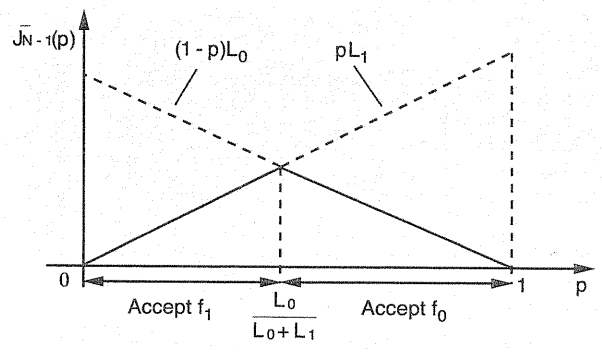


Figure 5.4.5 Determining the optimal policy for the last period.

We now prove that the functions $A_k : [0,1] \to R$ of Eq. (5.50) are concave, and satisfy for all k and $p \in [0,1]$

$$A_k(0) = A_k(1) = 0, \tag{5.51}$$

$$A_{k-1}(p) \leq A_k(p). \tag{5.52}$$

Indeed, we have for all $p \in [0,1]$

$$\overline{J}_{N-2}(p) \leq \min\big[(1-p)L_0,\, pL_0\big] = \overline{J}_{N-1}(p).$$

By making use of the stationarity of the system and the monotonicity property of DP (Exercise 1.23 in Chapter 1), we obtain

$$\overline{J}_k(p) \leq \overline{J}_{k+1}(p)$$

for all k and $p \in [0,1]$. Using Eq. (5.50), we obtain $A_{k+1}(p) \leq A_k(p)$ for all k and $p \in [0,1]$.

To prove concavity of A_k in view of Eqs. (5.48) and (5.49), it is sufficient to show that concavity of $\overline{J}_{k+1}$ implies concavity of A_k through relation (5.50). Indeed, assume that $\overline{J}_{k+1}$ is concave over $[0,1]$. Let $z^1, z^2, \ldots, z^n$ denote the elements of the observation space Z. We have from Eq. (5.50) that

$$A_k(p) = \sum_{i=1}^{n} \big(pf_0(z^i) + (1-p)f_1(z^i)\big)\overline{J}_{k+1}\left(\frac{pf_0(z^i)}{pf_0(z^i) + (1-p)f_1(z^i)}\right).$$

Hence it is sufficient to show that concavity of $\overline{J}_{k+1}$ implies concavity of each of the functions

$$H_i(p) = \big(pf_0(z^i) + (1-p)f_1(z^i)\big)\overline{J}_{k+1}\left(\frac{pf_0(z^i)}{pf_0(z^i) + (1-p)f_1(z^i)}\right).$$

To show concavity of H_i, we must show that for every $\lambda \in [0,1]$, p_1, $p_2 \in [0,1]$ we have

$$\lambda H_i(p_1) + (1-\lambda)H_i(p_2) \leq H_i\big(\lambda p_1 + (1-\lambda)p_2\big).$$

Using the notation

$$\xi_1 = p_1 f_0(z^i) + (1-p_1)f_1(z^i), \qquad \xi_2 = p_2 f_0(z^i) + (1-p_2)f_1(z^i),$$

this inequality is equivalent to

$$\frac{\lambda\xi_1}{\lambda\xi_1 + (1-\lambda)\xi_2}\overline{J}_{k+1}\left(\frac{p_1 f_0(z^i)}{\xi_1}\right) + \frac{(1-\lambda)\xi_2}{\lambda\xi_1 + (1-\lambda)\xi_2}\overline{J}_{k+1}\left(\frac{p_2 f_0(z^i)}{\xi_2}\right)$$
$$\leq \overline{J}_{k+1}\left(\frac{\big(\lambda p_1 + (1-\lambda)p_2\big)f_0(z^i)}{\lambda\xi_1 + (1-\lambda)\xi_2}\right).$$

This relation, however, is implied by the concavity of $\overline{J}_{k+1}$.

Using Eqs. (5.51) and (5.52), we obtain (see Fig. 5.4.6) that if

$$C + A_{N-2}\left(\frac{L_0}{L_0 + L_1}\right) < \frac{L_0 L_1}{L_0 + L_1},$$

then an optimal policy for each period k is of the form

accept f_0 if $p_k \geq \alpha_k$,

accept f_1 if $p_k \leq \beta_k$,

continue the observations if $\beta_k < p_k < \alpha_k$,

where the scalars α_k, β_k are determined from the relations

$$\beta_k L_1 = C + A_k(\beta_k),$$
$$(1-\alpha_k)L_0 = C + A_k(\alpha_k).$$

Furthermore, we have

$$\cdots \leq \alpha_{k+1} \leq \alpha_k \leq \alpha_{k-1} \leq \cdots \leq 1 - \frac{C}{L_0},$$

$$\cdots \geq \beta_{k+1} \geq \beta_k \geq \beta_{k-1} \geq \cdots \geq \frac{C}{L_1}.$$

Hence as $N \to \infty$ the sequences $\{\alpha_{N-i}\}$, $\{\beta_{N-i}\}$ converge to scalars $\overline{\alpha}$, $\overline{\beta}$, respectively, and the optimal policy is approximated by the stationary policy

accept f_0 if $p_k \geq \overline{\alpha}$, (5.53)

accept f_1 if $p_k \leq \overline{\beta}$, (5.54)

continue the observations if $\overline{\beta} < p_k < \overline{\alpha}$. (5.55)

Now the conditional probability p_k is given by

$$p_k = \frac{pf_0(z_0)\cdots f_0(z_k)}{pf_0(z_0)\cdots f_0(z_k) + (1-p)f_1(z_0)\cdots f_1(z_k)}, \tag{5.56}$$

where p is the a priori probability that f_0 is the true hypothesis. Using Eq. (5.56), the stationary policy (5.53)-(5.55) can be written in the form

accept f_0 if $R_k \geq A$, (5.57)

accept f_1 if $R_k \leq B$, (5.58)

continue the observations if $B < R_k < A$, (5.59)

where

$$A = \frac{(1-p)\overline{\alpha}}{p(1-\overline{\alpha})},$$

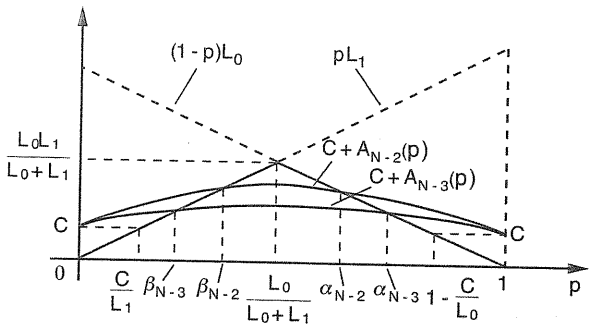


Figure 5.4.6 Determining the optimal hypothesis testing policy.

$$B = \frac{(1-p)\overline{\beta}}{p(1-\overline{\beta})},$$

and

$$R_k = \frac{f_0(z_0)\cdots f_0(z_k)}{f_1(z_0)\cdots f_1(z_k)}.$$

Note that R_k can be easily generated by means of the recursive equation

$$R_{k+1} = \frac{f_0(z_{k+1})}{f_1(z_{k+1})} R_k.$$

The policy (5.57)-(5.59) is known as the *sequential probability ratio test*, and was among the first formal methods studied in statistical sequential analysis by Wald [Wal47]. The optimality of this policy for the infinite horizon version of the problem will be shown in Vol. II, Chapter 3.

5.5 NOTES, SOURCES, AND EXERCISES

For literature on linear-quadratic problems with imperfect state information, see the references quoted for Section 4.1 and the survey paper by Witsenhausen [Wit71]. The Kalman filtering algorithm is discussed in many textbooks, such as Anderson and Moore [AnM79], Ljung and Soderstrom [LjS83]. For linear-quadratic problems with Gaussian uncertainties and observation cost in the spirit of Exercise 5.6, see Aoki and Li [AoL69]. Exercise 5.1, which indicates the form of the certainty equivalence principle when the random disturbances are correlated, is based on an unpublished report by the author [Ber70]. The minimum variance approach is described in Aström and Wittenmark [AsW84], and Whittle [Whi63].

Problems with exponential cost functions are discussed in James, Baras, and Elliott [JBE94], and Fernandez-Gaucherand and Markus [FeM94].

The idea of data reduction via a sufficient statistic gained wide attention following the paper by Striebel [Str65]; see also Shiryaev [Shi64], [Shi66]. For the analog of the sufficient statistic idea in sequential minimax problems with set membership description of the uncertainty, see Bertsekas and Rhodes [BeR73].

Sufficient statistics have been used for analysis of finite-state problems with imperfect state information by Eckles [Eck68], and by Smallwood and Sondik [SmS73], [Son71]. The proof of piecewise linearity of the cost-to-go functions, and an algorithm for their computation are given by Smallwood and Sondik [SmS73], [Son71]. For further material on finite-state problems with imperfect state information, see Arapostathis et. al. [ABF93], Lovejoy [Lov91a], [Lov91b], and White and Scherer [WhS89].

The instruction model described in Example 5.4.3 has been considered (with some variations) by a number of authors such as Atkinson, Bower, and Crothers [ABC65], Groen and Atkinson [GrA66], Karush and Dear [KaD66], and Smallwood [Sma71].

For a discussion of the sequential probability ratio test (cf. Example 5.4.4) and related subjects, see Chernoff [Che72], DeGroot [DeG70], [Whi82], and the references quoted therein. The treatment given here stems from Arrow, Blackwell, and Girshick [ABG49].

EXERCISES

5.1 (Linear Quadratic Problems – Correlated Disturbances) WWW

Consider the linear system and measurement equation of Section 5.2 and consider the problem of finding a policy $\{\mu_0^*(I_0), \ldots, \mu_{N-1}^*(I_{N-1})\}$ that minimizes the quadratic cost

$$E\left\{x_N' Q x_N + \sum_{k=0}^{N-1} u_k' R_k u_k\right\}.$$

Assume, however, that the random vectors $x_0, w_0, \ldots, w_{N-1}$, $v_0, \ldots, v_{N-1}$ are correlated and have a given joint probability distribution, and finite first and second moments. Show that the optimal policy is given by

$$\mu_k^*(I_k) = L_k E\{y_k \mid I_k\},$$

where the gain matrices L_k are obtained from the algorithm

$$L_k = -(B_k' K_{k+1} B_k + R_k)^{-1} B_k' K_{k+1} A_k,$$
$$K_N = Q,$$
$$K_k = A_k' \big(K_{k+1} - K_{k+1} B_k (B_k' K_{k+1} B_k + R_k)^{-1} B_k' K_{k+1}\big) A_k,$$

and the vectors y_k are given by

$$y_k = x_k + A_k^{-1} w_k + A_k^{-1} A_{k+1}^{-1} w_{k+1} + \cdots + A_k^{-1} \cdots A_{N-1}^{-1} w_{N-1}$$

(assuming the matrices $A_0, A_1, \ldots, A_{N-1}$ are invertible). *Hint*: Show that the cost can be written as

$$E\left\{ y_0' K_0 y_0 + \sum_{k=0}^{N-1} (u_k - L_k y_k)' P_k (u_k - L_k y_k) \right\},$$

where

$$P_k = B_k' K_{k+1} B_k + R_k.$$

5.2

Consider the scalar system

$$x_{k+1} = x_k + u_k + w_k,$$
$$z_k = x_k + v_k,$$

where we assume that the initial condition x_0, and the disturbances w_k and v_k are all independent. Let the cost be

$$E\left\{ x_N^2 + \sum_{k=0}^{N-1} \big(x_k^2 + u_k^2\big) \right\},$$

and let the given probability distributions be

$$p(x_0 = 2) = \frac{1}{2}, \qquad p(w_k = 1) = \frac{1}{2}, \qquad p\left(v_k = \frac{1}{4}\right) = \frac{1}{2},$$
$$p(x_0 = -2) = \frac{1}{2}, \qquad p(w_k = -1) = \frac{1}{2}, \qquad p\left(v_k = -\frac{1}{4}\right) = \frac{1}{2}.$$

(a) Determine the optimal policy. *Hint*: For this problem, $E\{x_k \mid I_k\}$ can be determined from $E\{x_{k-1} \mid I_{k-1}\}$, u_{k-1}, and z_k.

(b) Determine the policy that is identical to the optimal except that it uses a linear least squares estimator of x_k given I_k in place of $E\{x_k \mid I_k\}$ (see Appendix E – this policy can be shown to be optimal within the class of all policies that are linear functions of the measurements).

(c) Determine the asymptotic form of the policies in parts (a) and (b) as $N \to \infty$.

5.3 WWW

A linear system with Gaussian disturbances and Gaussian initial state

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

is to be controlled so as to minimize a quadratic cost similar to that in Section 5.2. The difference is that the controller has the option of choosing at each time k one of two types of measurements for the next stage $(k+1)$:

$$\text{first type:} \qquad z_{k+1} = C^1 x_{k+1} + v_{k+1}^1$$
$$\text{second type:} \qquad z_{k+1} = C^2 x_{k+1} + v_{k+1}^2.$$

Here C^1 and C^2 are given matrices of appropriate dimension, and $\{v_k^1\}$ and $\{v_k^2\}$ are zero-mean, independent, random sequences with given finite covariances that do not depend on x_0 and $\{w_k\}$. There is a cost g_1 (or g_2) each time a measurement of type 1 (or type 2) is taken. The problem is to find the optimal control and measurement selection policy that minimizes the expected value of the sum of the quadratic cost

$$x_N' Q x_N + \sum_{k=0}^{N-1} (x_k' Q x_k + u_k' R u_k)$$

and the total measurement cost. Assume for convenience that $N = 2$ and that the first measurement z_0 is of type 1. Show that the optimal measurement selection at $k = 0$ and $k = 1$ does not depend on the value of the information vectors I_0 and I_1, and can be determined a priori. Describe the nature of the optimal policy.

5.4

Consider a scalar single-input, single-output system given by

$$y_k + a_1 y_{k-1} + \cdots + a_m y_{k-m} = b_M u_{k-M} + \cdots + b_m u_{k-m} + \epsilon_k + c_1 \epsilon_{k-1} + \cdots + c_m \epsilon_{k-m} + v_{k-n},$$

where $1 \le M \le m$, $0 \le n \le m$, and v_k is generated by an equation of the form

$$v_k + d_1 v_{k-1} + \cdots + d_m v_{k-m} = \xi_k + \epsilon_1 \xi_{k-1} + \cdots + \epsilon_m \xi_{k-m},$$

and the polynomials $(1 + c_1 s + \cdots + c_m s^m)$, $(1 + d_1 s + \cdots + d_m s^m)$, and $(1 + \epsilon_1 s + \cdots + \epsilon_m s^m)$ have roots strictly outside the unit circle. The value of the scalar v_k is observed by the controller at time k together with y_k. The sequences $\{\epsilon_k\}$ and $\{\xi_k\}$ are zero mean independent identically distributed with finite variances. Find an easily implementable approximation to the minimum variance controller minimizing $E\{\sum_{k=0}^{N} (y_k)^2\}$. Discuss the stability properties of the closed-loop system.

5.5

(a) Within the framework of the basic problem with imperfect state information, consider the case where the system and the observations are linear:

$$x_{k+1} = A_k x_k + B_k u_k + w_k,$$

$$z_k = C_k x_k + v_k.$$

The initial state x_0 and the disturbances w_k and v_k are assumed Gaussian and mutually independent. Their covariances are given, and w_k and v_k have zero mean. Show that $E\{x_0 \mid I_0\}, \ldots, E\{x_{N-1} \mid I_{N-1}\}$ constitute a sufficient statistic for this problem.

(b) Use the result of part (a) to obtain an optimal policy for the special case of the single-stage problem involving the scalar system and observation

$$x_1 = x_0 + u_0,$$

$$z_0 = x_0 + v_0,$$

and the cost function $E\{|x_k|\}$.

(c) Generalize part (b) for the case of the scalar system

$$x_{k+1} = a x_k + u_k,$$

$$z_k = c x_k + v_k,$$

and the cost function $E\{\sum_{k=1}^{N} |x_k|\}$. The scalars a and c are given. *Note*: You may find useful the following "differentiation of an integral" formula:

$$\frac{d}{dy}\int_{\alpha(y)}^{\beta(y)} f(y,\xi)d\xi = \int_{\alpha(y)}^{\beta(y)} \frac{df(y,\xi)}{dy}d\xi + f\big(y,\beta(y)\big)\frac{d\beta(y)}{dy} - f\big(y,\alpha(y)\big)\frac{d\alpha(y)}{dy}.$$

5.6

Consider a machine that can be in one of two states, good or bad. Suppose that the machine produces an item at the end of each period. The item produced is either good or bad depending on whether the machine is in a good or bad state at the beginning of the corresponding period, respectively. We suppose that once the machine is in a bad state it remains in that state until it is replaced. If the machine is in a good state at the beginning of a certain period, then with probability t it will be in the bad state at the end of the period. Once an item is produced, we may inspect the item at a cost I or not inspect. If an inspected item is found to be bad, the machine is replaced with a machine in good state at a cost R. The cost for producing a bad item is $C > 0$. Write a DP algorithm for obtaining an optimal inspection policy assuming a machine initially in good state and a horizon of N periods. Solve the problem for $t = 0.2$, $I = 1$, $R = 3$, $C = 2$, and $N = 8$. (The optimal policy is to inspect at the end of the third period and not inspect in any other period.)

5.7 (Finite-State Systems – Imperfect State Information) WWW

Consider a system that at any time can be in any one of a finite number of states $1, 2, \ldots, n$. When a control u is applied, the system moves from state i to state j with probability $p_{ij}(u)$. The control u is chosen from a finite collection $u^1, u^2, \ldots, u^m$. Following each state transition, an observation is made by the controller. There is a finite number of possible observation outcomes $z^1, z^2, \ldots, z^q$. The probability of occurrence of z^θ, given that the current state is j and the preceding control was u, is denoted by $r_j(u,\theta)$, $\theta = 1, \ldots, q$.

(a) Consider the column vector of conditional probabilities

$$P_k = [p_k^1, \ldots, p_k^n]',$$

where

$$p_k^j = P(x_k = j \mid z_0, \ldots, z_k, u_0, \ldots, u_{k-1}), \qquad j = 1, \ldots, n,$$

and show that it can be updated according to

$$p_{k+1}^j = \frac{\sum_{i=1}^n p_k^i p_{ij}(u_k) r_j(u_k, z_{k+1})}{\sum_{s=1}^n \sum_{i=1}^n p_k^i p_{is}(u_k) r_s(u_k, z_{k+1})}, \qquad j = 1, \ldots, n.$$

Write this equation in the compact form

$$P_{k+1} = \frac{\big[r(u_k, z_{k+1})\big] * \big[P(u_k)' P_k\big]}{r(u_k, z_{k+1})' P(u_k)' P_k},$$

where prime denotes transposition and

$P(u_k)$ is the $n \times n$ transition probability matrix with ijth element $p_{ij}(u_k)$,

$r(u_k, z_{k+1})$ is the column vector with jth coordinate $r_j(u_k, z_{k+1})$,

$\big[P(u_k)' P_k\big]_j$ is the jth coordinate of the vector $P(u_k)' P_k$,

$\big[r(u_k, z_{k+1})\big] * \big[P(u_k)' P_k\big]$ is the vector whose jth coordinate is the scalar $r_j(u_k, z_{k+1})\big[P(u_k)' P_k\big]_j$.

(b) Assume that there is a cost for each stage k denoted $g_k(i,u,j)$ and associated with the control u and a transition from i to j. There is no terminal cost. Consider the problem of finding a policy that minimizes the sum of expected costs per stage over N periods. Show that the corresponding DP algorithm is given by

$$\overline{J}_{N-1}(P_{N-1}) = \min_{u \in \{u^1, \ldots, u^m\}} P_{N-1}' G_{N-1}(u),$$

$$\overline{J}_k(P_k) = \min_{u \in \{u^1, \ldots, u^m\}} \left[P_k' G_k(u) + \sum_{\theta=1}^{q} r(u,\theta)' P(u)' P_k \overline{J}_{k+1}\left(\frac{\big[r(u,\theta)\big] * \big[P(u)' P_k\big]}{r(u,\theta)' P(u)' P_k} \right) \right],$$

where $G_k(u)$ is the vector of expected kth stage costs given by

$$G_k(u) = \begin{pmatrix} \sum_{j=1}^n p_{1j}(u) g_k(1,u,j) \\ \vdots \\ \sum_{j=1}^n p_{nj}(u) g_k(n,u,j) \end{pmatrix}.$$

(c) Show that, for all k, $\overline{J}_k$ when viewed as a function defined on the set of vectors with nonnegative coordinates, is *positively homogeneous*; that is,

$$\overline{J}_k(\lambda P_k) = \lambda \overline{J}_k(P_k)$$

for all $\lambda > 0$. Use this fact to write the DP algorithm in the simpler form

$$\overline{J}_k(P_k) = \min_{u \in \{u^1, \ldots, u^m\}} \left[P_k' G_k(u) + \sum_{\theta=1}^q \overline{J}_{k+1}\left(\left[r(u,\theta) \right] * \left[P(u)' P_k \right] \right) \right].$$

(d) Show by induction that, for all k, $\overline{J}_k$ has the form

$$\overline{J}_k(P_k) = \min\left[P_k' \alpha_k^1, \ldots, P_k' \alpha_k^{m_k} \right],$$

where $\alpha_k^1, \ldots, \alpha_k^{m_k}$ are some vectors in $\Re^n$.

5.8

Consider the functions $\overline{J}_k(p_k)$ in the instruction problem of Example 5.4.3. Show inductively that each of these functions is piecewise linear, concave, and of the form

$$\overline{J}_k(p_k) = \min\left[\alpha_k^1 + \beta_k^1 p_k, \alpha_k^2 + \beta_k^2 p_k, \ldots, \alpha_k^{m_k} + \beta_k^{m_k} p_k \right],$$

where $\alpha_k^1, \ldots, \alpha_k^{m_k}$, $\beta_k^1, \ldots, \beta_k^{m_k}$ are suitable scalars.

5.9

A person is offered N free plays to be distributed as he pleases between two slot machines A and B. Machine A pays α dollars with known probability s and nothing with probability $(1-s)$. Machine B pays β dollars with probability p and nothing with probability $(1-p)$. The person does not know p but instead has an a priori probability distribution $F(p)$ for p. The problem is to find a playing policy that maximizes expected profit. Let $(m+n)$ denote the number of plays in machine B after k free plays $(m+n \leq k)$, and let m denote the number of successes and n the number of failures. Show that a DP algorithm for this problem is given for $m+n \leq k$ by

$$\overline{J}_{N-1}(m,n) = \max\left[s\alpha,\ p(m,n)\beta \right],$$

$$\overline{J}_k(m,n) = \max\Big[s\big(\alpha + \overline{J}_{k+1}(m,n)\big) + (1-s)\overline{J}_{k+1}(m,n),$$
$$p(m,n)\big(\beta + \overline{J}_{k+1}(m+1,n)\big) + \big(1 - p(m,n)\big)\overline{J}_{k+1}(m,n+1)\Big]$$

where

$$p(m,n) = \frac{\int_0^1 p^{m+1}(1-p)^n dF(p)}{\int_0^1 p^m (1-p)^n dF(p)}.$$

Solve the problem for $N = 6$, $\alpha = \beta = 1$, $s = 0.6$, $dF(p)/dp = 1$ for $0 \leq p \leq 1$. [The answer is to play machine B for the following pairs (m,n) : $(0,0)$, $(1,0)$, $(2,0)$, $(3,0)$, $(4,0)$, $(5,0)$, $(2,1)$, $(3,1)$, $(4,1)$. Otherwise, machine A should be played.]

5.10

A person is offered 2 to 1 odds in a coin-tossing game where he wins whenever a tail occurs. However, he suspects that the coin is biased and has an a priori probability distribution $F(p)$ for the probability p that a head occurs at each toss. The problem is to find an optimal policy of deciding whether to continue or stop participating in the game given the outcomes of the game so far. A maximum of N tossings is allowed. Indicate how such a policy can be found by means of DP.

5.11

Consider the ARMAX model of Section 5.3 where instead of $E\left\{\sum_{k=1}^N (y_k)^2\right\}$, the cost is

$$E\left\{ \sum_{k=1}^N (y_k - \overline{y})^2 \right\},$$

where $\overline{y}$ is a given scalar. Generalize the minimum variance policy for this case.

5.12

Consider the ARMAX model

$$y_k + a y_{k-1} = u_{k-M} + \epsilon_k,$$

where $M \geq 1$. Show that the minimum variance controller is

$$\mu_k(I_k) = a u_{k-1} - a^2 u_{k-2} + \cdots - (-1)^{M-1} a^{M-1} u_{k-M+1} - (-1)^M a^M y_k,$$

that the resulting closed-loop system is

$$y_k = \epsilon_k - a\epsilon_{k-1} + a^2 \epsilon_{k-2} - \cdots + (-1)^{M-1} a^{M-1} \epsilon_{k-M+1},$$

and that the long-term output variance is

$$E\left\{(y_k)^2\right\} = \frac{1-a^{2M}}{1-a^2} E\left\{(\epsilon_k)^2\right\}.$$

Discuss the qualitative difference between the cases $|a| < 1$ and $|a| > 1$, and relate it to the stability properties of the uncontrolled system $y_k + a y_{k-1} = \epsilon_k$ and the size of the delay M.

5.13 (Linear-Quadratic Problems with Disturbance Estimation)

Consider the linear-quadratic problem discussed in Section 4.1 (A_k, B_k: known). The state x_k is perfectly observed at each stage, and the demands w_k are independent, identically distributed random vectors. However, the (common) distribution of the w_k is unknown. Instead it is known that this distribution is one out of two given distributions F_1 and F_2, and that the a priori probability that F_1 is the correct distribution is a given scalar q, with $0 < q < 1$. For convenience, assume that w_k can take a finite number of values under each of F_1 and F_2.

(a) Formulate this as an imperfect state information problem, and identify the state, control, system disturbance, observation, and observation disturbance.

(b) Show that (x_k, q_k), where

$$q_k = P(\text{distribution is } F_1 \mid w_0, \ldots, w_{k-1}),$$

is a suitable sufficient statistic, and write a corresponding DP algorithm.

(c) Show that the optimal control law is of the form

$$\mu_k(x_k, q_k) = -(B_k' K_{k+1} B_k + R_k)^{-1} B_k' K_{k+1} A_k x_k + c_k(q_k),$$

where the matrices K_k are given by the Riccati equation, and $c_k(q_k)$ are appropriate functions of q_k. *Hint*: Show that the cost-to-go function has the form

$$J_k(x_k, q_k) = x_k' K_k x_k + a_k(q_k)' x_k + b_k(q_k),$$

where $a_k(q_k)$ and $b_k(q_k)$ are appropriate functions of q_k.

5.14 (Asset Selling Problem with Offer Estimation)

Consider the asset selling problem of Section 4.4. The offers w_k are independent and identically distributed. However, the (common) distribution of the w_k is unknown. Instead it is known that this distribution is one out of two given distributions F_1 and F_2, and that the a priori probability that F_1 is the correct distribution is a given scalar q, with $0 < q < 1$.

(a) Formulate this as an imperfect state information problem, and identify the state, control, system disturbance, observation, and observation disturbance.

(b) Show that (x_k, q_k), where

$$q_k = P(\text{distribution is } F_1 \mid w_0, \ldots, w_{k-1}),$$

is a suitable sufficient statistic, write a corresponding DP algorithm, and derive the form of the optimal selling policy.

5.15

Consider an inventory control problem where stock evolves according to

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

and the cost of stage k is

$$cu_k + h \max(0, w_k - x_k - u_k) + p \max(0, x_k + u_k - w_k),$$

where c, h, and p are positive scalars with $p > c$. There is no terminal cost. The stock x_k is perfectly observed at each stage. The demands w_k are independent, identically distributed, nonnegative random variables. However, the (common) distribution of the w_k is unknown. Instead it is known that this distribution is one out of two given distributions F_1 and F_2, and that the a priori probability that F_1 is the correct distribution is a given scalar q, with $0 < q < 1$.

(a) Formulate this as an imperfect state information problem, and identify the state, control, system disturbance, observation, and observation disturbance.

(b) Write a DP algorithm in terms of a suitable sufficient statistic.

(c) Characterize as best as you can the optimal policy.

5.16

Consider the search problem of Example 5.4.1 for different values of the search horizon N.

(a) Show that for any value of the a priori probability p_0 that is strictly less than 1, there is a threshold value of N, call it $\overline{N}$, such that the optimal reward function $J_0(p_0)$ is independent of N as long as $N > \overline{N}$.

(b) For N greater than the threshold $\overline{N}$ of part (a) and for a given value of p_0, give a method to calculate the value of $J_0(p_0)$ that does not use the DP algorithm.

(c) Suppose that there are two sites that can be searched, instead of one. The sites may contain a treasure of corresponding values V^1 and V^2 (independently of each other), and the probabilities of a successful search are β_1 and β_2, respectively. After finding a treasure in one site, one may continue searching for the treasure in the other site (but of course each search costs C). Write a DP algorithm involving the probabilities p_k^1 and p_k^2 that a treasure is present at sites 1 and 2, respectively.

(d) Under the assumptions of part (c), show that for any values of the a priori probabilities p_0^1, p_0^2, there is a threshold value of N, call it $\overline{N}$, such that the optimal cost-to-go function $J_0(p_0^1, p_0^2)$ is independent of N as long as $N > \overline{N}$. Find the optimal search policy if $N > \overline{N}$.

6

Approximate Dynamic Programming

Contents

We have seen that it is sometimes possible to use the DP algorithm to obtain an optimal policy in closed form. However, this tends to be the exception. In most cases a numerical solution is necessary. The associated computational requirements are often overwhelming, and for many problems a complete solution of the problem by DP is impossible. To a great extent, the reason lies in what Bellman has called the "curse of dimensionality." This refers to an exponential increase of the required computation as the problem's size increases.

Consider for example a problem where the state, control, and disturbance spaces are the Euclidean spaces $\Re^n$, $\Re^m$, and $\Re^r$, respectively. In a straightforward numerical approach, these spaces are discretized. Taking d discretization points per state axis results in a state space grid with d^n points. For each of these points the minimization in the right-hand side of the DP equation must be carried out numerically, which involves comparison of as many as d^m numbers. To calculate each of these numbers, one must calculate an expected value over the disturbance, which is the weighted sum of as many as d^r numbers. Finally, the calculations must be done for each of the N stages. Thus as a first approximation, the number of computational operations is at least of the order of Nd^n and can be of the order of Nd^{n+m+r}. It follows that for perfect state information problems with Euclidean state and control spaces, DP can be applied numerically only if the dimensions of the spaces are relatively small. Based on the analysis of the preceding chapter, we can also conclude that for problems of imperfect state information the situation is hopeless, except for very simple or very special cases.

In the real world, there is an additional aspect of optimal control problems that can have a profound impact on the feasibility of DP as a practical solution method. In particular, there are many circumstances where the structure of the given problem is known well in advance, but some of the problem data, such as various system parameters, may be unknown until shortly before control is needed, thus seriously constraining the amount of time available for the DP computation. Usually this occurs as a result of one or both of the following situations:

(a) *A family of problems is addressed, rather than a single problem*, and we do not get to know the exact problem to be solved until shortly before the control process begins. As an example, consider a problem of planning the daily route of a utility vehicle within a street network so that it passes through a number of points where it must perform some service. The street network and the vehicle characteristics may be known well in advance, but the service points may vary from day to day, and may not become known until shortly before the vehicle begins its route. This example is typical of situations, where the same problem must periodically be solved with small variations in its data. Yet, if DP is to be used, the solution of one instance of the problem may not help appreciably in solving a different instance.

(b) *The problem data changes as the system is being controlled*. As an example, consider the route planning example in case (a) above, and assume that new service points to be visited arise as the vehicle is on its way. It is possible in principle to model these data changes in terms of stochastic disturbances, but then we may end up with a problem that is too complicated for analysis or solution by DP. A frequently employed alternative is to use *on-line replanning*, whereby the problem is resolved on-line with the new data, as soon as these data become available, and control continues with a policy that corresponds to the new data.

A common feature of the above situations, which can seriously impact the solution, is that there may be stringent time constraints for the computation of the controls. This may substantially exacerbate the "curse of dimensionality" problem mentioned above.

As indicated by the above discussion, in practice one often has to settle for a suboptimal control scheme that strikes a reasonable balance between convenient implementation and adequate performance. In this chapter we discuss some general approaches for suboptimal control, which are based on approximations to the DP algorithm. We begin with two general schemes to simplify the DP computation, certainty equivalent control (Section 6.1), which replaces the stochastic quantities of the problem by deterministic nominal values, and open-loop-feedback control (Section 6.2), which ignores in part the availability of information in the future. These two schemes set the stage for limited lookahead control, which together with its many variations (Sections 6.3-6.5), is one of the principal approaches for suboptimal control. We also discuss adaptive control in the context of certainty equivalent control. This discussion is not used in subsequent developments, so the reader may skip Sections 6.1.1-6.1.4 if desired.

6.1 CERTAINTY EQUIVALENT AND ADAPTIVE CONTROL

The *certainty equivalent controller* (CEC) is a suboptimal control scheme that is inspired by linear-quadratic control theory. It applies at each stage the control that would be optimal if the uncertain quantities were fixed at some "typical" values; that is, it acts as if a form of the certainty equivalence principle were holding.

The advantage of the CEC is that it replaces the DP algorithm with what is often a much less demanding computation: the solution of a *deterministic* optimal control problem at each stage. This problem yields an optimal control sequence, the first component of which is used at the current stage, while the remaining components are discarded. The main attractive characteristic of the CEC is its ability to deal with stochastic

and even imperfect information problems by using the mature and effective methodology of deterministic optimal control.

We describe the CEC for the general problem with imperfect state information of Section 5.1. As can be expected, the implementation is considerably simpler if the controller has perfect state information. Suppose that we have an "estimator" that uses the information vector I_k to produce a "typical" value $\overline{x}_k(I_k)$ of the state. Assume also that for every state-control pair (x_k, u_k) we have selected a "typical" value of the disturbance, which we denote by $\overline{w}_k(x_k, u_k)$. For example, if the state spaces and disturbance spaces are convex subsets of Euclidean spaces, the expected values

$$\overline{x}_k(I_k) = E\{x_k \mid I_k\}, \qquad \overline{w}_k(x_k, u_k) = E\{w_k \mid x_k, u_k\},$$

can serve as typical values.

The control input $\overline{\mu}_k(I_k)$ applied by the CEC at each time k is determined by the following rule:

(1) Given the information vector I_k, compute the state estimate $\overline{x}_k(I_k)$.

(2) Find a control sequence $\{\overline{u}_k, \overline{u}_{k+1}, \ldots, \overline{u}_{N-1}\}$ that solves the deterministic problem obtained by fixing the uncertain quantities x_k and $w_k, \ldots, w_{N-1}$ at their typical values:

$$\text{minimize} \quad g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, u_i, \overline{w}_i(x_i, u_i)\big)$$

subject to the initial condition $x_k = \overline{x}_k(I_k)$ and the constraints

$$u_i \in U_i, \qquad x_{i+1} = f_i\big(x_i, u_i, \overline{w}_i(x_i, u_i)\big), \qquad i = k, k+1, \ldots, N-1.$$

(3) Use as control the first element in the control sequence found:

$$\overline{\mu}_k(I_k) = \overline{u}_k.$$

Note that step (1) is unnecessary if we have perfect state information; in this case we simply use the known value of the x_k. The deterministic optimization problem in step (2) must be solved at each time k, once the initial state $\overline{x}_k(I_k)$ becomes known by means of an estimation (or perfect observation) procedure. A total of N such problems must be solved by the CEC at every system run. In many cases of interest, these deterministic problems can be solved by powerful numerical methods such as conjugate gradient, Newton's method, augmented Lagrangian, and sequential quadratic programming methods; see e.g. Luenberger [Lue84] or Bertsekas [Ber99]. Furthermore, the implementation of the CEC requires no storage of the type required for the optimal feedback controller.

An alternative to solving N optimal control problems in an "on-line" fashion is to solve these problems a priori. This is accomplished by computing an optimal feedback controller for the deterministic optimal control problem obtained from the original problem by replacing all uncertain quantities by their typical values. It is easy to verify, based on the equivalence of open-loop and feedback implementation of optimal controllers for deterministic problems, that the implementation of the CEC given earlier is equivalent to the following.

Let $\{\mu_0^d(x_0), \ldots, \mu_{N-1}^d(x_{N-1})\}$ be an optimal controller obtained from the DP algorithm for the deterministic problem

$$\text{minimize} \quad g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), \overline{w}_k(x_k, u_k)\big)$$

$$\text{subject to} \quad x_{k+1} = f_k\big(x_k, \mu_k(x_k), \overline{w}_k(x_k, u_k)\big), \quad \mu_k(x_k) \in U_k, \quad k \geq 0.$$

Then the control input $\overline{\mu}_k(I_k)$ applied by the CEC at time k is given by

$$\overline{\mu}_k(I_k) = \mu_k^d\big(\overline{x}_k(I_k)\big)$$

as shown in Fig. 6.1.1.

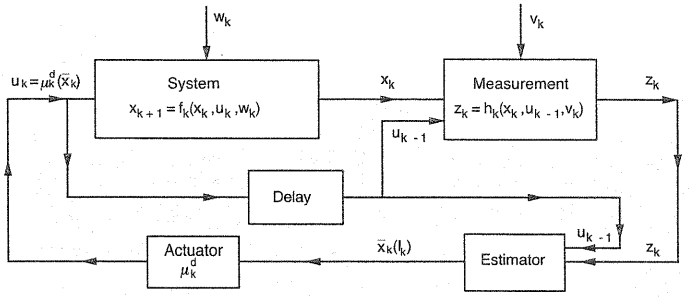


Figure 6.1.1 Structure of the certainty equivalent controller when implemented in feedback form.

In other words, an equivalent alternative implementation of the CEC consists of finding a feedback controller $\{\mu_0^d, \mu_1^d, \ldots, \mu_{N-1}^d\}$ that is optimal for a corresponding deterministic problem, and subsequently using this controller for control of the uncertain system [modulo substitution of the state x_k by its estimate $\overline{x}_k(I_k)$]. Either one of the definitions given for the CEC can serve as a basis for its implementation. Depending on the nature of the problem, one method may be preferable to the other.

The CEC approach often performs well in practice and yields near-optimal policies. In fact, for the linear-quadratic problems of Sections 4.1

and 5.2, the CEC is identical to the optimal controller (certainty equivalence principle). It is possible, however, that a CEC performs strictly worse than the optimal open-loop controller (see Exercise 6.2).

In what follows in this section, we will discuss a few variants of the CEC, and we will then focus on one particular type of methodology, adaptive control of systems with unknown parameters.

Certainty Equivalent Control with Heuristics

Even though the CEC approach simplifies a great deal the computations, it still requires the solution of a deterministic optimal control problem at each stage. This problem may be difficult, and a more convenient approach may be to solve it suboptimally using a heuristic algorithm. To simplify notation, let us assume perfect state information [the ideas to be discussed can also be applied to imperfect state information problems, by substituting x_k with its estimate $\overline{x}_k(I_k)$]. Then, in this approach, given x_k, we use some (easily implementable) heuristic to find a suboptimal control sequence $\{\overline{u}_k, \overline{u}_{k+1}, \ldots, \overline{u}_{N-1}\}$ for the problem

$$\text{minimize} \quad g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, u_i, \overline{w}_i(x_i, u_i)\big)$$

subject to

$$u_i \in U_i(x_i), \qquad x_{i+1} = f_i\big(x_i, u_i, \overline{w}_i(x_i, u_i)\big), \qquad i = k, k+1, \ldots, N-1.$$

We then use $\overline{u}_k$ as the control for stage k.

An important enhancement of this idea is to use minimization over the first control u_k and to use the heuristic only for the remaining stages $k+1, \ldots, N-1$. To implement this variant of the CEC, we must apply at time k a control $\overline{u}_k$ that minimizes over $u_k \in U_k(x_k)$ the expression

$$g_k\big(x_k, u_k, \overline{w}_k(x_k, u_k)\big) + H_{k+1}\Big(f_k\big(x_k, u_k, \overline{w}_k(x_k, u_k)\big)\Big), \qquad (6.1)$$

where H_{k+1} is the cost-to-go function corresponding to the heuristic, i.e., $H_{k+1}(x_{k+1})$ is the cost incurred over the remaining stages $k+1, \ldots, N-1$ starting from a state x_{k+1}, using the heuristic, and assuming that the future disturbances will be equal to their typical values $\overline{w}_i(x_i, u_i)$. Note that for any next-stage state x_{k+1}, it is not necessary to have a closed-form expression for the heuristic cost-to-go $H_{k+1}(x_{k+1})$. Instead we can generate this cost by running the system forward from x_{k+1} and accumulating the corresponding single-stage costs. Since the heuristic must be run for each possible value of the control u_k to calculate the costs $H_{k+1}\big(f_k\big(x_k, u_k, \overline{w}_k(x_k, u_k)\big)\big)$ needed in the minimization, it is necessary to discretize the control constraint set if it is not already finite.

Note that the general structure of the preceding variant of the CEC is similar to the one of standard DP. It involves minimization of the expression (6.1), which is the sum of a current stage cost and a cost-to-go starting from the next state. The difference with DP is that the optimal cost-to-go $J^*_{k+1}(x_{k+1})$ is replaced by the heuristic cost $H_{k+1}(x_{k+1})$, and the disturbance w_k is replaced by its typical value $\overline{w}_k(x_k, u_k)$ (so that there is no need to take expectation over w_k). We thus encounter for the first time an important suboptimal control idea, based on an approximation to the DP algorithm: *minimizing at each stage k the sum of approximations to the current stage cost and the optimal cost-to-go*. This idea is central in other types of suboptimal control such as the limited lookahead, rollout, and model predictive control approaches, which will be discussed in Sections 6.3-6.5.

Partially Stochastic Certainty Equivalent Control

In the preceding descriptions of the CEC all future disturbances are fixed at their typical values. A useful variation for some imperfect state information problems is to take into account the stochastic nature of these disturbances, and to treat the problem as one of perfect state information, using an estimate $\overline{x}_k(I_k)$ of x_k as if it were exact. Thus, if $\{\mu^p_0(x_0), \ldots, \mu^p_{N-1}(x_{N-1})\}$ is an optimal policy obtained from the DP algorithm for the stochastic *perfect state information* problem

$$\text{minimize} \quad E\left\{g_N(x_N) + \sum_{k=0}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big)\right\}$$

$$\text{subject to} \quad x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big), \quad \mu_k(x_k) \in U_k, \quad k = 0, \ldots, N-1,$$

then the control input $\overline{\mu}_k(I_k)$ applied by this variant of CEC at time k is given by

$$\overline{\mu}_k(I_k) = \mu^p_k\big(\overline{x}_k(I_k)\big).$$

Generally, there are several variants of the CEC, where the stochastic uncertainty about some of the unknown quantities is explicitly dealt with, while all other unknown quantities are replaced by estimates obtained in a variety of ways. Let us provide some examples.

Example 6.1.1 (Multiaccess Communication)

Consider the slotted Aloha system described in Example 5.1.1. It is very difficult to obtain an optimal policy for this problem, primarily because there is no simple characterization of the conditional distribution of the state (the system backlog), given the channel transmission history. We therefore resort to a suboptimal policy. As discussed in Section 5.1, the perfect state information version of the problem admits a simple optimal policy:

$$\mu_k(x_k) = \frac{1}{x_k}, \qquad \text{for all } x_k \geq 1.$$

As a result, there is a natural partially stochastic CEC,

$$\overline{\mu}_k(I_k) = \min\left[1, \frac{1}{\overline{x}_k(I_k)}\right],$$

where $\overline{x}_k(I_k)$ is an estimate of the current packet backlog based on the entire past channel history of successes, idles, and collisions (which is I_k). Recursive estimators for generating $\overline{x}_k(I_k)$ are discussed by Mikhailov [Mik79], Hajek and van Loon [HaL82], Tsitsiklis [Tsi87], and Bertsekas and Gallager [BeG92].

Example 6.1.2 (Finite-State Systems with Imperfect State Information)

Consider the case where the system is a finite-state Markov chain under imperfect state information. The partially stochastic CEC approach is to solve the corresponding problem of perfect state information, and then use the controller thus obtained for control of the imperfectly observed system, modulo substitution of the exact state by an estimate obtained via the Viterbi algorithm described in Section 2.2.2. In particular, suppose that $\{\mu_0^p, \ldots, \mu_{N-1}^p\}$ is an optimal policy for the corresponding problem where the state is perfectly observed. Then the partially stochastic CEC, given the information vector I_k, uses the Viterbi algorithm to obtain (in real time) an estimate $\overline{x}(I_k)$ of the current state x_k, and applies the control

$$\overline{\mu}_k(I_k) = \mu_k^p\big(\overline{x}_k(I_k)\big).$$

Example 6.1.3 (Systems with Unknown Parameters)

We have been dealing so far with systems having a known system equation. In practice, however, there are many cases where the system parameters are not known exactly or change over time. One possible approach is to estimate the unknown parameters from input-output records of the system by using system identification techniques. This is a broad and important methodology, for which we refer to textbooks such as Kumar and Varaiya [KuV86], Ljung and Soderstrom [LjS83], and Ljung [Lju86]. However, system identification can be time consuming, and thus difficult to apply in an on-line control context. Furthermore, the estimation must be repeated if the parameters change.

The alternative is to formulate the stochastic control problem so that unknown parameters are dealt with directly. It can be shown that problems involving unknown system parameters can be embedded within the framework of our basic problem with imperfect state information by using state augmentation. Indeed, let the system equation be of the form

$$x_{k+1} = f_k(x_k, \theta, u_k, w_k),$$

where θ is a vector of unknown parameters with a given a priori probability distribution. We introduce an additional state variable $y_k = \theta$ and obtain a system equation of the form

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, y_k, u_k, w_k) \\ y_k \end{pmatrix}.$$

This equation can be written compactly as

$$\tilde{x}_{k+1} = \tilde{f}_k(\tilde{x}_k, u_k, w_k),$$

where $\tilde{x}_k = (x_k, y_k)$ is the new state, and $\tilde{f}_k$ is an appropriate function. The initial state is

$$\tilde{x}_0 = (x_0, \theta).$$

With a suitable reformulation of the cost function, the resulting problem becomes one that fits our usual framework.

Unfortunately, however, since y_k (i.e., θ) is unobservable, we are faced with a problem of imperfect state information even if the controller knows the state x_k exactly. Thus, typically an optimal solution cannot be found. Nonetheless, the partially stochastic CEC approach is often convenient. In particular, suppose that for a fixed parameter vector θ, we can compute the corresponding optimal policy

$$\big\{\mu_0^*(I_0, \theta), \ldots, \mu_{N-1}^*(I_{N-1}, \theta)\big\};$$

this is true for example if for a fixed θ, the problem is linear-quadratic of the type considered in Sections 4.1 and 5.2. Then a partially stochastic CEC takes the form

$$\overline{\mu}_k(I_k) = \mu_k^*(I_k, \hat{\theta}_k),$$

where $\hat{\theta}_k$ is some estimate of θ based on the information vector I_k. Thus, in this approach, the system is identified while it is being controlled. However, the estimates of the unknown parameters are used as if they were exact.

The approach of the preceding example is one of the principal methods of *adaptive control*, that is, control that adapts itself to changing values of system parameters. In the remainder of this section, we discuss some of the associated issues. Because adaptive control is somewhat disjoint from other material in the chapter, the reader may skip directly to Section 6.2.

6.1.1 Caution, Probing, and Dual Control

Suboptimal control is often guided by the qualitative nature of optimal control. It is therefore important to try to understand some of the characteristic features of the latter in the case where some of the system parameters are unknown. One of these is the need for balance between "caution" (the need for conservatism in applying control, since the system is not fully known), and "probing" (the need for aggressiveness in applying control, in order to excite the system enough to be able to identify it). These notions cannot be easily quantified, but often manifest themselves in specific control schemes. The following example provides some orientation; see also Bar-Shalom [Bar81].

Example 6.1.4 [Kum83]

Consider the linear scalar system

$$x_{k+1} = x_k + bu_k + w_k, \qquad k = 0, 1, \ldots, N-1,$$

and the quadratic terminal cost $E\{(x_N)^2\}$. Here everything is as in Section 4.1 (perfect state information) except that the control coefficient b is unknown. Instead, it is known that the a priori probability distribution of b is Gaussian with mean and variance

$$\bar{b} = E\{b\} > 0, \qquad \sigma_b^2 = E\{(b - \bar{b})^2\}.$$

Furthermore, w_k is zero mean Gaussian with variance σ_w^2 for each k.

Consider first the case where $N = 1$, so the cost is calculated to be

$$E\{(x_1)^2\} = E\{(x_0 + bu_0 + w_0)^2\} = x_0^2 + 2\bar{b}x_0u_0 + (\bar{b}^2 + \sigma_b^2)u_0^2 + \sigma_w^2.$$

The minimum over u_0 is attained at

$$u_0 = -\frac{\bar{b}}{\bar{b}^2 + \sigma_b^2}x_0,$$

and the optimal cost is verified by straightforward calculation to be

$$\frac{\sigma_b^2}{\bar{b}^2 + \sigma_b^2}x_0^2 + \sigma_w^2.$$

Therefore, the optimal control here is *cautious* in that the optimum $|u_0|$ decreases as the uncertainty in b (i.e., σ_b^2) increases.

Consider next the case where $N = 2$. The optimal cost-to-go at stage 1 is obtained by the preceding calculation:

$$J_1(I_1) = \frac{\sigma_b^2(1)}{\big(\bar{b}(1)\big)^2 + \sigma_b^2(1)}x_1^2 + \sigma_w^2, \tag{6.2}$$

where $I_1 = (x_0, u_0, x_1)$ is the information vector and

$$\bar{b}(1) = E\{b \mid I_1\}, \qquad \sigma_b^2(1) = E\big\{\big(b - \bar{b}(1)\big)^2 \mid I_1\big\}.$$

Let us focus on the term $\sigma_b^2(1)$ in the expression (6.2) for $J_1(I_1)$. We can obtain $\sigma_b^2(1)$ from the equation $x_1 = x_0 + bu_0 + w_0$ (which we view as a noise-corrupted measurement of b) and least-squares estimation theory (see Appendix E). The formula for $\sigma_b^2(1)$ will be of no further use to us, so we just state it without going into the calculation:

$$\sigma_b^2(1) = \frac{\sigma_b^2\sigma_w^2}{u_0^2\sigma_b^2 + \sigma_w^2}.$$

The salient feature of this equation is that $\sigma_b^2(1)$ is affected by the control u_0. Basically, if $|u_0|$ is small, the measurement $x_1 = x_0 + bu_0 + w_0$ is dominated by w_0 and the "signal-to-noise ratio" is small. Thus to achieve small error variance $\sigma_b^2(1)$ [which is desirable in view of Eq. (6.2)], we must apply a control u_0 that is large in absolute value. A choice of large control to enhance parameter identification is often referred to as *probing*. On the other hand, if $|u_0|$ is large, $|x_1|$ will also be large, and this is not desirable in view of Eq. (6.2). Therefore, in choosing u_0 we must strike a balance between caution (choosing a small value to keep x_1 reasonably small) and probing (choosing a large value to improve the signal-to-noise ratio and enhance estimation of b).

The tradeoff between the control objective and the parameter estimation objective is commonly referred to as *dual control*.

6.1.2 Two-Phase Control and Identifiability

An apparently reasonable form of suboptimal control in the presence of unknown parameters (cf. Example 6.1.3) is to separate the control process into two phases, a *parameter identification phase* and a *control phase*. In the first phase the unknown parameters are identified, while the control takes no account of the interim results of identification. The final parameter estimates from the first phase are then used to implement an optimal control law in the second phase. This alternation of identification and control phases may be repeated several times during any system run in order to take into account subsequent changes of the parameters.

One drawback of this approach is that information gathered during the identification phase is not used to adjust the control law until the beginning of the second phase. Furthermore, it is not always easy to determine when to terminate one phase and start the other.

A second difficulty, of a more fundamental nature, is due to the fact that the control process may make some of the unknown parameters invisible to the identification process. This is the problem of parameter *identifiability*, discussed by Ljung [Lju86], which is best explained by means of an example.

Example 6.1.5

Consider the scalar system

$$x_{k+1} = ax_k + bu_k + w_k, \qquad k = 0, 1, \ldots, N-1,$$

with the quadratic cost

$$E\left\{\sum_{k=1}^{N}(x_k)^2\right\}.$$

We assume perfect state information, so if the parameters a and b are known, this is a minimum variance control problem (cf. Section 5.3), and the optimal

control law is

$$\mu_k^*(x_k) = -\frac{a}{b}x_k.$$

Assume now that the parameters a and b are unknown, and consider the two-phase method. During the first phase the control law

$$\tilde{\mu}_k(x_k) = \gamma x_k \tag{6.3}$$

is used (γ is some scalar; for example, $\gamma = -\overline{a}/\overline{b}$, where $\overline{a}$ and $\overline{b}$ are a priori estimates of a and b, respectively). At the end of the first phase, the control law is changed to

$$\overline{\mu}_k(x_k) = -\frac{\hat{a}}{\hat{b}}x_k,$$

where $\hat{a}$ and $\hat{b}$ are the estimates obtained from the identification process. However, with the control law (6.3), the closed-loop system is

$$x_{k+1} = (a + b\gamma)x_k + w_k,$$

so the identification process can at best identify the value of $(a + b\gamma)$ but not the values of both a and b. In other words, the identification process cannot discriminate between pairs of values (a_1, b_1) and (a_2, b_2) such that $a_1 + b_1\gamma = a_2 + b_2\gamma$. Therefore, a and b are not identifiable when feedback control of the form (6.3) is applied.

One way to correct the difficulty is to add an additional known input δ_k to the control law (6.3); that is, use

$$\tilde{\mu}_k(x_k) = \gamma x_k + \delta_k.$$

Then the closed-loop system becomes

$$x_{k+1} = (a + b\gamma)x_k + b\delta_k + w_k,$$

and the knowledge of $\{x_k\}$ and $\{\delta_k\}$ makes it possible to identify $(a+b\gamma)$ and b. Given γ, one can then obtain estimates of a and b. Actually, to guarantee this in a more general context where the system is of higher dimension, the sequence $\{\delta_k\}$ must satisfy certain conditions: it must be "persistently exciting" (see for example Ljung and Soderstrom [LjS83] for further explanation of this concept).

A second possibility to bypass the identifiability problem is to change the structure of the system by artificially introducing a one-unit delay in the control feedback. Thus, instead of considering control laws of the form $\tilde{\mu}_k(x_k) = \gamma x_k$, as in Eq. (6.3), we consider controls of the form

$$u_k = \hat{\mu}_k(x_{k-1}) = \gamma x_{k-1}.$$

The closed-loop system then becomes

$$x_{k+1} = ax_k + b\gamma x_{k-1} + w_k,$$

and given γ, it is possible to identify both parameters a and b. This technique can be generalized for systems of arbitrary order, but artificially introducing a control delay makes the system less responsive to control.

6.1.3 Certainty Equivalent Control and Identifiability

At the opposite extreme of the two-phase method we have the certainty equivalent control approach, where the parameter estimates are incorporated into the control law as they are generated, and they are treated as if they were true values. In terms of the system

$$x_{k+1} = f_k(x_k, \theta, u_k, w_k)$$

considered in Example 6.1.3, suppose that, for each possible value of θ, the control law $\pi^*(\theta) = \{\mu_0^*(\cdot, \theta), \ldots, \mu_{N-1}^*(\cdot, \theta)\}$ is optimal with respect to a certain cost $J_\pi(x_0, \theta)$. Then the (suboptimal) control used at time k is

$$\hat{\mu}_k(I_k) = \mu_k^*(x_k, \hat{\theta}_k),$$

where $\hat{\theta}_k$ is an estimate of θ based on the information

$$I_k = \{x_0, x_1, \ldots, x_k, u_0, u_1, \ldots, u_{k-1}\}$$

available at time k; for example,

$$\hat{\theta}_k = E\{\theta \mid I_k\}$$

or, more likely in practice, an estimate obtained via an on-line system identification method (see [KuV86], [LjS83], [Lju86]).

One would hope that when the horizon is very long, the parameter estimates $\hat{\theta}_k$ will converge to the true value θ, so the certainty equivalent controller will become asymptotically optimal. Unfortunately, we will see that difficulties related to identifiability arise here as well.

Suppose for simplicity that the system is stationary with a priori known transition probabilities $P\{x_{k+1} \mid x_k, u_k, \theta\}$ and that the control law used is also stationary:

$$\hat{\mu}_k(I_k) = \mu^*(x_k, \hat{\theta}_k), \qquad k = 0, 1, \ldots$$

There are three systems of interest here (cf. Fig. 6.1.2):

(a) The system (perhaps falsely) believed by the controller to be true, which evolves probabilistically according to

$$P\{x_{k+1} \mid x_k, \mu^*(x_k, \hat{\theta}_k), \hat{\theta}_k\}.$$

(b) The true closed-loop system, which evolves probabilistically according to

$$P\{x_{k+1} \mid x_k, \mu^*(x_k, \hat{\theta}_k), \theta\}.$$

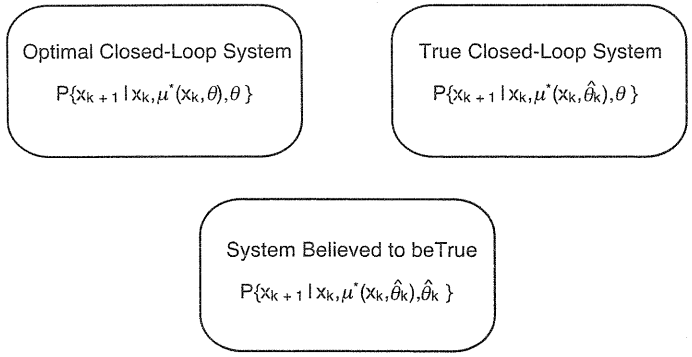


Figure 6.1.2 The three systems involved in certainty equivalent control, where θ is the true parameter and $\hat{\theta}_k$ is the parameter estimate at time k. Loss of optimality occurs when the true system differs asymptotically from the optimal closed-loop system. If the parameter estimates converge to some value $\hat{\theta}$, the true system typically becomes asymptotically equal to the system believed to be true. However, the parameter estimates need not converge, and even if they do, both systems may be different asymptotically from the optimal.

(c) The optimal closed-loop system that corresponds to the true value of the parameter, which evolves probabilistically according to

$$P\{x_{k+1} \mid x_k, \mu^*(x_k,\theta),\theta\}.$$

For asymptotic optimality, we would like the last two systems to be equal asymptotically. This will certainly be true if $\hat{\theta}_k \to \theta$. However, it is quite possible that either

(1) $\hat{\theta}_k$ does not converge to anything, or that

(2) $\hat{\theta}_k$ converges to a parameter $\hat{\theta} \neq \theta$.

There is not much we can say about the first case, so we concentrate on the second. To see how the parameter estimates can converge to a wrong value, assume that for some $\hat{\theta} \neq \theta$ and all x_{k+1}, x_k, we have

$$P\{x_{k+1} \mid x_k, \mu^*(x_k,\hat{\theta}),\hat{\theta}\} = P\{x_{k+1} \mid x_k, \mu^*(x_k,\hat{\theta}),\theta\}. \qquad (6.4)$$

In words, *there is a false value of parameter for which the system under closed-loop control looks exactly as if the false value were true.* Then, if the controller estimates at some time the parameter to be $\hat{\theta}$, subsequent data will tend to reinforce this erroneous estimate. As a result, a situation may develop where the identification procedure locks onto a wrong parameter value, regardless of how long information is collected. This is a difficulty with identifiability of the type discussed earlier in connection with two-phase control.

On the other hand, if the parameter estimates converge to some (possibly wrong) value, we can argue intuitively that the first two systems (believed and true) typically become equal in the limit as $k \to \infty$, since, generally, parameter estimate convergence in identification methods implies that the data obtained are asymptotically consistent with the view of the system one has based on the current estimates. However, the believed and true systems may or may not become asymptotically equal to the optimal closed-loop system. We first present two examples that illustrate how, even when the parameter estimates converge, the true closed-loop system can differ asymptotically from the optimal, thereby resulting in a certainty equivalent controller that is strictly suboptimal. We then discuss the special case of the self-tuning regulator for ARMAX models with unknown parameters, where, remarkably, it turns out that all three of the above systems are typically equal in the limit, even though the parameter estimates typically converge to false values.

Example 6.1.6 [BoV79]

Consider a two-state system with two controls u^1 and u^2. The transition probabilities depend on the control applied as well as a parameter θ, which is known to take one of two values θ^* and $\hat{\theta}$. They are as shown in Fig. 6.1.3. There is zero cost for a transition from state 1 to itself and a unit cost for all other transitions. Therefore, the optimal control at state 1 is the one that maximizes the probability of the state remaining at 1. Assume that the true parameter is θ^* and that

$$p_{11}(u^1,\hat{\theta}) > p_{11}(u^2,\hat{\theta}), \qquad p_{11}(u^1,\theta^*) < p_{11}(u^2,\theta^*).$$

Then the optimal control is u^2, but if the controller *thinks* that the true parameter is $\hat{\theta}$, it will apply u^1. Suppose also that

$$p_{11}(u^1,\hat{\theta}) = p_{11}(u^1,\theta^*).$$

Then, under u^1 the system looks identical for both values of the parameter, so if the controller estimates the parameter to be $\hat{\theta}$ and applies u^1, subsequent data will tend to reinforce the controller's belief that the true parameter is indeed $\hat{\theta}$.

More precisely, suppose that we estimate θ by selecting at each time k the value that maximizes

$$P\{\theta \mid I_k\} = \frac{P\{I_k \mid \theta\}P(\theta)}{P(I_k)},$$

where $P(\theta)$ is the a priori probability that the true parameter is θ (this is a popular estimation method). Then if $P(\hat{\theta}) > P(\theta^*)$, it can be seen, by using induction, that at each time k, the controller will estimate falsely θ to be $\hat{\theta}$ and apply the incorrect control u^1. To avoid the difficulty illustrated in this example, it has been suggested to occasionally deviate from the certainty

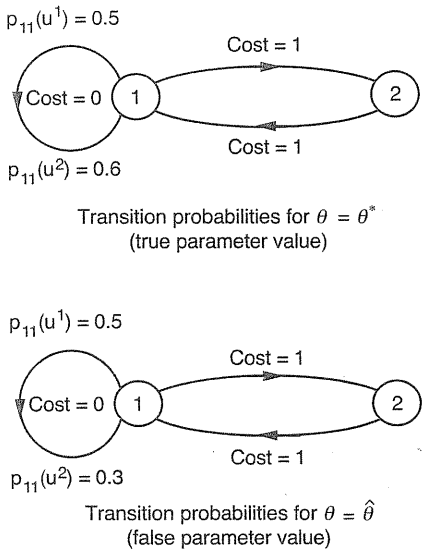


Figure 6.1.3 Transition probabilities for the two-state system of Example 6.1.6. Under the nonoptimal control u^1, the system looks identical under the true and the false values of the parameter θ.

equivalent control, applying other controls that enhance the identification of the unknown parameter (see Doshi and Shreve [DoS80], and Kumar and Lin [KuL82]). For example, by making sure that the control u^2 is used infrequently but infinitely often, we can guarantee that the correct parameter value will be identified by the preceding estimation scheme.

Example 6.1.7 [Kum83]

Consider the linear scalar system

$$x_{k+1} = ax_k + bu_k + w_k,$$

where we know that the parameters are either $(a, b) = (1, 1)$ or $(a, b) = (0, -1)$. The sequence $\{w_k\}$ is independent, stationary, zero mean, and Gaussian. The cost is quadratic of the form

$$\sum_{k=0}^{N-1} \big((x_k)^2 + 2(u_k)^2\big),$$

where N is very large, so the stationary form of the optimal control law is used (see Section 4.1). This control law can be calculated via the Riccati

equation to be

$$\mu^*(x_k) = \begin{cases} -\frac{x_k}{2} & \text{if } (a, b) = (1, 1), \\ 0 & \text{if } (a, b) = (0, -1). \end{cases}$$

To estimate (a, b), we use a least-squares identification method. The value of the least-squares criterion at time k is given by

$$V_k(1, 1) = \sum_{i=0}^{k-1} (x_{i+1} - x_i - u_i)^2, \qquad \text{for } (a, b) = (1, 1), \tag{6.5}$$

$$V_k(0, -1) = \sum_{i=0}^{k-1} (x_{i+1} + u_i)^2, \qquad \text{for } (a, b) = (0, -1). \tag{6.6}$$

The control applied at time k is

$$u_k = \tilde{\mu}_k(I_k) = \begin{cases} -\frac{x_k}{2} & \text{if } V_k(1, 1) < V_k(0, -1), \\ 0 & \text{if } V_k(1, 1) > V_k(0, -1). \end{cases}$$

Suppose the true parameters are $\theta = (0, -1)$. Then the true system evolves according to

$$x_{k+1} = -u_k + w_k. \tag{6.7}$$

If at time k the controller estimates incorrectly the parameters to be $\hat{\theta} = (1, 1)$, because $V_k(\hat{\theta}) < V_k(\theta)$, the control applied will be $u_k = -x_k/2$ and the *true* closed-loop system will evolve according to

$$x_{k+1} = \frac{x_k}{2} + w_k. \tag{6.8}$$

On the other hand, the controller *thinks* (given the estimate $\hat{\theta}$) that the closed-loop system will evolve according to

$$x_{k+1} = x_k + u_k + w_k = x_k - \frac{x_k}{2} + w_k = \frac{x_k}{2} + w_k, \tag{6.9}$$

so from Eqs. (6.7) and (6.8) we see that *under the control law $u_k = -x_k/2$, the closed-loop system evolves identically for both the true and the false values of the parameters* [cf. Eq. (6.4)].

To see what can go wrong, note that if $V_k(\hat{\theta}) < V_k(\theta)$ for some k we will have, from Eqs. (6.5)-(6.9),

$$x_{k+1} + u_k = x_{k+1} - x_k - u_k,$$

so from Eqs. (6.5) and (6.6) we obtain

$$V_{k+1}(\hat{\theta}) < V_{k+1}(\theta).$$

Therefore, if $V_1(\hat{\theta}) < V_1(\theta)$, the least-squares identification method will yield the wrong estimate $\hat{\theta}$ *for every* k. To see that this can happen with positive probability, note that, since the true system is $x_{k+1} = -u_k + w_k$, we have

$$V_1(\hat{\theta}) = (x_1 - x_0 - u_0)^2 = (w_0 - x_0 - 2u_0)^2,$$
$$V_1(\theta) = (x_1 + u_0)^2 = w_0^2.$$

Therefore, the inequality $V_1(\hat{\theta}) < V_1(\theta)$ is equivalent to

$$(x_0 + 2u_0)^2 < 2w_0(x_0 + 2u_0),$$

which will occur with positive probability since w_0 is Gaussian.

The preceding examples illustrate that loss of identifiability is a serious problem that frequently arises in the context of certainty equivalent control.

6.1.4 Self-Tuning Regulators

We described earlier the nature of the identifiability issue in certainty equivalent control: under closed-loop control, incorrect parameter estimates can make the system behave as if these estimates were correct [cf. Eq. (6.4)]. As a result, the identification scheme may lock onto false parameter values. This is not necessarily bad, however, since it may happen that the control law implemented on the basis of the false parameter values is near optimal. Indeed, through a fortuitous coincidence, it turns out that *in the practically important minimum variance control formulation (Section 5.3), when the parameter estimates converge, they typically converge to false values, but the resulting control law typically converges to the optimal.* We can get an idea about this phenomenon by means of an example.

Example 6.1.8

Consider the simplest ARMAX model:

$$y_{k+1} + ay_k = bu_k + \epsilon_{k+1}.$$

The minimum variance control law when a and b are known is

$$u_k = \mu_k(I_k) = \frac{a}{b} y_k.$$

Suppose now that a and b are not known but are identified on-line by means of some scheme. The control applied is

$$u_k = \frac{\hat{a}_k}{\hat{b}_k} y_k, \tag{6.10}$$

where $\hat{a}_k$ and $\hat{b}_k$ are the estimates obtained at time k. Then the difficulty with identifiability occurs when

$$\hat{a}_k \to \hat{a}, \qquad \hat{b}_k \to \hat{b},$$

where $\hat{a}$ and $\hat{b}$ are such that the true closed-loop system given by

$$y_{k+1} + ay_k = \frac{b\hat{a}}{\hat{b}} y_k + \epsilon_{k+1}$$

coincides with the closed-loop system that the controller thinks is true on the basis of the estimates $\hat{a}$ and $\hat{b}$. This latter system is

$$y_{k+1} = \epsilon_{k+1}.$$

For these two systems to be identical, we must have

$$\frac{a}{b} = \frac{\hat{a}}{\hat{b}},$$

which means that the control law (6.10) asymptotically becomes optimal despite the fact that the asymptotic estimates $\hat{a}$ and $\hat{b}$ may be incorrect.

Example 6.1.8 can be extended to the general ARMAX model of Section 5.3 with no delay:

$$y_k + \sum_{i=1}^{m} a_i y_{k-i} = \sum_{i=1}^{m} b_i u_{k-i} + \epsilon_k + \sum_{i=1}^{m} c_i \epsilon_{k-i}.$$

If the parameter estimates converge (regardless of the identification method used and regardless of whether the limit values are correct), then a minimum variance controller *thinks* that the closed-loop system is asymptotically

$$y_k = \epsilon_k.$$

Furthermore, parameter estimate convergence intuitively means that the true closed-loop system is also asymptotically $y_k = \epsilon_k$, and this is clearly the optimal closed-loop system. Results of this type have been proved in the literature in connection with several popular methods for parameter estimation. In fact, surprisingly, in some of these results, the model adopted by the controller is allowed to be incorrect to some extent.

One issue that we have not discussed is whether the parameter estimates indeed converge. A complete analysis of this issue is quite difficult. We refer to the survey paper by Kumar [Kum85], and the textbooks by Goodwin and Sin [GoS84], Kumar and Varaiya [KuV86], and Aström and Wittenmark [AsW90] for a discussion and sources on this subject. However, extensive simulations have shown that with proper implementation, these estimates typically converge for the type of systems likely to arise in many applications.

6.2 OPEN-LOOP FEEDBACK CONTROL

Generally, in a problem with imperfect state information, the performance of the optimal policy improves when extra information is available. However, the use of this information may make the DP calculation of the optimal policy intractable. This motivates an approximation, based on a more tractable computation that in part ignores the availability of extra information.

Let us consider the imperfect state information problem under the assumption of Section 5.4.1, which guarantees that the conditional state distribution is a sufficient statistic, i.e., that the probability distribution of the observation disturbance v_{k+1} depends explicitly only on the immediately preceding state, control, and system disturbance x_k, u_k, w_k, and not on $x_{k-1}, \ldots, x_0$, $u_{k-1}, \ldots, u_0$, $w_{k-1} \ldots, w_0$, $v_{k-1}, \ldots, v_0$.

We introduce a suboptimal policy known as the *open-loop feedback controller* (OLFC), which uses the current information vector I_k to determine $P_{x_k|I_k}$. However, it calculates the control u_k as if no further measurements will be received, by using an open-loop optimization over the future evolution of the system. In particular, u_k is determined as follows:

(1) Given the information vector I_k, compute the conditional probability distribution $P_{x_k|I_k}$ (in the case of perfect state information, where I_k includes x_k, this step is unnecessary).

(2) Find a control sequence $\{\overline{u}_k, \overline{u}_{k+1}, \ldots, \overline{u}_{N-1}\}$ that solves the open-loop problem of minimizing

$$E\left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, u_i, w_i) \;\middle|\; I_k \right\}$$

subject to the constraints

$$x_{i+1} = f_i(x_i, u_i, w_i), \qquad u_i \in U_i, \qquad i = k, k+1, \ldots, N-1.$$

(3) Apply the control input

$$\overline{\mu}_k(I_k) = \overline{u}_k.$$

Thus the OLFC uses at time k the new measurement z_k to calculate the conditional probability distribution $P_{x_k|I_k}$. However, it selects the control input as if future measurements will be disregarded.

Similar to the CEC, the OLFC requires the solution of N optimal control problems. Each problem may again be solved by deterministic optimal control techniques. The computations, however, may be more complicated than those for the CEC, since now the cost involves an expectation with

respect to the uncertain quantities. The main difficulty in the implementation of the OLFC is the computation of $P_{x_k|I_k}$. In many cases one cannot compute $P_{x_k|I_k}$ exactly, in which case some "reasonable" approximation scheme must be used. Of course, if we have perfect state information, this difficulty does not arise.

In any suboptimal control scheme, one would like to be assured that measurements are advantageously used. By this we mean that the scheme performs at least as well as any open-loop policy that uses a sequence of controls that is independent of the values of the measurements received. An optimal open-loop policy can be obtained by finding a sequence $\{u_0^*, u_1^*, \ldots, u_{N-1}^*\}$ that minimizes

$$\overline{J}(u_0, u_1, \ldots, u_{N-1}) = E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\}$$

subject to the constraints

$$x_{k+1} = f_k(x_k, u_k, w_k), \qquad u_k \in U_k, \qquad k = 0, 1, \ldots, N-1.$$

A nice property of the OLFC is that it performs at least as well as an optimal open-loop policy, as shown by the following proposition. By contrast, the CEC does not have this property (for a one-stage problem, the optimal open-loop controller and the OLFC are both optimal, but the CEC may be strictly suboptimal; see also Exercise 6.2).

Proposition 6.2.1: The cost $J_{\overline{\pi}}$ corresponding to an OLFC satisfies

$$J_{\overline{\pi}} \leq J_0^*, \tag{6.11}$$

where J_0^* is the cost corresponding to an optimal open-loop policy.

Proof: We assume throughout the proof that all expected values appearing are well defined and finite, and that the minimum in the following Eq. (6.14) is attained for every I_k. Let $\overline{\pi} = \{\overline{\mu}_0, \overline{\mu}_1, \ldots, \overline{\mu}_{N-1}\}$ be the OLFC. Its cost is given by

$$J_{\overline{\pi}} = \mathop{E}_{z_0}\{\overline{J}_0(I_0)\} = \mathop{E}_{z_0}\{\overline{J}_0(z_0)\}, \tag{6.12}$$

where $\overline{J}_0$ is obtained from the recursive algorithm

$$\overline{J}_{N-1}(I_{N-1}) = \mathop{E}_{x_{N-1}, w_{N-1}} \Big\{ g_N\big(f_{N-1}(x_{N-1}, \overline{\mu}_{N-1}(I_{N-1}), w_{N-1})\big) + g_{N-1}\big(x_{N-1}, \overline{\mu}_{N-1}(I_{N-1}), w_{N-1}\big) \mid I_{N-1} \Big\},$$

$$\overline{J}_k(I_k) = \mathop{E}_{x_k,w_k,v_{k+1}} \Big\{ g_k\big(x_k, \overline{\mu}_k(I_k), w_k\big) + \overline{J}_{k+1}\big(I_k, h_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(I_k), w_k\big), \overline{\mu}_k(I_k), v_{k+1}\big), \overline{\mu}_k(I_k)\big) \mid I_k \Big\}, \quad k = 0, \ldots, N-1, \tag{6.13}$$

where h_k is the function involved in the measurement equation as in the basic problem with imperfect state information of Section 5.1.

Consider the functions $J_k^c(I_k)$, $k = 0, 1, \ldots, N-1$, defined by

$$J_k^c(I_k) = \min_{\substack{u_i \in U_i \\ i=k,\ldots,N-1}} E\left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, u_i, w_i) \;\Big|\; I_k \right\}. \tag{6.14}$$

The minimization problem in this equation is the one that must be solved at time k in order to calculate the control $\overline{\mu}_k(I_k)$ of the OLFC. Clearly, $J_k^c(I_k)$ can be interpreted as the optimal open-loop cost from time k to time N when the current information vector is I_k. It can be seen that

$$\mathop{E}_{z_0}\big\{J_0^c(z_0)\big\} \leq J_0^*, \tag{6.15}$$

since J_0^* is the minimum over $u_0, \ldots, u_{N-1}$ of the total expected cost and can be written as

$$\min_{u_0,\ldots,u_{N-1}} \mathop{E}_{z_0}\big\{E\{\text{cost} \mid z_0\}\big\},$$

while $E_{z_0}\big\{J_0^c(z_0)\big\}$ can be written as

$$\mathop{E}_{z_0}\left\{ \min_{u_0,\ldots,u_{N-1}} E\{\text{cost} \mid z_0\} \right\}$$

(we generally have $E\{\min[\cdot]\} \leq \min[E\{\cdot\}]$). We will prove that

$$\overline{J}_k(I_k) \leq J_k^c(I_k), \qquad \text{for all } I_k \text{ and } k. \tag{6.16}$$

Then from Eqs. (6.12), (6.15), and (6.16), it will follow that

$$J_{\overline{\pi}} \leq J_0^*,$$

which is the relation to be proved. We show Eq. (6.16) by induction.

By the definition of the OLFC and Eq. (6.14), we have

$$\overline{J}_{N-1}(I_{N-1}) = J_{N-1}^c(I_{N-1}), \qquad \text{for all } I_{N-1},$$

and hence Eq. (6.16) holds for $k = N-1$. Assume that

$$\overline{J}_{k+1}(I_{k+1}) \leq J_{k+1}^c(I_{k+1}), \qquad \text{for all } I_{k+1}. \tag{6.17}$$

Then from Eqs. (6.13), (6.14), and (6.17), we have

$$\begin{aligned}
\overline{J}_k(I_k) &= \mathop{E}_{x_k,w_k,v_{k+1}} \Big\{ g_k\big(x_k, \overline{\mu}_k(I_k), w_k\big) \\
&\quad + \overline{J}_{k+1}\big(I_k, h_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(I_k), w_k\big), \overline{\mu}_k(I_k), v_{k+1}\big), \overline{\mu}_k(I_k)\big) \mid I_k \Big\} \\
&\leq \mathop{E}_{x_k,w_k,v_{k+1}} \Big\{ g_k(x_k, \overline{\mu}_k(I_k), w_k) \\
&\quad + J_{k+1}^c\big(I_k, h_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(I_k), w_k\big), \overline{\mu}_k(I_k), v_{k+1}\big), \overline{\mu}_k(I_k)\big) \mid I_k \Big\} \\
&= \mathop{E}_{x_k,w_k,v_{k+1}} \left\{ \min_{\substack{u_i \in U_i \\ i=k+1,\ldots,N-1}} \mathop{E}_{\substack{x_{k+1},w_i \\ x_{i+1}=f_i(x_i,u_i,w_i) \\ i=k+1,\ldots,N-1}} \Big\{ g_k\big(x_k, \overline{\mu}_k(I_k), w_k\big) \right. \\
&\quad \left. + \sum_{i=k+1}^{N-1} g_i(x_i, u_i, w_i) + g_N(x_N) \mid I_{k+1} \Big\} \;\Big|\; I_k \right\} \\
&\leq \min_{\substack{u_i \in U_i \\ i=k+1,\ldots,N-1}} \mathop{E}_{\substack{x_k,w_k,w_i \\ x_{i+1}=f_i(x_i,u_i,w_i) \\ i=k+1,\ldots,N-1 \\ x_{k+1}=f_k(x_k,\overline{\mu}_k(I_k),w_k)}} \Big\{ g_N(x_N) + g_k\big(x_k, \overline{\mu}_k(I_k), w_k\big) \\
&\quad + \sum_{i=k+1}^{N-1} g_i(x_i, u_i, w_i) \;\Big|\; I_k \Big\} \\
&= J_k^c(I_k).
\end{aligned}$$

The second inequality follows by interchanging expectation and minimization (since we generally have $E\{\min[\cdot]\} \leq \min[E\{\cdot\}]$) and by "integrating out" v_{k+1}. The last equality follows from the definition of OLFC. Thus Eq. (6.16) is proved for all k and the desired result is shown. **Q.E.D.**

The preceding proposition shows that the OLFC uses the measurements advantageously even though it selects at each period the present control input as if no further measurements will be taken in the future. It is worth noting that by Eq. (6.16), $J_k^c(I_k)$, which is the calculated open-loop optimal cost from time k to time N, provides a readily obtainable performance bound for the OLFC.

Partial Open-Loop Feedback Control

A form of suboptimal control that is intermediate between the optimal feedback controller and the OLFC is provided by a generalization of the OLFC called the *partial open-loop feedback controller* (POLFC for short). This controller uses past measurements to compute $P_{x|I_k}$, but calculates the control input on the basis that *some* (but not necessarily all) of the

measurements will in fact be taken in the future, and the remaining measurements will not be taken.

This method often allows one to deal with those measurements that are troublesome and complicate the solution. As an example consider an inventory problem such as the one considered in Section 4.2, where forecasts in the form of a probability distribution of each of the future demands become available over time. A reasonable form of POLFC calculates at each stage an optimal (s, S) policy based on the current forecast of future demands and follows this policy until a new forecast becomes available. When this happens, the current policy is abandoned in favor of a new one that is calculated on the basis of the new probability distribution of future demands, etc. Thus the complications due to the forecasts are bypassed at the expense of suboptimality of the policy obtained.

We note that an analog of Prop. 6.2.1 can be shown for the POLFC (see Bertsekas [Ber76]). In fact the corresponding error bound is potentially much better than the bound (6.11), reflecting the fact that the POLFC takes into account the future availability of some of the measurements.

We will discuss further the idea of ignoring a portion of the information for the purpose of obtaining a tractable suboptimal policy in Section 6.5.3. There we will generalize the OLFC and the POLFC by embedding them within a more general suboptimal scheme.

6.3 LIMITED LOOKAHEAD POLICIES

An effective way to reduce the computation required by DP is to truncate the time horizon and use at each stage a decision based on lookahead of a small number of stages. The simplest possibility is to use a *one-step lookahead policy* whereby at stage k and state x_k one uses the control $\overline{\mu}_k(x_k)$, which attains the minimum in the expression

$$\min_{u_k \in U_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}, \tag{6.18}$$

where $\tilde{J}_{k+1}$ is some approximation of the true cost-to-go function J_{k+1}, with $\tilde{J}_N = g_N$. Similarly, a *two-step lookahead policy* applies at time k and state x_k, the control $\overline{\mu}_k(x_k)$ attaining the minimum in the preceding equation, where now $\tilde{J}_{k+1}$ is obtained itself on the basis of a one-step lookahead approximation. In other words, for all possible states x_{k+1} that can be generated via the system equation starting from x_k,

$$x_{k+1} = f_k(x_k, u_k, w_k),$$

we have

$$\tilde{J}_{k+1}(x_{k+1}) = \min_{u_{k+1} \in U_{k+1}(x_{k+1})} E\Big\{ g_{k+1}(x_{k+1}, u_{k+1}, w_{k+1}) + \tilde{J}_{k+2}\big(f_{k+1}(x_{k+1}, u_{k+1}, w_{k+1})\big) \Big\},$$

where $\tilde{J}_{k+2}$ is some approximation of the cost-to-go function J_{k+2}. Policies with lookahead of more than two stages are similarly defined.

Note that the limited lookahead approach can be used equally well when the horizon is infinite. One simply uses as the terminal cost-to-go function an approximation to the optimal cost of the infinite horizon problem that starts at the end of the lookahead. Thus the following discussion, with a few straightforward modifications, applies to infinite horizon problems as well.

Given the approximations $\tilde{J}_k$ to the optimal costs-to-go, the computational savings of the limited lookahead approach are evident. For a one-step lookahead policy, only a single minimization problem has to be solved per stage, while in a two-step policy the corresponding number of minimization problems is one plus the number of all possible next states x_{k+1} that can be generated from the current state x_k.

However, even with readily available cost-to-go approximations $\tilde{J}_k$, the minimization over $u_k \in U_k(x_k)$ in the calculation of the one-step lookahead control [cf. Eq. (6.18)] may involve substantial computation. In a variant of the method that aims at reducing this computation, the minimization is done over a subset

$$\overline{U}_k(x_k) \subset U_k(x_k).$$

Thus, the control $\overline{\mu}_k(x_k)$ used in this variant is one that attains the minimum in the expression

$$\min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}. \tag{6.19}$$

A practical example of this approach is when by using some heuristic or approximate optimization, we identify a subset $\overline{U}_k(x_k)$ of promising controls, and to save computation, we restrict attention to this subset in the one-step lookahead minimization.

6.3.1 Performance Bounds for Limited Lookahead Policies

Let us denote by $\overline{J}_k(x_k)$ the expected cost-to-go incurred by a limited lookahead policy $\{\overline{\mu}_0, \overline{\mu}_1, \ldots, \overline{\mu}_{N-1}\}$ starting from state x_k at time k [$\overline{J}_k(x_k)$ should be distinguished from $\tilde{J}_k(x_k)$, the approximation of the cost-to-go that is used to compute the limited lookahead policy via the minimization in Eq. (6.19)]. It is generally difficult to evaluate analytically the functions $\overline{J}_k$, even when the functions $\tilde{J}_k$ are readily available. We thus aim to obtain some estimates of $\overline{J}_k(x_k)$. The following proposition gives a condition under which the one-step lookahead policy achieves a cost $\overline{J}_k(x_k)$ that is better than the approximation $\tilde{J}_k(x_k)$. The proposition also provides a readily computable upper bound to $\overline{J}_k(x_k)$.

Proposition 6.3.1: Assume that for all x_k and k, we have

$$\min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\} \leq \tilde{J}_k(x_k). \quad (6.20)$$

Then the cost-to-go functions $\overline{J}_k$ corresponding to a one-step lookahead policy that uses $\tilde{J}_k$ and $\overline{U}_k(x_k)$ [cf. Eq. (6.19)] satisfy for all x_k and k

$$\overline{J}_k(x_k) \leq \min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}. \quad (6.21)$$

Proof: For $k = 0, \ldots, N-1$, denote

$$\hat{J}_k(x_k) = \min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}, \quad (6.22)$$

and let $\hat{J}_N = g_N$. We must show that for all x_k and k, we have

$$\overline{J}_k(x_k) \leq \hat{J}_k(x_k).$$

We use backwards induction on k. In particular, we have $\overline{J}_N(x_N) = \hat{J}_N(x_N) = \tilde{J}_N(x_N) = g_N(x_N)$ for all x_N. Assuming that $\overline{J}_{k+1}(x_{k+1}) \leq \hat{J}_{k+1}(x_{k+1})$ for all x_{k+1}, we have

$$\begin{aligned}
\overline{J}_k(x_k) &= E\big\{ g_k\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \overline{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big) \big\} \\
&\leq E\big\{ g\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \hat{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big) \big\} \\
&\leq E\big\{ g\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big) \big\} \\
&= \hat{J}_k(x_k),
\end{aligned}$$

for all x_k. The first equality above follows from the DP algorithm that defines the costs-to-go $\overline{J}_k$ of the limited lookahead policy, while the first inequality follows from the induction hypothesis, and the second inequality follows from the assumption (6.20). This completes the induction proof. **Q.E.D.**

Note that by Eq. (6.21), the value $\hat{J}_k(x_k)$ of Eq. (6.22), which is the calculated one-step lookahead cost from state x_k at time k, provides a readily obtainable performance bound for the cost-to-go $\overline{J}_k(x_k)$ of the one-step lookahead policy. Furthermore, using also the assumption (6.20), we obtain for all x_k and k,

$$\overline{J}_k(x_k) \leq \tilde{J}_k(x_k),$$

i.e., *the cost-to-go of the one-step lookahead policy is no greater than the lookahead approximation on which it is based.* The critical assumption (6.20) in Prop. 6.3.1 can be verified in a few interesting special cases, as indicated by the following examples.

Example 6.3.1 (Rollout Algorithm)

Suppose that $\tilde{J}_k(x_k)$ is the cost-to-go of some given (suboptimal) heuristic policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ and that the set $\overline{U}_k(x_k)$ contains the control $\mu_k(x_k)$ for all x_k and k. The resulting one-step lookahead algorithm is called the *rollout algorithm* and will be discussed extensively in Section 6.4. From the DP algorithm (restricted to the given policy π), we have

$$\tilde{J}_k(x_k) = E\Big\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k(x_k, \mu_k(x_k), w_k)\big) \Big\},$$

which in view of the assumption $\mu_k(x_k) \in \overline{U}_k(x_k)$, yields

$$\tilde{J}_k(x_k) \geq \min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}.$$

Thus, the assumption of Prop. 6.3.1 is satisfied, and it follows that the rollout algorithm performs better than the heuristic on which it is based, starting from any state and stage.

Example 6.3.2 (Rollout Algorithm with Multiple Heuristics)

Consider a scheme that is similar to the one of the preceding example, except that $\tilde{J}_k(x_k)$ is the minimum of the cost-to-go functions corresponding to m heuristics, i.e.,

$$\tilde{J}_k(x_k) = \min\big\{ J_{\pi_1,k}(x_k), \ldots, J_{\pi_m,k}(x_k) \big\},$$

where for each j, $J_{\pi_j,k}(x_k)$ is the cost-to-go of a policy $\pi_j = \{\mu_{j,0}, \ldots, \mu_{j,N-1}\}$, starting from state x_k at stage k. From the DP algorithm, we have, for all j,

$$J_{\pi_j,k}(x_k) = E\Big\{ g_k\big(x_k, \mu_{j,k}(x_k), w_k\big) + J_{\pi_j,k+1}\big(f_k(x_k, \mu_{j,k}(x_k), w_k)\big) \Big\},$$

from which, using the definition of $\tilde{J}_k$, it follows that

$$\begin{aligned}
J_{\pi_j,k}(x_k) &\geq E\Big\{ g_k\big(x_k, \mu_{j,k}(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k(x_k, \mu_{j,k}(x_k), w_k)\big) \Big\} \\
&\geq \min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}.
\end{aligned}$$

Taking the minimum of the left-hand side over j, we obtain

$$\tilde{J}_k(x_k) \geq \min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}.$$

Thus, Prop. 6.3.1 implies that the one-step lookahead algorithm based on the heuristic algorithms' costs-to-go $J_{\pi_1,k}(x_k), \ldots, J_{\pi_m,k}(x_k)$ performs better than all these heuristics, starting from any state and stage.

Generally, the cost-to-go approximation functions $\tilde{J}_k$ need not satisfy the assumption (6.20) of Prop. 6.3.1. The following proposition does not require this assumption. It is useful in some contexts, including the case where the minimization involved in the calculation in the one-step-lookahead policy is not exact.

Proposition 6.3.2: Let $\tilde{J}_k$, $k = 0, 1, \ldots, N$, be functions of x_k with $\tilde{J}_N(x_N) = g_N(x_N)$ for all x_N, and let $\pi = \{\overline{\mu}_0, \overline{\mu}_1, \ldots, \overline{\mu}_{N-1}\}$ be a policy such that for all x_k and k, we have

$$E\Big\{g_k\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\Big\} \leq \tilde{J}_k(x_k) + \delta_k, \tag{6.23}$$

where $\delta_0, \delta_1, \ldots, \delta_{N-1}$ are some scalars. Then for all x_k and k, we have

$$J_{\pi,k}(x_k) \leq \tilde{J}_k(x_k) + \sum_{i=k}^{N-1} \delta_i,$$

where $J_{\pi,k}(x_k)$ is the cost-to-go of π starting from state x_k at stage k.

Proof: We use backwards induction on k. In particular, we have $J_{\pi,N}(x_N) = \tilde{J}_N(x_N) = g_N(x_N)$ for all x_N. Assuming that

$$J_{\pi,k+1}(x_{k+1}) \leq \tilde{J}_{k+1}(x_{k+1}) + \sum_{i=k+1}^{N-1} \delta_i$$

for all x_{k+1}, we have

$$\begin{aligned} J_{\pi,k}(x_k) &= E\Big\{g_k\big(x_k, \overline{\mu}_k(x_k), w_k\big) + J_{\pi,k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\Big\} \\ &\leq E\Big\{g\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\Big\} + \sum_{i=k+1}^{N-1} \delta_i \\ &\leq \tilde{J}_k(x_k) + \delta_k + \sum_{i=k+1}^{N-1} \delta_i, \end{aligned}$$

for all x_k. The first equality above follows from the DP algorithm that defines the costs-to-go $J_{\pi,k}$ of π, while the first inequality follows from the induction hypothesis, and the second inequality follows from the assumption (6.23). This completes the induction proof. **Q.E.D.**

Example 6.3.3 (Certainty Equivalent Control)

Consider the CEC for the case of a perfect state information problem, where each disturbance w_k is fixed at a nominal value $\overline{w}_k$, $k = 0, \ldots, N-1$, which is independent of x_k and u_k. Consider the optimal value of the problem solved by the CEC at state x_k and stage k,

$$\tilde{J}_k(x_k) = \min_{\substack{x_{i+1} = f_i(x_i, u_i, \overline{w}_i) \\ u_i \in U_i(x_i),\ i=k,\ldots,N-1}} \left[g_N(x_N) + \sum_{i=k}^{N-1} g_i(x_i, u_i, \overline{w}_i)\right],$$

and let $\tilde{J}_N(x_N) = g_N(x_N)$ for all x_N. Recall that the CEC applies the control $\overline{\mu}_k(x_k) = \overline{u}_k$ after finding an optimal control sequence $\{\overline{u}_k, \ldots, \overline{u}_{N-1}\}$ for the deterministic problem in the right-hand side above. Note also that the following DP equation

$$\tilde{J}_k(x_k) = \min_{u_k \in U_k(x_k)} \Big[g_k(x_k, u_k, \overline{w}_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, \overline{w}_k)\big)\Big]$$

holds, and that the control $\overline{u}_k$ applied by the CEC minimizes in the right-hand side.

Let us now apply Prop. 6.3.2 to derive a performance bound for the CEC. We have for all x_k and k,

$$\begin{aligned} \tilde{J}_k(x_k) &= g_k\big(x_k, \overline{\mu}_k(x_k), \overline{w}_k\big) + \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), \overline{w}_k\big)\big) \\ &= E\Big\{g\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\Big\} - \gamma_k(x_k) \end{aligned}$$

where $\gamma_k(x_k)$ is defined by

$$\begin{aligned} \gamma_k(x_k) = E\Big\{&g\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\Big\} \\ &- g_k\big(x_k, \overline{\mu}_k(x_k), \overline{w}_k\big) - \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), \overline{w}_k\big)\big). \end{aligned}$$

It follows that

$$E\Big\{g\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \tilde{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\Big\} \leq \tilde{J}_k(x_k) + \delta_k,$$

where

$$\delta_k = \max_{x_k} \gamma_k(x_k),$$

and by Prop. 6.3.2, we obtain the following bound for the cost-to-go function $\overline{J}_k(x_k)$ of the CEC:

$$\overline{J}_k(x_k) \leq \tilde{J}_k(x_k) + \sum_{i=k}^{N-1} \delta_i.$$

The preceding performance bound is helpful when it can be shown that $\delta_k \leq 0$ for all k, in which case we have $\overline{J}_k(x_k) \leq \tilde{J}_k(x_k)$ for all x_k and k. This is true for example if for all x_k and u_k, we have

$$E\big\{g(x_k, u_k, w_k)\big\} \leq g_k(x_k, u_k, \overline{w}_k),$$

and

$$E\{\tilde{J}_{k+1}(f_k(x_k,u_k,w_k))\} \leq \tilde{J}_{k+1}(f_k(x_k,u_k,\overline{w}_k)).$$

The most common way to assert that inequalities of this type hold is via some kind of concavity assumptions; for example, the inequalities hold if the state, control, and disturbance spaces are Euclidean spaces, $\overline{w}_k$ is the expected value of w_k, and the functions $g(x_k,u_k,\cdot)$ and $\tilde{J}_{k+1}(f_k(x_k,u_k,\cdot))$, viewed as functions of w_k, are concave (this is known as Jensen's inequality, and at least in the case where w_k takes a finite number of values, follows easily from the definition of concavity). It can be shown that the concavity conditions just described are guaranteed if the system is linear with respect to x_k and w_k, the cost functions g_k are concave with respect to x_k and w_k for each fixed u_k, the terminal cost function g_N is concave, and the control constraint sets U_k do not depend on x_k.

6.3.2 Computational Issues in Limited Lookahead

We now discuss the computation of the cost-to-go approximations and the corresponding minimization of the one-step lookahead costs.

Minimization Using Nonlinear Programming

One approach to obtain the control $\overline{\mu}_k(x_k)$ used by the one-step lookahead policy is to exhaustively calculate and compare the one-step lookahead costs of all the controls in the set $\overline{U}_k(x_k)$. In some cases, there is a more efficient alternative, which is to solve a suitable nonlinear programming problem. In particular, if the control space is the Euclidean space $\Re^m$, then for a one-step lookahead control calculation, we are faced with a minimization over a subset of $\Re^m$, which may be approached by continuous optimization/nonlinear programming techniques.

It turns out that even a multistage lookahead control calculation can be approached by nonlinear programming. In particular, assume that the disturbance can take a finite number of values, say r. Then, it can be shown that for a given initial state, an l-stage perfect state information problem (which corresponds to an l-step lookahead control calculation) can be formulated as a nonlinear programming problem with $m(1+r^{l-1})$ variables. We illustrate this by means of an important example where $l=2$ and then discuss the general case.

Example 6.3.4 (Two-Stage Stochastic Programming)

Here we want to find an optimal two-stage decision rule for the following situation: In the first stage we will choose a vector u_0 from a subset $U_0 \subset \Re^m$ with cost $g_0(u_0)$. Then an uncertain event represented by a random variable w will occur, where w will take one of the values $w^1,\ldots,w^r$ with corresponding probabilities $p^1,\ldots,p^r$. We will know the value w^j once it occurs, and we must

then choose a vector u_1^j from a subset $U_1(u_0,w^j) \subset \Re^m$ at a cost $g_1(u_1^j,w^j)$. The objective is to minimize the expected cost

$$g_0(u_0) + \sum_{j=1}^{r} p^j g_1(u_1^j,w^j)$$

subject to

$$u_0 \in U_0, \qquad u_1^j \in U_1(u_0,w^j), \quad j=1,\ldots,r.$$

This is a nonlinear programming problem of dimension $m(1+r)$ (the optimization variables are $u_0,u_1^1,\ldots,u_1^r$). It can also be viewed as a two-stage perfect state information problem, where $x_1=w_0$ is the state equation, w_0 can take the values $w^1,\ldots,w^r$ with probabilities $p^1,\ldots,p^r$, the cost of the first stage is $g_0(u_0)$, and the cost of the second stage is $g_1(x_1,u_1)$.

The preceding example can be generalized. Consider the basic problem of Chapter 1 for the case where there are only two stages ($l=2$) and the disturbances w_0 and w_1 can independently take one of the r values $w^1,\ldots,w^r$ with corresponding probabilities $p^1,\ldots,p^r$. The optimal cost function $J_0(x_0)$ is given by the two-stage DP algorithm

$$J_0(x_0) = \min_{u_0\in U_0(x_0)} \left[\sum_{j=1}^{r} p^j \left\{ g_0(x_0,u_0,w^j) + \min_{u_1^j \in U_1(f_0(x_0,u_0,w^j))} \left[\sum_{i=1}^{r} p^i \left\{ g_1\big(f_0(x_0,u_0,w^j),u_1^j,w^i\big) + g_2\big(f_1\big(f_0(x_0,u_0,w^j),u_1^j,w^i\big)\big)\right\}\right]\right\}\right].$$

This DP algorithm is equivalent to solving the nonlinear programming problem

$$\text{minimize} \quad \sum_{j=1}^{r} p^j \left\{ g_0(x_0,u_0,w^j) + \sum_{i=1}^{r} p^i \left\{ g_1\big(f_0(x_0,u_0,w^j),u_1^j,w^i\big) + g_2\big(f_1\big(f_0(x_0,u_0,w^j),u_1^j,w^i\big)\big)\right\}\right\}$$

$$\text{subject to } u_0 \in U_0(x_0), \qquad u_1^j \in U_1\big(f_0(x_0,u_0,w^j)\big), \quad j=1,\ldots,r.$$

If the controls u_0 and u_1 are elements of $\Re^m$, the number of variables in the above problem is $m(1+r)$. More generally, for an l-stage perfect state information problem a similar reformulation as a nonlinear programming problem requires $m(1+r^{l-1})$ variables. Thus if the number of lookahead stages is relatively small, a nonlinear programming approach may be the preferred option in calculating suboptimal limited lookahead policies.

Choosing the Approximate Cost-to-Go

A key issue in implementing a limited lookahead policy is the selection of the cost-to-go approximation at the final step. It may appear important at first that the true cost-to-go function should be approximated well over the range of relevant states; however, this is not necessarily true. What is important is that the *cost-to-go differentials (or relative values) be approximated well*; that is, for an l-step lookahead policy it is important to have

$$\tilde{J}_{k+l}(x) - \tilde{J}_{k+l}(x') \approx J_{k+l}(x) - J_{k+l}(x'),$$

for any two states x and x' that can be generated l steps ahead from the current state. For example, if equality were to hold above for all x, x', then $\tilde{J}_{k+l}(x)$ and $J_{k+l}(x)$ would differ by the same constant for each relevant x and the l-step lookahead policy would be optimal.

The manner in which the cost-to-go approximation is selected depends very much on the problem being solved. There is a wide variety of possibilities here. We will discuss three such approaches:

(a) *Problem Approximation*: The idea here is to approximate the optimal cost-to-go with some cost derived from a related but simpler problem (for example the optimal cost-to-go of that problem). This possibility is discussed and is illustrated with examples in Sections 6.3.3 and 6.3.4.

(b) *Parametric Cost-to-Go Approximation*: The idea here is to approximate the optimal cost-to-go with a function of a suitable parametric form, whose parameters are tuned by some heuristic or systematic scheme. This possibility is discussed in Section 6.3.5 and is illustrated using the computer chess paradigm. Additional methods of this type are discussed in Vol. II.

(c) *Rollout Approach*: Here the optimal cost-to-go is approximated by the cost of some suboptimal policy, which is calculated either analytically, or more commonly, by simulation. Generally, if a reasonably good suboptimal policy is known (e.g., a certainty equivalent or open-loop-feedback controller, or some other problem-dependent heuristic), it can be used to obtain a cost-to-go approximation. This approach is also particularly well-suited for deterministic and combinatorial problems. It is discussed at length in Section 6.4.

6.3.3 Problem Approximation - Enforced Decomposition

An often convenient approach for cost-to-go approximation is based on solution of a simpler problem that is tractable computationally or analytically. Here is an illustrative example, involving a convenient modification of the probabilistic structure of the problem.

Example 6.3.5

Consider the problem of an unscrupulous innkeeper who charges one of m different rates $r_1, \ldots, r_m$ for a room as the day progresses, depending on whether he has many or few vacancies, so as to maximize his expected total income during the day (Exercise 1.25 in Chapter 1). A quote of a rate r_i is accepted with probability p_i and is rejected with probability $1 - p_i$, in which case the customer departs, never to return during that day. When the number y of customers that will ask for a room during the rest of the day (including the customer currently asking for a room) is known and the number of vacancies is x, the optimal expected income $J(x, y)$ of the innkeeper is given by the DP recursion

$$J(x,y) = \max_{i=1,\ldots,m} \Big[p_i\big(r_i + J(x-1, y-1)\big) + (1-p_i)J(x, y-1)\Big],$$

for all $x \geq 1$ and $y \geq 1$, with initial conditions

$$J(x,0) = J(0,y) = 0, \qquad \text{for all } x \text{ and } y.$$

On the other hand, when the innkeeper does not know y at the times of decision, but instead only has a probability distribution for y, it can be seen that the problem becomes a difficult imperfect state information problem. Yet a reasonable one-step lookahead policy is based on approximating the optimal cost-to-go of subsequent decisions with $J(x-1, \overline{y}-1)$ or $J(x, \overline{y}-1)$, where the function J is calculated by the above recursion and $\overline{y}$ is the closest integer to the expected value of y. In particular, according to this one-step lookahead policy, when the innkeeper has a number of vacancies $x \geq 1$, he quotes to the current customer the rate that maximizes $p_i\big(r_i + J(x-1, \overline{y}-1) - J(x, \overline{y}-1)\big)$.

The preceding example is based on replacing the problem uncertainty (the random variable y) with a "certainty equivalent" (the scalar $\overline{y}$). The next example describes a generalization of this type of approximation, based on simplifying the stochastic structure of the problem.

Example 6.3.6 (Approximation Using Scenarios)

One possibility to approximate the optimal cost-to-go is to use certainty equivalence, in the spirit of Section 6.1. In particular, for a given state x_{k+1} at time $k+1$, we fix the remaining disturbances at some nominal values $\overline{w}_{k+1}, \ldots, \overline{w}_{N-1}$, and we compute an optimal control trajectory starting from x_{k+1} at time $k+1$. The corresponding cost, denoted by $\tilde{J}_{k+1}(x_{k+1})$, is used to approximate the optimal cost-to-go $J_{k+1}(x_{k+1})$ for the purpose of computing the corresponding one-step lookahead policy. Thus to compute the one-step lookahead control at state x_k, we need to solve a deterministic optimal control problem from all possible next states $f_k(x_k, u_k, w_k)$ and to evaluate the corresponding optimal cost $\tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big)$ based on the nominal values of the uncertainty.

A simpler but less effective variant of this approach is to compute $\tilde{J}_{k+1}(x_{k+1})$ as the cost-to-go of a given heuristic (rather than optimal) policy

for the deterministic problem that corresponds to the nominal values of the uncertainty and the starting state x_{k+1}. The advantage of using certainty equivalence here is that the potentially costly calculation of the expected value of the cost is replaced by a single state-control trajectory calculation.

The certainty equivalent approximation involves a single nominal trajectory of the remaining uncertainty. To strengthen this approach, it is natural to consider multiple trajectories of the uncertainty, called *scenarios*, and to construct an approximation to the optimal cost-to-go that involves, for every one of the scenarios, the cost of either an optimal or a given heuristic policy. Mathematically, we assume that we have a method, which at a given state x_{k+1}, generates M uncertainty sequences

$$w^m(x_{k+1}) = (w^m_{k+1}, \ldots, w^m_{N-1}), \qquad m = 1, \ldots, M.$$

These are the scenarios considered at state x_{k+1}. The cost $J_{k+1}(x_{k+1})$ is approximated by

$$\tilde{J}_{k+1}(x_{k+1}, r) = r_0 + \sum_{m=1}^{M} r_m C_m(x_{k+1}), \tag{6.24}$$

where $r = (r_0, r_1, \ldots, r_M)$ is a vector of parameters, and $C_m(x_{k+1})$ is the cost corresponding to an occurrence of the scenario $w^m(x_{k+1})$, when starting from state x_{k+1} and using either an optimal or a given heuristic policy.

The parameters $r_0, r_1, \ldots, r_M$ may depend on the time index, and in more sophisticated schemes, they may depend on some characteristics of the state (see our subsequent discussion of feature-based architectures in Section 6.3.5). We may interpret the parameter r_m as an "aggregate weight" that encodes the aggregate effect on the cost-to-go function of uncertainty sequences that are similar to the scenario $w^m(x_{k+1})$. Note that, if $r_0 = 0$, the approximation (6.24) may also be viewed as a calculation by *limited simulation*, based on just the M scenarios $w^m(x_{k+1})$, and using the weights r_m as "aggregate probabilities." One difficulty with this approach is that we have to choose the parameters $(r_0, r_1, \ldots, r_M)$. For this, we may either use some heuristic scheme based on trial and error, or some of the more systematic schemes of neuro-dynamic programming, discussed in Vol. II.

We finally mention a variation of the scenario-based approximation method, whereby only a portion of the future uncertain quantities are fixed at nominal scenario values, while the remaining uncertain quantities are explicitly viewed as random. The cost of scenario m at state x_{k+1} is now a random variable, and the quantity $C_m(x_{k+1})$ used in Eq. (6.24) should be the *expected* cost of this random variable. This variation is appropriate and makes practical sense as long as the computation of the corresponding expected scenario costs $C_m(x_{k+1})$ is convenient.

Enforced Decomposition of Weakly Coupled Systems

The simplification/approximation approach is often well-suited for problems that involve a number of subsystems that may be coupled through

the system equation, or the cost function, or the control constraints, but the degree of coupling is "relatively weak." It is difficult to define precisely what constitutes "weak coupling," but in specific problem contexts, usually this type of structure is easily recognized. For such problems it is often sensible to introduce approximations by artificially decoupling the subsystems in some way, thereby creating either a simpler problem or a simpler cost calculation, where subsystems can be dealt with in isolation. There are a number of different ways to effect this type of artificial decomposition, and the best approach is often problem-dependent.

As an example consider a deterministic problem, where the control u_k at time k consists of m components, $u_k = \{u^1_k, \ldots, u^m_k\}$, with u^i_k corresponding to the ith subsystem. Then to compute a cost-to-go approximation at a given state x_k, one may try a one-subsystem-at-a-time approach: first optimize over the control sequence $\{u^1_k, u^1_{k+1}, \ldots, u^1_{N-1}\}$ of the first subsystem, while keeping the controls of the remaining subsystems at some nominal values, then minimize over the controls of the second subsystem, while keeping the controls of the first subsystem at the "optimal" values just computed and the controls of subsystems $3, \ldots, m$ to the nominal values, and continue in this manner. There are several possible variations, for example to make the order in which the subsystems are considered subject to optimization as well. Let us illustrate this approach by means of an example.

Example 6.3.7 (Vehicle Routing)

Consider m vehicles that move along the arcs of a given graph. Each node of the graph has a given "value" and the first vehicle that will pass through the node will collect its value, while vehicles that pass subsequently through the node do not collect any value. This may serve as a model of a situation where there are various valuable tasks to be performed at the nodes of a transportation network, and each task can be performed at most once and by a single vehicle. We assume that each vehicle starts at a given node and after at most a given number of arc moves, it must return to some other given node. The problem is to find a route for each vehicle satisfying these constraints, so that the total value collected by the vehicles is maximized.

This is a difficult combinatorial problem that in principle can be approached by DP. In particular, we can view as the state the set of current positions of the vehicles together with the list of nodes that have been traversed by some vehicle in the past, and have thus "lost" their value. Unfortunately, the number of these states is enormous (it increases exponentially with the number of nodes and the number of vehicles). The version of the problem involving a single vehicle, while still difficult in principle, can often be solved in reasonable time either exactly by DP or fairly accurately using a heuristic. Thus a one step-lookahead policy suggests itself, with the value-to-go approximation obtained by solving single vehicle problems.

In particular, in a one step-lookahead scheme, at a given time k and from a given state we consider all possible kth moves by the vehicles, and at

the resulting states we approximate the optimal value-to-go with the value corresponding to a suboptimal set of paths. These paths are obtained as follows: we fix an order of the vehicles and we calculate a path for the first vehicle, assuming the other vehicles do not move. (This is done either optimally by DP, or nearly optimally using some heuristic.) Then we calculate a path for the second vehicle in the order, taking into account the value collected by the first vehicle, and we similarly continue: for each vehicle, we calculate in the given order a path, taking into account the value collected by the preceding vehicles. We end up with a set of paths that have a certain total value associated with them, and which correspond to the particular order for considering the vehicles. We can also calculate other sets of paths and their corresponding total values, for other orders (possibly all orders) for considering the vehicles. We then use as the value-to-go approximation at the given state the maximal value over all the sets of paths computed from that state.

Another context where enforced decomposition may be an attractive possibility, is when the subsystems are coupled only through the disturbance. In particular, consider m subsystems of the form

$$x_{k+1}^i = f^i(x_k^i, u_k^i, w_k^i), \qquad i = 1, \ldots, m.$$

Here the ith subsystem has its own state x_k^i, control u_k^i, and cost per stage $g^i(x_k^i, u_k^i, w_k^i)$, but the probability distribution of w_k^i depends on the full state

$$x_k = (x_k^1, \ldots, x_k^m).$$

A natural form of suboptimal control is to solve at each stage k and for each i, the ith subsystem optimization problem where the probability distribution of each of the future disturbances $w_{k+1}^i, \ldots, w_{N-1}^i$ is fixed at some distribution that depends only on the corresponding "local" states $x_{k+1}^i, \ldots, x_{N-1}^i$. This distribution may be derived on the basis of some nominal values $\overline{x}_{k+1}^j, \ldots, \overline{x}_{N-1}^j$, $j \neq i$, of the future states of the other subsystems, and these nominal values may in turn depend on the full current state x_k. The first control $\overline{u}_k^i$ in the optimal policy thus obtained is applied at the ith subsystem in stage k, and the remaining portion of this policy is discarded.

Let us also discuss in some detail an example of subsystem decomposition where the coupling comes through the control constraint.

Example 6.3.8 (Flexible Manufacturing)

Flexible manufacturing systems (FMS) provide a popular approach for increasing productivity in manufacturing small batches of related parts. There are several workstations in an FMS, and each is capable of carrying out a variety of operations. This allows the simultaneous manufacturing of more than one part type, reduces idle time, and allows production to continue even when a workstation is out of service because of failure or maintenance.

Consider a work center in which n part types are produced. Denote

u_k^i: the amount of part i produced in period k.

d_k^i: a known demand for part i in period k.

x_k^i: the cumulative difference of amount of part i produced and demanded up to period k.

Let us denote also by x_k, u_k, d_k the n-dimensional vectors with coordinates x_k^i, u_k^i, d_k^i, respectively. We then have

$$x_{k+1} = x_k + u_k - d_k. \tag{6.25}$$

The work center consists of a number of workstations that fail and get repaired in random fashion, thereby affecting the productive capacity of the system (i.e., the constraints on u_k). Roughly, our problem is to schedule part production so that x_k is kept around zero to the extent possible.

The productive capacity of the system depends on a random variable α_k that reflects the status of the workstations. In particular, we assume that the production vector u_k must belong to a constraint set $U(\alpha_k)$. We model the evolution of α_k by a Markov chain with known transition probabilities $P(\alpha_{k+1} \mid \alpha_k)$. In practice, these probabilities must be estimated from individual station failure and repair rates, but we will not go into the matter further. Note also that in practice these probabilities may depend on u_k. This dependence is ignored for the purpose of development of a cost-to-go approximation. It may be taken into account when the actual suboptimal control is computed.

We select as system state the pair (x_k, α_k), where x_k evolves according to Eq. (6.25) and α_k evolves according to the Markov chain described earlier. The problem is to find for every state (x_k, α_k) a production vector $u_k \in U(\alpha_k)$ such that a cost function of the form

$$J_\pi(x_0) = E\left\{\sum_{k=0}^{N-1}\sum_{i=1}^{n} g^i(x_k^i)\right\}$$

is minimized. The function g^i expresses the desire to keep the current backlog or surplus of part i near zero. Two examples are $g^i(x^i) = \beta_i|x^i|$ and $g^i(x^i) = \beta_i(x^i)^2$, where $\beta_i > 0$.

The DP algorithm for this problem is

$$\begin{aligned} J_k(x_k, \alpha_k) = {} & \sum_{i=1}^{n} g^i(x_k^i) \\ & + \min_{u_k \in U(\alpha_k)} E_{\alpha_{k+1}}\left\{J_{k+1}(x_k + u_k - d_k, \alpha_{k+1}) \mid \alpha_k\right\}. \end{aligned} \tag{6.26}$$

If there is only one part type ($n = 1$), the optimal policy can be fairly easily determined from this algorithm (see Exercise 6.7). However, in general, the algorithm requires a prohibitive amount of calculation for an FMS of realistic size (say for $n > 10$ part types). We thus consider a one-step lookahead policy

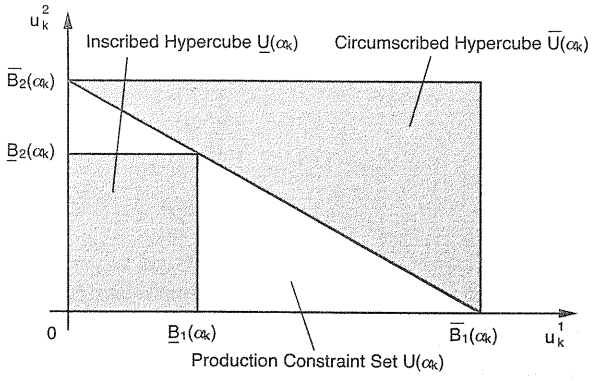


Figure 6.3.1 Inner and outer approximations of the production capacity constraint set by hypercubes in the flexible manufacturing example.

with the cost-to-go J_{k+1} replaced by an approximation $\tilde{J}_{k+1}$ that exploits the nearly separable structure of our problem.

In particular, we note that the problem can to a large extent be decomposed with respect to individual part types. Indeed, the system equation (6.25) and the cost per stage are decoupled, and the only coupling between parts comes from the constraint $u_k \in U(\alpha_k)$. Suppose we approximate $U(\alpha_k)$ by inner and outer approximating hypercubes $\underline{U}(\alpha_k)$ and $\overline{U}(\alpha_k)$ of the form

$$\underline{U}(\alpha_k) = \left\{u_k^i \mid 0 \leq u_k^i \leq \underline{B}_i(\alpha_k)\right\},$$

$$\overline{U}(\alpha_k) = \left\{u_k^i \mid 0 \leq u_k^i \leq \overline{B}_i(\alpha_k)\right\},$$

$$\underline{U}(\alpha_k) \subset U(\alpha_k) \subset \overline{U}(\alpha_k),$$

as shown in Fig. 6.3.1. If $U(\alpha_k)$ is replaced for each α_k by either $\overline{U}(\alpha_k)$ or $\underline{U}(\alpha_k)$, then the problem is decomposed completely with respect to part types. For every part i the DP algorithm for the outer approximation is given by

$$\overline{J}_k^i(x_k^i, \alpha_k) = g^i(x_k^i) + \min_{0 \leq u_k^i \leq \overline{B}_i(\alpha_k)} E_{\alpha_{k+1}} \left\{\overline{J}_k^i(x_k^i + u_k^i - d_k^i, \alpha_{k+1}) \mid \alpha_k\right\}, \tag{6.27}$$

and for the inner approximation it is given by

$$\underline{J}_k^i(x_k^i, \alpha_k) = g^i(x_k^i) + \min_{0 \leq u_k^i \leq \underline{B}_i(\alpha_k)} E_{\alpha_{k+1}} \left\{\underline{J}_{k+1}^i(x_k^i + u_k^i - d_k^i, \alpha_{k+1}) \mid \alpha_k\right\}. \tag{6.28}$$

Furthermore, since $\underline{U}(\alpha_k) \subset U(\alpha_k) \subset \overline{U}(\alpha_k)$, the cost-to-go functions $\overline{J}_k^i$ and $\underline{J}_k^i$ provide lower and upper bounds to the true cost-to-go function J_k,

$$\sum_{i=1}^n \overline{J}_k^i(x_k^i, \alpha_k) \leq J_k(x_k, \alpha_k) \leq \sum_{i=1}^n \underline{J}_k^i(x_k^i, \alpha_k),$$

and can be used to construct approximations to J_k that are suitable for a one-step lookahead policy. A simple possibility is to adopt the averaging approximation

$$\tilde{J}_k(x_k, \alpha_k) = \frac{1}{2} \sum_{i=1}^n \left(\overline{J}_k^i(x_k^i, \alpha_k) + \underline{J}_k^i(x_k^i, \alpha_k)\right)$$

and use at state (x_k, α_k) the control $\tilde{u}_k$ that minimizes [cf. Eq. (6.26)]

$$E\left\{\sum_{i=1}^n \left(\overline{J}_{k+1}^i(x_k^i + u_k^i - d_k^i, \alpha_{k+1}) + \underline{J}_{k+1}^i(x_k^i + u_k^i - d_k^i, \alpha_{k+1})\right) \;\middle|\; \alpha_k\right\} \tag{6.29}$$

over all $u_k \in U(\alpha_k)$. Multiple upper bound approximations, based on multiple choices of $\underline{B}_i(\alpha_k)$, can also be used.

To implement this scheme, it is necessary to compute and store the approximate cost-to-go functions $\overline{J}_k^i$ and $\underline{J}_k^i$ in tables, so that they can be used in the real-time computation of the suboptimal control via the minimization of expression (6.29). The corresponding calculations [cf. the DP algorithms (6.27) and (6.28)] are nontrivial, but they can be carried out off-line, and in any case they are much less than what would be required to compute the optimal controller. The feasibility and the benefits of the overall approach have been demonstrated by simulation in the thesis by Kimemia [Kim82], on which this example is based. See also Kimemia, Gershwin, and Bertsekas [KGB82], and Tsitsiklis [Tsi84a].

For some other examples of decomposition approaches, see Wu and Bertsekas [WuB99], which deals with admission control in cellular communication networks, and Meuleau et. al. [MHK98], which deals with problems of resource allocation.

6.3.4 Aggregation

An alternative method for constructing a simpler and more tractable problem is based on reducing the number of states by "combining" many of them together into *aggregate states*. This results in an *aggregate problem*, with fewer states, which may be solvable by exact DP methods. The optimal cost-to-go functions of the aggregate problem is then used to construct a one-step-lookahead cost approximation for the original problem. The precise form of the aggregate problem may depend on intuition and/or heuristic reasoning, based on our understanding of the original problem.

In this subsection, we will discuss various aggregation methods, starting with the case of a finite-state problem. We will focus on defining the transition probabilities and costs of the aggregate problem, and to simplify notation, we suppress the time indexing in what follows. We generically denote:

I, $\overline{I}$: The set of states of the original system at the current and the next stage, respectively.

$p_{ij}(u)$: The transition probability of the original system from state $i \in I$ to state $j \in \overline{I}$ under control u.

$g(i,u,j)$: The transition cost of the original system from state $i \in I$ to state $j \in \overline{I}$ under control u.

S, $\overline{S}$: The set of states of the aggregate system at the current and the next stage, respectively.

$r_{st}(u)$: The transition probability of the aggregate system from state $s \in S$ to state $t \in \overline{S}$ under control u.

$h(s,u)$: The expected transition cost of the aggregate system from state $s \in S$ under control u.

For simplicity, we assume that the control constraint set $U(i)$ is the same for all states $i \in I$. This common control constraint set, denoted by U, is chosen as the control constraint set at all states $s \in S$ of the aggregate problem.

There are several types of aggregation methods, which bring to bear intuition about the problem's structure in different ways. All these methods are based on two (somewhat arbitrary) choices of probabilities, which relate the original system states with the aggregate states:

(1) For each aggregate state $s \in S$ and original system state $i \in I$, we specify the *disaggregation probability* q_{si} (we have $\sum_{i \in I} q_{si} = 1$ for each $s \in S$). Roughly, q_{si} may be interpreted as the "degree to which s is represented by i."

(2) For each original system state $j \in \overline{I}$ and aggregate state $t \in \overline{S}$, we specify the *aggregation probability* w_{jt} (we have $\sum_{t \in \overline{S}} w_{jt} = 1$ for each $j \in \overline{I}$). Roughly, w_{jt} may be interpreted as the "degree of membership of j in the aggregate state t."

Note that in general, the disaggregation and aggregation probabilities may change at each stage (since the state space may change at each stage). On the other hand, for a stationary problem, where state and control spaces, system equation, and cost per stage that are the same for all stages, the disaggregation and aggregation probabilities will ordinarily also be the same for all stages.

As an illustration consider the following example of aggregation.

Example 6.3.9 (Hard Aggregation)

We are given a partition of the original system state spaces I and $\overline{I}$ into subsets of states (each state belongs to one and only one subset). We view each subset as an aggregate state. This corresponds to aggregation probabilities

$$w_{jt} = 1 \qquad \text{if state } j \in \overline{I} \text{ belongs to aggregate state/subset } t \in \overline{S},$$

and (assuming all states that belong to aggregate state/subset s are "equally representative") disaggregation probabilities

$$q_{si} = 1/n_s \qquad \text{if state } i \in I \text{ belongs to aggregate state/subset } s \in S,$$

where n_s is the number of states of s.

Given the disaggregation and aggregation probabilities, q_{si} and w_{jt}, and the original transition probabilities $p_{ij}(u)$, we envisage an aggregate system where state transitions occur as follows:

(i) From aggregate state s, generate state i according to q_{si}.

(ii) Generate a transition from i to j according to $p_{ij}(u)$, with cost $g(i,u,j)$.

(iii) From state j, generate aggregate state t according to w_{jt}.

Then, the transition probability from aggregate state s to aggregate state t under u, and the corresponding expected transition cost, are given by

$$r_{st}(u) = \sum_{i \in I} \sum_{j \in \overline{I}} q_{si} p_{ij}(u) w_{jt},$$

$$h(s,u) = \sum_{i \in I} \sum_{j \in \overline{I}} q_{si} p_{ij}(u) g(i,u,j).$$

These transition probabilities and costs define the aggregate problem. After solving for the optimal costs-to-go $\hat{J}(t)$, $t \in \overline{S}$, of the aggregate problem, the costs of the original problem are approximated by

$$\tilde{J}(j) = \sum_{t \in \overline{S}} w_{jt} \hat{J}(t), \qquad j \in \overline{I}. \tag{6.30}$$

As an illustration, for the preceding hard aggregation Example 6.3.9, the aggregate system transition process works as follows: Starting from an aggregate state/subset s, we generate with equal probability a state i in s, then a next state $j \in \overline{I}$ according to the transition probabilities $p_{ij}(u)$, and then we declare as next aggregate state the subset t to which j belongs. The corresponding transition probability and expected transition cost are

$$r_{st}(u) = \frac{1}{n_s} \sum_{i \in s} \sum_{j \in t} p_{ij}(u),$$

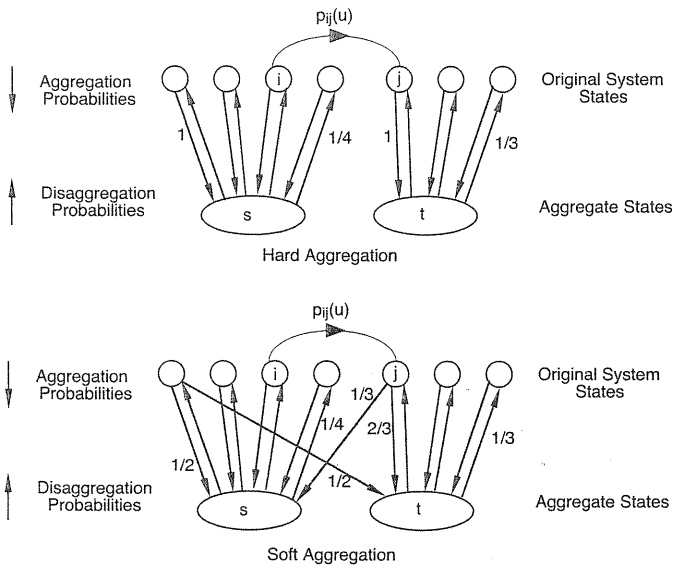


Figure 6.3.1 Disaggregation and aggregation probabilities in the hard and soft aggregation Examples 6.3.9 and 6.3.10. The difference is that in soft aggregation, some of the aggregation probabilities are strictly between 0 and 1, i.e., a state of the original system may be associated with multiple aggregate states.

$$h(s,u) = \frac{1}{n_s} \sum_{i \in s} \sum_{j \in \overline{I}} p_{ij}(u) g(i,u,j).$$

After computing the optimal costs $\hat{J}(t)$ of the aggregate problem, we use Eq. (6.30) to obtain the approximate cost function $\tilde{J}(j)$, which for this hard aggregation example is piecewise constant, with all states $j \in \overline{S}$ that belong to the same aggregate state/subset t having the same value of $\tilde{J}(j)$.

Example 6.3.10 (Soft Aggregation)

In hard aggregation, the aggregate states/subsets are disjoint, and each original system state is associated with a single aggregate state. A generalization is to allow the aggregate states/subsets to overlap, with the aggregation probabilities w_{jt} quantifying the "degree of membership" of j in the aggregate state t. Thus, an original system state j may be a member of multiple aggregate states/subsets t, and if this is so, the aggregation probabilities w_{jt} will be positive but less than 1 for all t that contain j (we should still have $\sum_{t \in \overline{S}} w_{jt} = 1$); see Fig. 6.3.1.

For example, assume that we are dealing with a queue that has space for 100 customers, and that the state is the number of spaces occupied at a given time. Suppose that we introduce four aggregate states: "nearly empty" (0-10 customers), "lightly loaded" (11-50 customers), "heavily loaded" (51-90 customers), and "nearly full" (91-100 customers). Then it makes sense to use soft aggregation, so that a state with close to 50 customers is classified neither as "lightly loaded" nor as "heavily loaded," but is viewed instead as associated with both of these aggregate states, to some degree.

It can be seen from Eq. (6.30), that in soft aggregation, the approximate cost-to-go function $\tilde{J}$ is not piecewise constant, as in the case of hard aggregation, and varies "smoothly" along the boundaries separating aggregate states. This is because original system states that belong to multiple aggregate states/subsets have approximate cost-to-go that is a convex combination of the costs-to-go of these aggregate states.

The choices of aggregate states, and aggregation and disaggregation probabilities are often guided by insight into the problem's structure. The following is an example.

Example 6.3.11 (Admission Control in a Service Facility)

Consider a facility that serves m types of customers. At each time period, a single customer of each type arrives, and requires a level of service per unit time that is random and has a given distribution (which depends on the customer type). The facility must decide which of the arriving customers to admit at each time period. The total service capacity of the facility is given, and must at all times be no less than the sum of service levels of the customers currently within the facility. An admitted customer of a given type has a given probability of leaving the facility at each time period, independently of his required level of service and of how long he has already been in the facility. An admitted customer also pays to the facility a given amount per period, which is proportional to his required level of service (with the constant of proportionality depending on the customer type). The objective here is to maximize the expected revenue of the facility over a given horizon. Thus the tradeoff is to provide selective preference to high-paying customer types, or alternatively expressed, to avoid filling up the facility with low-paying long-staying customers, thereby potentially blocking out some high-paying customers.

In a problem of this type, the system state, just prior to making an admission decision, is the entire list of customers of each type within the facility as well as their service levels (together with the list of service levels of the customers that have just arrived). There is clearly an extremely large number of states. Intuition suggests here that it is appropriate to aggregate all customers of a given type, and represent them with their total service level. Thus the aggregate state in this approach is the list of total required service level of each type within the facility (together with the list of service levels of the customers that have just arrived - an uncontrollable state component, cf. Section 1.4). Clearly it is much easier to deal with such a space of aggregate states in a DP context.

The choice of aggregation probabilities here is clear, since any original system state maps naturally into a unique aggregate state. The rationale for specifying the disaggregation probabilities, while somewhat arbitrary, is

guided by intuition into the problem structure. Given an aggregate state (a total level of service for each customer type within the facility), we must generate through the disaggregation probabilities, a "representative" list of customers of each type. It is not difficult to devise reasonable heuristic methods for doing so. The disaggregation and aggregation probabilities, together with the transition probabilities of the original system, specify the transition probabilities and the expected cost per stage of the aggregate problem.

Some simplification techniques, aimed at reducing the complexity of the DP computation, can be interpreted in terms of aggregation. The following is an example.

Example 6.3.12 (Using a Coarse Grid)

A technique often used to reduce the computational requirements of DP is to select a small collection S of states from I and a small collection $\overline{S}$ of states from $\overline{I}$, and define an aggregate problem whose states are those in S and $\overline{S}$. The aggregate problem is then solved by DP and its optimal costs are used to define approximate costs for all states in I and $\overline{I}$. This process, is known as *coarse grid approximation*, and is motivated by the case where the original state spaces I and $\overline{I}$ are dense grids of points obtained by discretization of continuous state spaces, while the collections S and $\overline{S}$ represent coarser subgrids.

The aggregate problem may be formalized by specifying the disaggregation probabilities to associate the states of S with themselves (since $S \subset I$):

$$q_{ss} = 1, \qquad q_{si} = 0 \text{ if } i \neq s, \qquad s \in S.$$

The aggregation probabilities are used to represent each state in $\overline{I}$ as a probabilistic mixture of states in $\overline{S}$, respectively, possibly using some geometrical attribute of the state space.

The aggregation methodology also applies to problems with an infinite number of states. The only difference is that for each aggregate state, the disaggregation probabilities are replaced by a *disaggregation distribution* over the original system's state space. Among other possibilities, this type of aggregation provides methods for discretizing continuous state space problems, as illustrated by the following example.

Example 6.3.13 (Discretization of Continuous State Spaces)

Assume for simplicity a stationary problem, where the state space of the original problem is a bounded region of a Euclidean space. The idea here is to discretize this state space using some finite grid $\{x^1, \ldots, x^M\}$, and then to express each nongrid state by a linear interpolation of nearby grid states. By this we mean that the grid states $x^1, \ldots, x^M$ are suitably selected within the state space, and each nongrid state x is expressed as

$$x = \sum_{m=1}^{M} w^m(x) x^m,$$

for some nonnegative weights $w^m(x)$, which add to 1 and are chosen on the basis of some geometric considerations. We view the weights $w^m(x)$ as aggregation probabilities, and we specify the disaggregation probabilities to associate the grid states with themselves, i.e.,

$$q_{x^m x^m} = 1, \qquad \text{for all } m,$$

similar to the coarse grid approach of Example 6.3.12.

The aggregation and disaggregation probabilities just given specify the aggregate problem, which has a finite state space, the set $\{x^1, \ldots, x^M\}$, and can be solved by DP to obtain the corresponding optimal costs-to-go $\hat{J}_k(x^m)$, $m = 1, \ldots, M$, for each stage k. Then the cost-to-go of each nongrid state x at stage k is approximated by

$$\tilde{J}_k(x) = \sum_{m=1}^{M} w^m(x) \hat{J}_k(x^m).$$

We finally note that one may address the solution of the aggregate problem itself by some suboptimal method, thereby introducing an additional layer of approximation in the solution of the original problem.

6.3.5 Parametric Cost-to-Go Approximation

The idea here is to select from within a parametric class of functions, some cost-to-go approximations $\tilde{J}_k$, which will be used in a limited lookahead scheme in place of the optimal cost-to-go functions J_k. Such parametric classes of functions are called *approximation architectures*, and are generically denoted by $\tilde{J}(x, r)$, where x is the current state and $r = (r_1, \ldots, r_m)$ is a vector of "tunable" scalar parameters, also called *weights* (to simplify notation, we suppress the time indexing in what follows). By adjusting the weights, one can change the "shape" of the approximation $\tilde{J}$ so that it is reasonably close to the true optimal cost-to-go function.

There is an extensive methodology for the selection of the weights. The simplest and often tried approach is to do some form of semi-exhaustive or semi-random search in the space of weight vectors and adopt the weights that result in best performance of the associated one-step lookahead controller. Other more systematic approaches are based on various forms of cost-to-go evaluation and least squares fit. We will discuss such approaches briefly here and more extensively in Vol. II in the context of the methodology of neuro-dynamic programming; see also the books by Bertsekas and Tsitsiklis [BeT96], and Sutton and Barto [SuB98].

There is also a large variety of approximation architectures, involving for example polynomials, neural networks, wavelets, various types of basis functions, etc. We provide a brief discussion of architectures based on extraction of features, and we refer to the specialized literature (e.g., Bertsekas and Tsitsiklis [BeT96], Bishop [Bis95], Haykin [Hay99], Sutton and Barto [SuB98]) for detailed discussions.

Approximation Architectures Based on Feature Extraction

Clearly, for the success of the cost function approximation approach, it is very important to select a class of functions $\tilde{J}(x,r)$ that is suitable for the problem at hand. One particularly interesting type of cost approximation is provided by *feature extraction*, a process that maps the state x into some other vector $y(x)$, called the *feature vector* associated with state x. The vector $y(x)$ consists of scalar components $y_1(x),\ldots,y_m(x)$, called *features*. These features are usually handcrafted, based on whatever human intelligence, insight, or experience is available, and are meant to capture the most important aspects of the current state x. A feature-based cost approximation has the form

$$\tilde{J}(x,r) = \hat{J}\big(y(x),r\big),$$

where r is a parameter vector. Thus, the cost approximation depends on the state x through its feature vector $y(x)$ (see Fig. 6.3.2).

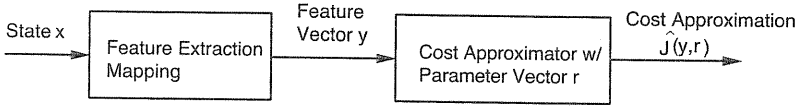


Figure 6.3.2 Using a feature extraction mapping to generate an input to a cost approximator.

The idea is that the cost-to-go function J to be approximated may be a highly complicated nonlinear map and it is sensible to try to break its complexity into smaller, less complex pieces. Ideally, the features will encode much of the nonlinearity that is inherent in J, and the approximation may be quite accurate without requiring a complicated function $\hat{J}$. For example, with a well-chosen feature vector $y(x)$, a good approximation to the cost-to-go is often provided by *linearly* weighting the features, i.e.,

$$\tilde{J}(x,r) = \hat{J}\big(y(x),r\big) = \sum_{i=1}^{m} r_i y_i(x),$$

where $r_1,\ldots,r_m$ is a set of tunable scalar weights.

Note that the use of a feature vector implicitly involves the grouping of states into the subsets that share the same feature vector, i.e., the subsets

$$S_v = \big\{x \mid y(x) = v\big\},$$

where v is possible value of $y(x)$. These subsets form a partition of the state space, and the approximate cost-to-go function $\tilde{J}\big(y(x),r\big)$ is piecewise

constant with respect to this partition; that is, it assigns the same cost-to-go value $\tilde{J}(v,r)$ to all states in the set S_v. This suggests that a feature extraction mapping is well-chosen if states that have the same feature vector have roughly similar optimal cost-to-go.

A feature-based architecture is also related to the aggregation methodology of Section 6.3.4. In particular, suppose that we introduce m aggregate states, $1,\ldots,m$, and associated aggregation and disaggregation probabilities. Let r_i be the optimal cost-to-go associated with aggregate state i. Then, the aggregation methodology yields the linear parametric approximation

$$\tilde{J}(x,r) = \sum_{i=1}^{m} r_i y_i(x),$$

where $y_i(x)$ is the aggregation probability associated with state x and aggregate state i. Thus, within this context, the aggregation probability $y_i(x)$ may be viewed as a feature, which, roughly speaking, specifies the "degree of membership of x to aggregate state i."

We now illustrate the preceding concepts with a detailed discussion of computer chess, where feature-based approximation architectures play an important role.

Computer Chess

Chess-playing computer programs are one of the more visible successes of artificial intelligence. Their underlying methodology relies provides an interesting case study in the use of approximate DP. It involves the idea of limited lookahead, but also illustrates some DP ideas that we have not had much opportunity to look at in detail. These are the idea of *forward depth-first search*, an important memory-saving technique that was discussed in Section 2.3 in the context of label correcting methods, and the idea of *alpha-beta pruning*, which is an effective method for reducing the amount of computation needed to find optimal strategies in competitive games.

The fundamental paper on which all computer chess programs are based was written by one of the most illustrious modern-day applied mathematicians, C. Shannon [Sha50]. It was argued by Shannon that whether the starting chess position is a win, loss, or draw is a question that can be answered in principle, but the answer will probably never be known. He estimated that, based on the chess rule requiring at least one pawn advance or capture within every 50 moves (otherwise a draw is declared), there are on the order of 10^{120} different possible sequences of moves in a chess game. He concluded that to examine these and select the best initial move for White would require 10^{90} years of a "fast" computer's time (fast here relates to the standards of the '50s, but the number 10^{90} is overwhelming even by today's standards). As an alternative, Shannon proposed a *limited lookahead* of a few moves and *evaluating the end positions by means of a*

scoring function. The scoring function may involve, for example, the calculation of a numerical value for each of a set of major features of a position (such as material balance, mobility, pawn structure, and other positional factors), together with a method to combine these numerical values into a single score. Thus, we may view a scoring function as a feature-based architecture for evaluating a chess position/state.

Consider first a *one-move lookahead strategy* for selecting the first move in a given position P. Let $M_1, \ldots, M_r$ be all the legal moves that can be made in position P by the side to move. Denote the resulting positions by $M_1P, \ldots, M_rP$, and let $S(M_1P), \ldots, S(M_rP)$ be the corresponding scores (the convention here is that White is favored in positions with high score, while Black is favored in positions with low score). Then the move selected by White (Black) in position P is the move with maximum (minimum) score. This is known as the *backed-up score* of P and is given by

$$BS(P) = \begin{cases} \max\{S(M_1P), \ldots, S(M_rP)\} & \text{if White is to move in } P, \\ \min\{S(M_1P), \ldots, S(M_rP)\} & \text{if Black is to move in } P. \end{cases}$$

This process is illustrated in Fig. 6.3.3.

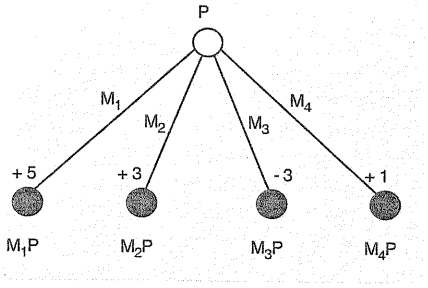


Figure 6.3.3 A one-move lookahead tree. If White moves at position P, the best move is M_1 and the backed-up score is $+5$. If Black moves in position P, the best move is M_3, and the backed-up score of P is -3.

Consider next a *two-move lookahead strategy* in a given position P. Assume for concreteness that White moves, and let the legal moves be $M_1, \ldots, M_r$ and the corresponding positions be $M_1P, \ldots, M_rP$. Then in each of the positions M_iP, $i = 1, \ldots, r$, apply the one-move lookahead strategy with Black to move. This gives a best move and a backed-up score $BS(M_iP)$ for Black in each of the positions M_iP, $i = 1, \ldots, r$. Finally, based on the backed-up scores $BS(M_1P), \ldots, BS(M_rP)$, apply a one-move lookahead strategy for White, thereby obtaining the best move at position P and a backed-up score for position P of

$$BS(P) = \max\{BS(M_1P), \ldots, BS(M_rP)\}.$$

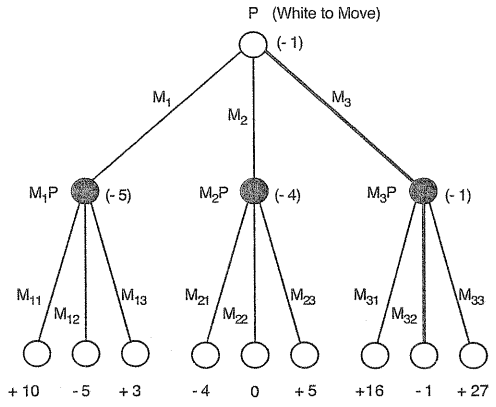


Figure 6.3.4 A two-move lookahead tree with White to move. The backed-up scores are shown in parentheses. The best initial move is M_3 and the principal continuation is (M_3, M_{32}).

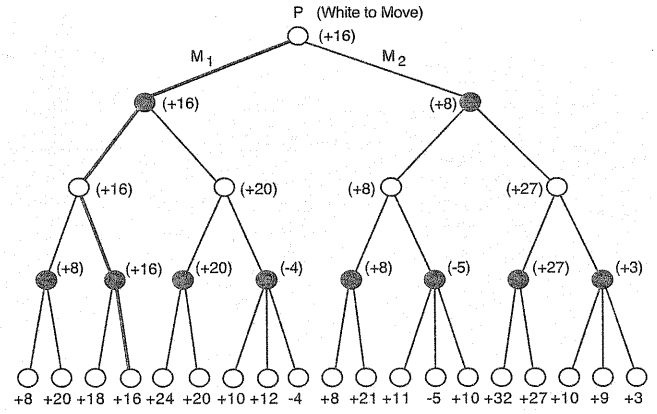


Figure 6.3.5 A four-move lookahead tree with White to move. The backed-up scores are shown in parentheses. The best initial move is M_1. The principal continuation is heavily shaded.

The sequence of best moves is known as the *principal continuation*. The process is illustrated in Fig. 6.3.4. It is clear that Shannon's method as just described can be generalized for an arbitrary number of lookahead moves (see Fig. 6.3.5).

Generally, to evaluate the best move at a given position and the corresponding backed-up score using lookahead of n moves, one can use the following DP-like procedure:

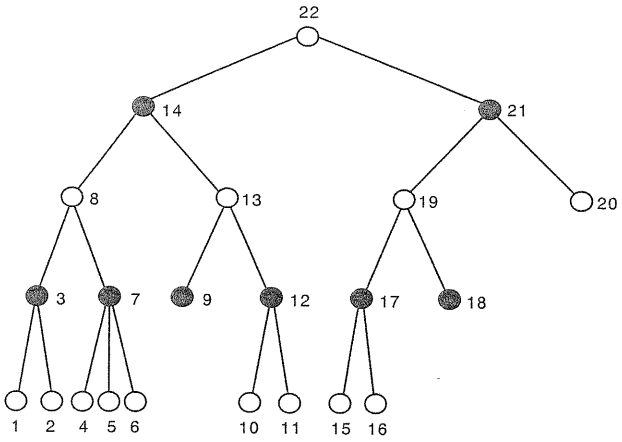


Figure 6.3.6 Traversing a tree in depth-first fashion. The numbers indicate the order in which the scores of the terminal positions and the backed-up scores of the intermediate positions are evaluated.

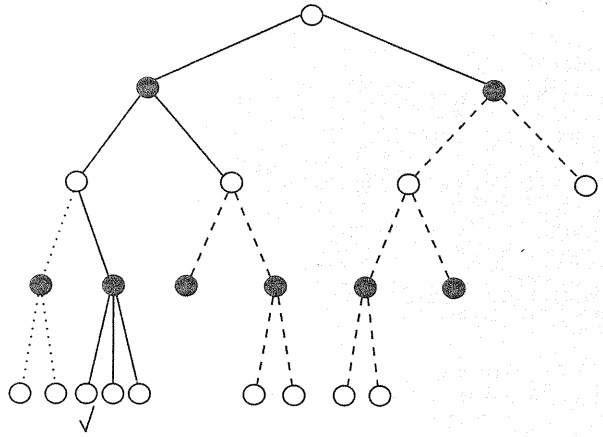


Figure 6.3.7 Storage requirements of the depth-first version of the minimax algorithm for the tree of Fig. 6.3.6. At the time that the terminal position marked by a checkmark is scored, only the solid-line moves are stored in memory. The dotted-line moves have been generated and purged from memory. The broken-line moves have not been generated as yet. The memory required grows linearly with the depth of the lookahead.

1. Evaluate the scores of all possible positions that are n moves ahead from the given position P.
2. Using the scores of the terminal positions just evaluated, calculate the backed-up scores of all possible positions that are $n-1$ moves ahead from P.
3. For $k = 1, \ldots, n-1$, using the backed-up scores of all possible positions that are $n-k$ moves ahead from P, calculate the backed-up scores of all positions that are $n-k-1$ moves ahead from P.

The above procedure requires a lot of memory storage even for a modest number of lookahead moves. Shannon pointed out that with an alternative but mathematically equivalent calculation, the amount of memory required grows only *linearly* with the depth of lookahead, thereby allowing chess programs to operate in limited-memory microprocessor systems. This is accomplished by searching the tree of moves in *depth-first fashion*, and by generating new moves only when needed, as illustrated in Figs. 6.3.6 and 6.3.7. It is only necessary to store the *one* move sequence under current examination together with one list of legal moves at each level of the search tree. The precise algorithm is described by the following routine, which calls itself recursively.

Minimax Algorithm

To determine the backed-up score $BS(n)$ of position n, do the following:

1. If n is a terminal position return its score. Otherwise:
2. Generate the list of legal moves at position n and let the corresponding positions be $n_1, \ldots, n_r$. Set the tentative backed-up score $TBS(n)$ of position n to ∞ if it is White's turn to move at n and to $-\infty$ if it is Black's turn to move at n.
3. For $i = 1, \ldots, r$, do:
   a. Determine the backed-up score $BS(n_i)$ of position n_i.
   b. If it is White's move at position n, set
   $$TBS(n) := \max\{TBS(n),\ BS(n_i)\};$$
   if it is Black's move at position n, set
   $$TBS(n) := \min\{TBS(n),\ BS(n_i)\}.$$
4. Return $BS(n) = TBS(n)$.

The idea of solving one-step lookahead problems with a terminal cost (or backed-up score) that summarizes future costs is of course central in the DP algorithm. Indeed, it can be seen that the minimax algorithm just described is nothing but the DP algorithm for minimax problems (see Section 1.6). Here, positions and moves can be identified with states and controls, respectively, there are only terminal costs (the scores of the terminal positions), and the backed-up score of a position is nothing but the optimal cost-to-go at the corresponding state.

The minimax algorithm is also known as the *type A strategy*. Shannon argued that with this strategy, one could not expect a computer to seriously challenge human players of even moderate strength. In a typical chess position there are around 30 to 35 legal moves. It follows that for an n-move lookahead there will be around 30^n to 35^n terminal positions to be scored. Thus the number of terminal positions grows exponentially with the size of lookahead, practically limiting n to being of the order of 10 with present computers. Unfortunately, in some chess positions it is essential to look a substantially larger number of moves ahead. In particular, in dynamic positions involving many captures and countercaptures, the necessary size of lookahead can be very large.

These considerations led Shannon to consider another strategy, called *type B*, whereby the depth of the search tree is variable. He suggested that at each position the computer give all legal moves a preliminary examination and discard those that are "obviously bad." A scoring function together with some heuristic strategy can be used for this purpose. Similarly, he suggested that some positions that are dynamic, such as those involving many captures or checkmate threats, be explored further beyond the nominal depth of the search.

Chess-playing computer programs typically use a combination of Shannon's type A and B strategies. These programs use scoring functions, the forms of which have evolved by trial-and-error, but they also use sophisticated heuristics to evaluate dynamic terminal positions in detail. In particular, an effective algorithm, known as *swapoff*, is used to quickly analyze long sequences of captures and countercaptures, thereby making it possible to score realistically complex, dynamic positions (see Levy [Lev84] for a description). One may view such heuristics as either defining a sophisticated scoring function, or as implementing a type B strategy.

The efficiency of the minimax algorithm can be substantially improved by using the *alpha-beta pruning procedure* (denoted α-β for short), which can be used to forego some calculations involving positions that cannot affect the selection of the best move. To understand the α-β procedure, consider a chess player pondering the next move at position P. Suppose that the player has already exhaustively analyzed one relatively good move M_1 with corresponding score $BS(M_1P)$ and proceeds to examine the next move M_2. Suppose that as the opponent's replies are examined, a particularly strong response is found, which assures that the score of M_2 will be

worse than that of M_1. Such a response, called a *refutation* of move M_2, makes further consideration of move M_2 unnecessary (i.e., the portion of the search tree that descends from move M_2 can be discarded). An example is shown in Fig. 6.3.8.

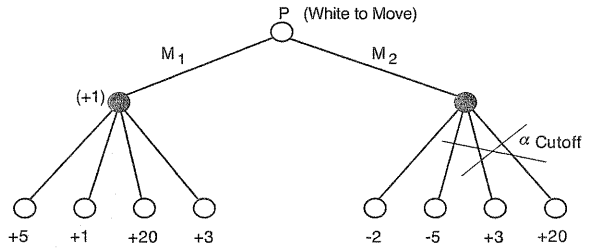


Figure 6.3.8 The α-β procedure. White has evaluated move M_1 to have backed-up score $(+1)$, and starts evaluating move M_2. The first reply of Black is a refutation of M_2, since it leads to a temporary score of -2, less than the backed-up score of M_1. Since the backed-up score of M_2 will be -2 or less, M_2 will be inferior to M_1. Therefore, it is not necessary to evaluate move M_2 further.

The α-β procedure can be generalized to trees of arbitrary or irregular depth and can be incorporated very simply into the minimax algorithm. Generally, if in the process of updating the backed-up score of a given position (step 3b) this score crosses a certain bound, then no further calculation is needed regarding that position. The cutoff bounds are adjusted dynamically as follows:

1. The cutoff bound in position n, where Black has to move, is denoted α and equals the highest current score of all ancestor positions of n where White has to move. The exploration of position n can be terminated as soon as its temporary backed-up score equals or falls below α.

2. The cutoff bound in position n, where White has to move, is denoted β and equals the lowest current value of all ancestor positions of n where Black has the move. The exploration of position n can be terminated as soon as its temporary backed-up score rises above β.

The process is illustrated in Fig. 6.3.9. *It can be shown that the backed-up score and optimal move at the starting position are unaffected by the incorporation of the α-β procedure in the minimax algorithm.* We leave the verification of this fact to the reader (Exercise 6.8). It can also be seen that *the α-β procedure will be more effective if the best moves in each position are explored first.* This tends to keep the α bounds high and the β bounds low, thus saving a maximum amount of calculation. Current

chess programs use sophisticated techniques for ordering moves so as to maximize the effectiveness of the α-β procedure. We discuss briefly two of these techniques: *iterative deepening* and the *killer heuristic*.

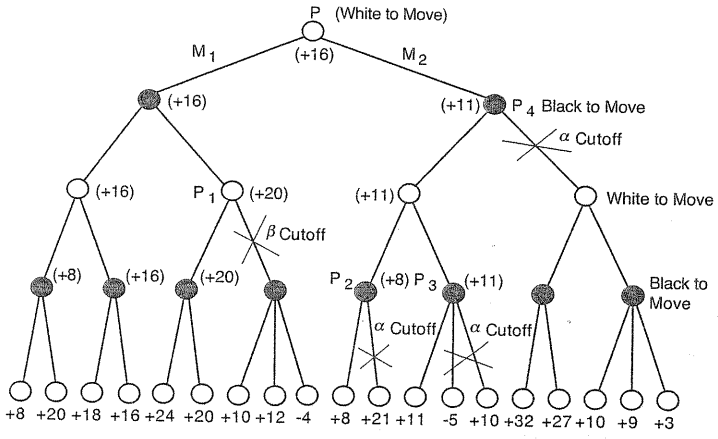


Figure 6.3.9 The α-β procedure applied to the tree of Fig. 6.3.5. For example, the β-cutoff in position P_1 is due to the fact that its temporary score (+20) exceeds its current β-bound (+16). The α-cutoffs in positions P_2, P_3, and P_4 are due to the fact that the corresponding temporary scores, +8, +11, and +11, have fallen below the current α-bound, which is +16, the current temporary score in position P.

Iterative deepening, in its pure form, consists of first conducting a search based on lookahead of one move; then carrying out (from scratch) a search based on lookahead of two moves; then carrying out a search based on lookahead of three moves and so on. This process is continued either up to a fixed level of lookahead or until some limit on computation time is exceeded. At each iteration associated with a certain level of lookahead, one obtains a best move at the starting position, which is examined first in the subsequent iteration that requires one extra move of lookahead. This enhances the power of the α-β procedure, thereby more than making up for the extra computation involved in doing a short lookahead search before doing a longer one. (Actually, given that the number of terminal positions increases on the average by a factor of the order of 30 with each additional level of lookahead, the extra computation is relatively small.) An additional benefit of this method is that a best move is maintained throughout the search and can be produced at any time as needed. This comes in handy in commercial programs that incorporate a feature whereby the computer is forced to move either upon exhausting a given time allocation or upon command by a human opponent. An improvement of the method is to

obtain a thoroughly sorted list of moves at the starting position via a one-move lookahead, and then use the improved ordering in subsequent iterations to enhance the performance of the α-β procedure.

The killer heuristic is similar to iterative deepening in that it aims at examining first the most powerful moves at each position, thereby enhancing the pruning power of the α-β procedure. To understand the idea, suppose that in some position, White selects the first move M_1 from a candidate list $\{M_1, M_2, M_3, \ldots\}$, and upon examining Black's responses to M_1 finds that a particular move, which we will refer to as the *killer move*, is by far Black's best. Then it is often true that the killer move is also Black's best response to the second and subsequent moves $M_2, M_3, \ldots$ in White's list. It is therefore a good idea from the point of view of α-β pruning to consider the killer move first as a potential response to the remaining moves $M_2, M_3, \ldots$ Of course, this does not always work as hoped, in which case it is advisable to change the killer move depending on subsequent results of the computation. In fact, some programs maintain lists of more than one killer move at each level of lookahead.

The α-β procedure is safe in the sense that searching a game tree with it and without it will produce the same result. Some computer chess programs use more drastic tree-pruning procedures, which usually require less computation for a given level of lookahead, but may miss on occasion the strongest move. There is some debate at present regarding the merits of such procedures. The books by Levy [Lev84] and Newborn [New75] consider this subject, and provide a broader discussion of the limitations of computer chess programs. A fascinating account of the development of a checkers computer program that implements many of the ideas discussed here is given by Schaeffer [Sch97].

6.4 ROLLOUT ALGORITHMS

We now discuss a specific type of cost-to-go approximation within the context of a limited lookahead scheme. Recall that in the one-step lookahead method, at stage k and state x_k we use the control $\overline{\mu}_k(x_k)$ that attains the minimum in the expression

$$\min_{u_k \in U_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\},$$

where $\tilde{J}_{k+1}$ is some approximation of the true cost-to-go function J_{k+1}. In the rollout algorithm, the approximating function $\tilde{J}_{k+1}$ is the cost-to-go of some known heuristic/suboptimal policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, called *base policy* (see also Example 6.3.1). The policy thus obtained is called the *rollout policy* based on π. Thus *the rollout policy is a one-step lookahead*

policy, with the optimal cost-to-go approximated by the cost-to-go of the base policy.

The process of starting from some suboptimal policy and generating another policy using the one-step lookahead process described above is also called *policy improvement*. This process will be discussed in Section 7.2 and in Vol. II in the context of the *policy iteration* method, which is a primary method for solving infinite horizon problems.

Note that it is also possible to define rollout policies that make use of multistep (say, l-step) lookahead. Here we assign to every state x that can be reached in l steps, the exact cost-to-go of the base policy, as computed by Monte Carlo simulation of several trajectories that start at x. Clearly, such multistep lookahead involves much more on-line computation, but it may yield better performance than its single-step counterpart. In what follows, we concentrate on rollout policies with single-step lookahead.

The viability of a rollout policy depends on how much time is available to choose the control following the transition to state x and on how expensive the Monte Carlo evaluation of the expected value

$$E\Big\{g_k(x_k,u_k,w_k)+\tilde{J}_{k+1}\big(f_k(x_k,u_k,w_k)\big)\Big\}$$

is. In particular, it must be possible to perform the Monte Carlo simulations and calculate the rollout control within the real-time constraints of the problem. If the problem is deterministic, a single simulation trajectory suffices, and the calculations are greatly simplified, but in general, the computational overhead can be substantial.

It is possible, however, to speed up the calculation of the rollout policy if we are willing to accept some potential performance degradation. For example, we may use an approximation $\hat{J}_{k+1}$ of $\tilde{J}_{k+1}$ to identify a few promising controls through a minimization of the form

$$\min_{u_k\in U_k(x_k)} E\Big\{g_k(x_k,u_k,w_k)+\hat{J}_{k+1}\big(f_k(x_k,u_k,w_k)\big)\Big\},$$

and then restrict attention to these controls, using fairly accurate Monte Carlo simulation. In particular, the required values of $\hat{J}_{k+1}$ may be obtained by performing approximately the Monte Carlo simulation, using a limited number of representative trajectories. Adaptive variants of this approach are also possible, whereby we use some heuristics to adjust the accuracy of the Monte Carlo simulation depending on the results of the computation.

Generally, it is important to use as base policy one whose expected cost-to-go is conveniently calculated. The following is an example.

Example 6.4.1 (The Quiz Problem)

Consider the quiz problem of Example 4.5.1, where a person is given a list of N questions and can answer these questions in any order he/she chooses.

Question i will be answered correctly with probability p_i, and the person will then receive a reward v_i. At the first incorrect answer, the quiz terminates and the person is allowed to keep his or her previous rewards. The problem is to choose the ordering of questions so as to maximize the total expected reward.

We saw that the optimal sequence can be obtained using an interchange argument: questions should be answered in decreasing order of the "index of preference" $p_iv_i/(1-p_i)$. We refer to this as the *index policy*. Unfortunately, with only minor changes in the structure of the problem, the index policy need not be optimal. Examples of difficult variations of the problem may involve one or more of the following characteristics:

(a) A limit on the maximum number of questions that can be answered, which is smaller than the number of questions N. To see that the index policy is not optimal anymore, consider the case where there are two questions, only one of which may be answered. Then it is optimal to answer the question that offers the maximum expected reward p_iv_i.

(b) A time window for each question, which constrains the set of time slots when each question may be answered. Time windows may also be combined with the option to refuse answering a question at a given period, when either no question is available during the period, or answering any one of the available questions involves excessive risk.

(c) Precedence constraints, whereby the set of questions that can be answered in a given time slot depends on the immediately preceding question, and possibly on some earlier answered questions.

(d) Sequence-dependent rewards, whereby the reward from answering correctly a given question depends on the immediately preceding question, and possibly on some questions answered earlier.

Nonetheless, even when the index policy is not optimal, it can conveniently be used as a base policy for the rollout algorithm. The reason is that at a given state, the index policy together with its expected reward can be easily calculated. In particular, each feasible question order $(i_1,\ldots,i_N)$ has expected reward equal to

$$p_{i_1}\Big(v_{i_1}+p_{i_2}\big(v_{i_2}+p_{i_3}(\cdots+p_{i_N}v_{i_N})\cdots\big)\Big).$$

Thus the rollout algorithm based on the index heuristic operates as follows: at a state where a given subset of questions have already been answered, we consider the set of questions J that are eligible to be answered next. For each question $j\in J$, we consider a sequence of questions that starts with j and continues with the remaining questions chosen according to the index rule. We compute the expected reward of the sequence, denoted $R(j)$, using the above formula. Then among the questions $j\in J$, we choose to answer next the one with maximal $R(j)$. A computational study of rollout algorithms for the quiz problem and some variations, using several methods for approximating the cost-to-go of the base policy, is given by Bertsekas and Castanon [BeC99].

Cost Improvement with a Rollout Algorithm

Rollout policies have a nice property: in their pure form, they always result in improved performance over the corresponding base policy. This is essentially a consequence of Prop. 6.3.1 (see Example 6.3.1), but for the purpose of convenient reference, we adapt the proof of that proposition to the rollout context. Let $\overline{J}_k(x_k)$ and $H_k(x_k)$ be the costs-to-go of the rollout and the base policies, respectively, starting from a state x_k at time k. We will show that $\overline{J}_k(x_k) \leq H_k(x_k)$ for all x_k and k, so that the rollout policy $\overline{\pi}$ is an improved policy over the base policy π. We have $\overline{J}_N(x_N) = H_N(x_N) = g_N(x_N)$ for all x_N. Assuming that $\overline{J}_{k+1}(x_{k+1}) \leq H_{k+1}(x_{k+1})$ for all x_{k+1}, we have

$$\begin{aligned}\overline{J}_k(x_k) &= E\big\{g_k\big(x_k, \overline{\mu}_k(x_k), w_k\big) + \overline{J}_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\big\} \\ &\leq E\big\{g_k\big(x_k, \overline{\mu}_k(x_k), w_k\big) + H_{k+1}\big(f_k\big(x_k, \overline{\mu}_k(x_k), w_k\big)\big)\big\} \\ &\leq E\big\{g_k\big(x_k, \mu_k(x_k), w_k\big) + H_{k+1}\big(f_k\big(x_k, \mu_k(x_k), w_k\big)\big)\big\} \\ &= H_k(x_k),\end{aligned}$$

for all x_k. The first inequality above follows from the induction hypothesis, the second inequality follows from the definition of the rollout policy, and the first and second equalities follow from the DP algorithm that defines the costs-to-go of the rollout and the base policies, respectively. This completes the induction proof that $\overline{\pi}$ is an improved policy over π.

Empirically, it has been observed that the rollout policy typically produces considerable (and often dramatic) cost improvement over the base policy. However, there is no solid theoretical support for this observation. The following example provides some insight into the nature of cost improvement.

Example 6.4.2 (The Breakthrough Problem)

Consider a binary tree with N stages as shown in Fig. 6.4.1. Stage k of the tree has 2^k nodes, with the node of stage 0 called *root* and the nodes of stage N called *leaves*. There are two types of tree arcs: *free* and *blocked*. A free (or blocked) arc can (cannot, respectively) be traversed in the direction from the root to the leaves. The objective is to break through the graph with a sequence of free arcs (a free path) starting from the root, and ending at one of the leaves.

One may use DP to discover a free path (if one exists) by starting from the last stage and by proceeding backwards to the root node. The kth step of the algorithm determines for each node of stage $N-k$ whether there is a free path from that node to some leaf node, by using the results of the preceding step. The amount of calculation at the kth step is $O(2^{N-k})$. Adding the calculations for the N stages, we see that the total amount of calculation is $O(N2^N)$, so it increases exponentially with the number of stages. For this reason it is interesting to consider heuristics that require calculation that is

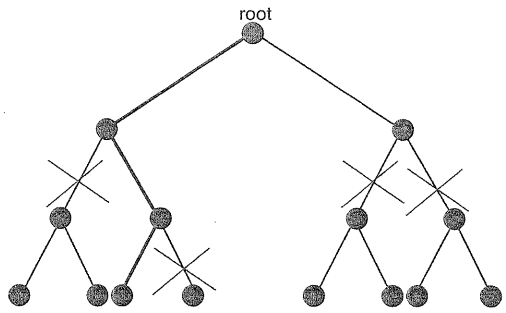


Figure 6.4.1 Binary tree for the breakthrough problem. Each arc is either free or is blocked (crossed out in the figure). The problem is to find a path from the root to one of the leaves, which is free (such as the one shown with thick lines).

linear or polynomial in N, but may sometimes fail to determine a free path, even when a free path exists.

Thus, one may suboptimally use a *greedy* algorithm, which starts at the root node, selects a free outgoing arc (if one is available), and tries to construct a free path by adding successively nodes to the path. Generally, at the current node, if one of the outgoing arcs is free and the other is blocked, the greedy algorithm selects the free arc. Otherwise, it selects one of the two outgoing arcs according to some fixed rule that depends only on the current node (and not on the status of other arcs). Clearly, the greedy algorithm may fail to find a free path even if such a path exists, as can be seen from Fig. 6.4.1. On the other hand the amount of computation associated with the greedy algorithm is $O(N)$, which is much faster than the $O(N2^N)$ computation of the DP algorithm. Thus we may view the greedy algorithm as a fast heuristic, which is suboptimal in the sense that there are problem instances where it fails while the DP algorithm succeeds.

Let us also consider the rollout algorithm that uses the greedy algorithm as the base heuristic. This algorithm starts at the root and tries to construct a free path by exploring alternative paths constructed by the greedy algorithm. At the current node, it proceeds according to the following two cases:

(a) If at least one of the two outgoing arcs of the current node is blocked, the rollout algorithm adds to the current path the arc that the greedy algorithm would select at the current node.

(b) If both outgoing arcs of the current node are free, the rollout algorithm considers the two end nodes of these arcs, and from each of them it runs the greedy algorithm. If the greedy algorithm succeeds in finding a free path that starts from at least one of these nodes, the rollout algorithm stops with a free path having been found; otherwise, the rollout algorithm moves to the node that the greedy algorithm would select at the current node.

Thus, when both outgoing arcs are free, the rollout algorithm explores further the suitability of these arcs, as in case (b) above. Because of this additional discriminatory capability, the rollout algorithm always does at least as well as the greedy (it always finds a free path when the greedy algorithm does, and it also finds a free path in some cases where the greedy algorithm does not). This is consistent with our earlier discussion of the generic cost improvement property of the rollout algorithm over the base heuristic. On the other hand, the rollout algorithm applies the greedy heuristic as many as $2N$ times, so that it requires $O(N^2)$ amount of computation – this is intermediate between the $O(N)$ computation of the greedy and the $O(N2^N)$ computation of the DP algorithm.

Let us now calculate the probabilities that the algorithms will find a free path given a randomly chosen breakthrough problem. In particular, we generate the graph of the problem randomly, by selecting each of its arcs to be free with probability p, independently of the other arcs. We then calculate the corresponding probabilities of success for the greedy and the rollout algorithms.

The probability G_k that the greedy algorithm will find a free path in a graph of k stages is the probability of a "success" in each of the k stages, where a success is counted whenever at least one of the two arcs involved is free, an event of probability $1-(1-p)^2$ or $p(2-p)$. Thus we have, using the independence of the blocked/unblocked status of the arcs,

$$G_k = \big(p(2-p)\big)^k.$$

The probability R_k that the rollout algorithm will find a free path in a graph of k stages can be calculated by means of a recursion, as we now show. At a given node n_0 with k stages to go, consider the path $(n_0, n_1, \ldots, n_k)$ generated by the greedy algorithm, and let (n_0, n_1) and (n_0, n_1') denote the incident arcs of node n_0. Let P_1 denote the probability that exactly one of the arcs (n_0, n_1) and (n_0, n_1') is free, so

$$P_1 = 2p(1-p),$$

and let P_2 denote the probability that both arcs (n_0, n_1) and (n_0, n_1') are free, so

$$P_2 = p^2.$$

To calculate the probability R_k of the event that the rollout algorithm succeeds in finding a free path, we partition this event into the following four mutually exclusive events, and we calculate their probabilities:

(1) *Event* E_1. Exactly one of the arcs (n_0, n_1) and (n_0, n_1') is free [by necessity (n_0, n_1) since it is chosen by the greedy algorithm] and the rollout algorithm finds a free path starting from n_1. The probability of this event is $P(E_1) = P_1 R_{k-1}$.

(2) *Event* E_2. Both arcs (n_0, n_1) and (n_0, n_1') are free and the greedy algorithm finds a free path starting from n_1. The probability of this event is $P(E_2) = P_2 G_{k-1}$.

(3) *Event* E_3. Both the arcs (n_0, n_1) and (n_0, n_1') are free, and the greedy algorithm does not find a free path starting from n_1 but finds a free path from n_1'. The probability of this event is $P(E_3) = P_2(1-G_{k-1})G_{k-1}$.

(4) *Event* E_4. Both the arcs (n_0, n_1) and (n_0, n_1') are free, the greedy algorithm does not find a free path starting from either n_1 or n_1', but the rollout algorithm finds a free path from n_1. The probability of this event is $P_2(1-G_{k-1})^2 H_{k-1}$, where H_{k-1} is the conditional probability that the rollout finds a free path from n_1 given that the greedy does not find a free path from n_1. We have

$$R_{k-1} = G_{k-1} + (1-G_{k-1})H_{k-1},$$

so $(1-G_{k-1})H_{k-1} = R_{k-1} - G_{k-1}$, and the probability of the event E_4 is

$$P(E_4) = P_2(1-G_{k-1})^2 H_{k-1} = P_2(1-G_{k-1})(R_{k-1} - G_{k-1}).$$

Thus, by adding the probabilities of the above mutually exclusive and collectively exhaustive events, we have

$$\begin{aligned} R_k &= P(E_1) + P(E_2) + P(E_3) + P(E_4) \\ &= P_1 R_{k-1} + P_2\big(G_{k-1} + (1-G_{k-1})G_{k-1} + (1-G_{k-1})(R_{k-1} - G_{k-1})\big) \\ &= \big(P_1 + P_2(1-G_{k-1})\big)R_{k-1} + P_2 G_{k-1}. \end{aligned}$$

From this, by substituting the expressions $P_1 = 2p(1-p)$ and $P_2 = p^2$, we obtain

$$\begin{aligned} R_k &= \big(2p(1-p) + p^2(1-G_{k-1})\big)R_{k-1} + p^2 G_{k-1} \\ &= p(2-p)R_{k-1} + p^2 G_{k-1}(1-R_{k-1}), \end{aligned}$$

with the initial condition $R_0 = 1$. Since $\lim_{k\to\infty} G_k = 0$ and $p(2-p) < 1$, it follows from the above equation that $\lim_{k\to\infty} R_k = 0$. Furthermore, by dividing with $G_k = p(2-p)G_{k-1}$, we have

$$\frac{R_k}{G_k} = \frac{R_{k-1}}{G_{k-1}} + \frac{p}{2-p}(1-R_{k-1}),$$

so since $\lim_{k\to\infty} R_k = 0$, we obtain for large N

$$\frac{R_N}{G_N} = O\left(N\frac{p}{2-p}\right).$$

Thus, asymptotically, the rollout algorithm requires $O(N)$ times more computation, but has an $O(N)$ times larger probability of finding a free path than the greedy algorithm. This type of tradeoff appears to be qualitatively typical: the rollout algorithm achieves a substantial performance improvement over the base heuristic at the expense of extra computation that is equal to the computation time of the base heuristic times a factor that is a low order polynomial of the problem size.

Computational Issues in Rollout Algorithms

We now consider in more detail implementation issues and specific properties of rollout algorithms in a variety of settings. To compute the rollout control $\overline{\mu}_k(x_k)$, we need for all $u_k \in U_k(x_k)$ the value of

$$Q_k(x_k, u_k) = E\Big\{g_k(x_k, u_k, w_k) + H_{k+1}\big(f_k(x_k, u_k, w_k)\big)\Big\},$$

known as the *Q-factor* of (x_k, u_k) at time k. Alternatively, for the computation of $\overline{\mu}_k(x_k)$ we need the value of the cost-to-go

$$H_{k+1}\big(f_k(x_k, u_k, w_k)\big)$$

of the base policy at all possible next states $f_k(x_k, u_k, w_k)$, from which we can compute the required Q-factors.

We will focus on the case where a closed form expression of the Q-factor is not available. We assume instead that we can simulate the system under the base policy π, and in particular, that we can generate sample system trajectories and corresponding costs consistently with the probabilistic data of the problem. We will consider several cases and possibilities, we will point out their advantages and drawbacks, and we will discuss the contexts within which they are most appropriate. These cases are:

(1) *The deterministic problem case*, where w_k takes a single known value at each stage. We provide an extensive discussion of this case, focusing not only on traditional deterministic optimal control problems, but also on quite general combinatorial optimization problems, for which the rollout approach has proved convenient and effective.

(2) *The stochastic problem case with Q-factors evaluated by Monte-Carlo simulation*. Here, once we are at state x_k, the Q-factors $Q_k(x_k, u_k)$ are evaluated on-line by Monte-Carlo simulation, for all $u_k \in U_k(x_k)$.

(3) *The stochastic problem case with Q-factors approximated in some way*. One possibility is to use a certainty equivalence approximation, where the problem is genuinely stochastic, but the values $H_k(x_k)$ are approximated by the cost-to-go of π that would be incurred if the system were replaced by a suitable deterministic system from state x_k and time k onward (assumed certainty equivalence). There are also other possibilities based on the use of an approximation architecture and some form of least squares.

We examine each of these three cases in the next three subsections.

6.4.1 Discrete Deterministic Problems

Let us assume that the problem is deterministic, i.e., that w_k can take only one value at each stage k. Then, starting from state x_k at stage k, the

base policy π produces deterministic sequences of states $\{x_{k+1}, \ldots, x_N\}$ and controls $\{u_k, \ldots, u_{N-1}\}$ such that

$$x_{i+1} = f(x_i, u_i), \qquad i = k, \ldots, N-1,$$

and a cost

$$g_k(x_k, u_k) + \cdots + g_{N-1}(x_{N-1}, u_{N-1}) + g_N(x_N).$$

Thus the Q-factor

$$Q_k(x_k, u_k) = g_k(x_k, u_k) + H_{k+1}\big(f_k(x_k, u_k)\big)$$

can be obtained by running π starting from state $f_k(x_k, u_k)$ and time $k+1$, and recording the corresponding cost $H_{k+1}\big(f_k(x_k, u_k)\big)$. The rollout control $\overline{\mu}_k(x_k)$ is obtained by calculating in this manner the Q-factors $Q_k(x_k, u_k)$ for all $u_k \in U_k(x_k)$, and setting

$$\overline{\mu}_k(x_k) = \arg \min_{u_k \in U_k(x_k)} Q_k(x_k, u_k).$$

Aside for being convenient for the deterministic special case of the basic problem of Chapter 1, this rollout method can be adapted for general discrete or combinatorial optimization problems that do not necessarily have the strong sequential character of the basic problem. For such problems the rollout approach provides a convenient and broadly applicable suboptimal solution method that goes beyond and indeed enhances the common types of heuristics, such as greedy algorithms, local search, genetic algorithms, tabu search, and others.

To illustrate the ideas involved, let us consider the problem

$$\begin{aligned} &\text{minimize} \quad G(u) \\ &\text{subject to} \quad u \in U \end{aligned} \tag{6.31}$$

where U is a finite set of feasible solutions and $G(u)$ is a cost function. We assume that each solution u has N components; that is, it has the form $u = (u_1, u_2, \ldots, u_N)$, where N is a positive integer. Under this assumption, we can view the problem as a sequential decision problem, where the components $u_1, \ldots, u_N$ are selected one-at-a-time. An n-tuple $(u_1, \ldots, u_n)$ consisting of the first n components of a solution is called an *n-solution*. We associate n-solutions with the nth stage of a DP problem. In particular, for $n = 1, \ldots, N$, the states of the nth stage are of the form $(u_1, \ldots, u_n)$. The initial state is a dummy (artificial) state. From this state we may move to any state (u_1), with u_1 belonging to the set

$$U_1 = \big\{\tilde{u}_1 \mid \text{there exists a solution of the form } (\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_N) \in U\big\}.$$

Thus U_1 is the set of choices of u_1 that are consistent with feasibility.

More generally, from a state of the form $(u_1, \ldots, u_{n-1})$, we may move to any state of the form $(u_1, \ldots, u_{n-1}, u_n)$, with u_n belonging to the set

$$U_n(u_1, \ldots, u_{n-1}) = \big\{\tilde{u}_n \mid \text{there exists a solution of the form } (u_1, \ldots, u_{n-1}, \tilde{u}_n, \ldots, \tilde{u}_N) \in U\big\}. \tag{6.32}$$

The choices available at state $(u_1, \ldots, u_{n-1})$ are $u_n \in U_n(u_1, \ldots, u_{n-1})$. These are the choices of u_n that are consistent with the preceding choices $u_1, \ldots, u_{n-1}$, and are also consistent with feasibility. The terminal states correspond to the N-solutions $(u_1, \ldots, u_N)$, and the only nonzero cost is the terminal cost $G(u_1, \ldots, u_N)$.

Let $J^*(u_1, \ldots, u_n)$ denote the optimal cost starting from the n-solution $(u_1, \ldots, u_n)$, i.e., the optimal cost of the problem over solutions whose first n components are constrained to be equal to u_i, $i = 1, \ldots, n$, respectively. If we knew the optimal cost-to-go function $J^*(u_1, \ldots, u_n)$, we could construct an optimal solution by a sequence of N single component minimizations. In particular, an optimal solution $(u_1^*, \ldots, u_N^*)$ could be obtained through the algorithm

$$u_i^* = \arg \min_{u_i \in U_i(u_1^*, \ldots, u_{i-1}^*)} J^*(u_1^*, \ldots, u_{i-1}^*, u_i), \qquad i = 1, \ldots, N.$$

Unfortunately, this is seldom viable, because of the prohibitive computation required to obtain $J^*(u_1, \ldots, u_n)$.

Assume now that we have a heuristic, which starting from an n-solution $(u_1, \ldots, u_n)$, produces an N-solution $(u_1, \ldots, u_n, u_{n+1} \ldots, u_N)$ whose cost is denoted by $H(u_1, \ldots, u_n)$. Such a heuristic may be viewed as a base policy for the problem, in the sense that given the current state $(u_1, \ldots, u_n)$ it generates the next decision u_{n+1} as the first component of the remaining portion $(u_{n+1}, \ldots, u_N)$ of the solution. Let us consider the corresponding rollout algorithm. It can be seen that this algorithm selects as the first solution component

$$\overline{u}_1 = \arg \min_{u_1 \in U_1} H(u_1),$$

and operates sequentially, for $n = 1, \ldots, N-1$, as follows:

> Given a partial solution $(\overline{u}_1, \ldots, \overline{u}_n)$, it runs the heuristic starting from the partial solutions $(\overline{u}_1, \ldots, \overline{u}_n, u_{n+1})$ corresponding to all the possible next solution components $u_{n+1} \in U_{n+1}(\overline{u}_1, \ldots, \overline{u}_n)$, and selects as next component
>
> $$\overline{u}_{n+1} = \arg \min_{u_{n+1} \in U_{n+1}(\overline{u}_1, \ldots, \overline{u}_n)} H(\overline{u}_1, \ldots, \overline{u}_n, u_{n+1}).$$

In order to analyze most economically the preceding algorithm and its variants, we will embed it within a more general and flexible framework for

discrete optimization. To this end, we introduce a graph search problem, which contains as special cases broad classes of discrete/integer optimization problems, and will serve as the context of our methodology. We will then describe and analyze a basic form of a one-step lookahead algorithm, we will discuss some of its variations, we will illustrate it by means of some examples, and we will discuss its connection with the DP context.

As we explain later (see the end of Section 6.4.1), the algorithm to be introduced is not quite a rollout algorithm in the sense discussed so far, because, strictly speaking, it does not use the cost of a heuristic policy as a one-step lookahead cost approximation, except under a special assumption (the sequential consistency assumption, to be described later). The basic idea of the algorithm is, however, very close to a rollout: it is a one-step lookahead policy with cost approximation *derived* from a heuristic. Thus, with a stretch of terminology, we will call this algorithm "rollout" as well.

The Basic Rollout Algorithm for Discrete Optimization

Let us introduce a graph search problem that will serve as a general model for discrete optimization. We are given a graph with set of nodes $\mathcal{N}$, set of arcs $\mathcal{A}$, and a special node s, which we call the *origin*. The arcs are directed in the sense that arc (i, j) is distinct from arc (j, i). We are also given a subset $\overline{\mathcal{N}}$ of nodes, called *destinations*, and a cost $g(i)$ for each destination i. The destination nodes are terminal in the sense that they have no outgoing arcs. For simplicity, we assume that the node and arc sets, $\mathcal{N}$ and $\mathcal{A}$ contain a finite number of elements. However, the following analysis and discussion applies, with minor modifications in language, to the case of a countably infinite number of nodes and a finite set of outgoing arcs from each node. We want to find a path that starts at the origin s, ends at one of the destination nodes $i \in \overline{\mathcal{N}}$, and is such that the cost $g(i)$ is minimized.

In the context of the discrete optimization problem (6.31), nodes i correspond to n-tuples $(u_1, \ldots, u_n)$ consisting of the first n components of a solution, where $n = 1, \ldots, N$. Arcs lead from nodes of the form $(u_1, \ldots, u_{n-1})$ to nodes of the form $(u_1, \ldots, u_{n-1}, u_n)$, and there is an arc for each u_n of the form (6.32). An interesting property of this special case is that its associated graph is acyclic.

In our terminology, a path is a sequence of arcs

$$(i_1, i_2), (i_2, i_3), \ldots, (i_{m-1}, i_m),$$

all of which are oriented in the forward direction. The nodes i_1 and i_m are called the *start node* and the *end node* of the path, respectively. For convenience, and without loss of generality,† we will assume that given an

† In the case where there are multiple arcs connecting a node pair, we can merge all these arcs to a single arc, since the set of destination nodes that can be reached from any non-destination node will not be affected.

ordered pair of nodes (i,j), there is at most one arc with start node i and end node j, which (if it exists) will be denoted by (i,j). In this way, a path consisting of arcs $(i_1,i_2),(i_2,i_3),\ldots,(i_{m-1},i_m)$ is unambiguously specified as the sequence of nodes $(i_1,i_2,\ldots,i_m)$.

Let us assume that we have a heuristic path construction algorithm, denoted $\mathcal{H}$, which given a non-destination node $i \notin \overline{N}$, constructs a path $(i,i_1,\ldots,i_m,\bar{i})$ starting at i and ending at one of the destination nodes $\bar{i}$. Implicit in this assumption is that for every non-destination node, there exists at least one path starting at that node and ending at some destination node. We refer to the algorithm $\mathcal{H}$ as the *base heuristic*, since we will use this algorithm as the basic building block for constructing the rollout algorithm to be introduced shortly.

The end node $\bar{i}$ of the path constructed by the base heuristic $\mathcal{H}$ is completely specified by the start node i. We call $\bar{i}$ the *projection of i under* $\mathcal{H}$, and we denote it by $p(i)$. We denote the corresponding cost by $H(i)$,

$$H(i) = g\big(p(i)\big).$$

The projection of a destination node is the node itself by convention, so that for all $i \in \overline{N}$ we have $i = p(i)$ and $H(i) = g(i)$. Note that while the base heuristic $\mathcal{H}$ will generally yield a suboptimal solution, the path that it constructs may involve a fairly sophisticated suboptimization. For example, $\mathcal{H}$ may construct several paths ending at destination nodes according to some heuristics, and then select the path that yields minimal cost.

One possibility for suboptimal solution of the problem is to start at the origin s and use the base heuristic $\mathcal{H}$ to obtain the projection $p(s)$. We instead propose to use $\mathcal{H}$ to construct a path to a destination node sequentially. At the typical step of the sequence, we consider all downstream neighbors j of a node i, we run the base heuristic $\mathcal{H}$ starting from each of these neighbors, and obtain the corresponding projections and costs. We then move to the neighbor that gives the best projection. This sequential version of $\mathcal{H}$ is called the *rollout algorithm based on* $\mathcal{H}$, and is denoted by $\mathcal{RH}$.

To formally describe the rollout algorithm, let $N(i)$ denote the set of downstream neighbors of node i,

$$N(i) = \big\{j \mid (i,j) \text{ is an arc}\big\}.$$

Note that $N(i)$ is nonempty for every non-destination node i, since by assumption there exists a path starting at i and ending at its projection $p(i)$. The rollout algorithm $\mathcal{RH}$ starts with the origin node s. At the typical step, given a node sequence $(s,i_1,\ldots,i_m)$, where i_m is not a destination, $\mathcal{RH}$ adds to the sequence a node i_{m+1} such that

$$i_{m+1} = \arg\min_{j\in N(i_m)} H(j). \tag{6.33}$$

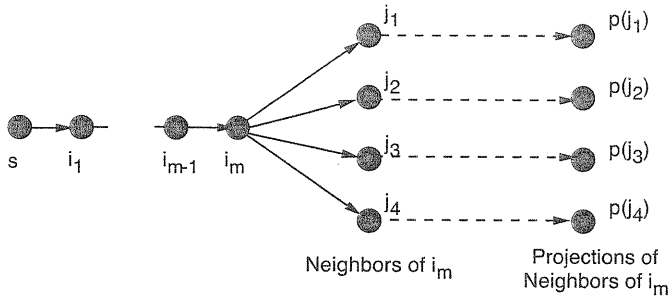


Figure 6.4.2 Illustration of the rollout algorithm. After m steps of the algorithm, we have the path $(s,i_1,\ldots,i_m)$. To extend this path at the next step, we generate the set $N(i_m)$ of neighbors of the terminal node i_m, and select from this set the neighbor that has the best projection, i.e.

$$i_{m+1} = \arg\min_{j\in N(i_m)} H(j) = \arg\min_{j\in N(i_m)} g\big(p(j)\big).$$

If i_{m+1} is a destination node, $\mathcal{RH}$ terminates. Otherwise, the process is repeated with the sequence $(s,i_1,\ldots,i_m,i_{m+1})$ replacing $(s,i_1,\ldots,i_m)$; see Fig. 6.4.2.

Note that once $\mathcal{RH}$ has terminated with a path $(s,i_1,\ldots,i_m)$, we will have obtained the projection $p(i_k)$ of each of the nodes i_k, $k=1,\ldots,m$. The best of these projections yields a cost

$$\min_{k=1,\ldots,m} H(i_k) = \min_{k=1,\ldots,m} g\big(p(i_k)\big),$$

and the projection that corresponds to the minimum above may be taken as the final (suboptimal) solution produced by the rollout algorithm. We may also compare the above minimal cost with the cost $g\big(p(s)\big)$ of the projection $p(s)$ of the origin, and use $p(s)$ as the final solution if it produces a smaller cost. This will ensure that the rollout algorithm will produce a solution that is no worse than the one produced by the base heuristic.

Example 6.4.3 (Traveling Salesman Problem)

Let us consider the traveling salesman problem, whereby a salesman wants to find a minimum mileage/cost tour that visits each of N given cities exactly once and returns to the city he started from. We associate a node with each city $i = 1,\ldots,N$, and we introduce an arc (i,j) with traversal cost a_{ij} for each ordered pair of nodes i and j. Note that we assume that the graph is complete; that is, there exists an arc for each ordered pair of nodes. There is no loss of generality in doing so because we can assign a very high cost a_{ij} to

an arc (i,j) that is precluded from participation in the solution. The problem is to find a cycle that goes through all the nodes exactly once and whose sum of arc costs is minimum.

There are many heuristic approaches for solving the traveling salesman problem. For illustration purposes, let us restrict attention to the simple *nearest neighbor* heuristic. Here, we start from a path consisting of just a single node i_1 and at each iteration, we enlarge the path with a node that does not close a cycle and minimizes the cost of the enlargement. In particular, after k iterations, we have a path $\{i_1,\ldots,i_k\}$ consisting of distinct nodes, and at the next iteration, we add an arc (i_k,i_{k+1}) that minimizes $a_{i_k i}$ over all arcs (i_k,i) with $i \neq i_1,\ldots,i_k$. After $N-1$ iterations, all nodes are included in the path, which is then converted to a tour by adding the final arc (i_N,i_1).

We can formulate the traveling salesman problem as a graph search problem as follows: There is a chosen starting city, say i_1 corresponding to the origin of the graph search problem. Each node of the graph search problem corresponds to a path $(i_1,i_2,\ldots,i_k)$, where $i_1,i_2,\ldots,i_k$ are distinct cities. The neighbor nodes of the path $(i_1,i_2,\ldots,i_k)$ are paths of the form $(i_1,i_2,\ldots,i_k,i_{k+1})$ which correspond to adding one more unvisited city $i_{k+1} \neq i_1,i_2,\ldots,i_k$ at the end of the path. The destinations are the cycles of the form $(i_1,i_2,\ldots,i_N)$, and the cost of a destination in the graph search problem is the cost of the corresponding cycle. Thus a path from the origin to a destination in the graph search problem corresponds to constructing a cycle in $N-1$ arc addition steps, and at the end incurring the cost of the cycle.

Let us now use as base heuristic the nearest neighbor method. The corresponding rollout algorithm operates as follows: After k iterations, we have a path $\{i_1,\ldots,i_k\}$ consisting of distinct nodes. At the next iteration, we run the nearest neighbor heuristic starting from each of the paths of the form $\{i_1,\ldots,i_k,i\}$ where $i \neq i_1,\ldots,i_k$, and obtain a corresponding cycle. We then select as next node i_{k+1} of the path the node i that corresponds to the best cycle thus obtained.

Termination and Sequential Consistency

We say that the rollout algorithm $\mathcal{RH}$ is *terminating* if it is guaranteed to terminate finitely starting from any node. Contrary to the base heuristic $\mathcal{H}$, which by definition, has the property that it yields a path terminating at a destination starting from any node, the rollout algorithm $\mathcal{RH}$ need not have this property in the absence of additional conditions. The termination question can usually be resolved quite easily, and we will now discuss a few different methods by which this can be done.

One important case where $\mathcal{RH}$ is terminating is *when the graph is acyclic*, since then the nodes of the path generated by $\mathcal{RH}$ cannot be repeated within the path, and their number is bounded by the number of nodes in $\mathcal{N}$. As a first step towards developing another case where $\mathcal{RH}$ is terminating, we introduce the following definition, which will also set the stage for further analysis of the properties of $\mathcal{RH}$.

Definition 6.4.1: We say that the base heuristic $\mathcal{H}$ is *sequentially consistent* if for every node i, it has the following property: If $\mathcal{H}$ generates the path $(i,i_1,\ldots,i_m,\bar{i})$ when it starts at i, it generates the path $(i_1,\ldots,i_m,\bar{i})$ when it starts at the node i_1.

Thus $\mathcal{H}$ is sequentially consistent if all the nodes of a path that it generates have the same projection. There are many examples of sequentially consistent algorithms that are used as heuristics in combinatorial optimization, including the following.

Example 6.4.4 (Greedy Algorithms as Base Heuristics)

Suppose that we have a function F, which for each node i, provides a scalar estimate $F(i)$ of the optimal cost starting from i, that is, the minimal cost $g(\bar{i})$ that can be obtained with a path that starts at i and ends at one of the destination nodes $\bar{i} \in \overline{\mathcal{N}}$. Then F can be used to define a base heuristic, called the *greedy algorithm with respect to* F, as follows:

The greedy algorithm starts at a node i with the (degenerate) path that consists of just node i. At the typical step, given a path $(i,i_1,\ldots,i_m)$, where i_m is not a destination, the algorithm adds to the path a node i_{m+1} such that

$$i_{m+1} = \arg\min_{j \in N(i_m)} F(j). \tag{6.34}$$

In the case where i_{m+1} is a destination, the algorithm terminates with the path $(i,i_1,\ldots,i_m,i_{m+1})$. Otherwise, the process is repeated with the path $(i,i_1,\ldots,i_m,i_{m+1})$ replacing $(i,i_1,\ldots,i_m)$.

An example of a greedy algorithm is the nearest neighbor heuristic for the traveling salesman problem (cf. Example 6.4.3). Recall from that example that nodes of the graph search problem correspond to paths (sequences of distinct cities), and a transition to a neighbor node corresponds to adding one more unvisited city to the end of the current path. The function F in the nearest neighbor heuristic specifies the cost of the addition of the new city.

It is also interesting to note that by viewing F as a cost-to-go approximation, we may consider the greedy algorithm to be a special type of one-step lookahead policy. Furthermore, if $F(j)$ is chosen to be the cost obtained by some base heuristic starting from j, then the greedy algorithm becomes the corresponding rollout algorithm. Thus, it may be said that the rollout algorithm is a special case of a greedy algorithm. However, the particular choice of F used in the rollout algorithm is responsible for special properties that are not shared by other types of greedy algorithms.

Let us denote by $\mathcal{H}$ the greedy algorithm described above and assume that it terminates starting from every node (this has to be verified independently). Let us also assume that whenever there is a tie in the minimization of Eq. (6.34), $\mathcal{H}$ resolves the tie in a manner that is fixed and independent of the starting node i of the path, e.g., by resolving the tie in favor of the

numerically smallest node j that attains the minimum in Eq. (6.34). Then it can be seen that $\mathcal{H}$ is sequentially consistent, since by construction, every node on a path generated by $\mathcal{H}$ has the same projection.

For a sequentially consistent base heuristic $\mathcal{H}$, we will assume a restriction in the way the rollout algorithm $\mathcal{RH}$ resolves ties in selecting the next node on its path; this restriction will guarantee that $\mathcal{RH}$ is terminating. In particular, suppose that after m steps, $\mathcal{RH}$ has produced the node sequence $(s, i_1, \ldots, i_m)$, and that the path generated by $\mathcal{H}$ starting from i_m is $(i_m, i_{m+1}, i_{m+2}, \ldots, \bar{i})$. Suppose that among the neighbor set $N(i_m)$, the node i_{m+1} attains the minimum in the selection test

$$\min_{j \in N(i_m)} H(j), \tag{6.35}$$

but there are also some other nodes, in addition to i_{m+1}, that attain this minimum. Then, we require that the tie is broken in favor of i_{m+1}, i.e., that the next node added to the current sequence $(s, i_1, \ldots, i_m)$ is i_{m+1}. Under this convention for tie-breaking, we show in the following proposition that the rollout algorithm $\mathcal{RH}$ terminates at a destination and yields a cost that is no larger than the cost yielded by the base heuristic $\mathcal{H}$.†

Proposition 6.4.1: Let the base heuristic $\mathcal{H}$ be sequentially consistent. Then the rollout algorithm $\mathcal{RH}$ is terminating. Furthermore, if $(i_1, \ldots, i_{\tilde{m}})$ is the path generated by $\mathcal{RH}$ starting from a non-destination node i_1 and ending at a destination node $i_{\tilde{m}}$, the cost of $\mathcal{RH}$ starting from i_1 is less or equal to the cost of $\mathcal{H}$ starting from i_1. In particular, we have

$$H(i_1) \geq H(i_2) \geq \cdots \geq H(i_{\tilde{m}-1}) \geq H(i_{\tilde{m}}). \tag{6.36}$$

† For an example where this convention for tie-breaking is not observed and as a consequence $\mathcal{RH}$ does not terminate, assume that there is a single destination d and that all other nodes are arranged in a cycle. Each non-destination node i has two outgoing arcs: one arc that belongs to the cycle, and another arc which is (i, d). Let $\mathcal{H}$ be the (sequentially consistent) base heuristic that starting from a node $i \neq d$, generates the path (i, d). When the terminal node of the path is node i, the rollout algorithm $\mathcal{RH}$ compares the two neighbors of i, which are d and the node next to i on the cycle, call it j. Both neighbors have d as their projection, so there is tie in Eq. (6.35). It can be seen that if $\mathcal{RH}$ breaks ties in favor of the neighbor j that lies on the cycle, then $\mathcal{RH}$ continually repeats the cycle and never terminates.

Furthermore, for all $m = 1, \ldots, \tilde{m}$,

$$H(i_m) = \min \left\{ H(i_1), \min_{j \in N(i_1)} H(j), \ldots, \min_{j \in N(i_{m-1})} H(j) \right\}. \tag{6.37}$$

Proof: Let i_m and i_{m+1} be two successive nodes generated by $\mathcal{RH}$, and let $(i_m, i'_{m+1}, i'_{m+2}, \ldots, \bar{i}_m)$ be the path generated by $\mathcal{H}$ starting from i_m, where $\bar{i}_m$ is the projection of i_m. Then, since $\mathcal{H}$ is sequentially consistent, we have

$$H(i_m) = H(i'_{m+1}) = g(\bar{i}_m).$$

Furthermore, since $i'_{m+1} \in N(i_m)$, we have using the definition of $\mathcal{RH}$ [cf. Eq. (6.33)]

$$H(i'_{m+1}) \geq \min_{j \in N(i_m)} H(j) = H(i_{m+1}).$$

Combining the last two relations, we obtain

$$H(i_m) \geq H(i_{m+1}) = \min_{j \in N(i_m)} H(j). \tag{6.38}$$

To show that $\mathcal{RH}$ is terminating, note that in view of Eq. (6.38), either $H(i_m) > H(i_{m+1})$, or else $H(i_m) = H(i_{m+1})$. In the latter case, in view of the convention for breaking ties that occur in Eq. (6.35) and the sequential consistency of $\mathcal{H}$, the path generated by $\mathcal{H}$ starting from i_{m+1} is the tail portion of the path generated by $\mathcal{H}$ starting from i_m, and has one arc less. Thus the number of nodes generated by $\mathcal{RH}$ between successive times that the inequality $H(i_m) > H(i_{m+1})$ holds is finite. On the other hand, the inequality $H(i_m) > H(i_{m+1})$ can occur only a finite number of times, since the number of destination nodes is finite, and the destination node of the path generated by $\mathcal{H}$ starting from i_m cannot be repeated if the inequality $H(i_m) > H(i_{m+1})$ holds. Therefore, $\mathcal{RH}$ is terminating.

If $(i_1, \ldots, i_{\tilde{m}})$ is the path generated by $\mathcal{RH}$, the relation (6.38) implies the desired relations (6.36) and (6.37). **Q.E.D.**

Proposition 6.4.1 shows that in the sequentially consistent case, the rollout algorithm $\mathcal{RH}$ has an important "automatic cost sorting" property, whereby it follows the best path generated by the base heuristic $\mathcal{H}$. In particular, when $\mathcal{RH}$ generates a path $(i_1, \ldots, i_{\tilde{m}})$, it does so by using $\mathcal{H}$ to generate a collection of other paths and corresponding projections starting from all the successor nodes of the intermediate nodes $i_1, \ldots, i_{\tilde{m}-1}$. However, $(i_1, \ldots, i_{\tilde{m}})$ is guaranteed to be the best among this path collection and $i_{\tilde{m}}$ has minimal cost among all generated projections [cf. Eq. (6.37)]. Of course this does not guarantee that the path generated by $\mathcal{RH}$ will be a

near-optimal path, because the collection of paths generated by $\mathcal{H}$ may be "poor." Still, the property whereby $\mathcal{RH}$ at all times follows the best path found so far is intuitively reassuring.

The following example illustrates the preceding concepts.

Example 6.4.5 (One-Dimensional Walk)

Consider a person who walks on a straight line and at each time period takes either a unit step to the left or a unit step to the right. There is a cost function assigning cost $g(i)$ to each integer i. Given an integer starting point on the line, the person wants to minimize the cost of the point where he will end up after a given and fixed number N of steps.

We can formulate this problem as a graph search problem of the type discussed in the preceding section. In particular, without loss of generality, let us assume that the starting point is the origin, so that the person's position after n steps will be some integer in the interval $[-n, n]$. The nodes of the graph are identified with pairs (k, m), where k is the number of steps taken so far ($k = 1, \ldots, N$) and m is the person's position ($m \in [-k, k]$). A node (k, m) with $k < N$ has two outgoing arcs with end nodes $(k+1, m-1)$ (corresponding to a left step) and $(k+1, m+1)$ (corresponding to a right step). The starting state is $(0, 0)$ and the terminating states are of the form (N, m), where m is of the form $N - 2l$ and $l \in [0, N]$ is the number of left steps taken.

Let the base heuristic $\mathcal{H}$ be defined as the algorithm, which, starting at a node (k, m), takes $N - k$ successive steps to the right and terminates at the node $(N, m + N - k)$. Note that $\mathcal{H}$ is sequentially consistent. The rollout algorithm $\mathcal{RH}$, at node (k, m) compares the cost of the destination node $(N, m+N-k)$ (corresponding to taking a step to the right and then following $\mathcal{H}$) and the cost of the destination node $(N, m + N - k - 2)$ (corresponding to taking a step to the left and then following $\mathcal{H}$).

Let us say that an integer $i \in [-N + 2, N - 2]$ is a *local minimum* if $g(i-2) \geq g(i)$ and $g(i) \leq g(i+2)$. Let us also say that N (or $-N$) is a local minimum if $g(N-2) \leq g(N)$ [or $g(-N) \leq g(-N+2)$, respectively]. Then it can be seen that starting from the origin $(0, 0)$, $\mathcal{RH}$ obtains the local minimum that is closest to N, (see Fig. 6.4.3). This is no worse (and typically better) than the integer N obtained by $\mathcal{H}$. Note that if g has a unique local minimum in the set of integers in the range $[-N, N]$, the minimum must also be global, and it will be found by $\mathcal{RH}$. This example illustrates how $\mathcal{RH}$ may exhibit "intelligence" that is totally lacking from $\mathcal{H}$, and is in agreement with the result of Prop. 6.4.1.

Sequential Improvement

It is possible to show that the rollout algorithm improves on the base heuristic (cf. Prop. 6.4.1) under weaker conditions. To this end we introduce the following definition.

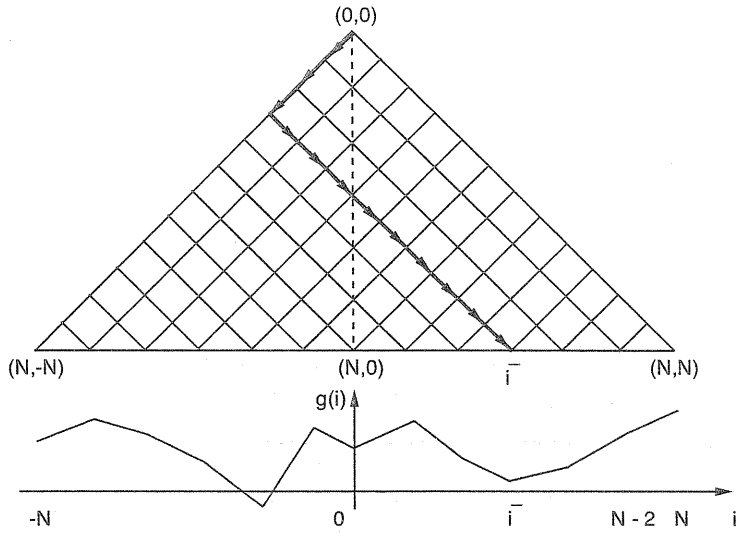


Figure 6.4.3 Illustration of the path generated by the rollout algorithm $\mathcal{RH}$ in Example 6.4.5. The algorithm keeps moving to the left up to the time where the base heuristic $\mathcal{H}$ generates two destinations $(N, \bar{i})$ and $(N, \bar{i}-2)$ with $g(\bar{i}) \leq g(\bar{i}-2)$. Then it continues to move to the right ending at the destination $(N, \bar{i})$, which corresponds to the local minimum closest to N.

Definition 6.4.2: We say that the base heuristic $\mathcal{H}$ is *sequentially improving* if for every non-destination node i, we have

$$H(i) \geq \min_{j \in N(i)} H(j). \tag{6.39}$$

It can be seen that a sequentially consistent $\mathcal{H}$ is also sequentially improving, since sequential consistency implies that $H(i)$ is equal to one of the values $H(j)$, $j \in N(i)$. We have the following generalization of Prop. 6.4.1, which also bears a relation with the general cost estimate for one-step lookahead policies of Prop. 6.3.1.

Proposition 6.4.2: Let the base heuristic $\mathcal{H}$ be sequentially improving, and assume that the rollout algorithm $\mathcal{RH}$ is terminating. Let $(i_1, \ldots, i_{\bar{m}})$ be the path generated by $\mathcal{RH}$ starting from a non-destination node i_1 and ending at a destination node $i_{\bar{m}}$. Then the

cost of $\mathcal{RH}$ starting from i_1 is less or equal to the cost of $\mathcal{H}$ starting from i_1. In particular, we have for all $m = 1, \ldots, \tilde{m}$,

$$H(i_m) = \min \left\{ H(i_1), \min_{j \in N(i_1)} H(j), \ldots, \min_{j \in N(i_{m-1})} H(j) \right\}. \qquad (6.40)$$

Proof: For each $m = 1, \ldots, \tilde{m} - 1$, we have

$$H(i_m) \geq \min_{j \in N(i_m)} H(j),$$

by the sequential improvement assumption, while we have

$$\min_{j \in N(i_m)} H(j) = H(i_{m+1}),$$

by the definition of the rollout algorithm. These two relations imply Eq. (6.40). Since the cost of $\mathcal{RH}$ starting from i_1 is $H(i_{\tilde{m}})$, the result follows. **Q.E.D.**

Example 6.4.6 (One-Dimensional Walk – Continued)

Consider the one-dimensional walk problem of Example 6.4.5, and let $\mathcal{H}$ be defined as the algorithm that, starting at a node (k, m), compares the cost $g(m+N-k)$ (corresponding to taking all of the remaining $N-k$ steps to the right) and the cost $g(m-N+k)$ (corresponding to taking all of the remaining $N-k$ steps to the left), and accordingly moves to node

$$(N, m+N-k) \qquad \text{if} \qquad g(m+N-k) \leq g(m-N+k),$$

or to node

$$(N, m-N+k) \qquad \text{if} \qquad g(m-N+k) < g(m+N-k).$$

It can be seen that $\mathcal{H}$ is not sequentially consistent, but is instead sequentially improving. Using Eq. (6.40), it follows that starting from the origin $(0, 0)$, $\mathcal{RH}$ obtains the global minimum of g in the interval $[-N, N]$, while $\mathcal{H}$ obtains the better of the two points $-N$ and N.

Proposition 6.4.2 actually follows from a general equation for the cost of the path generated by the rollout algorithm, which holds for any base heuristic (not necessarily one that is sequentially improving). This is given in the following proposition, which is related to Prop. 6.3.2.

Proposition 6.4.3: Assume that the rollout algorithm $\mathcal{RH}$ is terminating. Let $(i_1, \ldots, i_{\tilde{m}})$ be the path generated by $\mathcal{RH}$ starting from a non-destination node i_1 and ending at a destination node $i_{\tilde{m}}$. Then the cost of $\mathcal{RH}$ starting from i_1 is equal to

$$H(i_1) + \delta_{i_1} + \cdots + \delta_{i_{\tilde{m}-1}},$$

where for every non-destination node i, we denote

$$\delta_i = \min_{j \in N(i)} H(j) - H(i).$$

Proof: We have by the definition of the rollout algorithm

$$H(i_m) + \delta_{i_m} = \min_{j \in N(i_m)} H(j) = H(i_{m+1}), \qquad m = 1, \ldots, \tilde{m} - 1.$$

By adding these equations over m, we obtain

$$H(i_1) + \delta_{i_1} + \cdots + \delta_{i_{\tilde{m}-1}} = H(i_{\tilde{m}}).$$

Since the cost of $\mathcal{RH}$ starting from i_1 is $H(i_{\tilde{m}})$, the result follows. **Q.E.D.**

If the base heuristic is sequentially improving, we have $\delta_i \leq 0$ for all non-destination nodes i, so it follows from Prop. 6.4.3 that the cost of the rollout algorithm is less or equal to the cost of the base heuristic (cf. Prop. 6.4.2).

The Fortified Rollout Algorithm

We now describe a variant of the rollout algorithm that implicitly uses a sequentially improving base heuristic, so that it has the cost improvement property of Prop. 6.4.2. This variant, called the *fortified rollout algorithm*, and denoted by $\mathcal{R}\overline{\mathcal{H}}$, starts at the origin s, and after m steps, maintains, in addition to the current sequence of nodes $(s, i_1, \ldots, i_m)$, a path

$$P(i_m) = (i_m, i'_{m+1}, \ldots, i'_k),$$

ending at a destination i'_k. Roughly speaking, the path $P(i_m)$ is the tail portion of the best path found after the first m steps of the algorithm, in the sense that the destination i'_k has minimal cost over all the projections of nodes calculated thus far.

In particular, initially $P(s)$ is the path generated by the base heuristic $\mathcal{H}$ starting from s. At the typical step of the fortified rollout algorithm $\mathcal{R}\overline{\mathcal{H}}$,

we have a node sequence $(s, i_1, \ldots, i_m)$, where i_m is not a destination, and the path $P(i_m) = (i_m, i'_{m+1}, \ldots, i'_k)$. Then, if

$$\min_{j \in N(i_m)} H(j) < g(i'_k), \tag{6.41}$$

$\mathcal{R}\overline{\mathcal{H}}$ adds to the node sequence $(s, i_1, \ldots, i_m)$ the node

$$i_{m+1} = \arg \min_{j \in N(i_m)} H(j),$$

and sets $P(i_{m+1})$ to the path generated by $\mathcal{H}$, starting from i_{m+1}. On the other hand, if

$$\min_{j \in N(i_m)} H(j) \geq g(i'_k), \tag{6.42}$$

$\mathcal{R}\overline{\mathcal{H}}$ adds to the node sequence $(s, i_1, \ldots, i_m)$ the node

$$i_{m+1} = i'_{m+1},$$

and sets $P(i_{m+1})$ to the path $(i_{m+1}, i'_{m+2}, \ldots, i'_k)$. If i_{m+1} is a destination, $\mathcal{R}\overline{\mathcal{H}}$ terminates, and otherwise $\mathcal{R}\overline{\mathcal{H}}$ repeats the process with $(s, i_1, \ldots, i_{m+1})$ replacing $(s, i_1, \ldots, i_m)$, and $P(i_{m+1})$ replacing $P(i_m)$, respectively.

The idea behind the construction of $\mathcal{R}\overline{\mathcal{H}}$ is to follow the path $P(i_m)$ unless a path of lower cost is discovered through Eq. (6.41). We can show that $\mathcal{R}\overline{\mathcal{H}}$ may be viewed as the rollout algorithm $\mathcal{R}\mathcal{H}$ corresponding to a modified version of $\mathcal{H}$, called *fortified* $\mathcal{H}$, and denoted $\overline{\mathcal{H}}$. This algorithm is applied to a slightly modified version of the original problem, which involves an additional downstream neighbor for each node i_m that is generated in the course of the algorithm $\mathcal{R}\overline{\mathcal{H}}$ and for which the condition (6.42) holds. For every such node i_m, the additional neighbor is a copy of i'_{m+1}, and the path generated by $\overline{\mathcal{H}}$ starting from this copy is $(i'_{m+1}, \ldots, i'_k)$. From every other node, the path generated by $\overline{\mathcal{H}}$ is the same as the path generated by $\mathcal{H}$.

It can be seen that $\overline{\mathcal{H}}$ is sequentially improving, so that $\mathcal{R}\overline{\mathcal{H}}$ is terminating and has the automatic cost sorting property of Prop. 6.4.2; that is,

$$H(i_m) = \min \left\{ H(i_1), \min_{j \in N(i_1)} H(j), \ldots, \min_{j \in N(i_{m-1})} H(j) \right\}.$$

The above property can also be easily verified directly, using the definition of $\mathcal{R}\overline{\mathcal{H}}$. Finally, it can be seen that when $\mathcal{H}$ is sequentially consistent, the rollout algorithm $\mathcal{R}\mathcal{H}$ and its fortified version $\mathcal{R}\overline{\mathcal{H}}$ coincide.

Using Multiple Path Construction Algorithms

In many problems, several promising path construction heuristics may be available. It is then possible to use all of these heuristics in the rollout framework. In particular, let us assume that we have K algorithms $\mathcal{H}_1, \ldots, \mathcal{H}_K$. The kth of these algorithms, given a non-destination node i, produces a path $(i, i_1, \ldots, i_m, \bar{i})$ that ends at a destination node $\bar{i}$, and the corresponding cost is denoted by $H_k(i) = g(\bar{i})$. We can incorporate the K algorithms in a generalized version of the rollout algorithm, which uses the minimal cost

$$H(i) = \min_{k=1,\ldots,K} H_k(i), \tag{6.43}$$

in place of the cost obtained by any one of the K algorithms $\mathcal{H}_1, \ldots, \mathcal{H}_K$.

In particular, the algorithm starts with the origin node s. At the typical step, given a node sequence $(s, i_1, \ldots, i_m)$, where i_m is not a destination, the algorithm adds to the sequence a node i_{m+1} such that

$$i_{m+1} = \arg \min_{j \in N(i_m)} H(j).$$

If i_{m+1} is a destination node, the algorithm terminates, and otherwise the process is repeated with the sequence $(s, i_1, \ldots, i_m, i_{m+1})$ replacing $(s, i_1, \ldots, i_m)$.

An interesting property, which can be readily verified by using the definitions, is that if all the algorithms $\mathcal{H}_1, \ldots, \mathcal{H}_K$ are sequentially improving, the same is true for $\mathcal{H}$. This is consistent with the analysis of Example 6.3.2.

The fortified version of the rollout algorithm $\mathcal{R}\overline{\mathcal{H}}$ easily generalizes for the case of Eq. (6.43), by defining the path generated starting from a node i as the path generated by the path construction algorithm, which attains the minimum in Eq. (6.43).

In an alternative version of the rollout algorithm that uses multiple path construction heuristics, the results of the K algorithms $\mathcal{H}_1, \ldots, \mathcal{H}_K$ are weighted with some fixed scalar weights r_k to compute $H(i)$ for use in Eq. (6.33):

$$H(i) = \sum_{k=1}^{K} r_k H_k(i). \tag{6.44}$$

The weights r_k may be adjusted by trial and error. An alternative and more sophisticated possibility, is to use weights that depend on the node i and which are obtained by training using the neuro-dynamic programming methodology described in Vol. II.

Extension for Intermediate Arc Costs

Let us consider a variant of the graph search problem where in addition to the terminal cost $g(i)$, there is a cost $c(i, j)$ for a path to traverse an arc

(i,j). Within this context, the cost of a path $(i_1, i_2, \ldots, i_n)$ that starts at i_1 and ends at a destination node i_n is redefined to be

$$g(i_n) + \sum_{k=1}^{n-1} c(i_k, i_{k+1}). \tag{6.45}$$

Note that when the cost $g(i)$ is zero for all destination nodes i, this is the problem of finding a shortest path from the origin node s to one of the destination nodes, with $c(i,j)$ viewed as the length of arc (i,j). We have seen in Chapter 2 that there are efficient algorithms for solving this problem. However, here we are interested in problems where the number of nodes is very large, and the use of the shortest path algorithms of Chapter 2 is impractical.

One way to transform the problem with arc costs into one involving a terminal cost only is to redefine the graph of the problem so that nodes correspond to sequences of nodes in the original problem graph. Thus if we have arrived at node i_k using path $(i_1, \ldots, i_k)$, the choice of i_{k+1} as the next node is viewed as a transition from $(i_1, \ldots, i_k)$ to $(i_1, \ldots, i_k, i_{k+1})$. Both nodes $(i_1, \ldots, i_k)$ and $(i_1, \ldots, i_k, i_{k+1})$ are viewed as nodes of a redefined graph. Furthermore, in this redefined graph, a destination node has the form $(i_1, i_2, \ldots, i_n)$, where i_n is a destination node of the original graph, and has a cost given by Eq. (6.45).

After the details are worked out, we see that to recover our earlier algorithms and analysis, we need to modify the cost of the heuristic algorithm $\mathcal{H}$ as follows: If the path $(i_1, \ldots, i_n)$ is generated by $\mathcal{H}$ starting at i_1, then

$$H(i_1) = g(i_n) + \sum_{k=1}^{n-1} c(i_k, i_{k+1}).$$

Furthermore, the rollout algorithm $\mathcal{RH}$ at node i_m selects as next node i_{m+1} the node

$$i_{m+1} = \arg \min_{j \in N(i_m)} \big[c(i_m, j) + H(j)\big];$$

[cf. Eq. (6.33)]. The definition of a sequentially consistent algorithm remains unchanged. Furthermore, Prop. 6.4.1 remains unchanged except that Eqs. (6.36) and (6.37) are modified to read

$$H(i_k) \geq c(i_k, i_{k+1}) + H(i_{k+1}) = \min_{j \in N(i_k)} \big[c(i_k, j) + H(j)\big], \quad k = 1, \ldots, m-1.$$

A sequentially improving algorithm should now be characterized by the property

$$H(i_k) \geq c(i_k, i_{k+1}) + H(i_{k+1}),$$

where i_{k+1} is the next node on the path generated by $\mathcal{H}$ starting from i_k. Furthermore, Prop. 6.4.2 remains unchanged, except that Eq. (6.40) is modified to read

$$H(i_k) \geq \min_{j \in N(i_k)} \big[c(i_k, j) + H(j)\big], \qquad k = 1, \ldots, m-1.$$

Finally, the criterion $\min_{j \in N(i_m)} H(j) < g(i'_k)$ [cf. Eq. (6.41)] used in the fortified rollout algorithm, given the sequence $(s, i_1, \ldots, i_m)$, where $i_m \notin \overline{\mathcal{N}}$, and the path $P(i_m) = (i_m, i'_{m+1}, \ldots, i'_k)$, should be replaced by

$$\min_{j \in N(i_m)} \big[c(i_m, j) + H(j)\big] < g(i'_k) + c(i_m, i'_{m+1}) + \sum_{l=m+1}^{k-1} c(i'_l, i'_{l+1}).$$

Rollout Algorithms with Multistep Lookahead

We may incorporate *multistep lookahead* into the rollout framework. To describe the case of 2-step lookahead, suppose that after m steps of the rollout algorithm, we have the current node sequence $(s, i_1, \ldots, i_m)$. We then consider the set of all 2-step-ahead neighbors of i_m, defined as

$$N_2(i_m) = \big\{ j \in \mathcal{N} \mid j \in N(i_m) \text{ and } j \in \overline{\mathcal{N}}, \text{ or } j \in N(n) \text{ for some } n \in N(i_m) \big\}.$$

We run the base heuristic $\mathcal{H}$ starting from each $j \in N_2(i_m)$ and we find the node $\overline{j} \in N_2(i_m)$ that has projection of minimum cost. Let $i_{m+1} \in N(i_m)$ be the node next to i_m on the (one- or two-arc) path from i_m to $\overline{j}$. If i_{m+1} is a destination node, the algorithm terminates. Otherwise, the process is repeated with the sequence $(s, i_1, \ldots, i_m, i_{m+1})$ replacing $(s, i_1, \ldots, i_m)$.

Note that a fortified version of the rollout algorithm described above is possible along the lines described earlier. Also, it is possible to eliminate from the set $N_2(i_m)$ some of the 2-step neighbors of i_m that are judged less promising according to some heuristic criterion, in order to limit the number of applications of the base heuristic. This may be viewed as *selective depth lookahead*. Finally, the extension of the algorithm to lookahead of more than two steps is straightforward: we simply replace the 2-step-ahead neighbor set $N_2(i_m)$ with a suitably defined k-step ahead neighbor set $N_k(i_m)$.

Interpretation in Terms of DP

Let us now reinterpret the graph-based rollout algorithm within the context of deterministic DP. We will aim to view the base heuristic as a suboptimal

policy, and to view the rollout algorithm as a policy obtained by a process of policy improvement, provided the base heuristic is sequentially consistent.

To this end, we cast the graph search problem as a sequential decision problem, where each node corresponds to a state of a dynamic system. At each non-destination node/state i, a node j must be selected from the set of neighbors $N(i)$; then if j is a destination, the process terminates with cost $g(j)$, and otherwise the process is repeated with j becoming the new state. The DP algorithm calculates for every node i, the minimal cost that can be achieved starting from i, that is, the smallest value of $g(\bar{i})$ that can be obtained using paths that start from i and end at destination nodes $\bar{i}$. This value, denoted $J^*(i)$, is the optimal cost-to-go starting at node i. Once $J^*(i)$ is computed for all nodes i, an optimal path $(i_1, i_2, \ldots, i_m)$ can be constructed starting from any initial node/state i_1 by successively generating nodes using the relation

$$i_{k+1} = \arg \min_{j \in N(i_k)} J^*(j), \qquad k = 1, \ldots, m-1, \tag{6.46}$$

up to the point where a destination node i_m is encountered.†

A base heuristic $\mathcal{H}$ defines a policy π, i.e., an assignment of a successor node to any non-destination node. However, starting from a given node i, the cost of π need not be equal to $H(i)$ because if a path $(i_1, i_2, i_3, \ldots, i_m)$ is generated by $\mathcal{H}$ starting from node i_1, it is not necessarily true that the path $(i_2, i_3, \ldots, i_m)$ is generated by the base heuristic starting from i_2. Thus the successor node chosen at node i_2 by policy π may be different than the one used in the calculation of $H(i_1)$. On the other hand, if $\mathcal{H}$ is sequentially consistent, the cost of policy π starting from a node i is $H(i)$, since sequential consistency implies that the path that the base heuristic generates starting at the successor node is part of the path it generates at the predecessor node. Thus the cost improvement property of the rollout algorithm in the sequentially consistent case also follows from the cost improvement property shown earlier in the DP context.

Generally, we can view the rollout algorithm $\mathcal{RH}$ as a one-step lookahead policy that uses $H(j)$ as a cost-to-go approximation from state j. In some cases, $H(j)$ is the cost of some policy (in the DP sense), such as for example when $\mathcal{H}$ is sequentially consistent, as explained above. In general, however, this need not be so, in which case we can view $H(j)$ as a convenient cost-to-go approximation that is derived from the base heuristic. Still, the rollout algorithm $\mathcal{RH}$ may improve on the cost of the base heuristic (e.g., when $\mathcal{H}$ is sequentially improving, cf. Prop. 6.4.2) just as a general one-step lookahead policy may improve on the corresponding one-step lookahead cost approximation (cf. Prop. 6.3.1).

† We assume here that there are no termination/cycling difficulties of the type illustrated in the footnote following Example 6.4.4.

6.4.2 Q-Factors Evaluated by Simulation

We now consider a stochastic problem and some computational issues regarding the implementation of the rollout policy based on a given heuristic policy. A conceptually straightforward approach to compute the rollout control at a given state x_k and time k is to use Monte-Carlo simulation. To implement this algorithm, we consider all possible controls $u_k \in U_k(x_k)$ and we generate a "large" number of simulated trajectories of the system starting from x_k, using u_k as the first control, and using the policy π thereafter. Thus a simulated trajectory has the form

$$x_{i+1} = f_i\big(x_i, \mu_i(x_i), w_i\big), \qquad i = k+1, \ldots, N-1,$$

where the first generated state is

$$x_{k+1} = f_k(x_k, u_k, w_k),$$

and each of the disturbances $w_k, \ldots, w_{N-1}$ is an independent random sample from the given distribution. The costs corresponding to these trajectories are averaged to compute an approximation $\tilde{Q}_k(x_k, u_k)$ to the Q-factor

$$E\Big\{g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big)\Big\}.$$

Here, $\tilde{Q}_k(x_k, u_k)$ is an approximation to $Q_k(x_k, u_k)$ because of the simulation error resulting from the use of a limited number of trajectories. The approximation becomes increasingly accurate as the number of simulated trajectories increases. Once the approximate Q-factor $\tilde{Q}_k(x_k, u_k)$ corresponding to each control $u \in U_k(x_k)$ is computed, we can obtain the (approximate) rollout control $\overline{\mu}_k(x_k)$ by the minimization

$$\overline{\mu}_k(x_k) = \arg \min_{u_k \in U_k(x_k)} \tilde{Q}_k(x_k, u_k).$$

There is a serious flaw with this approach, due to the simulation error involved in the calculation of the Q-factors. In particular, for the calculation of $\overline{\mu}_k(x_k)$ to be accurate, the Q-factor differences

$$Q_k(x_k, u_k) - Q_k(x_k, \hat{u}_k)$$

must be computed accurately for all pairs of controls u_k and $\hat{u}_k$, so that these controls can be accurately compared. On the other hand, the simulation/approximation errors in the computation of the individual Q-factors $Q_k(x_k, u_k)$ may be magnified through the preceding differencing operation.

An alternative approach is to approximate by simulation the Q-factor difference $Q_k(x_k, u_k) - Q_k(x_k, \hat{u}_k)$ by sampling the difference

$$C_k(x_k, u_k, \mathbf{w}_k) - C_k(x_k, \hat{u}_k, \mathbf{w}_k),$$

where $\mathbf{w}_k = (w_k, w_{k+1}, \ldots, w_{N-1})$ and

$$C_k(x_k, u_k, \mathbf{w}_k) = g_N(x_N) + g_k(x_k, u_k, w_k) + \sum_{i=k+1}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big).$$

This approximation may be far more accurate than the one obtained by differencing independent samples of $C_k(x_k, u_k, \mathbf{w}_k)$ and $C_k(x_k, \hat{u}_k, \mathbf{w}_k)$. Indeed, by introducing the zero mean sample errors

$$D_k(x_k, u_k, \mathbf{w}_k) = C_k(x_k, u_k, \mathbf{w}_k) - Q_k(x_k, u_k),$$

it can be seen that the variance of the error in estimating $Q_k(x_k, u_k) - Q_k(x_k, \hat{u}_k)$ with the former method will be smaller than with the latter method if and only if

$$E_{\mathbf{w}_k, \hat{\mathbf{w}}_k}\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k) - D_k(x_k, \hat{u}_k, \hat{\mathbf{w}}_k)\big|^2\Big\} > E_{\mathbf{w_k}}\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k) - D_k(x_k, \hat{u}_k, \mathbf{w}_k)\big|^2\Big\},$$

or equivalently

$$E\big\{D_k(x_k, u_k, \mathbf{w}_k) D_k(x_k, \hat{u}_k, \mathbf{w}_k)\big\} > 0; \tag{6.47}$$

i.e., if and only if the correlation between the errors $D_k(x_k, u_k, \mathbf{w}_k)$ and $D_k(x_k, \hat{u}_k, \mathbf{w}_k)$ is positive. A little thought should convince the reader that this property is likely to hold in many types of problems. Roughly speaking, the relation (6.47) holds if changes in the value of u_k (at the first stage) have little effect on the value of the error $D_k(x_k, u_k, \mathbf{w}_k)$ relative to the effect induced by the randomness of $\mathbf{w}_k$. In particular, suppose that there exists a scalar $\gamma < 1$ such that, for all x_k, u_k, and $\hat{u}_k$, there holds

$$E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k) - D_k(x_k, \hat{u}_k, \mathbf{w}_k)\big|^2\Big\} \leq \gamma E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2\Big\}. \tag{6.48}$$

Then we have

$$\begin{aligned} &D_k(x_k, u_k, \mathbf{w}_k) D_k(x_k, \hat{u}_k, \mathbf{w}_k) \\ &\quad = \big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2 \\ &\qquad + D_k(x_k, u_k, \mathbf{w}_k)\big(D_k(x_k, \hat{u}_k, \mathbf{w}_k) - D_k(x_k, u_k, \mathbf{w}_k)\big) \\ &\quad \geq \big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2 \\ &\qquad - \big|D_k(x_k, u_k, \mathbf{w}_k)\big| \cdot \big|D_k(x_k, \hat{u}_k, \mathbf{w}_k) - D_k(x_k, u_k, \mathbf{w}_k)\big|, \end{aligned}$$

from which we obtain, using also Eq. (6.48),

$$\begin{aligned} &E\big\{D_k(x_k, u_k, \mathbf{w}_k) D_k(x_k, \hat{u}_k, \mathbf{w}_k)\big\} \\ &\qquad \geq E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2\Big\} \\ &\qquad\quad - E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k)\big| \cdot \big|D_k(x_k, \hat{u}_k, \mathbf{w}_k) - D_k(x_k, u_k, \mathbf{w}_k)\big|\Big\} \\ &\qquad \geq E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2\Big\} - \frac{1}{2} E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2\Big\} \\ &\qquad\quad - \frac{1}{2} E\Big\{\big|D_k(x_k, \hat{u}_k, \mathbf{w}_k) - D_k(x_k, u_k, \mathbf{w}_k)\big|^2\Big\} \\ &\qquad \geq \frac{1-\gamma}{2} E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2\Big\}. \end{aligned}$$

Thus, under the assumption (6.48) and the assumption

$$E\Big\{\big|D_k(x_k, u_k, \mathbf{w}_k)\big|^2\Big\} > 0,$$

the condition (6.47) holds and guarantees that by averaging cost difference samples rather than differencing (independently obtained) averages of cost samples, the simulation error variance decreases.

6.4.3 Q-Factor Approximation

Let us now consider the case of a stochastic problem and various possibilities for approximating the costs-to-go $H_k(x_k)$, $k = 1, \ldots, N-1$, of the base policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$, rather than calculating them by Monte-Carlo simulation. For example, in a certainty equivalence approach, given a state x_k at time k, we fix the remaining disturbances at some "typical" values $\overline{w}_{k+1}, \ldots, \overline{w}_{N-1}$, and we approximate the true Q-factor

$$Q_k(x_k, u_k) = E\Big\{g_k(x_k, u_k, w_k) + H_{k+1}\big(f_k(x_k, u_k, w_k)\big)\Big\}$$

with

$$\tilde{Q}_k(x_k, u_k) = E\Big\{g_k(x_k, u_k, w_k) + \tilde{H}_{k+1}\big(f_k(x_k, u_k, w_k)\big)\Big\}, \tag{6.49}$$

where $\tilde{H}_{k+1}\big(f_k(x_k, u_k, w_k)\big)$ is obtained by

$$\tilde{H}_{k+1}\big(f_k(x_k, u_k, w_k)\big) = g_N(\overline{x}_N) + \sum_{i=k+1}^{N-1} g_i(\overline{x}_i, \mu_i(x_i), \overline{w}_i),$$

the initial state is

$$\overline{x}_{k+1} = f_k(x_k, u_k, w_k),$$

and the intermediate states are given by

$$\overline{x}_{i+1} = f_i\big(\overline{x}_i, \mu_i(x_i), \overline{w}_i\big), \qquad i = k+1, \ldots, N-1.$$

Thus, in this approach, the rollout control is approximated by

$$\tilde{\mu}_k(x_k) = \arg \min_{u_k \in U_k(x_k)} \tilde{Q}_k(x_k, u_k).$$

Note that the approximate cost-to-go $\tilde{H}_{k+1}(x_{k+1})$ represents an approximation of the true cost-to-go $H_{k+1}(x_{k+1})$ of the base policy based on a single sample (the nominal disturbances $\overline{w}_{k+1}, \ldots, \overline{w}_{N-1}$). A potentially more accurate approximation is obtained using multiple nominal disturbance sequences and averaging the corresponding costs with appropriate nominal probabilities, similar to the scenario approximation approach of Example 6.3.6.

Let us also mention another approach for approximation of the cost-to-go H_{k+1} of the base policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$, using an approximation architecture. Here we calculate (possibly approximate) values for the cost-to-go of the base policy at a finite set of state-time pairs, and then we select the weights through a "least-squares fit" of these values.

In particular, suppose that we have calculated the correct value of the cost-to-go $H_{N-1}(x^i)$ at the next-to-last stage for s states x^i, $i = 1, \ldots, s$, through the DP formula

$$H_{N-1}(x_{N-1}) = E\Big\{g_{N-1}\big(x_{N-1}, \mu_{N-1}(x_{N-1}), w_{N-1}\big) + g_N\Big(f_{N-1}\big(x_{N-1}, \mu_{N-1}(x_{N-1}), w_{N-1}\big)\Big)\Big\},$$

and the given terminal cost function g_N. We can then approximate the entire function $H_{N-1}(x_{N-1})$ by a function of some given form

$$\tilde{H}_{N-1}(x_{N-1}, r_{N-1}),$$

where r_{N-1} is a vector of weights, which can be obtained by solving the problem

$$\min_r \sum_{i=1}^{s} \big|H_{N-1}(x^i) - \tilde{H}_{N-1}(x^i, r)\big|^2. \tag{6.50}$$

For example if $\tilde{H}_{N-1}$ is specified to be a linear function of m features $y_1(x), \ldots, y_m(x)$,

$$\tilde{H}_{N-1}(x, r) = \sum_{j=1}^{m} r_j y_j(x),$$

the least squares problem (6.50) is

$$\min_r \sum_{i=1}^{s} \left|H_{N-1}(x^i) - \sum_{j=1}^{m} r_j y_j(x^i)\right|^2.$$

This is a linear least squares problem that can be solved in closed form (its cost function is convex quadratic in the vector r).

Note that this approximation procedure can be enhanced if we have additional information on the true cost-to-go function $H_{N-1}(x_{N-1})$. For example, if we know that $H_{N-1}(x_{N-1}) \geq 0$ for all x_{N-1}, we may first compute the approximation $\tilde{H}_{N-1}(x_{N-1}, r_{N-1})$ by solving the least-squares problem (6.50) and then replace this approximation by

$$\max\big\{0, \tilde{H}_{N-1}(x_{N-1}, r_{N-1})\big\}.$$

Once an approximating function $\tilde{H}_{N-1}(x_{N-1}, r_{N-1})$ for the next-to-last stage has been obtained, it can be used to similarly obtain an approximating function $\tilde{H}_{N-2}(x_{N-2}, r_{N-2})$. In particular, (approximate) cost-to-go function values $\hat{H}_{N-2}(x^i)$ are obtained for s states x^i, $i = 1, \ldots, s$, through the (approximate) DP formula

$$\hat{H}_{N-2}(x_{N-2}) = E\Big\{g_{N-2}\big(x_{N-2}, \mu_{N-2}(x_{N-2}), w_{N-2}\big) + \tilde{H}_{N-1}\Big(f_{N-2}\big(x_{N-2}, \mu_{N-2}(x_{N-2}), w_{N-2}\big), r_{N-1}\Big)\Big\}.$$

These values are used to approximate the cost-to-go function $H_{N-2}(x_{N-2})$ by a function of some given form

$$\tilde{H}_{N-2}(x_{N-2}, r_{N-2}),$$

where r_{N-2} is a vector of parameters, which is obtained by solving the problem

$$\min_r \sum_{i=1}^{s} \big|\hat{H}_{N-2}(x^i) - \tilde{H}_{N-2}(x^i, r)\big|^2.$$

The process can be similarly continued to obtain $\tilde{H}_k(x_k, r_k)$ up to $k = 0$ by solving for each k the problem

$$\min_r \sum_{i=1}^{s} \big|\hat{H}_k(x^i) - \tilde{H}_k(x^i, r)\big|^2. \tag{6.51}$$

Given the approximations $\tilde{H}_0(x_0, r_0), \ldots, \tilde{H}_{N-1}(x_{N-1}, r_{N-1})$ to the cost-to-go of the base policy, one may obtain a suboptimal policy by using at state-time pair (x_k, k) the one-step lookahead control

$$\overline{\mu}_k(x_k) = \arg \min_{u_k \in U_k(x_k)} E\Big\{g_k(x_k, u_k, w_k) + \tilde{H}_{k+1}\Big(f_k(x_k, u_k, w_k), r_{k+1}\Big)\Big\}.$$

This control must be calculated on-line once the state x_k at time k becomes known.

6.5 MODEL PREDICTIVE CONTROL AND RELATED METHODS

In many control problems where the objective is to keep the state of a system near some desired point, the linear-quadratic models of Sections 4.1 and 5.3 are not satisfactory. There are two main reasons for this:

(a) The system may be nonlinear, and using for control purposes a model that is linearized around the desired point may be inappropriate.

(b) There may be control and/or state constraints, which are not handled adequately through a quadratic penalty on state and control. The reason may be special structure of the problem dictating that, for efficiency purposes, the system should often be operated at the boundary of its constraints. The solution obtained from a linear-quadratic model is not suitable for this, because the quadratic penalty on state and control tends to "blur" the boundaries of the constraints.

These inadequacies of the linear-quadratic model have motivated a form of suboptimal control, called *model predictive control* (MPC), which combines elements of several ideas that we have discussed so far: certainty equivalent control, multistage lookahead, and rollout algorithms. We will focus primarily on the most common form of MPC, where the system is either deterministic, or else it is stochastic, but it is replaced with a deterministic version by using typical values in place of all uncertain quantities, as in the certainty equivalent control approach. At each stage, a (deterministic) optimal control problem is solved over a fixed length horizon, starting from the current state. The first component of the corresponding optimal policy is then used as the control of the current stage, while the remaining components are discarded. The process is then repeated at the next stage, once the next state is revealed. We will also briefly discuss a version of MPC where there is uncertainty with a set-membership description.

The primary objective in MPC, aside from fulfilling the state and control constraints of the problem, is to obtain a stable closed-loop system. Note here that we may only be able to guarantee the stability of the deterministic model that forms the basis for the calculations of the MPC. This is consistent with a common practice in control theory: design a stable controller for a deterministic model of the system, and expect that it will provide some form of stability in a realistic stochastic environment as well.

In Section 6.5.2, we will discuss the mechanism by which stability is achieved by MPC under some reasonable conditions. We will first discuss in the next subsection some issues of multistage lookahead, which are relevant to MPC, but are also important in a broader context. In Section 6.5.3, we provide a general unifying framework for suboptimal control, which includes as special cases several approaches discussed in this chapter, OLFC, rollout, and MPC, and captures the mathematical essence of their attractive properties.

6.5.1 Rolling Horizon Approximations

Let us consider the l-step lookahead policy when the cost-to-go approximation is just zero. With this policy, at each stage we apply a control that would be optimal if the remaining horizon length were l and there were no terminal cost. Thus at the typical stage k we ignore the costs incurred in stages $k+l+1$ and beyond, and accordingly we neglect the corresponding long-range effects of our action. We call this the *rolling horizon* approach. In a variant of this approach, following the l steps of lookahead, we use a cost-to-go approximation that is equal to the terminal cost function g_N. This is essential if g_N is significant relative to the costs per stage accumulated over l stages.

We may also use a rolling horizon approach for infinite horizon problems. Then the length of the horizon of the problem solved at each stage stays the same at all stages. As a result, for a time-invariant system and cost per stage, the rolling horizon approach produces a stationary policy (the controls applied at the same state but in different stages are the same). This is a generic characteristic of infinite horizon control, as we have seen in the context of linear-quadratic problems (see also the discussion of Vol. II).

Naturally, a policy obtained using a rolling horizon is typically not optimal. One is tempted to conjecture that if the size of the lookahead l is increased, then the performance of the rolling horizon policy is improved. This, however, need not be true as the following example shows.

Example 6.5.1

This is an oversimplified problem, which, however, demonstrates the basic pitfall of the rolling horizon approach.

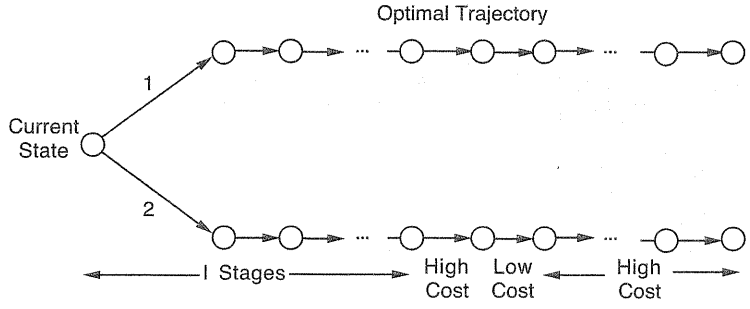


Figure 6.5.1 The problem of Example 6.5.1.

Consider a deterministic setting where at the initial state there are two possible controls, say 1 and 2 (see Fig. 6.5.1). At all other states there is only one available control, so a policy consists of just the initial choice between controls 1 and 2. Suppose that (based on the cost of the subsequent N stages) control 1 is optimal. Suppose also that if control 2 is chosen, an "unfavorable" (high cost) state results after l transitions, followed by a "particularly favorable" state, which is then followed by other "unfavorable" states. Then, in contrast with the l-step lookahead policy, the $(l+1)$-step lookahead policy may view the inferior control 2 as being better, because it may be "fooled" by the presence of the "particularly favorable" state $l+1$ transitions ahead.

The rolling horizon approach is also interesting in the context of a rollout algorithm, where we need to calculate the cost-to-go of the base policy at various states. It is possible to use a rolling horizon approximation in the calculation of this cost-to-go. Thus, from the given state, we calculate the cost of the base policy over a fixed number of stages, rather than over the entire remaining horizon. This can result in significant computational savings. Furthermore, there may also be an *improvement* in the performance of the rollout policy if a rolling horizon approximation is used. One reason is the phenomenon illustrated in the preceding example. In fact, because of the suboptimality of the base policy, this phenomenon can get exaggerated, as shown in the following example.

Example 6.5.2

Consider an N-stage stopping problem where at each stage we may either stop with a stopping cost equal to 0, or continue at a certain cost that is either $-\epsilon$ or 1, where $0 < \epsilon < 1/N$ (see Fig. 6.5.2). Let the first state with continuation cost equal to 1 be state m. Then the optimal policy is to stop after m steps at state m. The corresponding optimal cost is $-m\epsilon$. It can also be seen that an l-step rolling horizon approach with the cost evaluated optimally over the l steps (rather than suboptimally using a base heuristic) is optimal.

Consider now the rollout policy where the base heuristic is to continue at every state (except the last where stopping is mandatory). It can be seen that this policy will stop at the initial state at a cost of 0, since it will evaluate the continuation action as having positive cost, in view of the fact $1 - N\epsilon > 0$, and will thus prefer the stopping action. However, the rollout policy that uses a rolling horizon of l stages, with $l \leq m$, will continue up to the first $m - l + 1$ stages, thus compiling a cost of $-(m - l + 1)\epsilon$. Thus, as the length l of the rolling horizon becomes shorter, the performance of the rollout policy improves!

Another example of a rollout algorithm whose performance can be improved by using a rolling horizon approximation is the breakthrough problem of Example 6.4.2. In this case, the evolution of the system under

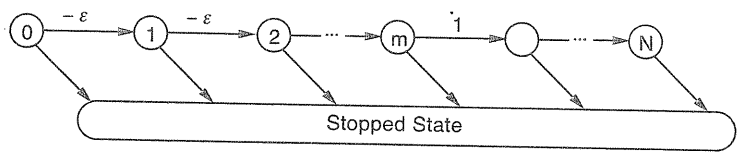


Figure 6.5.2 The problem of Example 6.5.2.

a rollout algorithm, where the greedy heuristic is evaluated using an l-step rolling horizon approximation, can be modeled using a Markov chain with $l+1$ states (see Exercise 6.18). Using this Markov chain, it is possible to ascertain that for a problem with a large number of steps N, the length of the rolling horizon that maximizes the breakthrough probability approaches an optimal value that is essentially independent of N.

6.5.2 Stability Issues in Model Predictive Control

As mentioned earlier, model predictive control (MPC) was initially motivated by the desire to introduce nonlinearities, and control and/or state constraints into the linear-quadratic framework, and obtain a suboptimal but stable closed-loop system. With this in mind, we will describe MPC for the case of a stationary, possibly nonlinear, deterministic system, where state and control belong to some Euclidean spaces. The system is

$$x_{k+1} = f(x_k, u_k), \qquad k = 0, 1, \ldots,$$

and the cost per stage is quadratic:

$$x_k' Q x_k + u_k' R u_k, \qquad k = 0, 1, \ldots,$$

where Q and R are positive definite symmetric matrices. We impose state and control constraints

$$x_k \in X, \qquad u_k \in U(x_k), \qquad k = 0, 1, \ldots,$$

and we assume that the set X contains the origin of the corresponding Euclidean space. Furthermore, if the system is at the origin, it can be kept there at no cost with control equal to 0, i.e., $0 \in U(0)$ and $f(0,0) = 0$. We want to derive a stationary feedback controller that applies control $\overline{\mu}(x)$ at state x, and is such that, for all initial states $x_0 \in X$, the state of the closed-loop system

$$x_{k+1} = f\big(x_k, \overline{\mu}(x_k)\big),$$

satisfies the state and control constraints, and the total cost over an infinite number of stages is finite:

$$\sum_{k=0}^{\infty} \big(x_k' Q x_k + \overline{\mu}(x_k)' R\, \overline{\mu}(x_k)\big) < \infty. \tag{6.52}$$

Note that because of the positive definiteness of Q and R, the feedback controller $\overline{\mu}$ is stable in the sense that $x_k \to 0$ and $\overline{\mu}(x_k) \to 0$ for all initial states $x_0 \in X$. (In the case of a linear system, the assumption of positive definiteness of Q may be relaxed to positive semidefiniteness, together with an observability assumption of the type introduced in Section 4.1 and Prop. 4.1.1.)

In order for such a controller to exist, it is evidently sufficient [in view of the assumption that $f(0,0) = 0$] that *there exists a positive integer m such that for every initial state $x_0 \in X$, one can find a sequence of controls u_k, $k = 0, 1, \ldots, m-1$, which drive to 0 the state x_m of the system at time m, while keeping all the preceding states $x_1, x_2, \ldots, x_{m-1}$ within X and satisfying the control constraints $u_0 \in U(x_0), \ldots, u_{m-1} \in U(x_{m-1})$*. We refer to this as the *constrained controllability assumption* (cf. the corresponding assumption of Section 4.1). In practical applications, this assumption can often be checked easily. Alternatively, the state and control constraints can be constructed in a way that the assumption is satisfied; the methodology of reachability of target tubes, discussed in Section 4.6.2, can be used for this purpose.

Let us now describe a form of MPC under the preceding assumption. At each stage k and state $x_k \in X$, it solves an m-stage deterministic optimal control problem involving the same quadratic cost *and the requirement that the state after m stages be exactly equal to 0*. This is the problem of minimizing

$$\sum_{i=k}^{k+m-1} (x_i' Q x_i + u_i' R u_i),$$

subject to the system equation constraints

$$x_{i+1} = f(x_i, u_i), \qquad i = k, k+1, \ldots, k+m-1,$$

the state and control constraints

$$x_i \in X, \qquad u_i \in U(x_i), \qquad i = k, k+1, \ldots, k+m-1,$$

and the terminal state constraint

$$x_{k+m} = 0.$$

By the constrained controllability assumption, this problem has a feasible solution. Let $\{\overline{u}_k, \overline{u}_{k+1}, \ldots, \overline{u}_{k+m-1}\}$ be a corresponding optimal control sequence. The MPC applies at stage k the first component of this sequence,

$$\overline{\mu}(x_k) = \overline{u}_k,$$

and discards the remaining components.

Example 6.5.3

Consider the scalar linear system

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

and the state and control constraints

$$x_k \in X = \{x \mid |x| \leq 1.5\}, \qquad u_k \in U(x_k) = \{u \mid |u| \leq 1\}.$$

Let also $Q = R = 1$. We select $m = 2$. For this value of m, the constrained controllability assumption is satisfied.

When at state $x_k \in X$, the MPC minimizes the two-stage cost

$$x_k^2 + u_k^2 + (x_k + u_k)^2 + u_{k+1}^2,$$

subject to the control constraints

$$|u_k| \leq 1, \qquad |u_{k+1}| \leq 1,$$

and the state constraints

$$x_{k+1} \in X, \qquad x_{k+2} = x_k + u_k + u_{k+1} = 0.$$

It is easily verified that this minimization yields $u_{k+1} = -(x_k + u_k)$ and $u_k = -(2/3)x_k$. Thus the MPC takes the form

$$\overline{\mu}(x_k) = -\frac{2}{3} x_k,$$

and the closed-loop system is

$$x_{k+1} = \frac{1}{3} x_k, \qquad k = 0, 1, \ldots$$

Note that while the closed-loop system is stable, its state is never driven to 0 if started from $x_0 \neq 0$.

We now show that the MPC satisfies the stability condition (6.52). Let $x_0, u_0, x_1, u_1, \ldots$ be the state and control sequence generated by MPC, so that

$$u_k = \overline{\mu}(x_k), \qquad x_{k+1} = f\big(x_k, \overline{\mu}(x_k)\big), \qquad k = 0, 1, \ldots$$

Denote $\hat{J}(x)$ the optimal cost of the m-stage problem solved by MPC when at a state $x \in X$. Let also $\tilde{J}(x)$ be the optimal cost starting at x of a corresponding $(m-1)$-stage problem, i.e., the optimal value of the quadratic cost

$$\sum_{k=0}^{m-2} (x_k' Q x_k + u_k' R u_k),$$

where $x_0 = x$, subject to the constraints

$$x_k \in \overline{X}, \qquad u_k \in \overline{U}(x_k), \qquad k = 0, 1, \ldots, m-2,$$

and

$$x_{m-1} = 0.$$

[For states $x \in X$ for which this problem does not have a feasible solution, we write $\tilde{J}(x) = \infty$.] Since having one less stage in our disposal to drive the state to 0 cannot decrease the optimal cost, we have for all $x \in X$

$$\hat{J}(x) \leq \tilde{J}(x). \tag{6.53}$$

From the definitions of $\hat{J}$ and $\tilde{J}$, we have for all k,

$$\min_{u \in \overline{U}(x)} \Big[x_k' Q x_k + u' R u + \tilde{J}\big(f(x_k, u)\big) \Big] = x_k' Q x_k + u_k' R u_k + \tilde{J}(x_{k+1}) = \hat{J}(x_k), \tag{6.54}$$

so using Eq. (6.53), we obtain

$$x_k' Q x_k + u_k' R u_k + \hat{J}(x_{k+1}) \leq \hat{J}(x_k), \qquad k = 0, 1, \ldots.$$

Adding this equation for all k in a range $[0, K]$, where $K = 0, 1, \ldots$, we obtain

$$\hat{J}(x_{K+1}) + \sum_{k=0}^{K} (x_k' Q x_k + u_k' R u_k) \leq \hat{J}(x_0).$$

Since $\hat{J}(x_{K+1}) \geq 0$, it follows that

$$\sum_{k=0}^{K} (x_k' Q x_k + u_k' R u_k) \leq \hat{J}(x_0), \qquad K = 0, 1, \ldots, \tag{6.55}$$

and taking the limit as $K \to \infty$,

$$\sum_{k=0}^{\infty} (x_k' Q x_k + u_k' R u_k) \leq \hat{J}(x_0) < \infty.$$

This shows the stability condition (6.52).

We note that *the one-step lookahead function $\tilde{J}$ implicitly used by MPC [cf. Eq. (6.54)] is the cost-to-go function of a certain policy*. This is the policy that drives to 0 the state after $m-1$ stages and keeps the state at 0 thereafter, while observing the state and control constraints $x_k \in X$ and $u_k \in \overline{U}(x_k)$, and minimizing the quadratic cost. Thus, we can also view MPC as a rollout algorithm with the policy just described viewed as the base heuristic. In fact the stability property of MPC is a special case of the cost improvement property of rollout algorithms, which in the

case of a quadratic cost, implies that *if the base heuristic results in a stable closed-loop system, the same is true for the corresponding rollout algorithm.*

Regarding the choice of the horizon length m for the MPC calculations, note that if the constrained controllability assumption is satisfied for some value of m, it is also satisfied for all larger values of m. Furthermore, it can be seen that the m-stage cost $\hat{J}(x)$, which by Eq. (6.55), is an upper bound to the cost of MPC, cannot increase with m. This argues for a larger value of m. On the other hand, the optimal control problem solved at each stage by the MPC becomes larger and hence more difficult as m increases. Thus, the horizon length is usually chosen on the basis of some experimentation: first use target tube reachability methods (cf. Section 4.6.2) to ensure that m is large enough for the constrained controllability assumption to hold with a target tube that is sufficiently large for the practical problem at hand, and then by further experimentation to ensure overall satisfactory performance.

The MPC scheme that we have described is just the starting point for a broad methodology with many variations, which often relate to the suboptimal control methods that we have discussed so far in this chapter. For example, in the problem solved by MPC at each stage, instead of the requirement of driving the system state to 0 in m steps, one may use a large penalty for the state being nonzero after m steps. Then, the preceding analysis goes through, as long as the terminal penalty is chosen so that Eq. (6.53) is satisfied. In another variant one may use a nonquadratic cost function, which is everywhere positive except at $(x, u) = (0, 0)$. In still another variant, instead of aiming to drive the state to 0 after m steps, one aims to reach a sufficiently small neighborhood of the origin, within which a stabilizing controller, designed by other methods, may be used. This variant is also well-suited for taking into account disturbances described by set membership, as we now proceed to explain.

MPC with Set-Membership Disturbances

To extend the MPC methodology to the case where there are disturbances w_k in the system equation

$$x_{k+1} = f(x_k, u_k, w_k),$$

we must first modify the stability objective. The reason is that in the presence of disturbances, the stability condition (6.52) is impossible to meet. A reasonable alternative is to introduce a set-membership constraint $w_k \in W(x_k, u_k)$ for the disturbance and a target set T for the state, and to require that the controller specified by MPC drives the state to T with finite quadratic cost.

To formulate the MPC, we assume that $T \subset X$, and that once the system state enters T, we will use some control law $\tilde{\mu}$ that keeps the state

within T for all possible values of the disturbances, i.e.,

$$f\big(x,\tilde{\mu}(x),w\big)\in T,\qquad \text{for all } x\in T, w\in W\big(x,\tilde{\mu}(x)\big). \tag{6.56}$$

The detailed methodology by which such a target set T and control law $\tilde{\mu}$ are obtained is outside our scope. We refer to the discussion of reachability of target tubes in Section 4.6.2 for orientation into this problem and references; see also Exercise 4.31, and Exercises 3.21 and 3.22 of Vol. II. We view T essentially as a cost-free and absorbing state, similar to our view of the origin in the earlier deterministic context. Consistent with this interpretation, we introduce the stage cost function

$$g(x,u)=\begin{cases} x'Qx+u'Ru & \text{if } x\notin T,\\ 0 & \text{if } x\in T.\end{cases}$$

The MPC is now defined as follows: At each stage k and state $x_k\in X$ with $x_k\notin T$, it solves the m-stage minimax control problem of finding a policy $\hat{\mu}_k,\hat{\mu}_{k+1},\ldots,\hat{\mu}_{k+m-1}$ that minimizes

$$\max_{\substack{w_i\in W(x_i,\hat{\mu}(x_i)),\\ i=k,k+1,\ldots,k+m-1}}\ \sum_{i=k}^{k+m-1} g\big(x_i,\mu(x_i)\big),$$

subject to the system equation constraints

$$x_{i+1}=f(x_i,u_i,w_i),\qquad i=k,k+1,\ldots,k+m-1,$$

the control and state constraints

$$x_i\in X,\qquad u_i\in U(x_i),\qquad i=k,k+1,\ldots,k+m-1,$$

and the terminal state constraint

$$x_i\in T,\qquad \text{for some } i\in[k+1,k+m].$$

These constraints must be satisfied for all disturbance sequences satisfying

$$w_i\in W(x_i,\hat{\mu}(x_i)),\qquad i=k,k+1,\ldots,k+m-1.$$

The MPC applies at stage k the first component of the policy $\hat{\mu}_k,\hat{\mu}_{k+1},\ldots,\hat{\mu}_{k+m-1}$ thus obtained,

$$\overline{\mu}(x_k)=\hat{\mu}_k(x_k),$$

and discards the remaining components. For states x within the target set T, the MPC applies the control $\tilde{\mu}(x)$ that keeps the state within T, as per Eq. (6.56), at no further cost [$\overline{\mu}(x)=\tilde{\mu}(x)$ for $x\in T$].

We make a constrained controllability assumption, namely that the problem solved at each stage by MPC has a feasible solution for all $x_k\in X$ with $x_k\notin T$ (this assumption can be checked using the target tube reachability methods of Section 4.6.2). Note that this problem is a potentially difficult minimax control problem, which generally must be solved by DP (cf. the algorithm of Section 1.6).

Example 6.5.4

This example is a version of the preceding one, modified to account for the presence of disturbances. We consider the scalar linear system

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

and the state and control constraints

$$x_k\in X=\big\{x\mid |x|\le 1.5\big\},\qquad u_k\in U(x_k)=\big\{u\mid |u|\le 1\big\},$$

and assume that the disturbances satisfy

$$w_k\in W(x_k,u_k)=\big\{w\mid |w|\le 0.2\big\}.$$

We select $m=2$, and it can be verified that for the target set

$$T=\big\{x\mid |x|\le 0.2\big\},$$

the constrained controllability assumption is satisfied, and the condition (6.56) is also satisfied using some control law $\tilde{\mu}$, namely $\tilde{\mu}(x)=-x$.

The associated 2-stage minimax control problem to be solved at each stage by MPC requires a DP solution. At the last stage, assuming $x\notin T$, the DP algorithm calculates

$$\tilde{J}(x)=\min_{\substack{|u|\le 1,\\ |x+u+\overline{w}|\le 0.2 \text{ for all } |\overline{w}|\le 0.2}}\Big[\max_{|w|\le 0.2}(x^2+u^2)\Big].$$

This is a straightforward minimization. It is feasible if and only if $|x|\le 1$, and it yields a minimizing policy for the last stage:

$$\hat{\mu}_1(x)=-x,\qquad \text{for all } x\notin T \text{ with } |x|\le 1,$$

and a cost-to-go

$$\tilde{J}(x)=2x^2,\qquad \text{for all } x\notin T \text{ with } |x|\le 1.$$

At the first stage, the DP algorithm calculates

$$\min_{\substack{|u|\le 1,\\ |x+u+\overline{w}|\le 1 \text{ for all } |\overline{w}|\le 0.2}}\Big[\max_{|w|\le 0.2}\big(x^2+u^2+\tilde{J}(x+u+w)\big)\Big],$$

or, since the maximum over w is attained for $w=0.2\,\mathrm{sgn}(x+u)$,

$$\min_{\substack{|u|\le 1,\\ |x+u+0.2\,sgn(x+u)|\le 1}}\Big[x^2+u^2+2\big(x+u+0.2\,\mathrm{sgn}(x+u)\big)^2\Big],$$

or

$$\min_{\substack{|u|\leq 1,\\ |x+u|\leq 0.8}} \Big[x^2+u^2+2\big(x^2+u^2+2xu+0.4|x+u|+0.04\big)\Big].$$

This minimization is again straightforward, and yields the MPC

$$\overline{\mu}(x)=\begin{cases} -\min\big[x,\ \frac{2}{3}(x+0.2)\big] & \text{if } x\in(0.2,1.5],\\ \min\big[-x,\ -\frac{2}{3}(x-0.2)\big] & \text{if } x\in[-1.5,-0.2).\end{cases}$$

This piecewise linear form of the MPC should be compared with the corresponding linear form, $\overline{\mu}(x)=-(2/3)x$, of Example 6.5.3, in which there are no disturbances.

The stability analysis of MPC (in the modified sense of reaching the target set T with finite quadratic cost, for all possible disturbance values) is similar to the one given earlier in the absence of disturbances. It is also possible to view MPC in the presence of disturbances as a special case of a rollout algorithm, suitably modified to take account of the set-membership description of the disturbances. The details of this analysis are sketched in Exercise 6.21.

6.5.3 Restricted Structure Policies

We will now introduce a general unifying suboptimal control scheme that contains as special cases several of the control schemes we have discussed: OLFC, POLFC, rollout, and MPC. The idea is to simplify the problem by selectively restricting the information and/or the controls available to the controller, thereby obtaining a restricted but more tractable problem structure, which can be used conveniently in a one-step lookahead context.

An example of such a structure is one where fewer observations are obtained, or one where the control constraint set is restricted to a single or a small number of given controls at each state. Generally, a restricted structure is associated with a problem where the optimal cost achievable is less favorable than in the given problem; this will be made specific in what follows. At each stage, we compute a policy that solves an optimal control problem involving the remaining stages and the restricted problem structure. The control applied at the given stage is the first component of the restricted policy thus obtained.

An example of a suboptimal control approach that uses a restricted structure is the OLFC, where one uses the information available at a given stage as the starting point for an open-loop computation (where future observations are ignored). Another example is the rollout algorithm, where at a given stage one restricts the controls available at future stages to be those applied by some heuristic policy. Still another example is MPC, which

under some conditions may be viewed as a form of rollout algorithm, as discussed in the preceding subsection.

For a problem with N stages, implementation of the suboptimal scheme to be discussed requires the solution of a problem involving the restricted structure at each stage. The horizon of this problem starts at the current stage, call it k, and extends up to the final stage N. This solution yields a control u_k for stage k and a policy for the remaining stages $k+1,\ldots,N-1$ (which must obey the constraints of the restricted structure). The control u_k is used at the current stage, while the policy for the remaining stages $k+1,\ldots,N-1$ is discarded. The process is repeated at the next stage $k+1$, using the additional information obtained between stages k and $k+1$; this is similar to CEC, OLFC, multistage lookahead, and MPC.

Similarly, for an infinite horizon model, implementation of the suboptimal scheme requires, at each stage k, the solution of a problem involving the restricted structure and a (rolling) horizon of fixed length. The solution yields a control u_k for stage k and a policy for each of the remaining stages. The control u_k is then used at stage k, and the policy for the remaining stages is discarded. For simplicity in what follows, we will focus attention to the finite horizon case, but the analysis applies, with minor modifications, to infinite horizon cases as well.

Our main result is that the performance of the suboptimal control scheme is no worse than the one of the restricted problem, i.e., the problem corresponding to the restricted structure. This result unifies and generalizes our analysis for open-loop-feedback control (which is known to improve the cost of the optimal open-loop policy, cf. Section 6.2), for the rollout algorithm (which is known to improve the cost of the corresponding heuristic policy, cf. Section 6.4), and for model predictive control (where under some reasonable assumptions, stability of the suboptimal closed-loop control scheme is guaranteed, cf. Section 6.5.2).

For simplicity, we focus on the imperfect state information framework for stationary finite-state Markov chains with N stages (cf. Section 5.4.2); the ideas apply to much more general problems with perfect and imperfect state information, as well problems with an infinite horizon. We assume that the system state is one of a finite number of states denoted $1,2,\ldots,n$. When a control u is applied, the system moves from state i to state j with probability $p_{ij}(u)$. The control u is chosen from a finite set U. Following a state transition, an observation is made by the controller. There is a finite number of possible observation outcomes, and the probability of each depends on the current state and the preceding control. The information available to the controller at stage k is the information vector

$$I_k=(z_1,\ldots,z_k,u_0,\ldots,u_{k-1}),$$

where for all i, z_i and u_i are the observation and control at stage i, respectively. Following the observation z_k, a control u_k is chosen by the controller,

and a cost $g(x_k, u_k)$ is incurred, where x_k is the current (hidden) state. The terminal cost for being at state x at the end of the N stages is denoted $G(x)$. We wish to minimize the expected value of the sum of costs incurred over the N stages.

As discussed in Section 5.4, we can reformulate the problem into a problem of perfect state information where the objective is to control the column vector of conditional probabilities

$$p_k = (p_k^1, \ldots, p_k^n)',$$

with

$$p_k^j = P(x_k = j \mid I_k), \qquad j = 1, \ldots, n.$$

We refer to p_k as the *belief state*, and we note that it evolves according to an equation of the form

$$p_{k+1} = \Phi(p_k, u_k, z_{k+1}).$$

The function Φ represents an estimator, as discussed in Section 5.4. The initial belief state p_0 is given.

The corresponding DP algorithm was given in Section 5.4, and has the form

$$J_k(p_k) = \min_{u_k \in U} \Big[p_k' g(u_k) + E_{z_{k+1}} \big\{ J_{k+1}\big(\Phi(p_k, u_k, z_{k+1})\big) \mid p_k, u_k \big\} \Big],$$

where $g(u_k)$ is the column vector with components $g(1, u_k), \ldots, g(n, u_k)$, and $p_k' g(u_k)$, the expected stage cost, is the inner product of the vectors p_k and $g(u_k)$. The algorithm starts at stage N, with

$$J_N(p_N) = p_N' G,$$

where G is the column vector with components $G(1), \ldots, G(n)$, and proceeds backwards.

We will also consider another control structure, where the information vector is

$$\overline{I}_k = (\overline{z}_1, \ldots, \overline{z}_k, u_0, \ldots, u_{k-1}), \qquad k = 0, \ldots, N-1,$$

with $\overline{z}_i$ being some observation for each i (possibly different from z_i), and the control constraint set at each p_k is a given set $\overline{U}(p_k)$. The probability distribution of $\overline{z}_k$ given x_k and u_{k-1} is known, and may be different than the one of z_k. Also $\overline{U}(p_k)$ may be different than U [in what follows, we will assume that $\overline{U}(p_k)$ is a subset of U].

We introduce a suboptimal policy, which at stage k, and starting with the current belief state p_k, applies a control $\overline{\mu}_k(p_k) \in U$, based on the assumption that the future observations and control constraints will be

according to the restricted structure. More specifically, this policy chooses the control at the typical stage k and state x_k as follows:

Restricted Structure Policy: At stage k and state x_k, apply the control

$$\overline{\mu}_k(p_k) = u_k,$$

where

$$\big(u_k, \hat{\mu}_{k+1}(\overline{z}_{k+1}, u_k), \ldots, \hat{\mu}_{N-1}(\overline{z}_{k+1}, \ldots, \overline{z}_{N-1}, u_k, \ldots, u_{N-2})\big)$$

is a policy that attains the optimal cost achievable from stage k onward with knowledge of p_k and with access to the future observations $\overline{z}_{k+1}, \ldots, \overline{z}_{N-1}$ (in addition to the future controls), and subject to the constraints

$$u_k \in U, \qquad \mu_{k+1}(p_{k+1}) \in \overline{U}(p_{k+1}), \ldots, \mu_{N-1}(p_{N-1}) \in \overline{U}(p_{N-1}).$$

Let $\overline{J}_k(p_k)$ be the cost-to-go, starting at belief state p_k at stage k, of the restricted structure policy $\{\overline{\mu}_0, \ldots, \overline{\mu}_{N-1}\}$ just described. This is given by the DP algorithm

$$\overline{J}_k(p_k) = p_k' g\big(\overline{\mu}_k(p_k)\big) + E_{z_{k+1}} \Big\{ \overline{J}_{k+1}\Big(\Phi\big(p_k, \overline{\mu}_k(p_k), z_{k+1}\big)\Big) \mid p_k, \overline{\mu}_k(p_k) \Big\} \tag{6.57}$$

for all p_k and k, with the terminal condition $\overline{J}_N(p_N) = p_N' G$ for all p_N.

Let us also denote by $J_k^r(p_k)$ the optimal cost-to-go of the restricted problem, i.e., the one where the observations and control constraints of the restricted structure are used exclusively. This is the optimal cost achievable, starting at belief state p_k at stage k, using the observations $\overline{z}_i$, $i = k+1, \ldots, N-1$, and subject to the constraints

$$u_k \in \overline{U}(p_k), \quad \mu_{k+1}(p_{k+1}) \in \overline{U}(p_{k+1}), \ldots, \mu_{N-1}(p_{N-1}) \in \overline{U}(p_{N-1}).$$

We will show, under certain assumptions to be introduced shortly, that

$$\overline{J}_k(p_k) \leq J_k^r(p_k), \qquad \forall\ p_k,\ k = 0, \ldots, N-1,$$

and we will also obtain a readily computable upper bound to $\overline{J}_k(p_k)$. To this end, for a given belief vector p_k and control $u_k \in U$, we consider three optimal costs-to-go corresponding to three different patterns of availability of information and control restriction over the remaining stages $k+1, \ldots, N-1$. We denote:

$Q_k(p_k, u_k)$: The cost achievable from stage k onward starting with p_k, applying u_k at stage k, and optimally choosing each future control u_i, $i = k+1, \ldots, N-1$, with knowledge of p_k, the observations $z_{k+1}, \ldots, z_i$ and the controls $u_k, \ldots, u_{i-1}$, and subject to the constraint $u_i \in U$.

$Q_k^c(p_k, u_k)$: The cost achievable from stage k onward starting with p_k, applying u_k at stage k, and optimally choosing each future control u_i, $i = k+1, \ldots, N-1$, with knowledge of p_k, the observations $\overline{z}_{k+1}, \ldots, \overline{z}_i$, and the controls $u_k, \ldots, u_{i-1}$, and subject to the constraint $u_i \in \overline{U}(p_i)$. Note that this definition is equivalent to

$$Q_k^c\big(p_k, \overline{\mu}_k(p_k)\big) = \min_{u_k \in U} Q_k^c(p_k, u_k), \tag{6.58}$$

where $\overline{\mu}_k(p_k)$ is the control applied by the restricted structure policy just described.

$\hat{Q}_k^c(p_k, u_k)$: The cost achievable from stage k onward starting with p_k, applying u_k at stage k, optimally choosing the control u_{k+1} with knowledge of p_k, the observation z_{k+1}, and the control u_k, subject to the constraint $u_{k+1} \in U$, and optimally choosing each of the remaining controls u_i, $i = k+2, \ldots, N-1$, with knowledge of p_k, the observations $z_{k+1}, \overline{z}_{k+2}, \ldots, \overline{z}_i$, and the controls $u_k, \ldots, u_{i-1}$, and subject to the constraints $u_i \in \overline{U}(p_i)$.

Thus, the difference between $Q_k^c(p_k, u_k)$ and $Q_k(p_k, u_k)$ is due to the difference in the control constraint and the information available to the controller at *all* future stages $k+1, \ldots, N-1$ [$\overline{U}(p_{k+1}), \ldots, \overline{U}(p_{N-1})$ versus U, and $\overline{z}_{k+1}, \ldots, \overline{z}_{N-1}$ versus $z_{k+1}, \ldots, z_{N-1}$, respectively]. The difference between $Q_k^c(p_k, u_k)$ and $\hat{Q}_k^c(p_k, u_k)$ is due to the difference in the control constraint and the information available to the controller at the *single* stage $k+1$ [$\overline{U}(p_{k+1})$ versus U, and $\overline{z}_{k+1}$ versus z_{k+1}, respectively]. Our key assumptions are that

$$\overline{U}(p_k) \subset U, \qquad \forall\, p_k,\ k = 0, \ldots, N-1, \tag{6.59}$$

$$Q_k(p_k, u_k) \leq \hat{Q}_k^c(p_k, u_k) \leq Q_k^c(p_k, u_k), \qquad \forall\, p_k,\ u_k \in U,\ k = 0, \ldots, N-1. \tag{6.60}$$

Roughly, this means that the control constraint $\overline{U}(p_k)$ is more stringent than U, and the observations $\overline{z}_{k+1}, \ldots, \overline{z}_{N-1}$ are "weaker" (no more valuable in terms of improving the cost) than the observations $z_{k+1}, \ldots, z_{N-1}$. Consequently, if Eqs. (6.59) and (6.60) hold, we may interpret a controller that uses in part the observations $\overline{z}_k$ and the control constraints $\overline{U}(p_k)$, in place of z_k and U, respectively, as "handicapped" or "restricted."

Let us denote:

$J_k(p_k)$: The optimal cost-to-go of the original problem, starting at belief state p_k at stage k. This is given by

$$J_k(p_k) = \min_{u_k \in U} Q_k(p_k, u_k). \tag{6.61}$$

$J_k^c(p_k)$: The optimal cost achievable, starting at belief state p_k at stage k, using the observations $\overline{z}_i$, $i = k+1, \ldots, N-1$, and subject to the constraints

$$u_k \in U, \quad \mu_{k+1}(p_{k+1}) \in \overline{U}(p_{k+1}), \ldots, \mu_{N-1}(p_{N-1}) \in \overline{U}(p_{N-1}).$$

This is given by

$$J_k^c(p_k) = \min_{u_k \in U} Q_k^c(p_k, u_k), \tag{6.62}$$

and it is the cost that is computed when solving the optimization problem of stage k in the restricted structure policy scheme. Note that we have for all p_k,

$$J_k^r(p_k) = \min_{u_k \in \overline{U}(p_k)} Q_k^c(p_k, u_k) \geq \min_{u_k \in U} Q_k^c(p_k, u_k) = J_k^c(p_k), \tag{6.63}$$

where the inequality holds in view of the assumption $\overline{U}(p_k) \subset U$.

Our main result is the following:

Proposition 6.5.1: Under the assumptions (6.59) and (6.60), there holds

$$J_k(p_k) \leq \overline{J}_k(p_k) \leq J_k^c(p_k) \leq J_k^r(p_k), \qquad \forall\, p_k,\ k = 0, \ldots, N-1.$$

Proof: The inequality $J_k(p_k) \leq \overline{J}_k(p_k)$ is evident, since $J_k(p_k)$ is the optimal cost-to-go over a class of policies that includes the restricted structure policy $\{\overline{\mu}_0, \ldots, \overline{\mu}_{N-1}\}$. Also the inequality $J_k^c(p_k) \leq J_k^r(p_k)$ follows from the definitions; see Eq. (6.63). We prove the remaining inequality $\overline{J}_k(p_k) \leq J_k^c(p_k)$ by induction on k.

We have $\overline{J}_N(p_N) = J_N^c(p_N) = 0$ for all p_N. Assume that for all p_{k+1}, we have

$$\overline{J}_{k+1}(p_{k+1}) \leq J_{k+1}^c(p_{k+1}).$$

Then, for all p_k,

$$\begin{aligned}
\overline{J}_k(p_k) &= p_k' g\big(\overline{\mu}_k(p_k)\big) + E_{z_{k+1}}\Big\{\overline{J}_{k+1}\big(\Phi\big(p_k, \overline{\mu}_k(p_k), z_{k+1}\big)\big) \mid p_k, \overline{\mu}_k(p_k)\Big\} \\
&\leq p_k' g\big(\overline{\mu}_k(p_k)\big) + E_{z_{k+1}}\Big\{J_{k+1}^c\big(\Phi\big(p_k, \overline{\mu}_k(p_k), z_{k+1}\big)\big) \mid p_k, \overline{\mu}_k(p_k)\Big\} \\
&= p_k' g\big(\overline{\mu}_k(p_k)\big) \\
&\quad + E_{z_{k+1}}\Big\{\min_{u_{k+1} \in U} Q_{k+1}^c\big(\Phi\big(p_k, \overline{\mu}_k(p_k), z_{k+1}\big), u_{k+1}\big) \Big| p_k, \overline{\mu}_k(p_k)\Big\} \\
&= \hat{Q}_k^c\big(p_k, \overline{\mu}_k(p_k)\big) \\
&\leq Q_k^c\big(p_k, \overline{\mu}_k(p_k)\big) \\
&= J_k^c(p_k),
\end{aligned}$$

where the first equality holds by Eq. (6.57), the first inequality holds by the induction hypothesis, the second equality holds by Eq. (6.62), the third equality holds by the definition of $\hat{Q}_k^c$, the second inequality holds by the assumption (6.60), and the last equality holds from the definition (6.58) of the restricted structure policy. The induction is complete. **Q.E.D.**

The main conclusion from the proposition is that the performance of the restricted structure policy $\{\overline{\mu}_0, \ldots, \overline{\mu}_{N-1}\}$ is no worse than the performance associated with the restricted control structure. Furthermore, at each stage k, the value $J_k^c(p_k)$, which is obtained as a byproduct of the online computation of the control $\overline{\mu}_k(p_k)$, is an upper bound to the cost-to-go $\overline{J}_k(p_k)$ of the suboptimal policy. This is consistent with Prop. 6.2.1, which shows the cost improvement property of the OLFC, and Prop. 6.3.1, which is the basis for the cost improvement property of the rollout algorithm and the stability property of MPC.

6.6 ADDITIONAL TOPICS IN APPROXIMATE DP

We close this chapter with a brief discussion of a few additional topics relating to approximate DP. We first address some of the discretization issues that arise when continuous state and control spaces are approximated by discrete spaces for DP computation purposes. We then describe some alternative suboptimal control approaches.

6.6.1 Discretization

An important practical issue is how to deal computationally with problems involving nondiscrete state and control spaces. In particular, problems with continuous state, control, or disturbance spaces must be discretized in order to execute the DP algorithm. Here each of the continuous spaces of the problem is replaced by a space with a finite number of elements, and the system equation is appropriately modified. Thus the resulting approximating problem involves a finite number of states, and a set of transition probabilities between these states. Once the discretization is done, the DP algorithm is executed to yield the optimal cost-to-go function and an optimal policy for the discrete approximating problem. The optimal cost function and/or the optimal policy for the discrete problem may then be extended to an approximate cost function or a suboptimal policy for the original continuous problem through some form of interpolation. We have already seen an example of such a process in the context of aggregation (cf. Example 6.3.13).

A prerequisite for success of this type of discretization is *consistency*. By this we mean that the optimal cost of the original problem should be achieved in the limit as the discretization becomes finer and finer. Consistency is typically guaranteed if there is a "sufficient amount of continuity" in the problem; for example, if the cost-to-go functions and the optimal policy of the original problem are continuous functions of the state. This in turn can be guaranteed through appropriate continuity assumptions on the original problem data (see the references given in Section 6.7).

Continuity of the cost-to-go functions may be sufficient to guarantee consistency, even if the optimal policy is discontinuous in the state. What may happen here is that for some states there may be a large discrepancy between the optimal policy of the continuous problem and the optimal policy of its discretized version, but this discrepancy may occur over a portion of the state space that diminishes as the discretization becomes finer. As an example consider the inventory control problem of Section 4.2 with nonzero fixed cost. We obtained an optimal policy of the (s, S) type

$$\mu_k^*(x_k) = \begin{cases} S_k - x_k & \text{if } x_k < s_k, \\ 0 & \text{if } x_k \geq s_k, \end{cases}$$

which is discontinuous at the lower threshold s_k. The optimal policy obtained from the discretized problem may not approximate well the optimal around the point of discontinuity s_k, but it is intuitively clear that the discrepancy has a diminishing effect on the optimal cost as the discretization becomes finer.

If the original problem is defined in continuous time, then the time must also be discretized, to obtain a discrete-time approximating problem. The issue of consistency becomes now considerably more complex, because the time discretization affects not only the system equation but also the control constraint set. In particular, the control constraint set may change considerably as we pass to the appropriate discrete-time approximation. As an example, consider the two-dimensional system

$$\dot{x}_1(t) = u_1(t), \qquad \dot{x}_2(t) = u_2(t),$$

with the control constraint

$$u_1(t) \in \{-1, 1\}, \qquad u_2(t) \in \{-1, 1\}.$$

It can be seen then that the state at time $t + \Delta t$ can be anywhere within the square centered at $x(t)$ with side of length $2\Delta t$ (note that the effect of any control in the interval $[-1, 1]$ can be obtained in the continuous-time system by "chattering" between the +1 and -1 controls). Thus, given Δt, the appropriate discrete-time approximation of the control constraint set should involve a discretized version of the entire unit square, the *convex hull* of the control constraint set of the continuous-time problem. An example that illustrates some of the pitfalls associated with the discretization process is given in Exercise 6.10.

A general method to address the discretization issues of continuous-time/space optimal control is the aggregation/discretization approach described in Example 6.3.13. The idea is to discretize, in addition to time, the state space using some finite grid, and then to approximate the cost-to-go of nongrid states by linear interpolation of the cost-to-go values of the nearby grid states. Thus, the grid states $x^1, \ldots, x^M$ are suitably selected within the state space, and each nongrid state x is expressed as

$$x = \sum_{m=1}^{M} w^m(x) x^m,$$

for some nonnegative weights $w^m(x)$, which add to 1. When this is worked out (cf. Example 6.3.13), one ends up with a *stochastic* optimal control problem having as states the finite number of grid states, and transition probabilities that are determined from the weights $w^m(x)$ above. If the original continuous-time optimal control problem has fixed terminal time, the resulting stochastic control approximation has finite horizon. If the terminal time of the original problem is free and subject to optimization, the stochastic control approximation is of the stochastic shortest path type to be discussed in Section 7.2. Finally, once the costs-to-go $\hat{J}_k(x^m)$ of the grid states in the stochastic approximating problem are computed, the cost-to-go of each nongrid state x at stage k is approximated by

$$\tilde{J}_k(x) = \sum_{m=1}^{M} w^m(x) \hat{J}_k(x^m).$$

We refer to the papers by Gonzalez and Rofman [GoR85], and by Falcone [Fal87] for an account of this approach, and to the survey paper by Kushner [Kus90], and the monograph by Kushner and Dupuis [KuD92] for a detailed analysis of the associated consistency issues.

An important special case is the continuous-space shortest path problem, described in Exercise 6.10. For the corresponding stochastic shortest path problem, a finitely terminating adaptation of the Dijkstra shortest path algorithm has been developed by Tsitsiklis [Tsi95]; see Exercises 2.10 and 2.11 in Chapter 2 of Vol. II. Other related works are the papers by Bertsekas, Guerriero, and Musmanno [BGM95], and Polymenakos, Bertsekas, and Tsitsiklis [PBT98], which develop continuous space versions of label correcting algorithms, such as the Small-Label-First algorithm discussed in Section 2.3.1.

6.6.2 Other Approximation Approaches

We mention briefly three additional approaches for using approximations. In the first approach, the optimal cost-to-go functions $J_k(x_k)$ are approximated with functions $\tilde{J}_k(x_k, r_k)$, where $r_0, r_1, \ldots, r_{N-1}$ are unknown parameter vectors, which are chosen to minimize some form of error in the

DP equations; for example by solving the problem

$$\min_{r_0,\ldots,r_{N-1}} \sum_{(x_k,k)\in\tilde{S}} \left| \tilde{J}_k(x_k, r_k) - \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k), r_{k+1}\big) \Big\} \right|^2 \tag{6.64}$$

where $\tilde{S}$ is a suitably chosen subset of "representative" state-time pairs. The above minimization can be attempted using some type of gradient method. Note that there is some difficulty in doing so because the cost function of Eq. (6.64) may be nondifferentiable for some values of r. However, there are adaptations of gradient methods that work with nondifferentiable cost functions, and for which we refer to the specialized literature. One possibility is to replace the nondifferentiable term

$$\min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k), r_{k+1}\big) \Big\}$$

by a smooth approximation (see Bertsekas [Ber82b], Ch. 3). The approach of approximating cost-to-go functions by minimizing the error in the DP equations will also be discussed in more detail within an infinite horizon context (see Vol. II, Section 2.3).

In the second approach, optimal policies are directly approximated. In particular, suppose that the control space is a Euclidean space, and that we obtain, for a finite number of states x^i, $i = 1, \ldots, m$, the minimizing controls

$$\hat{\mu}_k(x^i) = \arg \min_{u_k \in U_k(x^i)} \mathop{E}_{w_k} \Big\{ g_k(x^i, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x^i, u_k, w_k), r_{k+1}\big) \Big\}.$$

We can then approximate the optimal policy $\mu_k(x_k)$ by a function of some given form

$$\tilde{\mu}_k(x_k, s_k),$$

where s_k is a vector of parameters obtained by solving the problem

$$\min_{s_k} \sum_{i=1}^{m} \big\| \hat{\mu}_k(x^i) - \tilde{\mu}_k(x^i, s_k) \big\|^2. \tag{6.65}$$

In the case of deterministic optimal control problems, we can take advantage of the equivalence between open-loop and feedback control to carry out the approximation process more efficiently. In particular, for such problems we may select a representative finite subset of initial states, and generate an optimal open-loop trajectory starting from each of these states. (Gradient-based methods can often be used for this purpose.) Each

of these trajectories yields a sequence of pairs $\big(x_k, J_k(x_k)\big)$ and a sequence of pairs $\big(x_k, \mu_k(x_k)\big)$, which can be used in the least-squares approximation procedures discussed above. In particular, we can use the exact values $J_k(x^i)$ and $\mu_k(x^i)$ obtained from the optimal open-loop trajectories in place of $\hat{J}_k(x^i)$ and $\hat{\mu}_k(x^i)$, respectively, in the least-squares problems of Eqs. (6.51) and (6.65).

In the third approach, sometimes called *optimization in policy space*, we parameterize the set of policies by a vector $s = (s_0, s_1, \ldots, s_{N-1})$ and we optimize the corresponding cost over this vector. In particular, we consider policies of the form

$$\pi(s) = \big\{\tilde{\mu}_0(x_0, s_0), \ldots, \tilde{\mu}_{N-1}(x_{N-1}, s_{N-1})\big\},$$

where the $\tilde{\mu}_k(\cdot,\cdot)$ are functions of a given form. We then minimize over s the expected cost

$$E\big\{J_{\pi(s)}(x_0)\big\},$$

where $J_{\pi(s)}(x_0)$ is the cost of the policy $\pi(s)$ starting from the initial state x_0, and the expected value is taken with respect to a suitable probability distribution of x_0. One of the difficulties associated with this approach is that the optimization of $E\big\{J_{\pi(s)}(x_0)\big\}$ over s may be time-consuming, because it may require some brute force search, local search, or random search method. Sometimes, it is possible to use a gradient-based approach for optimizing $E\big\{J_{\pi(s)}(x_0)\big\}$ over s, but this can be time-consuming as well.

In an important special case of this approach, the parameterization of the policies is indirect through a parameterization of an approximate cost-to-go function. In particular, for a given parameter vector $s = (s_0, \ldots, s_{N-1})$, we define

$$\tilde{\mu}_k(x_k, s_k) = \arg \min_{u_k \in U_k(x_k)} E\Big\{g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k), s_k\big)\Big\},$$

where $\tilde{J}_{k+1}(\cdot,\cdot)$ is a function of a given form. For example, $\tilde{J}_{k+1}$ may represent a linear feature-based architecture, where s_k is a vector of adjustable scalar weights multiplying corresponding features of states x_{k+1} (cf. Section 6.3.5). Note that the policies

$$\pi(s) = \big\{\tilde{\mu}_0(x_0, s_0), \ldots, \tilde{\mu}_{N-1}(x_{N-1}, s_{N-1})\big\}$$

form a class of one-step lookahead policies parametrized by s. By optimizing over s the corresponding expected cost $E\big\{J_{\pi(s)}(x_0)\big\}$, we end up with a one-step lookahead policy that is optimal within this class.

6.7 NOTES, SOURCES, AND EXERCISES

Many schemes for suboptimal control have been discussed in this chapter, and it may be helpful to summarize them here. Most of these schemes are

based on one-step lookahead, whereby we apply at stage k and state x_k the control $\overline{\mu}_k(x_k)$ that minimizes over $u_k \in U_k(x_k)$

$$E\Big\{g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big)\Big\},$$

where $\tilde{J}_{k+1}$ is a suitable cost-to-go approximating function; in some cases, the control constraint set and/or the expected cost per stage are also approximated. The principal distinction between alternative approaches is the method for calculating $\tilde{J}_{k+1}$. There are several possibilities (and variations thereoff), the principal of which are:

(a) *Explicit cost-to-go approximation.* Here $\tilde{J}_{k+1}$ is computed *off-line* in one of a number of ways.

  (1) By solving a related problem, obtained for example by *aggregation* or *enforced decomposition*, and by deriving $\tilde{J}_{k+1}$ from the optimal cost-to-go of that problem.

  (2) By introducing a parametric approximation architecture, possibly using features. The parameters of the architecture are tuned by some form of heuristic or systematic method.

(b) *Implicit cost-to-go approximation.* Here the values of $\tilde{J}_{k+1}$ at the states $f_k(x_k, u_k, w_k)$ are computed *on-line* as needed, by using an open-loop computation (optimal or suboptimal/heuristic, with or without a rolling horizon). We focused on a few possibilities, all which were interpreted under the unifying framework of restricted structure policies in Section 6.5.3:

  (1) *Open-loop-feedback control*, where an optimal open-loop computation is used, starting from the state x_k (in the case of perfect state information) or the conditional probability distribution of the state (in the case of imperfect state information).

  (2) *Rollout*, where the cost-to-go of a suboptimal/heuristic policy is used as $\tilde{J}_{k+1}$. This cost is computed as necessary by on-line simulation (which in some variants may be approximate and/or use a rolling horizon).

  (3) *Model predictive control*, where an optimal control computation is used in conjunction with a rolling horizon. This computation is deterministic, possibly based on a simplification of the original problem via certainty equivalence, but there is also a minimax variant that implicitly involves reachability of target tube computations.

A few important variations of the preceding schemes should be mentioned. The first is the use of *multistep lookahead*, which aims to improve the performance of one-step lookahead, at the expense of increased on-line

computation. The second is the use of *certainty equivalence*, which simplifies the off-line and on-line computations by replacing the current and future unknown disturbances $w_k, \ldots, w_{N-1}$ with nominal values. A third variation, which applies to problems of imperfect state information, is to use one of the preceding schemes with the unknown state x_k replaced by some estimate.

While the idea of one-step lookahead is old, it has gained a lot of credibility recently, thanks to extensive research on approximate dynamic programming and wide acceptance of model predictive control in practical applications. With experience and research, the relative merits of different approaches have been clarified to some extent, and it is now understood that some schemes possess desirable theoretical performance guarantees, while others do not. In particular, in this chapter, we have discussed qualitative and/or quantitative performance guarantees for open-loop feedback control (cf. Prop. 6.2.1 and the discussion of Section 6.5.3), rollout (Examples 6.3.1 and 6.3.2), and model predictive control (the stability guarantee discussed in Section 6.5.2). The performance guarantee for certainty equivalent control (cf. Prop. 6.3.2 and Example 6.3.3) is weaker, and indeed for some stochastic problems, certainty equivalent control may be outperformed by open-loop control (see Exercise 6.2). For additional theoretical analysis on performance bounds, see Witsenhausen [Wit69], [Wit70]. Despite the recent progress in theory and practical experience, the methodology for performance analysis of suboptimal control schemes is not very satisfactory at present, and the validation of a suboptimal policy by simulation is often essential in practice. This is true of all approaches described in this chapter, including ones that are not based on one-step lookahead, such as approximation in policy space (cf. Section 6.6.2).

Excellent surveys of adaptive control, which contain many other references, are given by Aström [Ast83] and Kumar [Kum85]. Self-tuning regulators received wide attention following the paper by Aström and Wittenmark [AsW73]. For textbook treatments of adaptive control, see Aström and Wittenmark [AsW94], Goodwin and Sin [GoS84], Hernandez-Lerma [Her89], Ioannou and Sun [IoS96], Krstic, Kanellakopoulos, and Kokotovic [KKK95], Kumar and Varaiya [KuV86], Sastry, Bodson, and Bartram [SBB89], and Slotine and Li [SlL91].

Open-loop feedback control was suggested by Dreyfus [Dre65]. Its superiority over open-loop control (cf. Prop. 6.2.1) was established by the author in the context of minimax control [Ber72b]. A generalization of this result is given by White and Harrington [WhH80]. The POLFC was proposed in Bertsekas [Ber76].

Stochastic programming problems have been discussed in detail in the literature (see the texts by Birge and Louveaux [BiL97], Kall and Wallace [KaW94], and Prekopa [Pre95]). The connections between stochastic programming and stochastic optimal control have been highlighted by Varaiya and Wets [VaW89].

There is a long history of limited lookahead approximations in specific application contexts. The performance bounds for limited lookahead policies, given in Section 6.3.1 and Exercises 6.11-6.15 are new.

The main idea of rollout algorithms, obtaining an improved policy starting from some other suboptimal policy using a one-time policy improvement, has appeared in several DP application contexts. In the context of game-playing computer programs, it has been proposed by Abramson [Abr90] and by Tesauro [TeG96]. The name "rollout" was coined by Tesauro in specific reference to rolling the dice in the game of backgammon. In Tesauro's proposal, a given backgammon position is evaluated by "rolling out" many games starting from that position, using a simulator, and the results are averaged to provide a "score" for the position. The internet contains a lot of material on computer backgammon and the use of rollout, in some cases in conjunction with multistep lookahead and cost-to-go approximation.

The application of rollout algorithms to discrete optimization problems has its origin in the neuro-dynamic programming work of the author and J. Tsitsiklis [BeT96], and has been further formalized by Bertsekas, Tsitsiklis, and Wu [BTW97], Bertsekas [Ber97], and Bertsekas and Castanon [BeC99]. The analysis of the breakthrough problem (Example 6.4.2) is based on unpublished joint work of the author with D. Castanon and J. Tsitsiklis. An analysis of the optimal policy and some suboptimal policies for this problem is given by Pearl [Pea84]. A discussion of rollout algorithms as applied to network optimization problems may be found in the author's network optimization book [Ber98a]. The technique for variance reduction in the calculation of Q-factor differences (Section 6.4.2) is from Bertsekas [Ber97].

For work on rollout algorithms, see Christodouleas [Chr97], Secomandi [Sec00], [Sec01], [Sec03], Bertsimas and Demir [BeD02], Ferris and Voelker [FeV02], [FeV04], McGovern, Moss, and Barto [MMB02], Savagaonkar, Givan, and Chong [SGC02], Bertsimas and Popescu [BeP03], Guerriero and Mancini [GuM03], Tu and Pattipati [TuP03], Wu, Chong, and Givan [WCG03], Chang, Givan, and Chong [CGC04], Meloni, Pacciarelli, and Pranzo [MPP04], and Yan, Diaconis, Rusmevichientong, and Van Roy [YDR05]. These works discuss a broad variety of applications and case studies, and generally report positive computational experience.

The model predictive control approach has become popular in a variety of control system design contexts, and particularly in chemical process control, where meeting explicit control and state constraints is an important practical issue. Over time, there has been increasing awareness of the connection with the problem of reachability of target tubes, set-membership descriptions of uncertainty, and minimax control (see the discussion of Section 4.6). The stability analysis given here is based on the work of Keerthi and Gilbert [KeG88]. For extensive surveys of the field, see Morari and Lee [MoL99], and Mayne et. al. [MRR00], who give many references. For

related textbooks, see Camacho and Bordons [CaB04], and Maciejowski [Mac02]. The connection with rollout algorithms and one-time policy iteration reported in Section 6.5.2 is new. The material of Section 6.5.3 on the unifying suboptimal control framework based on restricted structure policies is also new.

The computational requirements for solving stochastic optimal control problems are discussed from the point of view of computational complexity in the survey by Blondel and Tsitsiklis [BlT00], who give several additional references; see also Rust [Rus97]. For consistency analyses of various discretization and approximation procedures for discrete-time stochastic optimal control problems, see Bertsekas [Ber75], [Ber76a], Chow and Tsitsiklis [ChT89], [ChT91], Fox [Fox71], and Whitt [Whi78], [Whi79]. A discretization method that takes advantage of the special structure of finite-state imperfect state information problems was first given by Lovejoy [Lov91a]; see also the survey [Lov91b]. For more recent work, based on the aggregation/discretization approach described in Example 6.3.13, see Yu and Bertsekas [YuB04]. The discretization issues of continuous-time/space optimal control problems have been the subject of considerable research; see Gonzalez and Rofman [GoR85], Falcone [Fal87], Kushner [Kus90], and Kushner and Dupuis [KuD92], which give additional sources.

There have been important algorithmic developments for certain types of continuous space shortest path problems. A finitely terminating adaptation of the label setting (Dijkstra) method has been developed by Tsitsiklis [Tsi95]. This method was rediscovered later, under the name "fast marching method," by Sethian [Set99a], [Set99b], who discusses several other related methods and many applications, as well as by Helmsen et al. [HPC96]. Efficient analogs of label correcting algorithms for continuous space shortest path problems were developed by Bertsekas, Guerriero, and Musmanno [BGM95], and Polymenakos, Bertsekas, and Tsitsiklis [PBT98].

EXERCISES

6.1

Consider a problem with perfect state information involving the n-dimensional linear system of Section 4.1:

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

and a cost function of the form

$$\underset{\substack{w_k \\ k=0,1,\ldots,N-1}}{E} \left\{ g_N(c' x_N) + \sum_{k=0}^{N-1} g_k(u_k) \right\},$$

where $c \in R^n$ is a given vector. Show that the DP algorithm for this problem can be carried out over a one-dimensional state space.

6.2

Argue that for a one-stage problem, the optimal open-loop controller and the OLFC are both optimal. Construct an example where the CEC may be strictly suboptimal. Also work out the following two-stage example, due to [ThW66], which involves the following two-dimensional linear system with scalar control and disturbance:

$$x_{k+1} = x_k + b u_k + d w_k, \qquad k = 0, 1,$$

where $b = (1, 0)'$ and $d = (1/2, \sqrt{2}/2)'$. The initial state is $x_0 = 0$. The controls u_0 and u_1 are unconstrained. The disturbances w_0 and w_1 are independent random variables and each takes the values 1 and -1 with equal probability $1/2$. Perfect state information prevails. The cost is

$$\underset{w_0, w_1}{E} \{ \|x_2\| \},$$

where $\|\cdot\|$ denotes the usual Euclidean norm. Show that the CEC with nominal values $\overline{w}_0 = \overline{w}_1 = 0$ has worse performance than the optimal open-loop controller. In particular, show that the optimal open-loop cost and the optimal closed-loop cost are both $\sqrt{3}/2$, but the cost corresponding to the CEC is 1.

6.3

Consider a two-stage problem with perfect state information involving the scalar system

$$x_0 = 1, \qquad x_1 = x_0 + u_0 + w_0, \qquad x_2 = f(x_1, u_1).$$

The control constraints are u_0, $u_1 \in \{0, -1\}$. The random variable w_0 takes the values 1 and -1 with equal probability $1/2$. The function f is defined by

$$f(1, 0) = f(1, -1) = f(-1, 0) = f(-1, -1) = 0.5,$$

$$f(2, 0) = 0, \quad f(2, -1) = 2, \quad f(0, -1) = 0.6, \quad f(0, 0) = 2.$$

The cost function is

$$\underset{w_0}{E} \{ x_2 \}.$$

(a) Show that one possible OLFC for this problem is

$$\overline{\mu}_0(x_0) = -1, \qquad \overline{\mu}_1(x_1) = \begin{cases} 0 & \text{if } x_1 = \pm 1, 2, \\ -1 & \text{if } x_1 = 0, \end{cases}$$

and the resulting cost is 0.5.

(b) Show that one possible CEC for this problem is

$$\overline{\mu}_0(x_0) = 0, \qquad \overline{\mu}_1(x_1) = \begin{cases} 0 & \text{if } x_1 = \pm 1, 2, \\ -1 & \text{if } x_1 = 0, \end{cases}$$

and the resulting cost is 0.3. Show also that this CEC is an optimal feedback controller.

6.4

Consider the system and cost function of Exercise 6.3 but with the difference that

$$f(0, -1) = 0.$$

(a) Show that the controller of part (a) of Exercise 6.3 is both an OLFC and a CEC, and that the corresponding cost is 0.5.

(b) Assume that the control constraint set for the first stage is $\{0\}$ rather than $\{0, -1\}$. Show that the controller of part (b) of Exercise 6.3 is both an OLFC and a CEC, and that the corresponding cost is 0. *Note*: This problem illustrates a pathology that occurs generically in suboptimal control; that is, if the control constraint set is restricted, the performance of a suboptimal scheme may be improved. To see this, consider a problem and a suboptimal control scheme that is not optimal for the problem. Let $\pi^* = \{\mu_0^*, \ldots, \mu_{N-1}^*\}$ be an optimal policy. Restrict the control constraint set so that only the optimal control $\mu_k^*(x_k)$ is allowed at state x_k. Then the cost attained by the suboptimal control scheme will be improved.

6.5

Consider the ARMAX model

$$y_{k+1} + a y_k = b u_k + \epsilon_{k+1} + c\epsilon_k,$$

where the parameters a, b, and c are unknown. The controller hypothesizes a model of the form

$$y_{k+1} + a y_k = u_k + \epsilon_{k+1}$$

and uses at each k the minimum variance/certainty equivalent control

$$u_k^* = \hat{a}_k y_k,$$

where $\hat{a}_k$ is the least-squares estimate of a obtained as

$$\hat{a}_k = \arg\min_a \sum_{n=1}^{k} \left(y_n + a y_{n-1} - u_{n-1}^*\right)^2.$$

Write a computer program to test the hypothesis that the sequence $\{\hat{a}_k\}$ converges to the optimal value, which is $(c - a)/b$. Experiment with values $|a| < 1$ and $|a| > 1$.

6.6 (Semilinear Systems)

Consider the basic problem for semilinear systems (Exercise 1.13 in Chapter 1). Show that the OLFC, and the CEC, with nominal values of the disturbances equal to their expected values, are optimal for this problem.

6.7

Consider the production control problem of Example 6.3.8 for the case where there is only one part type ($n = 1$), and assume that the cost per stage is a convex function g with $\lim_{|x|\to 1} g(x) = \infty$.

(a) Show that the cost-to-go function $J_k(x_k, \alpha_k)$ is convex as a function of x_k for each value of α_k.

(b) Show that for each k and α_k, there is a target value $\overline{x}_{k+1}$ such that for each x_k it is optimal to choose the control $u_k \in U_k(\alpha_k)$ that brings $x_{k+1} = x_k + u_k - d_k$ as close as possible to $\overline{x}_{k+1}$.

6.8 (WWW)

Provide a careful argument showing that searching a chess position with and without α-β pruning will give the same result.

6.9

In a version of the game of Nim, two players start with a stack of five pennies and take turns removing one, two, or three pennies from the stack. The player who removes the last penny loses. Construct the game tree and verify that the second player to move will win with optimal play.

6.10 (Continuous Space Shortest Path Problems)

Consider the two-dimensional system

$$\dot{x}_1(t) = u_1(t), \qquad \dot{x}_2(t) = u_2(t),$$

with the control constraint $\|u(t)\| = 1$. We want to find a state trajectory that starts at a given point $x(0)$, ends at another given point $x(T)$, and minimizes

$$\int_0^T r\big(x(t)\big)dt.$$

The function $r(\cdot)$ is nonnegative and continuous, and the final time T is subject to optimization. Suppose we discretize the plane with a mesh of size Δ that passes through $x(0)$ and $x(T)$, and we introduce a shortest path problem of going from

$x(0)$ to $x(T)$ using moves of the following type: from each mesh point $\overline{x} = (\overline{x}_1, \overline{x}_2)$ we can go to each of the mesh points $(\overline{x}_1 + \Delta, \overline{x}_2)$, $(\overline{x}_1 - \Delta, \overline{x}_2)$, $(\overline{x}_1, \overline{x}_2 + \Delta)$, and $(\overline{x}_1, \overline{x}_2 - \Delta)$, at a cost $r(\overline{x})\Delta$. Show by example that this is a bad discretization of the original problem in the sense that the shortest distance need not approach the optimal cost of the original problem as $\Delta \to 0$.

6.11 (Discretization of Convex Problems)

Consider a problem with state space S, where S is a convex subset of $\Re^n$. Suppose that $\hat{S} = \{y_1, \ldots, y_M\}$ is a finite subset of S such that S is the convex hull of $\hat{S}$, and consider a one-step lookahead policy based on approximate cost-to-go functions $\tilde{J}_0, \tilde{J}_1, \ldots, \tilde{J}_N$ defined as follows:

$$\tilde{J}_N(x) = g_N(x), \qquad \forall\, x \in S,$$

and for $k = 1, \ldots, N-1$,

$$\tilde{J}_k(x) = \min \left\{ \sum_{i=1}^{M} \lambda_i \hat{J}_k(y_i) \;\Big|\; \sum_{i=1}^{M} \lambda_i y_i = x,\ \sum_{i=1}^{M} \lambda_i = 1,\ \lambda_i \geq 0,\ i = 1, \ldots, M \right\},$$

where $\hat{J}_k(x)$ is defined by

$$\hat{J}_k(x) = \min_{u \in U_k(x)} E\Big\{ g_k(x, u, w_k) + \tilde{J}_{k+1}\big(f_k(x, u, w_k)\big) \Big\}.$$

Thus $\tilde{J}_k$ is obtained from $\tilde{J}_{k+1}$ as a "grid-based" convex piecewise linear approximation to $\hat{J}_k$ based on the M values

$$\hat{J}_k(y_1), \ldots, \hat{J}_k(y_M).$$

Assume that the cost functions g_k and the system functions f_k are such that the function $\hat{J}_k$ is real-valued and convex over S whenever $\tilde{J}_{k+1}$ is real-valued and convex over S. Use Prop. 6.3.1 to show that the cost-to-go functions $\overline{J}_k$ corresponding to the one-step lookahead policy satisfy for all $x \in S$

$$\overline{J}_k(x) \leq \hat{J}_k(x) \leq \tilde{J}_k(x), \qquad k = 0, 1, \ldots, N-1.$$

6.12 (One-Step Lookahead with Cost per Stage and Constraint Approximations)

Consider a one-step lookahead policy as in Section 6.3, where $\tilde{J}_k(x_k)$ is chosen to be the optimal cost-to-go of a different problem where the costs-per-stage and control constraint sets are $\tilde{g}_k(x_k, u_k, w_k)$ and $\tilde{U}_k(x_k)$, respectively, [rather than $g_k(x_k, u_k, w_k)$ and $U_k(x_k)$]. Assume that for all k, x_k, u_k, w_k, we have

$$g_k(x_k, u_k, w_k) \leq \tilde{g}_k(x_k, u_k, w_k), \qquad \tilde{U}_k(x_k) \subset \overline{U}_k(x_k).$$

Use Prop. 6.3.1 to show that the costs-to-go $\overline{J}_k$ of the one-step lookahead policy satisfy

$$\overline{J}_k(x_k) \leq \tilde{J}_k(x_k),$$

for all x_k and k. Extend this result for the case where $\tilde{g}_k$ satisfies instead

$$g_k(x_k, u_k, w_k) \leq \tilde{g}_k(x_k, u_k, w_k) + \delta_k,$$

where δ_k are some scalars that depend only on k.

6.13 (One-Step Lookahead/Rollout for Shortest Paths)

Consider a graph with nodes $1, \ldots, N$, and the problem of finding a shortest path from each of the nodes $1, \ldots, N-1$ to node N with respect to a given set of arc lengths a_{ij}. We assume that all cycles have positive length. Let $F(i)$, $i = 1, \ldots, N$, be some given scalars with $F(N) = 0$, and denote

$$\hat{F}(i) = \min_{j \in J_i} \big[a_{ij} + F(j)\big], \qquad i = 1, \ldots, N-1, \tag{6.66}$$

where for each i, J_i is a nonempty subset of the set of neighbor nodes $\big\{j \mid (i,j) \text{ is an arc}\big\}$.

(a) Assume that $\hat{F}(i) \leq F(i)$ for all $i = 1, \ldots, N-1$. Let $j(i)$ attain the minimum in Eq. (6.66) and consider the graph consisting of the $N-1$ arcs $\big(i, j(i)\big)$, $i = 1, \ldots, N-1$. Show that this graph contains no cycles and for each $i = 1, \ldots, N-1$, it contains a unique path P_i starting at i and ending at N. Show that the length of P_i is less or equal to $\hat{F}(i)$.

(b) Would the conclusion of part (a) hold if the cycles of the original graph are assumed to have nonnegative (rather than positive) length?

(c) Let $F(i)$ be the length of some given path $\overline{P}_i$ from node i to node N with $F(N) = 0$, and assume that for the first arc of $\overline{P}_i$, say (i, j_i), we have $j_i \in J_i$. Assume further that

$$F(i) \geq a_{ij_i} + F(j_i)$$

[this is satisfied with equality if $\overline{P}_i$ consists of arc (i, j_i) followed by path $\overline{P}_{j_i}$, which is true if the paths $\overline{P}_i$ form a tree rooted at the destination N; for example if the paths $\overline{P}_i$ were obtained by solving some related shortest path problem]. Show that $\hat{F}(i) \leq F(i)$ for all $i = 1, \ldots, N-1$.

(d) Assume that $J_i = \{j \mid (i,j) \text{ is an arc}\}$. Let P_i be the paths obtained as in part (a) when the scalars $F(i)$ are generated as in part (c). Interpret P_i as the result of a rollout algorithm that uses an appropriate heuristic, and show that for each i, the length of P_i is less or equal to the length of $\overline{P}_i$.

(e) Assume that $J_i = \{j \mid (i,j) \text{ is an arc}\}$. Let us view the scalars $F(i)$ as the node labels of a label correcting method. This method starts with labels $F(i) = \infty$ for all $i \neq N$ and $F(N) = 0$, and at each step sets

$$F(i) = \min_{\{j \mid (i,j) \text{is an arc}\}} \big[a_{ij} + F(j)\big]$$

for some node $i \neq N$ for which the above equality is violated (the method terminates if this equality holds for all $i \neq N$). Show that in the course of this method, the labels $F(i)$ satisfy the assumptions of part (c) at all times (at or before termination) for which $F(i) < \infty$ for all i.

6.14 (Performance Bounds for Two-Step Lookahead Policies)

Consider a two-step lookahead policy as in Section 6.3, and assume that for all x_k and k, we have

$$\hat{J}_k(x_k) \leq \tilde{J}_k(x_k),$$

where $\hat{J}_N = g_N$ and for $k = 0, \ldots, N-1$,

$$\hat{J}_k(x_k) = \min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big\}.$$

Consider the cost-to-go functions $\overline{J}_k$ corresponding to the two-step lookahead policy that uses $\tilde{J}_k$ and $\overline{U}_k(x_k)$. Show that for all x_k and k, we have

$$\overline{J}_k(x_k) \leq J_k^+(x_k) \leq \hat{J}_k(x_k) \leq \tilde{J}_k(x_k),$$

where J_k^+ is the function obtained by two DP iterations starting from $\tilde{J}_{k+2}$:

$$J_k^+(x_k) = \min_{u_k \in \overline{U}_k(x_k)} E\Big\{ g_k(x_k, u_k, w_k) + \min_{u_{k+1} \in \overline{U}_{k+1}(f_k(x_k, u_k, w_k))} E\Big\{ g_{k+1}\big(f_k(x_k, u_k, w_k), u_{k+1}, w_{k+1}\big) + \tilde{J}_{k+2}\big(f_{k+1}(f_k(x_k, u_k, w_k), u_{k+1}, w_{k+1})\big) \Big\} \Big\}.$$

6.15 (Rollout Algorithms with Errors)

Consider the graph search problem of Section 6.4.1 and let $\mathcal{H}$ be a sequentially improving base heuristic. Suppose that we generate a path $(i_1, \ldots, i_{\overline{m}})$ according to

$$i_{m+1} = \arg \min_{j \in N(i_m)} \hat{H}(j), \qquad m = 1, \ldots, \overline{m} - 1,$$

where $\hat{H}(j)$ differs from the cost $H(j)$ of the base heuristic by the error

$$e(j) = \hat{H}(j) - H(j).$$

(a) Assuming that $|e(j)| \leq \epsilon$ for all j, show that the cost of the generated path is less than or equal to $H(i_1) + 2(\overline{m} - 1)\epsilon$. *Hint*: Use the relation

$$H(i_{m+1}) - \epsilon \leq \hat{H}(i_{m+1}) = \min_{j \in N(i_m)} \hat{H}(j) \leq \min_{j \in N(i_m)} H(j) + \epsilon \leq H(i_m) + \epsilon.$$

(b) Modify the estimate of part (a) for the case where we have $0 \leq e(j) \leq \epsilon$ for all j, and for the case where we have $-\epsilon \leq e(j) \leq 0$ for all j.

(c) Consider the case where $\mathcal{H}$ is optimal so that $H(j) = J^*(j)$, and derive a bound on the difference between the cost of the generated path and the optimal cost starting from i_1.

6.16 (Breakthrough Problem with Random Heuristic)

Consider the breakthrough problem of Example 6.4.2 with the difference that instead of the greedy heuristic, we use the *random* heuristic, which at a given node selects one of the two outgoing arcs with equal probability. Denote by

$$D_k = p^k$$

the probability of success of the random heuristic in a graph of k stages, and by R_k the probability of success of the corresponding rollout algorithm. Show that for all k

$$R_k = p(2-p)R_{k-1} + p^2 D_{k-1}(1 - R_{k-1}).$$

and that

$$\frac{R_k}{D_k} = (2-p)\frac{R_{k-1}}{D_{k-1}} + p(1 - R_{k-1}).$$

Conclude that R_k/D_k increases exponentially with k.

6.17

Consider the breakthrough problem of Example 6.4.2 with the difference that there are three outgoing arcs from each node instead of two. Each arc is free with probability p, independently of other arcs. Derive an equation for the ratio R_k/G_k, where G_k is the probability of success of the greedy heuristic for a k-stage problem, and R_k is the probability of success of the corresponding rollout algorithm. Verify that the results of Example 6.4.2 still hold in a qualitative sense, and that R_k/G_k increases linearly with k.

6.18 (Breakthrough Problem with a Rolling Horizon Rollout)

Consider the breakthrough problem of Example 6.4.2 and consider a rolling horizon-type of rollout algorithm that uses a greedy base heuristic with l-step lookahead. This is the same algorithm as the one described in Example 6.4.2, except that if both outgoing arcs of the current node at stage k are free, the rollout algorithm considers the two end nodes of these arcs, and from each of them it runs the greedy algorithm for $\min\{l, N-k-1\}$ steps. Consider a Markov chain with $l+1$ states, where states $i = 0, \ldots, l-1$ correspond to the path generated by the greedy algorithm being blocked after i arcs. State l corresponds to the path generated by the greedy algorithm being unblocked after l arcs.

(a) Derive the transition probabilities for this Markov chain so that it models the operation of the rollout algorithm.

(b) Use computer simulation to generate the probability of a breakthrough, and to demonstrate that for large values of N, the optimal value of l is roughly constant and much smaller than N (this can also be justified analytically, by using properties of Markov chains).

6.19 (Rollout for Constrained DP)

Consider the deterministic constrained DP problem involving the system

$$x_{k+1} = f_k(x_k, u_k),$$

where we want to minimize the cost function

$$g_N^1(x_N) + \sum_{k=0}^{N-1} g_k^1(x_k, u_k)$$

subject to the constraints

$$g_N^m(x_N) + \sum_{k=0}^{N-1} g_k^m(x_k, u_k) \leq b^m, \qquad m = 2, \ldots, M;$$

cf. Section 2.3.4. We assume that each state x_k takes values in a finite set and each control u_k takes values in a finite constraint set $U_k(x_k)$ that depends on x_k. We describe an extension of the rollout algorithm, involving some base heuristic, which is feasible in the sense that when started from the given initial state x_0, it produces a state/control trajectory that satisfies the constraints of the problem.

Consider a rollout algorithm, which at stage k, maintains a partial state/control trajectory

$$T_k = (x_0, u_0, x_1, \ldots, u_{k-1}, x_k)$$

that starts at the given initial state x_0, and is such that $x_{i+1} = f_i(x_i, u_i)$ and $u_i \in U_i(x_i)$ for all $i = 0, 1, \ldots, k-1$. For such a trajectory, let $C^m(x_k)$ be the corresponding values of constraint functions

$$C^m(x_k) = \sum_{i=0}^{k-1} g_i^m(x_i, u_i), \qquad m = 2, \ldots, M.$$

For each $u_k \in U_k(x_k)$, let $x_{k+1} = f_k(x_k, u_k)$ be the next state, and let $\tilde{J}(x_{k+1})$ and $\tilde{C}^m(x_{k+1})$ be the cost-to-go and values of constraint functions of the base heuristic starting from x_{k+1}.

The algorithm starts with the partial trajectory T_0 that consists of just the initial state x_0. For each $k = 0, \ldots, N-1$, and given the current trajectory T_k, it forms the subset of controls $u_k \in U_k(x_k)$ that together with the corresponding states $x_{k+1} = f_k(x_k, u_k)$ satisfy

$$C^m(x_k) + g_k^m(x_k, u_k) + \tilde{C}^m(x_{k+1}) \leq b^m, \qquad m = 2, \ldots, M.$$

The algorithm selects from this set a control u_k and corresponding state x_{k+1} such that

$$g_k^1(x_k, u_k) + \tilde{J}(x_{k+1})$$

is minimum, and then it forms the trajectory T_{k+1} by adding (u_k, x_{k+1}) to T_k. Formulate analogs of the assumptions of sequential consistency and sequential improvement of Section 6.4.1, under which the algorithm is guaranteed to generate a feasible state/control trajectory that has no greater cost than the cost associated with the base heuristic. *Note:* For a description and analysis of a generalized version of this algorithm, see the author's report "Rollout Algorithms for Constrained Dynamic Programming," LIDS Report 2646, MIT, April 2005.

6.20 (Rollout for Minimax Problems)

Consider the minimax DP problem, as described in Section 1.6, and a one-step lookahead policy based on lookahead functions $\tilde{J}_1, \ldots, \tilde{J}_N$, with $\tilde{J}_N = g_N$. This is the policy obtained by minimizing at state x_k the expression

$$\max_{w_k \in W_k(x_k, u_k)} \Big[g_k(x_k, u_k, w_k) + \tilde{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \Big]$$

over $u_k \in U_k(x_k)$.

(a) State and prove analogs of Props. 6.3.1 and 6.3.2.

(b) Consider a rollout algorithm where $\tilde{J}_k$ are the cost-to-go functions corresponding to some base heuristic. Show that the cost-to-go functions $\overline{J}_k$ of the rollout algorithm satisfy $\overline{J}_k(x_k) \leq \tilde{J}_k(x_k)$ for all x_k and k.

6.21 (MPC with Disturbances)

Consider the MPC framework of Section 6.5.2, including disturbances with set-membership description. Let $\overline{\mu}$ be the policy obtained from MPC.

(a) Use the constrained controllability assumption to show that $\overline{\mu}$ attains reachability of the target tube $\{X, X, \ldots\}$ in the sense that

$$f\big(x, \overline{\mu}(x), w\big) \in X, \qquad \text{for all } x \in X \text{ and } w \in W\big(x, \overline{\mu}(x)\big).$$

(b) Consider any sequence $\{x_0, u_0, x_1, u_1, \ldots\}$ generated by MPC [i.e., $x_0 \in X$, $x_0 \notin T$, $u_k = \overline{\mu}(x_k)$, $x_{k+1} = f(x_k, u_k, w_k)$, and $w_k \in W(x_k, u_k)$]. Show that

$$\sum_{k=0}^{K_T - 1} (x_k' Q x_k + u_k' R u_k) \leq \hat{J}(x_0) < \infty,$$

where K_T is the smallest integer k such that $x_k \in T$ (with $K_T = \infty$ if $x_k \notin T$ for all k), and $\hat{J}(x)$ is the optimal cost starting at state $x \in X$ of the m-stage minimax control problem solved by MPC. *Hint*: Argue as in the case where there are no disturbances. Consider an optimal control problem that is similar to the one solved at each stage by MPC, but has one stage less. In particular, given $x \in X$ with $x \notin T$, consider the minimax control problem of finding a policy $\hat{\mu}_0, \hat{\mu}_1, \ldots, \hat{\mu}_{m-2}$ that minimizes

$$\max_{\substack{w_i \in W(x_i, \hat{\mu}(x_i)), \\ i=0,1,\ldots,m-2}} \sum_{i=0}^{m-2} g\big(x_i, \hat{\mu}_i(x_i)\big),$$

subject to the system equation constraints

$$x_{i+1} = f\big(x_i, \hat{\mu}_i(x_i), w_i\big), \qquad i = 0, 1, \ldots, m-2,$$

the control and state constraints

$$x_i \in X, \qquad \hat{\mu}_i(x_i) \in U(x_i), \qquad i = 0, 1, \ldots, m-2,$$

and the terminal state constraint

$$x_i \in T, \qquad \text{for some } i \in [1, m-1].$$

These constraints must be satisfied for all disturbance sequences with

$$w_i \in W\big(x_i, \hat{\mu}_i(x_i)\big), \qquad i = 0, 1, \ldots, m-2.$$

Let $\tilde{J}(x_0)$ be the corresponding optimal value, and define $\tilde{J}(x_0) = 0$ for $x_0 \in T$, and $\tilde{J}(x_0) = \infty$ for all $x_0 \notin T$ for which the problem has no feasible solution. Show that the control $\overline{\mu}(x)$ applied by MPC at a state $x \in X$ with $x \notin T$, minimizes over $u \in U(x)$

$$\max_{w \in W(x,u)} \Big[x'Qx + u'Ru + \tilde{J}\big(f(x,u,w)\big)\Big],$$

and use the fact $\hat{J}(x) \leq \tilde{J}(x)$ to show that for all $x \in X$ with $x \notin T$, we have

$$\max_{w \in W(x,u)} \left[x'Qx + \overline{\mu}(x)'R\,\overline{\mu}(x) + \hat{J}\Big(f\big(x, \overline{\mu}(x), w\big)\Big)\right] \leq \hat{J}(x).$$

Conclude that for all k such that $x_k \in X$ with $x_k \notin T$, we have

$$x_k'Qx_k + u_k'Ru_k + \hat{J}(x_{k+1}) \leq \hat{J}(x_k),$$

where $\hat{J}(x_{k+1}) = 0$ if $x_{k+1} \in T$. Add over $k = 0, 1, \ldots, K_T - 1$.

(c) Show that under MPC, the state x_k of the system must belong to T for all sufficiently large k, provided that $\min_{x \in X,\, x \notin T} x'Qx > 0$. Use Example 6.5.3 to show the need for this assumption.

(d) Interpret the policy $\overline{\mu}$ produced by MPC as a rollout policy with an appropriate base heuristic. *Hint*: View $\overline{\mu}$ as a one-step lookahead policy with one-step lookahead approximation function equal to $\tilde{J}$, defined in the hint to part (b).

7

Introduction to Infinite Horizon Problems

Contents

In this chapter, we provide an introduction to infinite horizon problems. These problems differ from those considered so far in two respects:

(a) The number of stages is infinite.

(b) The system is stationary, i.e., the system equation, the cost per stage, and the random disturbance statistics do not change from one stage to the next.

The assumption of an infinite number of stages is never satisfied in practice, but is a reasonable approximation for problems involving a finite but very large number of stages. The assumption of stationarity is often satisfied in practice, and in other cases it approximates well a situation where the system parameters vary slowly with time.

Infinite horizon problems are interesting because their analysis is elegant and insightful, and the implementation of optimal policies is often simple. For example, optimal policies are typically stationary, i.e., the optimal rule for choosing controls does not change from one stage to the next.

On the other hand, infinite horizon problems generally require more sophisticated analysis than their finite horizon counterparts, because of the need to analyze limiting behavior as the horizon tends to infinity. This analysis is often nontrivial and at times reveals surprising possibilities. Our treatment will be limited to finite-state problems. A far more detailed development, together with applications from a variety of fields can be found in Vol. II of this work.

7.1 AN OVERVIEW

There are four principal classes of infinite horizon problems. In the first three classes, we try to minimize the *total cost over an infinite number of stages*, given by

$$J_\pi(x_0) = \lim_{N\to\infty} \underset{\substack{w_k\\ k=0,1,\ldots}}{E} \left\{ \sum_{k=0}^{N-1} \alpha^k g\big(x_k, \mu_k(x_k), w_k\big) \right\}.$$

Here, $J_\pi(x_0)$ denotes the cost associated with an initial state x_0 and a policy $\pi = \{\mu_0, \mu_1, \ldots\}$, and α is a positive scalar with $0 < \alpha \leq 1$, called the *discount factor*. The meaning of $\alpha < 1$ is that future costs matter to us less than the same costs incurred at the present time. As an example, think of kth period dollars depreciated to initial period dollars by a factor of $(1+r)^{-k}$, where r is a rate of interest; here $\alpha = 1/(1+r)$. An important concern in total cost problems is that the limit in the definition of $J_\pi(x_0)$ be finite. In the first two of the following classes of problems, this is guaranteed

through various assumptions on the problem structure and the discount factor. In the third class, the analysis is adjusted to deal with infinite cost for some of the policies. In the fourth class, this sum need not be finite for any policy, and for this reason, the cost is appropriately redefined.

(a) *Stochastic shortest path problems.* Here, $\alpha = 1$ but there is a special cost-free termination state; once the system reaches that state it remains there at no further cost. We will assume a problem structure such that termination is inevitable (this assumption will be relaxed somewhat in Chapter 2 of Vol. II). Thus the horizon is in effect finite, but its length is random and may be affected by the policy being used. These problems will be considered in the next section and their analysis will provide the foundation for the analysis of the other types of problems considered in this chapter.

(b) *Discounted problems with bounded cost per stage.* Here, $\alpha < 1$ and the absolute cost per stage $|g(x,u,w)|$ is bounded from above by some constant M; this makes the cost $J_\pi(x_0)$ well defined because it is the infinite sum of a sequence of numbers that are bounded in absolute value by the decreasing geometric progression $\{\alpha^k M\}$. We will consider these problems in Section 7.3.

(c) *Discounted and undiscounted problems with unbounded cost per stage.* Here the discount factor α may or may not be less than 1, and the cost per stage may be unbounded. These problems require a complicated analysis because the possibility of infinite cost for some of the policies is explicitly dealt with. We will not consider these problems here; see Chapter 3 of Vol. II.

(d) *Average cost per stage problems.* Minimization of the total cost $J_\pi(x_0)$ makes sense only if $J_\pi(x_0)$ is finite for at least some admissible policies π and some initial states x_0. Frequently, however, it turns out that $J_\pi(x_0) = \infty$ for every policy π and initial state x_0 (think of the case where $\alpha = 1$, and the cost for every state and control is positive). It turns out that in many such problems the *average cost per stage*, defined by

$$\lim_{N\to\infty} \frac{1}{N} \underset{\substack{w_k\\ k=0,1,\ldots}}{E} \left\{ \sum_{k=0}^{N-1} g\big(x_k, \mu_k(x_k), w_k\big) \right\},$$

is well defined and finite. We will consider some of these problems in Section 7.4.

A Preview of Infinite Horizon Results

There are several analytical and computational issues regarding infinite horizon problems. Many of these revolve around the relation between the

optimal cost-to-go function J^* of the infinite horizon problem and the optimal cost-to-go functions of the corresponding N-stage problems. In particular, consider the case $\alpha = 1$ and let $J_N(x)$ denote the optimal cost of the problem involving N stages, initial state x, cost per stage $g(x,u,w)$, and zero terminal cost. The optimal N-stage cost is generated after N iterations of the DP algorithm

$$J_{k+1}(x) = \min_{u\in U(x)} E_w\Big\{g(x,u,w) + J_k\big(f(x,u,w)\big)\Big\}, \qquad k = 0,1,\ldots \quad (7.1)$$

starting from the initial condition $J_0(x) = 0$ for all x (note here that we have reversed the time indexing to suit our purposes). Since the infinite horizon cost of a given policy is, by definition, the limit of the corresponding N-stage costs as $N \to \infty$, it is natural to speculate that:

(1) The optimal infinite horizon cost is the limit of the corresponding N-stage optimal costs as $N \to \infty$; that is,

$$J^*(x) = \lim_{N\to\infty} J_N(x) \quad (7.2)$$

for all states x. This relation is extremely valuable computationally and analytically, and, fortunately, it typically holds. In particular, it holds for the models of the next two sections [categories (a) and (b) above]. However, there are some unusual exceptions for problems in category (c) above, and this illustrates that infinite horizon problems should be approached with some care. This issue is discussed in more detail in Vol. II.

(2) The following limiting form of the DP algorithm should hold for all states x,

$$J^*(x) = \min_{u\in U(x)} E_w\Big\{g(x,u,w) + J^*\big(f(x,u,w)\big)\Big\},$$

as suggested by Eqs. (7.1) and (7.2). This is not really an algorithm, but rather a system of equations (one equation per state), which has as solution the costs-to-go of all the states. It can also be viewed as a *functional equation* for the cost-to-go function J^*, and it is called *Bellman's equation*. Fortunately again, an appropriate form of this equation holds for every type of infinite horizon problem of interest.

(3) If $\mu(x)$ attains the minimum in the right-hand side of Bellman's equation for each x, then the policy $\{\mu,\mu,\ldots\}$ should be optimal. This is true for most infinite horizon problems of interest and in particular, for all the models discussed in this chapter.

Most of the analysis of infinite horizon problems revolves around the above three issues and also around the issue of efficient computation of J^* and an optimal policy. In the next three sections we will provide a discussion of these issues for some of the simpler infinite horizon problems, all of which involve a finite state space.

Total Cost Problem Formulation

Throughout this chapter we assume a controlled finite-state discrete-time dynamic system whereby, at state i, the use of a control u specifies the transition probability $p_{ij}(u)$ to the next state j. Here the state i is an element of a finite state space, and the control u is constrained to take values in a given finite constraint set $U(i)$, which may depend on the current state i. As discussed in Section 1.1, the underlying system equation is

$$x_{k+1} = w_k,$$

where w_k is the disturbance. We will generally suppress w_k from the cost to simplify notation. Thus we will assume a kth stage cost $g(x_k,u_k)$ for using control u_k at state x_k. This amounts to averaging the cost per stage over all successor states in our calculations, which makes no essential difference in the subsequent analysis. Thus, if $\tilde{g}(i,u,j)$ is the cost of using u at state i and moving to state j, we use as cost per stage the expected cost $g(i,u)$ given by

$$g(i,u) = \sum_j p_{ij}(u)\tilde{g}(i,u,j).$$

The total expected cost associated with an initial state i and a policy $\pi = \{\mu_0,\mu_1,\ldots\}$ is

$$J_\pi(i) = \lim_{N\to\infty} E\left\{\sum_{k=0}^{N-1} \alpha^k g\big(x_k,\mu_k(x_k)\big)\,\Big|\, x_0 = i\right\},$$

where α is a discount factor with $0 < \alpha \le 1$. In the following two sections, we will impose assumptions that guarantee the existence of the above limit. The optimal cost from state i, that is, the minimum of $J_\pi(i)$ over all admissible π, is denoted by $J^*(i)$. A *stationary policy* is an admissible policy of the form $\pi = \{\mu,\mu,\ldots\}$, and its corresponding cost function is denoted by $J_\mu(i)$. For brevity, we refer to $\{\mu,\mu,\ldots\}$ as the stationary policy μ. We say that μ is optimal if

$$J_\mu(i) = J^*(i) = \min_\pi J_\pi(i), \qquad \text{for all states } i.$$

7.2 STOCHASTIC SHORTEST PATH PROBLEMS

Here, we assume that there is no discounting ($\alpha = 1$), and to make the cost meaningful, we assume that *there is a special cost-free termination state* t. Once the system reaches that state, it remains there at no further cost, i.e., $p_{tt}(u) = 1$ and $g(t,u) = 0$ for all $u \in U(t)$. We denote by $1,\ldots,n$ the states other than the termination state t.

We are interested in problems where reaching the termination state is inevitable, at least under an optimal policy. Thus, the essence of the problem is to reach the termination state with minimum expected cost. We call this problem the *stochastic shortest path problem*. The deterministic shortest path problem is obtained as the special case where for each state-control pair (i, u), the transition probability $p_{ij}(u)$ is equal to 1 for a unique state j that depends on (i, u). The reader may also verify that the finite horizon problem of Chapter 1 can be obtained as a special case by viewing as state the pair (x_k, k) (see also Section 3.6 of Vol. II).

Certain conditions are required to guarantee that, at least under an optimal policy, termination occurs with probability 1. We will make the following assumption that guarantees eventual termination under all policies:

Assumption 7.2.1: There exists an integer m such that regardless of the policy used and the initial state, there is positive probability that the termination state will be reached after no more that m stages; that is, for all admissible policies π we have

$$\rho_\pi = \max_{i=1,\ldots,n} P\{x_m \neq t \mid x_0 = i, \pi\} < 1. \tag{7.3}$$

We note, however, that the results to be presented are valid under more general circumstances.† Furthermore, it can be shown that if there exists an integer m with the property of Assumption 7.2.1, then there also exists an integer less or equal to n with this property (Exercise 7.12). Thus, we can always use $m = n$ in Assumption 7.2.1, if no smaller value of m is

† Let us call a stationary policy π *proper* if the condition (7.3) is satisfied for some m, and call π *improper* otherwise. It can be shown that Assumption 7.2.1 is equivalent to the seemingly weaker assumption that all stationary policies are proper (see Vol. II, Exercise 2.3). However, the results of Prop. 7.2.1 can also be shown under the genuinely weaker assumption that there exists at least one proper policy, and furthermore, every improper policy results in infinite expected cost from at least one initial state (see Bertsekas and Tsitsiklis [BeT89], [BeT91], or Vol. II, Chapter 2). These assumptions, when specialized to deterministic shortest path problems, are similar to the ones we used in Chapter 2. They imply that there is at least one path to the destination from every starting node and that all cycles have positive cost. Still another set of assumptions under which the results of Prop. 7.2.1 hold is described in Exercise 7.28, where again improper policies are allowed, but the stage costs $g(i, u)$ are assumed nonnegative, and the optimal costs $J^*(i)$ are assumed finite.

known. Let

$$\rho = \max_\pi \rho_\pi.$$

Note that ρ_π depends only on the first m components of the policy π. Furthermore, since the number of controls available at each state is finite, the number of distinct m-stage policies is also finite. It follows that there can be only a finite number of distinct values of ρ_π so that

$$\rho < 1.$$

We therefore have for any π and any initial state i

$$\begin{aligned} P\{x_{2m} \neq t \mid x_0 = i, \pi\} &= P\{x_{2m} \neq t \mid x_m \neq t,\, x_0 = i, \pi\} \\ &\quad \cdot P\{x_m \neq t \mid x_0 = i, \pi\} \\ &\leq \rho^2. \end{aligned}$$

More generally, for each admissible policy π, the probability of not reaching the termination state after km stages diminishes like ρ^k regardless of the initial state, that is,

$$P\{x_{km} \neq t \mid x_0 = i, \pi\} \leq \rho^k, \qquad i = 1, \ldots, n. \tag{7.4}$$

Thus the limit defining the associated total cost vector J_π exists and is finite, since the expected cost incurred in the m periods between km and $(k+1)m - 1$ is bounded in absolute value by

$$m\rho^k \max_{\substack{i=1,\ldots,n \\ u \in U(i)}} \bigl|g(i, u)\bigr|.$$

In particular, we have

$$\bigl|J_\pi(i)\bigr| \leq \sum_{k=0}^{\infty} m\rho^k \max_{\substack{i=1,\ldots,n \\ u \in U(i)}} \bigl|g(i, u)\bigr| = \frac{m}{1-\rho} \max_{\substack{i=1,\ldots,n \\ u \in U(i)}} \bigl|g(i, u)\bigr|. \tag{7.5}$$

The results of the following proposition are basic and are typical of many infinite horizon problems. The key idea of the proof is that the "tail" of the cost series,

$$\sum_{k=mK}^{\infty} E\Bigl\{g\bigl(x_k, \mu_k(x_k)\bigr)\Bigr\}$$

vanishes as K increases to ∞, since the probability that $x_{mK} \neq t$ decreases like ρ^K [cf. Eq. (7.4)].

Proposition 7.2.1 Under Assumption 7.2.1, the following hold for the stochastic shortest path problem:

(a) Given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence $J_k(i)$ generated by the iteration

$$J_{k+1}(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right], \qquad i = 1, \ldots, n, \tag{7.6}$$

converges to the optimal cost $J^*(i)$ for each i. [Note that, by reversing the time index this iteration can be viewed as the DP algorithm for a finite horizon problem with terminal cost function equal to J_0. In fact, $J_k(i)$ is the optimal cost starting from state i of a k-stage problem with cost per stage given by g and terminal cost at the end of the k stages given by J_0.]

(b) The optimal costs $J^*(1), \ldots, J^*(n)$ satisfy Bellman's equation,

$$J^*(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J^*(j) \right], \qquad i = 1, \ldots, n, \tag{7.7}$$

and in fact they are the unique solution of this equation.

(c) For any stationary policy μ, the costs $J_\mu(1), \ldots, J_\mu(n)$ are the unique solution of the equation

$$J_\mu(i) = g\big(i, \mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) J_\mu(j), \qquad i = 1, \ldots, n.$$

Furthermore, given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence $J_k(i)$ generated by the DP iteration

$$J_{k+1}(i) = g\big(i, \mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) J_k(j), \qquad i = 1, \ldots, n,$$

converges to the cost $J_\mu(i)$ for each i.

(d) A stationary policy μ is optimal if and only if for every state i, $\mu(i)$ attains the minimum in Bellman's equation (7.7).

Proof: (a) For every positive integer K, initial state x_0, and policy $\pi = \{\mu_0, \mu_1, \ldots\}$, we break down the cost $J_\pi(x_0)$ into the portions incurred over

the first mK stages and over the remaining stages

$$\begin{aligned} J_\pi(x_0) &= \lim_{N \to \infty} E\left\{ \sum_{k=0}^{N-1} g\big(x_k, \mu_k(x_k)\big) \right\} \\ &= E\left\{ \sum_{k=0}^{mK-1} g\big(x_k, \mu_k(x_k)\big) \right\} \\ &\quad + \lim_{N \to \infty} E\left\{ \sum_{k=mK}^{N-1} g\big(x_k, \mu_k(x_k)\big) \right\}. \end{aligned}$$

Let M denote the following upper bound on the cost of an m-stage cycle, assuming termination does not occur during the cycle,

$$M = m \max_{\substack{i=1,\ldots,n \\ u \in U(i)}} \big|g(i,u)\big|.$$

The expected cost during the Kth m-stage cycle [stages Km to $(K+1)m-1$] is upper bounded by $M\rho^K$ [cf. Eqs. (7.4) and (7.5)], so that

$$\left| \lim_{N \to \infty} E\left\{ \sum_{k=mK}^{N-1} g\big(x_k, \mu_k(x_k)\big) \right\} \right| \leq M \sum_{k=K}^{\infty} \rho^k = \frac{\rho^K M}{1-\rho}.$$

Also, denoting $J_0(t) = 0$, let us view J_0 as a terminal cost function and bound its expected value under π after mK stages. We have

$$\begin{aligned} \big|E\{J_0(x_{mK})\}\big| &= \left| \sum_{i=1}^{n} P(x_{mK} = i \mid x_0, \pi) J_0(i) \right| \\ &\leq \left(\sum_{i=1}^{n} P(x_{mK} = i \mid x_0, \pi) \right) \max_{i=1,\ldots,n} \big|J_0(i)\big| \\ &\leq \rho^K \max_{i=1,\ldots,n} \big|J_0(i)\big|, \end{aligned}$$

since the probability that $x_{mK} \neq t$ is less or equal to ρ^K for any policy. Combining the preceding relations, we obtain

$$\begin{aligned} -\rho^K \max_{i=1,\ldots,n} \big|J_0(i)\big| + J_\pi(x_0) - \frac{\rho^K M}{1-\rho} \\ \leq E\left\{ J_0(x_{mK}) + \sum_{k=0}^{mK-1} g\big(x_k, \mu_k(x_k)\big) \right\} \qquad (7.8) \\ \leq \rho^K \max_{i=1,\ldots,n} \big|J_0(i)\big| + J_\pi(x_0) + \frac{\rho^K M}{1-\rho}. \end{aligned}$$

Note that the expected value in the middle term of the above inequalities is the mK-stage cost of policy π starting from state x_0, with a terminal cost $J_0(x_{mK})$; the minimum of this cost over all π is equal to the value $J_{mK}(x_0)$, which is generated by the DP recursion (7.6) after mK iterations. Thus, by taking the minimum over π in Eq. (7.8), we obtain for all x_0 and K,

$$\begin{aligned} -\rho^K \max_{i=1,\ldots,n} \left|J_0(i)\right| + J^*(x_0) - \frac{\rho^K M}{1-\rho} & \\ \leq J_{mK}(x_0) & \qquad (7.9) \\ \leq \rho^K \max_{i=1,\ldots,n} \left|J_0(i)\right| + J^*(x_0) + \frac{\rho^K M}{1-\rho}, & \end{aligned}$$

and by taking the limit as $K \to \infty$, we obtain $\lim_{K\to\infty} J_{mK}(x_0) = J^*(x_0)$ for all x_0. Since

$$\left|J_{mK+q}(x_0) - J_{mK}(x_0)\right| \leq \rho^K M, \qquad q = 1, \ldots, m,$$

we see that $\lim_{K\to\infty} J_{mK+q}(x_0)$ is the same for all $q = 1, \ldots, m$, so that we have $\lim_{k\to\infty} J_k(x_0) = J^*(x_0)$.

(b) By taking the limit as $k \to \infty$ in the DP iteration (7.6) and using the result of part (a), we see that $J^*(1), \ldots, J^*(n)$ satisfy Bellman's equation. To show uniqueness, observe that if $J(1), \ldots, J(n)$ satisfy Bellman's equation, then the DP iteration (7.6) starting from $J(1), \ldots, J(n)$ just replicates $J(1), \ldots, J(n)$. It follows from the convergence result of part (a) that $J(i) = J^*(i)$ for all i.

(c) Given the stationary policy μ, we can consider a modified stochastic shortest path problem, which is the same as the original except that the control constraint set contains only one element for each state i, the control $\mu(i)$; that is, the control constraint set is $\tilde{U}(i) = \{\mu(i)\}$ instead of $U(i)$. From part (b) we then obtain that $J_\mu(1), \ldots, J_\mu(n)$ solve uniquely Bellman's equation for this modified problem, that is,

$$J_\mu(i) = g\big(i, \mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) J_\mu(j), \qquad i = 1, \ldots, n,$$

and from part (a) it follows that the corresponding DP iteration converges to $J_\mu(i)$.

(d) We have that $\mu(i)$ attains the minimum in Eq. (7.7) if and only if we have

$$\begin{aligned} J^*(i) &= \min_{u\in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J^*(j) \right] \\ &= g\big(i, \mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) J^*(j), \qquad i = 1, \ldots, n. \end{aligned}$$

Part (c) and the above equation imply that $J_\mu(i) = J^*(i)$ for all i. Conversely, if $J_\mu(i) = J^*(i)$ for all i, parts (b) and (c) imply the above equation. Q.E.D.

Example 7.2.1 (Minimizing Expected Time to Termination)

The case where

$$g(i,u) = 1, \qquad i = 1, \ldots, n, \quad u \in U(i),$$

corresponds to a problem where the objective is to terminate as fast as possible on the average, while the corresponding optimal cost $J^*(i)$ is the minimum expected time to termination starting from state i. Under our assumptions, the costs $J^*(i)$ uniquely solve Bellman's equation, which has the form

$$J^*(i) = \min_{u\in U(i)} \left[1 + \sum_{j=1}^{n} p_{ij}(u) J^*(j) \right], \qquad i = 1, \ldots, n.$$

In the special case where there is only one control at each state, $J^*(i)$ represents the mean first passage time from i to t (see Appendix D). These times, denoted m_i, are the unique solution of the equations

$$m_i = 1 + \sum_{j=1}^{n} p_{ij} m_j, \qquad i = 1, \ldots, n.$$

Example 7.2.2

A spider and a fly move along a straight line at times $k = 0, 1, \ldots$ The initial positions of the fly and the spider are integer. At each time period, the fly moves one unit to the left with probability p, one unit to the right with probability p, and stays where it is with probability $1-2p$. The spider, knows the position of the fly at the beginning of each period, and will always move one unit towards the fly if its distance from the fly is more that one unit. If the spider is one unit away from the fly, it will either move one unit towards the fly or stay where it is. If the spider and the fly land in the same position at the end of a period, then the spider captures the fly and the process terminates. The spider's objective is to capture the fly in minimum expected time.

We view as state the distance between spider and fly. Then the problem can be formulated as a stochastic shortest path problem with states $0, 1, \ldots, n$, where n is the initial distance. State 0 is the termination state where the spider captures the fly. Let us denote $p_{1j}(M)$ and $p_{1j}(\overline{M})$ the transition probabilities from state 1 to state j if the spider moves and does not move, respectively, and let us denote by p_{ij} the transition probabilities from a state $i \geq 2$. We have

$$p_{ii} = p, \quad p_{i(i-1)} = 1 - 2p, \quad p_{i(i-2)} = p, \qquad i \geq 2,$$

$$p_{11}(M) = 2p, \quad p_{10}(M) = 1 - 2p,$$

$$p_{12}(\overline{M}) = p, \quad p_{11}(\overline{M}) = 1 - 2p, \quad p_{10}(\overline{M}) = p,$$

with all other transition probabilities being 0.

For states $i \geq 2$, Bellman's equation is written as

$$J^*(i) = 1 + pJ^*(i) + (1-2p)J^*(i-1) + pJ^*(i-2), \quad i \geq 2, \tag{7.10}$$

where $J^*(0) = 0$ by definition. The only state where the spider has a choice is when it is one unit away from the fly, and for that state Bellman's equation is given by

$$J^*(1) = 1 + \min\big[2pJ^*(1),\, pJ^*(2) + (1-2p)J^*(1)\big], \tag{7.11}$$

where the first and the second expression within the bracket above are associated with the spider moving and not moving, respectively. By writing Eq. (7.10) for $i = 2$, we obtain

$$J^*(2) = 1 + pJ^*(2) + (1-2p)J^*(1),$$

from which

$$J^*(2) = \frac{1}{1-p} + \frac{(1-2p)J^*(1)}{1-p}. \tag{7.12}$$

Substituting this expression in Eq. (7.11), we obtain

$$J^*(1) = 1 + \min\left[2pJ^*(1),\, \frac{p}{1-p} + \frac{p(1-2p)J^*(1)}{1-p} + (1-2p)J^*(1)\right],$$

or equivalently,

$$J^*(1) = 1 + \min\left[2pJ^*(1),\, \frac{p}{1-p} + \frac{(1-2p)J^*(1)}{1-p}\right].$$

To solve the above equation, we consider the two cases where the first expression within the bracket is larger and is smaller than the second expression. Thus we solve for $J^*(1)$ in the two cases where

$$J^*(1) = 1 + 2pJ^*(1), \tag{7.13}$$

$$2pJ^*(1) \leq \frac{p}{1-p} + \frac{(1-2p)J^*(1)}{1-p}, \tag{7.14}$$

and

$$J^*(1) = 1 + \frac{p}{1-p} + \frac{(1-2p)J^*(1)}{1-p}, \tag{7.15}$$

$$2pJ^*(1) \geq \frac{p}{1-p} + \frac{(1-2p)J^*(1)}{1-p}.$$

The solution of Eq. (7.13) is seen to be $J^*(1) = 1/(1-2p)$, and by substitution in Eq. (7.14), we find that this solution is valid when

$$\frac{2p}{1-2p} \leq \frac{p}{1-p} + \frac{1}{1-p},$$

or equivalently (after some calculation), $p \leq 1/3$. Thus for $p \leq 1/3$, it is optimal for the spider to move when it is one unit away from the fly.

Similarly, the solution of Eq. (7.15) is seen to be $J^*(1) = 1/p$, and by substitution in Eq. (7.14), we find that this solution is valid when

$$2 \geq \frac{p}{1-p} + \frac{1-2p}{p(1-p)},$$

or equivalently (after some calculation), $p \geq 1/3$. Thus, for $p \geq 1/3$ it is optimal for the spider not to move when it is one unit away from the fly.

The minimal expected number of steps for capture when the spider is one unit away from the fly was calculated earlier to be

$$J^*(1) = \begin{cases} 1/(1-2p) & \text{if } p \leq 1/3, \\ 1/p & \text{if } p \geq 1/3. \end{cases}$$

Given the value of $J^*(1)$, we can calculate from Eq. (7.12) the minimal expected number of steps for capture when two units away, $J^*(2)$, and we can then obtain the remaining values $J^*(i)$, $i = 3, \ldots, n$, from Eq. (7.10).

Value Iteration and Error Bounds

The DP iteration

$$J_{k+1}(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right], \qquad i = 1, \ldots, n, \tag{7.16}$$

is called *value iteration* and is a principal method for calculating the optimal cost function J^*. Generally, value iteration requires an infinite number of iterations, although there are important special cases where it terminates finitely (see Vol. II, Section 2.2). Note that from Eq. (7.9) we obtain that the error

$$\big|J_{mK}(i) - J^*(i)\big|$$

is bounded by a constant multiple of ρ^K.

The value iteration algorithm can sometimes be strengthened with the use of some error bounds. In particular, it can be shown (see Exercise 7.13) that for all k and j, we have

$$J_{k+1}(j) + \big(N^*(j) - 1\big)\, \underline{c}_k \leq J^*(j) \leq J_{\mu^k}(j) \leq J_{k+1}(j) + \big(N^k(j) - 1\big)\, \overline{c}_k, \tag{7.17}$$

where μ^k is such that $\mu^k(i)$ attains the minimum in the kth value iteration (7.16) for all i, and

$N^*(j)$: The average number of stages to reach t starting from j and using some optimal stationary policy,

$N^k(j)$: The average number of stages to reach t starting from j and using the stationary policy μ^k,

$$\underline{c}_k = \min_{i=1,\ldots,n}\big[J_{k+1}(i) - J_k(i)\big], \qquad \overline{c}_k = \max_{i=1,\ldots,n}\big[J_{k+1}(i) - J_k(i)\big].$$

Unfortunately, the values $N^*(j)$ and $N^k(j)$ are easily computed or approximated only in the presence of special problem structure (see for example the next section). Despite this fact, the bounds (7.17) often provide a useful guideline for stopping the value iteration algorithm while being assured that J_k approximates J^* with sufficient accuracy.

Policy Iteration

There is an alternative to value iteration, which always terminates finitely. This algorithm is called *policy iteration* and operates as follows: we start with a stationary policy μ^0, and we generate a sequence of new policies $\mu^1, \mu^2, \ldots$ Given the policy μ^k, we perform a *policy evaluation step*, that computes $J_{\mu^k}(i)$, $i = 1, \ldots, n$, as the solution of the (linear) system of equations

$$J(i) = g\big(i, \mu^k(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^k(i)\big) J(j), \qquad i = 1, \ldots, n, \tag{7.18}$$

in the n unknowns $J(1), \ldots, J(n)$ [cf. Prop. 7.2.1(c)]. We then perform a *policy improvement step*, which computes a new policy μ^{k+1} as

$$\mu^{k+1}(i) = \arg\min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_{\mu^k}(j) \right], \qquad i = 1, \ldots, n. \tag{7.19}$$

The process is repeated with μ^{k+1} used in place of μ^k, unless we have $J_{\mu^{k+1}}(i) = J_{\mu^k}(i)$ for all i, in which case the algorithm terminates with the policy μ^k. The following proposition establishes the validity of policy iteration.

Proposition 7.2.2 Under Assumption 7.2.1, the policy iteration algorithm for the stochastic shortest path problem generates an improving sequence of policies [that is, $J_{\mu^{k+1}}(i) \leq J_{\mu^k}(i)$ for all i and k] and terminates with an optimal policy.

Proof: For any k, consider the sequence generated by the recursion

$$J_{N+1}(i) = g\big(i, \mu^{k+1}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^{k+1}(i)\big) J_N(j), \qquad i = 1, \ldots, n,$$

where $N = 0, 1, \ldots$, and

$$J_0(i) = J_{\mu^k}(i), \qquad i = 1, \ldots, n.$$

From Eqs. (7.18) and (7.19), we have

$$\begin{aligned} J_0(i) &= g\big(i, \mu^k(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^k(i)\big) J_0(j) \\ &\geq g\big(i, \mu^{k+1}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^{k+1}(i)\big) J_0(j) \\ &= J_1(i), \end{aligned}$$

for all i. By using the above inequality we obtain (compare with the monotonicity property of DP, Exercise 1.23 in Chapter 1)

$$\begin{aligned} J_1(i) &= g\big(i, \mu^{k+1}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^{k+1}(i)\big) J_0(j) \\ &\geq g\big(i, \mu^{k+1}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^{k+1}(i)\big) J_1(j) \\ &= J_2(i), \end{aligned}$$

for all i, and by continuing similarly we have

$$J_0(i) \geq J_1(i) \geq \cdots \geq J_N(i) \geq J_{N+1}(i) \geq \cdots, \qquad i = 1, \ldots, n. \tag{7.20}$$

Since by Prop. 7.2.1(c), $J_N(i) \to J_{\mu^{k+1}}(i)$, we obtain $J_0(i) \geq J_{\mu^{k+1}}(i)$ or

$$J_{\mu^k}(i) \geq J_{\mu^{k+1}}(i), \qquad i = 1, \ldots, n, \quad k = 0, 1, \ldots.$$

Thus the sequence of generated policies is improving, and since the number of stationary policies is finite, we must after a finite number of iterations, say $k+1$, obtain $J_{\mu^k}(i) = J_{\mu^{k+1}}(i)$ for all i. Then we will have equality holding throughout in Eq. (7.20), which means that

$$J_{\mu^k}(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_{\mu^k}(j) \right], \qquad i = 1, \ldots, n.$$

Thus the costs $J_{\mu^k}(1), \ldots, J_{\mu^k}(n)$ solve Bellman's equation, and by Prop. 7.2.1(b), it follows that $J_{\mu^k}(i) = J^*(i)$ and that μ^k is optimal. Q.E.D.

The linear system of equations (7.18) of the policy evaluation step can be solved by standard methods such as Gaussian elimination, but when the number of states is large, this is cumbersome and time-consuming. A typically more efficient alternative is to approximate the policy evaluation step with a few value iterations aimed at solving the corresponding system (7.18). One can show that the policy iteration method that uses such approximate policy evaluation yields in the limit the optimal costs and an optimal stationary policy, even if we evaluate each policy using an arbitrary positive number of value iterations (see Vol. II, Section 1.3).

Another possibility for approximating the policy evaluation step is to use simulation, and this is a key idea in the rollout algorithm, discussed in Section 6.4. Simulation also plays an important role in the neuro-dynamic programming methodology, discussed in Ch. 2 of Vol. II. In particular, when the number of states is large, one can try to approximate the cost-to-go function J_{μ^k} by simulating a large number of trajectories under the policy μ^k, and perform some form of least squares fit of J_{μ^k} using an approximation architecture (cf. Section 6.3.4). These are a number of variations of this idea, which are discussed in more detail in Vol. II and in the research monograph by Bertsekas and Tsitsiklis [BeT96].

Linear Programming

Suppose that we use value iteration to generate a sequence of vectors $J_k = \big(J_k(1), \ldots, J_k(n)\big)$ starting with an initial condition vector $J_0 = \big(J_0(1), \ldots, J_0(n)\big)$ such that

$$J_0(i) \leq \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_0(j) \right], \qquad i = 1, \ldots, n.$$

Then we will have $J_k(i) \leq J_{k+1}(i)$ for all k and i (the monotonicity property of DP; Exercise 1.23 in Chapter 1). It follows from Prop. 7.2.1(a) that we will also have $J_0(i) \leq J^*(i)$ for all i. Thus J^* is the "largest" J that satisfies the constraint

$$J(i) \leq g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J(j), \qquad \text{for all } i = 1, \ldots, n \text{ and } u \in U(i).$$

In particular, $J^*(1), \ldots, J^*(n)$ solve the linear program of maximizing $\sum_{i=1}^{n} J(i)$ subject to the above constraint (see Fig. 7.2.1). Unfortunately, for large n the dimension of this program can be very large and its solution can be impractical, particularly in the absence of special structure.

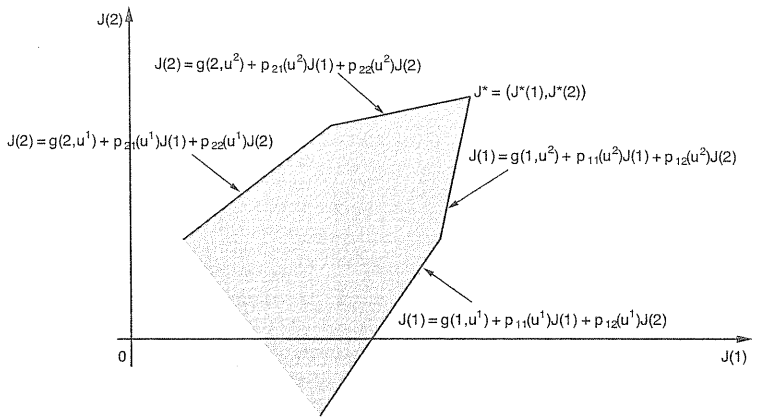


Figure 7.2.1 Linear program associated with a two-state stochastic shortest path problem. The constraint set is shaded, and the objective to maximize is $J(1) + J(2)$. Note that because we have $J(i) \leq J^*(i)$ for all i and vectors J in the constraint set, the vector J^* maximizes any linear cost function of the form $\sum_{i=1}^{n} \beta_i J(i)$ where $\beta_i \geq 0$ for all i. If $\beta_i > 0$ for all i, then J^* is the unique optimal solution of the corresponding linear program.

7.3 DISCOUNTED PROBLEMS

We now consider a discounted problem, where there is a discount factor $\alpha < 1$. We will show that this problem can be converted to a stochastic shortest path problem for which the analysis of the preceding section holds. To see this, let $i = 1, \ldots, n$ be the states, and consider an associated stochastic shortest path problem involving the states $1, \ldots, n$ plus an extra termination state t, with state transitions and costs obtained as follows: From a state $i \neq t$, when control u is applied, a cost $g(i,u)$ is incurred, and the next state is j with probability $\alpha p_{ij}(u)$ and t with probability $1-\alpha$; see Fig. 7.3.1. Note that Assumption 7.2.1 of the preceding section is satisfied for the associated stochastic shortest path problem.

Suppose now that we use the same policy in the discounted problem and in the associated stochastic shortest path problem. Then, as long as termination has not occurred, the state evolution in the two problems is governed by the same transition probabilities. Furthermore, the expected cost of the kth stage of the associated shortest path problem is $g\big(x_k, \mu_k(x_k)\big)$ multiplied by the probability that state t has not yet been reached, which is α^k. This is also the expected cost of the kth stage for the discounted problem. We conclude that the cost of any policy starting from a given state, is the same for the original discounted problem and for the associated stochastic shortest path problem. Furthermore, value iteration produces identical iterates for the two problems. We can thus apply

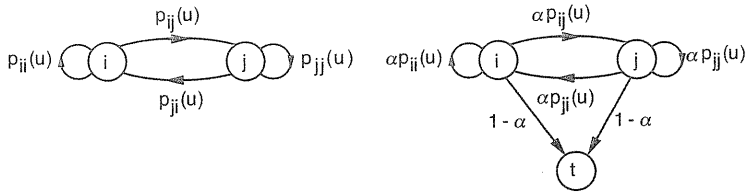


Figure 7.3.1 Transition probabilities for an α-discounted problem and its associated stochastic shortest path problem. In the latter problem, the probability that the state is not t after k stages is α^k. The expected cost at each state $i = 1, \ldots, n$ is $g(i, u)$ for both problems, but it must be multiplied by α^k because of discounting (in the discounted case) or because it is incurred with probability α^k when termination has not yet been reached (in the stochastic shortest path case).

the results of the preceding section to the latter problem and obtain the following:

Proposition 7.3.1 The following hold for the discounted problem:

(a) The value iteration algorithm

$$J_{k+1}(i) = \min_{u \in U(i)} \left[g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right], \qquad i = 1, \ldots, n, \tag{7.21}$$

converges to the optimal costs $J^*(i)$, $i = 1, \ldots, n$, starting from arbitrary initial conditions $J_0(1), \ldots, J_0(n)$.

(b) The optimal costs $J^*(1), \ldots, J^*(n)$ of the discounted problem satisfy Bellman's equation,

$$J^*(i) = \min_{u \in U(i)} \left[g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u) J^*(j) \right], \qquad i = 1, \ldots, n, \tag{7.22}$$

and in fact they are the unique solution of this equation.

(c) For any stationary policy μ, the costs $J_\mu(1), \ldots, J_\mu(n)$ are the unique solution of the equation

$$J_\mu(i) = g\big(i, \mu(i)\big) + \alpha \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) J_\mu(j), \qquad i = 1, \ldots, n.$$

Furthermore, given any initial conditions $J_0(1), \ldots, J_0(n)$, the sequence $J_k(i)$ generated by the DP iteration

$$J_{k+1}(i) = g\big(i, \mu(i)\big) + \alpha \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) J_k(j), \qquad i = 1, \ldots, n,$$

converges to the cost $J_\mu(i)$ for each i.

(d) A stationary policy μ is optimal if and only if for every state i, $\mu(i)$ attains the minimum in Bellman's equation (7.22).

(e) The policy iteration algorithm given by

$$\mu^{k+1}(i) = \arg \min_{u \in U(i)} \left[g(i,u) + \alpha \sum_{j=1}^{n} p_{ij}(u) J_{\mu^k}(j) \right], \quad i = 1, \ldots, n,$$

generates an improving sequence of policies and terminates with an optimal policy.

Proof: Parts (a)-(d) and part (e) are proved by applying parts (a)-(d) of Prop. 7.2.1, and Prop. 7.2.2, respectively, to the associated stochastic shortest path problem described above. **Q.E.D.**

Bellman's equation (7.22) has a familiar DP interpretation. At state i, the optimal cost $J^*(i)$ is the minimum over all controls of the sum of the expected current stage cost and the expected optimal cost of all future stages. The former cost is $g(i, u)$. The latter cost is $J^*(j)$, but since this cost starts accumulating after one stage, it is discounted by multiplication with α.

As in the case of stochastic shortest path problems [see Eq. (7.9) and the discussion following the proof of Prop. 7.2.1], we can show that the error

$$\big| J_k(i) - J^*(i) \big|$$

is bounded by a constant times α^k. Furthermore, the error bounds (7.17) become

$$J_{k+1}(j) + \frac{\alpha}{1-\alpha}\, \underline{c}_k \leq J^*(j) \leq J_{\mu^k}(j) \leq J_{k+1}(j) + \frac{\alpha}{1-\alpha}\, \overline{c}_k, \tag{7.23}$$

where μ^k is such that $\mu^k(i)$ attains the minimum in the kth value iteration (7.21) for all i, and

$$\underline{c}_k = \min_{i=1,\ldots,n} \big[J_{k+1}(i) - J_k(i) \big], \qquad \overline{c}_k = \max_{i=1,\ldots,n} \big[J_{k+1}(i) - J_k(i) \big],$$

since for the associated stochastic shortest path problem it can be shown that for every policy and starting state, the expected number of stages to reach the termination state t is $1/(1-\alpha)$, so that the terms $N^*(j)-1$ and $N^k(j)-1$ appearing in Eq. (7.17) are equal to $\alpha/(1-\alpha)$. We note also that there are a number of additional enhancements to the value iteration algorithm for the discounted problem (see Section 1.3 of Vol. II). There are also discounted cost variants of the approximate policy iteration and linear programming approaches discussed for stochastic shortest path problems.

Example 7.3.1 (Asset Selling)

Consider an infinite horizon version of the asset selling example of Section 4.4, assuming the set of possible offers is finite. Here, if accepted, the amount x_k offered in period k, will be invested at a rate of interest r. By depreciating the sale amount to period 0 dollars, we view $(1+r)^{-k}x_k$ as the reward for selling the asset in period k at a price x_k, where $r>0$ is the rate of interest. Then we have a total discounted reward problem with discount factor $\alpha = 1/(1+r)$. The analysis of the present section is applicable, and the optimal value function J^* is the unique solution of Bellman's equation

$$J^*(x) = \max\left[x, \frac{E\{J^*(w)\}}{1+r}\right],$$

(see Section 4.4). The optimal reward function is characterized by the critical number

$$\bar{\alpha} = \frac{E\{J^*(w)\}}{1+r},$$

which can be calculated as in Section 4.4. An optimal policy is to sell if and only if the current offer x_k is greater than or equal to $\bar{\alpha}$.

Example 7.3.2

A manufacturer at each time period receives an order for her product with probability p and receives no order with probability $1-p$. At any period she has a choice of processing all the unfilled orders in a batch, or process no order at all. The cost per unfilled order at each time period is $c>0$, and the setup cost to process the unfilled orders is $K>0$. The manufacturer wants to find a processing policy that minimizes the total expected cost, assuming the discount factor is $\alpha<1$ and the maximum number of orders that can remain unfilled is n.

Here the state is the number of unfilled orders at the beginning of each period, and Bellman's equation takes the form

$$J^*(i) = \min\big[K+\alpha(1-p)J^*(0)+\alpha pJ^*(1),\ ci+\alpha(1-p)J^*(i)+\alpha pJ^*(i+1)\big], \tag{7.24}$$

for the states $i=0,1,\ldots,n-1$, and takes the form

$$J^*(n) = K+\alpha(1-p)J^*(0)+\alpha pJ^*(1) \tag{7.25}$$

for state n. The first expression within brackets in Eq. (7.24) corresponds to processing the i unfilled orders, while the second expression corresponds to leaving the orders unfilled for one more period. When the maximum n of unfilled orders is reached, the orders must necessarily be processed, as indicated by Eq. (7.25).

To solve the problem, we observe that the optimal cost $J^*(i)$ is monotonically nondecreasing in i. This is intuitively clear, and can be rigorously proved by using the value iteration method. In particular, we can show by using the (finite horizon) DP algorithm that the k-stage optimal cost functions $J_k(i)$ are monotonically nondecreasing in i for all k (Exercise 7.7), and then argue that the optimal infinite horizon cost function $J^*(i)$ is also monotonically nondecreasing in i, since

$$J^*(i) = \lim_{k\to\infty} J_k(i)$$

by Prop. 7.3.1(a). Given that $J^*(i)$ is monotonically nondecreasing in i, from Eq. (7.24) we have that if processing a batch of m orders is optimal, that is,

$$K+\alpha(1-p)J^*(0)+\alpha pJ^*(1) \le cm+\alpha(1-p)J^*(m)+\alpha pJ^*(m+1),$$

then processing a batch of $m+1$ orders is also optimal. Therefore a *threshold policy*, that is, a policy that processes the orders if their number exceeds some threshold integer m^*, is optimal.

We leave it as Exercise 7.8 for the reader to verify that if we start the policy iteration algorithm with a threshold policy, every subsequently generated policy will be a threshold policy. Since there are $n+1$ distinct threshold policies, and the sequence of generated policies is improving, it follows that the policy iteration algorithm will yield an optimal policy after at most n iterations.

7.4 AVERAGE COST PER STAGE PROBLEMS

The methodology of the last two sections applies mainly to problems where the optimal total expected cost is finite either because of discounting or because of a cost-free termination state that the system eventually enters. In many situations, however, discounting is inappropriate and there is no natural cost-free termination state. In such situations it is often meaningful to optimize the average cost per stage starting from a state i, which is defined by

$$J_\pi(i) = \lim_{N\to\infty}\frac{1}{N}E\left\{\sum_{k=0}^{N-1} g\big(x_k,\mu_k(x_k)\big)\,\middle|\, x_0=i\right\}.$$

Let us first argue heuristically that *for most problems of interest the average cost per stage of a policy and the optimal average cost per stage are independent of the initial state.*

To this end we note that the average cost per stage of a policy primarily expresses cost incurred in the long term. Costs incurred in the early stages do not matter since their contribution to the average cost per stage is reduced to zero as $N \to \infty$; that is,

$$\lim_{N\to\infty} \frac{1}{N} E\left\{ \sum_{k=0}^{K} g\big(x_k, \mu_k(x_k)\big) \right\} = 0, \tag{7.26}$$

for any fixed K. Consider now a stationary policy μ and two states i and j such that the system will, under μ, eventually reach j with probability 1 starting from i. Then intuitively, it is clear that the average costs per stage starting from i and from j cannot be different, since the costs incurred in the process of reaching j from i do not contribute essentially to the average cost per stage. More precisely, let $K_{ij}(\mu)$ be the first passage time from i to j under μ, that is, the first index k for which $x_k = j$ starting from $x_0 = i$ under μ (see Appendix D). Then the average cost per stage corresponding to initial condition $x_0 = i$ can be expressed as

$$J_\mu(i) = \lim_{N\to\infty} \frac{1}{N} E\left\{ \sum_{k=0}^{K_{ij}(\mu)-1} g\big(x_k, \mu(x_k)\big) \right\}$$

$$+ \lim_{N\to\infty} \frac{1}{N} E\left\{ \sum_{k=K_{ij}(\mu)}^{N} g\big(x_k, \mu(x_k)\big) \right\}.$$

If $E\big\{K_{ij}(\mu)\big\} < \infty$ (which is equivalent to assuming that the system eventually reaches j starting from i with probability 1; see Appendix D), then it can be seen that the first limit is zero [cf. Eq. (7.26)], while the second limit is equal to $J_\mu(j)$. Therefore,

$$J_\mu(i) = J_\mu(j), \qquad \text{for all } i, j \text{ with } E\big\{K_{ij}(\mu)\big\} < \infty.$$

The preceding argument suggests that the optimal cost $J^*(i)$ should also be independent of the initial state i under normal circumstances. To see this, assume that for any two states i and j, there exists a stationary policy μ (dependent on i and j) such that j can be reached from i with probability 1 under μ. Then it is impossible that

$$J^*(j) < J^*(i),$$

since when starting from i we can adopt the policy μ up to the time when j is first reached and then switch to a policy that is optimal when starting from j, thereby achieving an average cost starting from i that is equal to $J^*(j)$. Indeed, it can be shown that

$$J^*(i) = J^*(j), \qquad \text{for all } i, j = 1, \ldots, n,$$

under the preceding assumption (see Vol. II, Section 4.2).

The Associated Stochastic Shortest Path Problem

The results of this section can be proved under a variety of different assumptions (see Chapter 4 in Vol. II). Here, we will make the following assumption, which will allow us to use the stochastic shortest path analysis of Section 7.2.

> **Assumption 7.4.1:** One of the states, by convention state n, is such that for some integer $m > 0$, and for all initial states and all policies, n is visited with positive probability at least once within the first m stages.

Assumption 7.4.1 can be shown to be equivalent to the assumption that the special state n is recurrent in the Markov chain corresponding to each stationary policy (see Appendix D for the definition of a recurrent state, and Exercise 2.3 of Chapter 2 in Vol. II for a proof of this equivalence).

Under Assumption 7.4.1 we will make an important connection of the average cost problem with an associated stochastic shortest path problem. To motivate this connection, consider a sequence of generated states, and divide it into "independent" cycles marked by successive visits to the state n. The first cycle includes the transitions from the initial state to the first visit to state n, and the kth cycle, $k = 2, 3, \ldots$, includes the transitions from the $(k-1)$th to the kth visit to state n. Each of the cycles can be viewed as a state trajectory of a corresponding stochastic shortest path problem with the termination state being essentially n.

More precisely, this problem is obtained by leaving unchanged all transition probabilities $p_{ij}(u)$ for $j \neq n$, by setting all transition probabilities $p_{in}(u)$ to 0, and by introducing an artificial termination state t to which we move from each state i with probability $p_{in}(u)$; see Fig. 7.4.1. Note that Assumption 7.4.1 is equivalent to the Assumption 7.2.1 of Section 7.2 under which the results of Section 7.2 on stochastic shortest path problems were shown.

We have specified the probabilistic structure of the stochastic shortest path problem so that its state trajectories replicate the state trajectories of a single cycle of the average cost problem. We will next argue that if we fix the expected stage cost incurred at state i to be

$$g(i, u) - \lambda^*,$$

where λ^* is the optimal average cost per stage starting from the special state n, then the associated stochastic shortest path problem becomes essentially equivalent to the original average cost per stage problem. Furthermore, Bellman's equation for the associated stochastic shortest path problem can

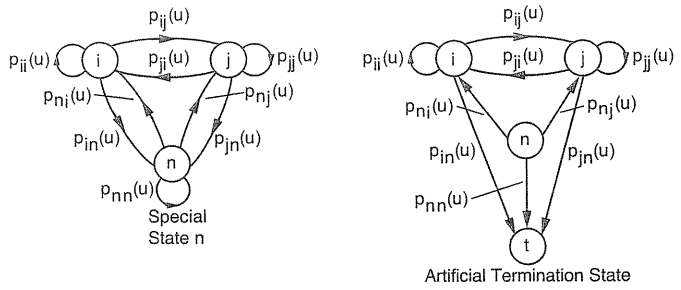


Figure 7.4.1 Transition probabilities for an average cost problem and its associated stochastic shortest path problem. The latter problem is obtained by introducing, in addition to $1, \ldots, n$, an artificial termination state t. The corresponding transition probabilities are obtained from the transition probabilities of the original average cost problem as follows: the probabilities of transition from the states $i \neq t$ to state t are set equal to $p_{in}(u)$, the probabilities of transition from all states to state n are set to 0, and all other transition probabilities are left unchanged.

be viewed as Bellman's equation for the original average cost per stage problem.

For a heuristic argument of why this is so, note that under all stationary policies there will be an infinite number of cycles marked by successive visits to state n. From this, it can be conjectured (and it can also be shown, as will be seen later) that the average cost problem is equivalent to a *minimum cycle cost problem*. This is the problem of finding a stationary policy μ that minimizes the average cycle cost

$$\frac{C_{nn}(\mu)}{N_{nn}(\mu)},$$

where for a fixed μ,

$C_{nn}(\mu)$: expected cost starting from n up to the first return to n,

$N_{nn}(\mu)$: expected number of stages to return to n starting from n.

The intuitive idea here is that the ratio $C_{nn}(\mu)/N_{nn}(\mu)$ is equal to the average cost of μ,† so the optimal average cost λ^* is equal to the optimal cycle cost. Therefore, we have

$$C_{nn}(\mu) - N_{nn}(\mu)\lambda^* \geq 0, \qquad \text{for all } \mu, \tag{7.27}$$

† For a heuristic argument, let λ_μ be the average cost per stage corresponding to a stationary policy μ, and consider a trajectory of the system under μ, starting from state n. If $C_1, C_2, \ldots C_m$ are the costs incurred in the first m cycles, and $N_1, N_2, \ldots, N_m$ are the corresponding numbers of stages of these cycles, we have

$$\lambda_\mu = \lim_{m\to\infty} \frac{\sum_{k=1}^m C_k}{\sum_{k=1}^m N_k} = \lim_{m\to\infty} \frac{\sum_{k=1}^m C_k}{m} \cdot \lim_{m\to\infty} \frac{m}{\sum_{k=1}^m N_k} = C_{nn}(\mu) \cdot \frac{1}{N_{nn}(\mu)}$$

(with probability one).

with equality holding if μ is optimal. Thus, to attain an optimal μ, we must minimize over μ the expression $C_{nn}(\mu) - N_{nn}(\mu)\lambda^*$, which is the expected cost of μ starting from n in the associated stochastic shortest path problem with stage costs

$$g(i,u) - \lambda^*, \qquad i = 1, \ldots, n.$$

Let us denote by $h^*(i)$ the optimal cost of this stochastic shortest path problem when starting at the nontermination states $i = 1, \ldots, n$. Then by Prop. 7.2.1(b), $h^*(1), \ldots, h^*(n)$ solve uniquely the corresponding Bellman's equation, which has the form

$$h^*(i) = \min_{u\in U(i)} \left[g(i,u) - \lambda^* + \sum_{j=1}^{n-1} p_{ij}(u) h^*(j) \right], \qquad i = 1, \ldots, n, \tag{7.28}$$

since in the stochastic shortest path problem, the transition probability from i to $j \neq n$ is $p_{ij}(u)$ and the transition probability from i to n is zero under all u. If μ^* is a stationary policy that minimizes the cycle cost, then this policy must satisfy,

$$C_{nn}(\mu^*) - N_{nn}(\mu^*)\lambda^* = 0,$$

and from Eq. (7.27), this policy must also be optimal for the associated stochastic shortest path problem. Thus, we must have

$$h^*(n) = C_{nn}(\mu^*) - N_{nn}(\mu^*)\lambda^* = 0.$$

By using this equation, we can now write Bellman's equation (7.28) as

$$\lambda^* + h^*(i) = \min_{u\in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h^*(j) \right], \qquad i = 1, \ldots, n. \tag{7.29}$$

Equation (7.29), which is really Bellman's equation for the associated stochastic shortest path problem, will be viewed as Bellman's equation for the average cost per stage problem. The preceding argument indicates that this equation has a unique solution as long as we impose the constraint $h^*(n) = 0$. Furthermore, by minimization of its right-hand side we should obtain an optimal stationary policy. We will now prove these facts formally.

Bellman's Equation

The following proposition provides the main results regarding Bellman's equation.

Proposition 7.4.1 Under Assumption 7.4.1 the following hold for the average cost per stage problem:

(a) The optimal average cost λ^* is the same for all initial states and together with some vector $h^* = \{h^*(1), \ldots, h^*(n)\}$ satisfies Bellman's equation

$$\lambda^* + h^*(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h^*(j) \right], \qquad i = 1, \ldots, n. \tag{7.30}$$

Furthermore, if $\mu(i)$ attains the minimum in the above equation for all i, the stationary policy μ is optimal. In addition, out of all vectors h^* satisfying this equation, there is a unique vector for which $h^*(n) = 0$.

(b) If a scalar λ and a vector $h = \{h(1), \ldots, h(n)\}$ satisfy Bellman's equation, then λ is the average optimal cost per stage for each initial state.

(c) Given a stationary policy μ with corresponding average cost per stage λ_μ, there is a unique vector $h_\mu = \{h_\mu(1), \ldots, h_\mu(n)\}$ such that $h_\mu(n) = 0$ and

$$\lambda_\mu + h_\mu(i) = g\big(i, \mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) h_\mu(j), \qquad i = 1, \ldots, n.$$

Proof: (a) Let us denote

$$\tilde{\lambda} = \min_{\mu} \frac{C_{nn}(\mu)}{N_{nn}(\mu)}, \tag{7.31}$$

where $C_{nn}(\mu)$ and $N_{nn}(\mu)$ have been defined earlier, and the minimum is taken over the finite set of all stationary policies. Note that $C_{nn}(\mu)$ and $N_{nn}(\mu)$ are finite in view of Assumption 7.4.1 and the results of Section 7.2. Then we have

$$C_{nn}(\mu) - N_{nn}(\mu)\tilde{\lambda} \geq 0,$$

with equality holding for all μ that attain the minimum in Eq. (7.31). Consider the associated stochastic shortest path problem when the expected

stage cost incurred at state i is

$$g(i,u) - \tilde{\lambda}.$$

Then by Prop. 7.2.1(b), the costs $h^*(1), \ldots, h^*(n)$ solve uniquely the corresponding Bellman's equation

$$h^*(i) = \min_{u \in U(i)} \left[g(i,u) - \tilde{\lambda} + \sum_{j=1}^{n-1} p_{ij}(u) h^*(j) \right], \tag{7.32}$$

since the transition probability from i to n is zero in the associated stochastic shortest path problem. An optimal stationary policy must minimize the cost $C_{nn}(\mu) - N_{nn}(\mu)\tilde{\lambda}$ and reduce it to zero [in view of Eq. (7.31)], so we see that

$$h^*(n) = 0. \tag{7.33}$$

Thus, Eq. (7.32) is written as

$$\tilde{\lambda} + h^*(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h^*(j) \right], \quad i = 1, \ldots, n. \tag{7.34}$$

We will show that this relation implies that $\tilde{\lambda} = \lambda^*$.

Indeed, let $\pi = \{\mu_0, \mu_1, \ldots\}$ be any admissible policy, let N be a positive integer, and for all $k = 0, \ldots, N-1$, define $J_k(i)$ using the following recursion

$$J_0(i) = h^*(i), \qquad i = 1, \ldots, n,$$

$$J_{k+1}(i) = g\big(i, \mu_{N-k-1}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu_{N-k-1}(i)\big) J_k(j), \quad i = 1, \ldots, n. \tag{7.35}$$

Note that $J_N(i)$ is the N-stage cost of π when the starting state is i and the terminal cost function is h^*. From Eq. (7.34), we have

$$\tilde{\lambda} + h^*(i) \leq g\big(i, \mu_{N-1}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu_{N-1}(i)\big) h^*(j), \quad i = 1, \ldots, n,$$

or equivalently, using Eq. (7.35) for $k = 0$ and the definition of J_0,

$$\tilde{\lambda} + J_0(i) \leq J_1(i), \qquad i = 1, \ldots, n.$$

Using this relation, we have

$$g\big(i, \mu_{N-2}(i)\big) + \tilde{\lambda} + \sum_{j=1}^{n} p_{ij}\big(\mu_{N-2}(i)\big) J_0(j)$$

$$\leq g\big(i, \mu_{N-2}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu_{N-2}(i)\big) J_1(j), \quad i = 1, \ldots, n.$$

By Eq. (7.34) and the definition $J_0(j) = h^*(j)$, the left-hand side of the above inequality is no less than $2\tilde{\lambda} + h^*(i)$, while by Eq. (7.35), the right-hand side is equal to $J_2(i)$. We thus obtain

$$2\tilde{\lambda} + h^*(i) \leq J_2(i), \qquad i = 1, \ldots, n.$$

By repeating this argument several times, we obtain

$$k\tilde{\lambda} + h^*(i) \leq J_k(i), \qquad k = 0, \ldots, N, \ i = 1, \ldots, n,$$

and in particular, for $k = N$,

$$\tilde{\lambda} + \frac{h^*(i)}{N} \leq \frac{1}{N} J_N(i), \qquad i = 1, \ldots, n. \tag{7.36}$$

Furthermore, equality holds in the above relation if $\mu_k(i)$ attains the minimum in Eq. (7.34) for all i and k.

Let us now take the limit as $N \to \infty$ in Eq. (7.36). The left-hand side tends to $\tilde{\lambda}$. We claim that the right-hand side tends to $J_\pi(i)$, the average cost per stage of π starting at state i. The reason is that from the definition (7.35), $J_N(i)$ is the N-stage cost of π starting at i, when the terminal cost function is h^*; when we take the limit of $(1/N)J_N(i)$, the dependence on the terminal cost function h^* disappears. Thus, by taking the limit as $N \to \infty$ in Eq. (7.36), we obtain

$$\tilde{\lambda} \leq J_\pi(i), \qquad i = 1, \ldots, n,$$

for all admissible π, with equality if π is a stationary policy μ such that $\mu(i)$ attains the minimum in Eq. (7.34) for all i and k. It follows that

$$\tilde{\lambda} = \min_\pi J_\pi(i) = \lambda^*, \qquad i = 1, \ldots, n,$$

and from Eq. (7.34), we obtain the desired Eq. (7.30).

Finally, Eqs. (7.33) and (7.34) are equivalent to Bellman's equation (7.32) for the associated stochastic shortest path problem. Since the solution to the latter equation is unique, the same is true for the solution of Eqs. (7.33) and (7.34).

(b) The proof of this part is obtained by using the argument of the proof of part (a) following Eq. (7.34).

(c) The proof of this part is obtained by specializing part (a) to the case where the constraint set at each state i is $\bar{U}(i) = \{\mu(i)\}$. **Q.E.D.**

An examination of the preceding proof shows that the unique vector h^* in Bellman's equation (7.30) for which $h^*(n) = 0$ is the optimal cost vector for the associated stochastic shortest path problem when the expected stage cost at state i is

$$g(i, u) - \lambda^*,$$

[cf. Eq. (7.32)]. Consequently, $h^*(i)$ has the interpretation of a *relative or differential cost*; it is the minimum of the difference between the expected cost to reach n from i for the first time and the cost that would be incurred if the cost per stage was the average λ^*. We note that the relation between optimal policies of the stochastic shortest path and the average cost problems is clarified in Exercise 7.15.

We finally mention that Prop. 7.4.1 can be shown under considerably weaker conditions (see Section 4.2 of Vol. II). In particular, Prop. 7.4.1 can be shown assuming that all stationary policies have a single recurrent class, even if their corresponding recurrent classes do not have state n in common. The proof, however, requires a more sophisticated use of the connection with an associated stochastic shortest path problem. Proposition 7.4.1 can also be shown assuming that for every pair of states i and j, there exists a stationary policy under which there is positive probability of reaching j starting from i. In this case, however, an associated stochastic shortest path problem cannot be defined and the corresponding connection with the average cost per stage problem cannot be made. The analysis of Chapter 4 of Vol. II relies on another connection that exists between the average cost per stage problem and the discounted cost problem, but to establish this connection and to fully explore its ramifications, a much more sophisticated analysis is required.

Example 7.4.1

Consider the average cost version of the manufacturer's problem of Example 7.3.2. Here, state 0 plays the role of the special state n in Assumption 7.4.1. Bellman's equation takes the form

$$\lambda^* + h^*(i) = \min\big[K + (1-p)h^*(0) + ph^*(1),\ ci + (1-p)h^*(i) + ph^*(i+1)\big], \tag{7.37}$$

for the states $i = 0, 1, \ldots, n-1$, and takes the form

$$\lambda^* + h^*(n) = K + (1-p)h^*(0) + ph^*(1)$$

for state n. The first expression within brackets in Eq. (7.37) corresponds to processing the i unfilled orders, while the second expression corresponds to leaving the orders unfilled for one more period. The optimal policy is to process i unfilled orders if

$$K + (1-p)h^*(0) + ph^*(1) \leq ci + (1-p)h^*(i) + ph^*(i+1).$$

If we view $h^*(i)$, $i = 1, \ldots, n$, as differential costs associated with an optimal policy, it is intuitively clear that $h^*(i)$ is monotonically nondecreasing with i [this can also be proved by interpreting $h^*(i)$ as optimal costs-to-go for the associate stochastic shortest path problem, or by using analysis based on the theory presented in Vol. II, Section 4.2]. As in Example 7.3.2, the monotonicity property of $h^*(i)$ implies that a threshold policy is optimal.

Value Iteration

The most natural version of the value iteration method for the average cost problem is simply to select arbitrarily a terminal cost function, say J_0, and to generate successively the corresponding optimal k-stage costs $J_k(i)$, $k = 1, 2, \ldots$ This can be done by executing the DP algorithm starting with J_0, that is, by using the recursion

$$J_{k+1}(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right], \qquad i = 1, \ldots, n. \tag{7.38}$$

It is natural to expect that the ratios $J_k(i)/k$ should converge to the optimal average cost per stage λ^* as $k \to \infty$, i.e.,

$$\lim_{k \to \infty} \frac{J_k(i)}{k} = \lambda^*.$$

To show this, let us define the recursion

$$J^*_{k+1}(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J^*_k(j) \right], \qquad i = 1, \ldots, n,$$

with the initial condition

$$J^*_0(i) = h^*(i), \qquad i = 1, \ldots, n,$$

where h^* is a differential cost vector satisfying Bellman's equation

$$\lambda^* + h^*(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h^*(j) \right], \qquad i = 1, \ldots, n. \tag{7.39}$$

Using this equation, it can be shown by induction that for all k we have

$$J^*_k(i) = k\lambda^* + h^*(i), \qquad i = 1, \ldots, n.$$

On the other hand, it can be seen that for all k,

$$\left| J_k(i) - J^*_k(i) \right| \leq \max_{j=1,\ldots,n} \left| J_0(j) - h^*(j) \right|, \qquad i = 1, \ldots, n.$$

The reason is that $J_k(i)$ and $J^*_k(i)$ are optimal costs for two k-stage problems that differ only in the corresponding terminal cost functions, which are J_0 and h^*, respectively. From the preceding two equations, we see that for all k,

$$\left| J_k(i) - k\lambda^* \right| \leq \max_{j=1,\ldots,n} \left| J_0(j) - h^*(j) \right| + \max_{j=1,\ldots,n} \left| h^*(j) \right|, \qquad i = 1, \ldots, n.$$

so that $J_k(i)/k$ converges to λ^* at the rate of a constant divided by k. Note that the above proof shows that $J_k(i)/k$ converges to λ^* under any conditions that guarantee that Bellman's equation (7.39) holds for some vector h^*.

The value iteration method just described is simple and straightforward, but has two drawbacks. First, since typically some of the components of J_k diverge to ∞ or $-\infty$, direct calculation of $\lim_{k\to\infty} J_k(i)/k$ is numerically cumbersome. Second, this method will not provide us with a corresponding differential cost vector h^*. We can bypass both difficulties by subtracting the same constant from all components of the vector J_k, so that the difference, call it h_k, remains bounded. In particular, we can consider the algorithm

$$h_k(i) = J_k(i) - J_k(s), \qquad i = 1, \ldots, n. \tag{7.40}$$

where s is some fixed state. By using Eq. (7.38), we then obtain

$$\begin{aligned} h_{k+1}(i) &= J_{k+1}(i) - J_{k+1}(s) \\ &= \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right] \\ &\quad - \min_{u \in U(s)} \left[g(s,u) + \sum_{j=1}^{n} p_{sj}(u) J_k(j) \right], \end{aligned}$$

from which in view of the relation $h_k(j) = J_k(j) - J_k(s)$, we have

$$\begin{aligned} h_{k+1}(i) = &\min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h_k(j) \right] \\ &- \min_{u \in U(s)} \left[g(s,u) + \sum_{j=1}^{n} p_{sj}(u) h_k(j) \right], \qquad i = 1, \ldots, n. \end{aligned} \tag{7.41}$$

The above algorithm, known as *relative value iteration*, is mathematically equivalent to the value iteration method (7.38) that generates $J_k(i)$. The iterates generated by the two methods merely differ by a constant [cf. Eq. (7.40)], and the minimization problems involved in the corresponding iterations of the two methods are mathematically equivalent. However, under Assumption 7.4.1, it can be shown that the iterates $h_k(i)$ generated by the relative value iteration method are bounded, while this is typically not true for the value iteration method.

It can be seen that if the relative value iteration (7.41) converges to some vector h, then we have

$$\lambda + h(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h(j) \right],$$

where

$$\lambda = \min_{u \in U(s)} \left[g(s,u) + \sum_{j=1}^{n} p_{sj}(u) h(j) \right].$$

By Prop. 7.4.1(b), this implies that λ is the optimal average cost per stage for all initial states, and h is an associated differential cost vector. Unfortunately, the convergence of the relative value iteration is not guaranteed under Assumption 7.4.1 (see Exercise 7.14 for a counterexample). A stronger assumption is required. It turns out, however, that there is a simple variant of the relative value iteration for which convergence is guaranteed under Assumption 7.4.1. This variant is given by

$$\begin{aligned} h_{k+1}(i) = (1-\tau)h_k(i) &+ \min_{u \in U(i)} \left[g(i,u) + \tau \sum_{j=1}^{n} p_{ij}(u) h_k(j) \right] \\ &- \min_{u \in U(s)} \left[g(s,u) + \tau \sum_{j=1}^{n} p_{sj}(u) h_k(j) \right], \quad i = 1, \ldots, n, \end{aligned} \tag{7.42}$$

where τ is a scalar such that $0 < \tau < 1$. Note that for $\tau = 1$, we obtain the relative value iteration (7.41). The convergence proof of this algorithm is somewhat complicated. It can be found in Section 4.3 of Vol. II.

Policy Iteration

It is possible to use a policy iteration algorithm for the average cost problem. This algorithm operates similar to the policy iteration algorithms of the preceding sections: given a stationary policy, we obtain an improved policy by means of a minimization process, and continue until no further improvement is possible. In particular, at the typical step of the algorithm, we have a stationary policy μ^k. We then perform a *policy evaluation* step; that is, we obtain corresponding average and differential costs λ^k and $h^k(i)$ satisfying

$$\lambda^k + h^k(i) = g\big(i, \mu^k(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^k(i)\big) h^k(j), \qquad i = 1, \ldots, n,$$

$$h^k(n) = 0.$$

We subsequently perform a *policy improvement* step; that is, we find a stationary policy μ^{k+1}, where for all i, $\mu^{k+1}(i)$ is such that

$$\begin{aligned} &g\big(i, \mu^{k+1}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu^{k+1}(i)\big) h^k(j) \\ &\qquad = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) h^k(j) \right]. \end{aligned}$$

If $\lambda^{k+1} = \lambda^k$ and $h^{k+1}(i) = h^k(i)$ for all i, the algorithm terminates; otherwise, the process is repeated with μ^{k+1} replacing μ^k.

To prove that the policy iteration algorithm terminates, it is sufficient that each iteration makes some irreversible progress towards optimality, since there are finitely many stationary policies. The type of irreversible progress that we can demonstrate is described in the following proposition, which also shows that an optimal policy is obtained upon termination.

Proposition 7.4.2 Under Assumption 7.4.1, in the policy iteration algorithm, for each k we either have

$$\lambda^{k+1} < \lambda^k$$

or else we have

$$\lambda^{k+1} = \lambda^k, \qquad h^{k+1}(i) \leq h^k(i), \quad i = 1, \ldots, n.$$

Furthermore, the algorithm terminates, and the policies μ^k and μ^{k+1} obtained upon termination are optimal.

Proof: To simplify notation, denote $\mu^k = \mu$, $\mu^{k+1} = \overline{\mu}$, $\lambda^k = \lambda$, $\lambda^{k+1} = \overline{\lambda}$, $h^k(i) = h(i)$, $h^{k+1}(i) = \overline{h}(i)$. Define for $N = 1, 2, \ldots$

$$h_N(i) = g\big(i, \overline{\mu}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\overline{\mu}(i)\big) h_{N-1}(j), \qquad i = 1, \ldots, n,$$

where

$$h_0(i) = h(i), \qquad i = 1, \ldots, n.$$

Note that $h_N(i)$ is the N-stage cost of policy $\overline{\mu}$ starting from i when the terminal cost function is h. Thus we have

$$\overline{\lambda} = J_{\overline{\mu}}(i) = \lim_{N \to \infty} \frac{1}{N} h_N(i), \qquad i = 1, \ldots, n, \tag{7.43}$$

since the contribution of the terminal cost to $(1/N)h_N(i)$ vanishes when $N \to \infty$. By the definition of $\overline{\mu}$ and Prop. 7.4.1(c), we have for all i

$$\begin{aligned} h_1(i) &= g\big(i, \overline{\mu}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\overline{\mu}(i)\big) h_0(j) \\ &\leq g\big(i, \mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big) h_0(j)\big) \\ &= \lambda + h_0(i). \end{aligned}$$

From the above equation, we also obtain

$$\begin{aligned} h_2(i) &= g\big(i,\overline{\mu}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\overline{\mu}(i)\big)h_1(j) \\ &\leq g\big(i,\overline{\mu}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\overline{\mu}(i)\big)\big(\lambda + h_0(j)\big) \\ &= \lambda + g\big(i,\overline{\mu}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\overline{\mu}(i)\big)h_0(j) \\ &\leq \lambda + g\big(i,\mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)h_0(j) \\ &= 2\lambda + h_0(i), \end{aligned}$$

and by proceeding similarly, we see that for all i and N we have

$$h_N(i) \leq N\lambda + h_0(i).$$

Thus,

$$\frac{1}{N}h_N(i) \leq \lambda + \frac{1}{N}h_0(i),$$

and by taking the limit as $N \to \infty$ and using Eq. (7.43), we obtain $\overline{\lambda} \leq \lambda$.

If $\overline{\lambda} = \lambda$, then it is seen that the iteration that produces μ^{k+1} is a policy improvement step for the associated stochastic shortest path problem with cost per stage

$$g(i,u) - \lambda.$$

Furthermore, $h(i)$ and $\overline{h}(i)$ are the optimal costs starting from i and corresponding to μ and $\overline{\mu}$, respectively, in this associated stochastic shortest path problem. Thus, by Prop. 7.2.2, we must have $\overline{h}(i) \leq h(i)$ for all i.

In view of the improvement properties just shown, no policy can be repeated without termination of the algorithm. Since there are only a finite number of policies, it follows that the algorithm will terminate. Let us now show that when the algorithm terminates with $\overline{\lambda} = \lambda$ and $\overline{h}(i) = h(i)$ for all i, the policies $\overline{\mu}$ and μ are optimal. Indeed, upon termination we have for all i

$$\begin{aligned} \lambda + h(i) &= \overline{\lambda} + \overline{h}(i) \\ &= g\big(i,\overline{\mu}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\overline{\mu}(i)\big)\overline{h}(j) \\ &= g\big(i,\overline{\mu}(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\overline{\mu}(i)\big)h(j) \\ &= \min_{u\in U(i)}\left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u)h(j)\right]. \end{aligned}$$

Thus λ and h satisfy Bellman's equation, and by Prop. 7.4.1(b), λ must be equal to the optimal average cost. Furthermore, $\overline{\mu}(i)$ attains the minimum in the right-hand side of Bellman's equation, so by Prop. 7.4.1(a), $\overline{\mu}$ is optimal. Since we also have for all i

$$\lambda + h(i) = g\big(i,\mu(i)\big) + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)h(j),$$

the same is true for μ. **Q.E.D.**

We note that policy iteration can be shown to terminate with an optimal stationary policy under less restrictive conditions than Assumption 7.4.1 (see Vol. II, Section 4.3).

7.5 SEMI-MARKOV PROBLEMS

We have considered so far problems where the cost per stage does not depend on the time required for transition from one state to the next. Such problems have a natural discrete-time representation. On the other hand, there are situations where controls are applied at discrete times but cost is continuously accumulated. Furthermore, the time between successive control choices is variable; it may be random or it may depend on the current state and the choice of control. For example, in queueing systems state transitions correspond to arrivals or departures of customers, and the corresponding times of transition are random. In this section, we discuss continuous-time, infinite horizon problems with a finite number of states. We will provide a fairly straightforward extension of our earlier infinite horizon analysis for discrete-time problems.

We assume that there are n states, denoted by $1,\ldots,n$, and that state transitions and control selections take place at discrete times, but the length of the time interval from one transition to the next is random. The state and control at any time t are denoted by $x(t)$ and $u(t)$, respectively, and stay constant between transitions. We use the following notation:

t_k: The time of occurrence of the kth transition. By convention, we denote $t_0 = 0$.

$x_k = x(t_k)$: We have $x(t) = x_k$ for $t_k \leq t < t_{k+1}$.

$u_k = u(t_k)$: We have $u(t) = u_k$ for $t_k \leq t < t_{k+1}$.

In place of transition probabilities, we have *transition distributions* $Q_{ij}(\tau,u)$, which for a given pair (i,u), specify the joint distribution of the transition interval and the next state:

$$Q_{ij}(\tau,u) = P\{t_{k+1} - t_k \leq \tau,\ x_{k+1} = j \mid x_k = i,\ u_k = u\}.$$

Note that the transition distributions specify the ordinary transition probabilities via

$$p_{ij}(u) = P\{x_{k+1} = j \mid x_k = i,\ u_k = u\} = \lim_{\tau \to \infty} Q_{ij}(\tau, u).$$

Note also that the conditional cumulative distribution function (CDF) of τ given i, j, u is

$$P\{t_{k+1} - t_k \leq \tau \mid x_k = i,\ x_{k+1} = j,\ u_k = u\} = \frac{Q_{ij}(\tau, u)}{p_{ij}(u)} \tag{7.44}$$

[assuming that $p_{ij}(u) > 0$]. Thus, $Q_{ij}(\tau, u)$ can be viewed as a "scaled CDF", i.e., a CDF multiplied by $p_{ij}(u)$ (see Fig. 7.4.2).

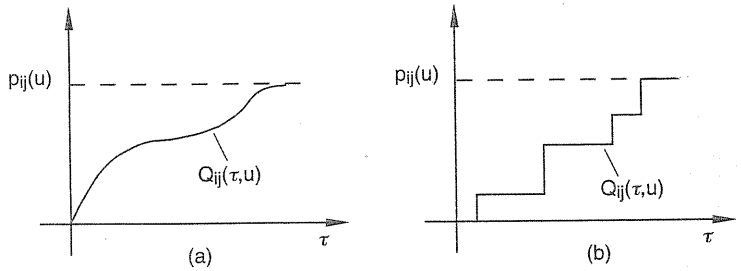


Figure 7.4.2 Illustration of the transition distributions $Q_{ij}(\tau, u)$ and the conditional CDF of τ. Figures (a) and (b) correspond to the cases where τ is a continuous and a discrete random variable, respectively.

An advantage of working with transition distributions $Q_{ij}(\tau, u)$ is that they can be used to model discrete, continuous, and mixed distributions for the transition time τ. Generally, expected values of functions of τ can be written as integrals involving the differential of Q_{ij} with respect to τ, denoted $dQ_{ij}(\tau, u)$. For example, the conditional expected value of τ given i, j, and u is written using the conditional CDF (7.44) as

$$E\{\tau \mid i, j, u\} = \int_0^\infty \tau \frac{dQ_{ij}(\tau, u)}{p_{ij}(u)}. \tag{7.45}$$

If $Q_{ij}(\tau, u)$ is continuous and piecewise differentiable with respect to τ, its partial derivative

$$q_{ij}(\tau, u) = \frac{dQ_{ij}(\tau, u)}{d\tau}$$

can be viewed as a "scaled" density function for τ. Then, $dQ_{ij}(\tau, u)$ may be replaced by $q_{ij}(\tau, u)d\tau$, and expected values of functions of τ can be written in terms of $q_{ij}(\tau, u)$. For example, Eq. (7.45) is written as

$$E\{\tau \mid i, j, u\} = \int_0^\infty \tau \frac{q_{ij}(\tau, u)}{p_{ij}(u)} d\tau.$$

If $Q_{ij}(\tau, u)$ is discontinuous and "staircase-like," then τ is a discrete random variable, and expected values of functions of τ can be written as summations.

We will assume that for each state i and control $u \in U(i)$, the expected transition time, denoted $\overline{\tau}_i(u)$, is nonzero and finite:

$$0 < \overline{\tau}_i(u) < \infty. \tag{7.46}$$

In view of Eq. (7.45), this expected transition time is given by

$$\overline{\tau}_i(u) = \sum_{j=1}^n p_{ij}(u) E\{\tau \mid i, j, u\} = \sum_{j=1}^n \int_0^\infty \tau\, dQ_{ij}(\tau, u).$$

Optimal control problems involving continuous-time Markov chains of the type described above are called *semi-Markov problems*. The reason is that, for a given policy, while at a transition time t_k the future of the system probabilistically depends only on the current state, at other times it may depend in addition on the time elapsed since the preceding transition. In fact, if we were to allow the control to depend continuously on the time t (rather than restricting the choice of control to just the transition times t_k), we would obtain a problem where there is genuine benefit to the controller for knowing the time elapsed since the preceding transition. We would then have to include this elapsed time as part of the state, and we would obtain a difficult (infinite state space) problem. This type of complication is avoided in our formulation by restricting the control to change only at the transition times t_k.

We note, however, that there is a special case where the future of the system depends only on its current state at all times, and there is no benefit in allowing the control to depend continuously on the time elapsed since the preceding transition. This is the case where the transition distributions are exponential, of the form

$$Q_{ij}(\tau, u) = p_{ij}(u)\big(1 - e^{-\nu_i(u)\tau}\big),$$

where $p_{ij}(u)$ are transition probabilities, and $\nu_i(u)$ are given positive scalars, called the *transition rates* at the corresponding states i. In this case, if the system is in state i and control u is applied, the next state will be j with probability $p_{ij}(u)$, and the time interval between the transition to state i and the transition to the next state is exponentially distributed with parameter $\nu_i(u)$; that is,

$$P\{\text{transition time interval } > \tau \mid i, u\} = e^{-\nu_i(u)\tau}.$$

The exponential distribution has the so called *memoryless property*, which in our context implies that, for any time t between the transition times t_k

and t_{k+1}, the additional time $t_{k+1} - t$ needed to effect the next transition is independent of the time $t - t_k$ that the system has been in the current state. To see this, use the following generic calculation

$$\begin{aligned} P\{\tau > r_1 + r_2 \mid \tau > r_1\} &= \frac{P\{\tau > r_1 + r_2\}}{P\{\tau > r_1\}} \\ &= \frac{e^{-\nu(r_1+r_2)}}{e^{-\nu r_1}} \\ &= e^{-\nu r_2} \\ &= P\{\tau > r_2\}, \end{aligned}$$

where $r_1 = t - t_k$, $r_2 = t_{k+1} - t$, and ν is the transition rate. Thus, when the transition distributions are exponential, the state evolves in continuous time as a Markov process, but this need not be true for a more general distribution.

We assume that for given state i and control $u \in U(i)$, the cost that is incurred in a small time interval dt is $g(i,u)dt$. Thus, we may view $g(i,u)$ as *cost per unit time*. Based on this generic cost structure, we will consider analogs of the discounted and average cost per stage problems of the preceding sections.

Discounted Problems

Here the cost function has the form

$$\lim_{T\to\infty} E\left\{\int_0^T e^{-\beta t} g\big(x(t),u(t)\big)dt\right\},$$

where β is a given positive discount parameter. Since the cost per unit time, $g\big(x(t),u(t)\big)$, remains constant between transitions, the expected cost of a single transition from state i under control u is given by

$$\begin{aligned} G(i,u) &= E\left\{\int_0^\tau e^{-\beta t} g(i,u)dt\right\} \\ &= g(i,u)\, E\left\{\int_0^\tau e^{-\beta t}dt\right\} \\ &= g(i,u)\, E_j\left\{E_\tau\left\{\int_0^\tau e^{-\beta t}dt \;\Big|\; j\right\}\right\} \\ &= g(i,u)\sum_{j=1}^n p_{ij}(u)\int_0^\infty \left(\int_0^\tau e^{-\beta t}dt\right)\frac{dQ_{ij}(\tau,u)}{p_{ij}(u)} \end{aligned}$$

or equivalently, since $\int_0^\tau e^{-\beta t}dt = (1 - e^{-\beta\tau})/\beta$,

$$G(i,u) = g(i,u)\sum_{j=1}^n \int_0^\infty \frac{1-e^{-\beta\tau}}{\beta}\, dQ_{ij}(\tau,u). \tag{7.47}$$

The cost of an admissible policy $\pi = \{\mu_0, \mu_1, \ldots\}$ starting from state i is given by

$$J_\pi(i) = \lim_{N\to\infty} \sum_{k=0}^{N-1} E\left\{\int_{t_k}^{t_{k+1}} e^{-\beta t} g\big(x_k,\mu_k(x_k)\big)dt \;\Big|\; x_0 = i\right\}.$$

This cost can be broken down into the sum of the expected cost of the first transition, which is $G\big(i,\mu_0(i)\big)$, plus the expected cost-to-go starting from the next state, discounted by the factor $e^{-\beta\tau}$, where τ is the (random) time when the first transition occurs:

$$J_\pi(i) = G\big(i,\mu_0(i)\big) + E\big\{e^{-\beta\tau} J_{\pi_1}(j) \mid x_0 = i, u_0 = \mu_0(i)\big\}. \tag{7.48}$$

The last term in the above equation can be calculated as

$$\begin{aligned} &E\big\{e^{-\beta\tau} J_{\pi_1}(j) \mid x_0 = i, u_0 = \mu_0(i)\big\} \\ &\quad = E\big\{E\{e^{-\beta\tau} \mid j\}\, J_{\pi_1}(j) \;\big|\; x_0 = i, u_0 = \mu_0(i)\big\} \\ &\quad = \sum_{j=1}^n p_{ij}\big(\mu_0(i)\big)\left(\int_0^\infty e^{-\beta\tau}\frac{dQ_{ij}\big(\tau,\mu_0(i)\big)}{p_{ij}\big(\mu_0(i)\big)}\right) J_{\pi_1}(j) \\ &\quad = \sum_{j=1}^n m_{ij}\big(\mu_0(i)\big) J_{\pi_1}(j), \end{aligned}$$

where for any $u \in U(i)$, $m_{ij}(u)$ is given by

$$m_{ij}(u) = \int_0^\infty e^{-\beta\tau} dQ_{ij}(\tau,u). \tag{7.49}$$

Thus, combining Eqs. (7.47)-(7.49), we see that $J_\pi(i)$ can be written as

$$J_\pi(i) = G\big(i,\mu_0(i)\big) + \sum_{j=1}^n m_{ij}\big(\mu_0(i)\big) J_{\pi_1}(j), \tag{7.50}$$

which is similar to the corresponding equation for discounted discrete-time problems [$m_{ij}\big(\mu_0(i)\big)$ replaces $\alpha p_{ij}\big(\mu_0(i)\big)$].

In analogy with the discrete-time case, we may associate Eq. (7.50) with a stochastic shortest path problem involving an artificial termination state t. Under control u, from state i the system moves to state j with probability $m_{ij}(u)$ and to the termination state t with probability

$$1 - \sum_{j=1}^n m_{ij}(u).$$

The assumption of a positive expected transition time [cf. Eq. (7.46)] implies that

$$\sum_{j=1}^{n} m_{ij}(u) < 1, \qquad \text{for all } i, u \in U(i),$$

so that the Assumption 7.2.1, which is required for the validity of our stochastic shortest path analysis of Section 7.2, is satisfied. By using an essentially identical approach to the one of Section 7.3, we can derive analogs of all the discounted cost results of Prop. 7.3.1. In particular, the optimal cost function J^* is the unique solution of Bellman's equation

$$J^*(i) = \min_{u \in U(i)} \left[G(i,u) + \sum_{j=1}^{n} m_{ij}(u) J^*(j) \right].$$

In addition, there are analogs of the computational methods of Section 7.3, including value iteration, policy iteration, and linear programming. What is happening here is that essentially we have the equivalent of a discrete-time discounted problem where the discount factor depends on i and u.

We finally note that in some problems, in addition to the cost per unit time g, there is an extra (instantaneous) one-stage cost $\hat{g}(i,u)$ that is incurred at the time the control u is chosen at state i, and is independent of the length of the transition interval. In this case, Bellman's equation takes the form

$$J^*(i) = \min_{u \in U(i)} \left[\hat{g}(i,u) + G(i,u) + \sum_{j=1}^{n} m_{ij}(u) J^*(j) \right], \tag{7.51}$$

and the various computational methods are appropriately adjusted. Another problem variation arises when the cost g depends on the next state j. Here, once the system goes into state i, a control $u \in U(i)$ is selected, the next state is determined to be j with probability $p_{ij}(u)$, and the cost incurred is $g(i,u,j)$. In this case, $G(i,u)$ should be defined by

$$G(i,u) = \sum_{j=1}^{n} \int_0^\infty g(i,u,j) \frac{1 - e^{-\beta\tau}}{\beta} \, dQ_{ij}(\tau, u),$$

[cf. Eq. (7.47)] and the preceding development goes through without modification.

Example 7.5.1

Consider the manufacturer's problem of Example 7.3.2, with the only difference that the times between the arrivals of successive orders are uniformly distributed in a given interval $[0, \tau_{\max}]$, and c is the cost per unit time of an unfilled order. Let F and NF denote the choices of filling and not filling the orders, respectively. The transition distributions are

$$Q_{ij}(\tau, F) = \begin{cases} \min\left[1, \frac{\tau}{\tau_{\max}}\right] & \text{if } j = 1, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$Q_{ij}(\tau, NF) = \begin{cases} \min\left[1, \frac{\tau}{\tau_{\max}}\right] & \text{if } j = i+1, \\ 0 & \text{otherwise.} \end{cases}$$

The one-stage expected cost G of Eq. (7.47) is given by

$$G(i,F) = 0, \qquad G(i,NF) = \gamma c i,$$

where

$$\gamma = \int_0^{\tau_{\max}} \frac{1 - e^{-\beta\tau}}{\beta \tau_{\max}} d\tau.$$

The scalars m_{ij} of Eq. (7.49) that are nonzero are

$$m_{i1}(F) = m_{i(i+1)}(NF) = \alpha,$$

where

$$\alpha = \int_0^{\tau_{\max}} \frac{e^{-\beta\tau}}{\tau_{\max}} d\tau = \frac{1 - e^{-\beta\tau_{\max}}}{\beta \tau_{\max}}.$$

Bellman's equation has the form [cf. Eq. (7.51)]

$$J(i) = \min\big[K + \alpha J(1), \gamma c i + \alpha J(i+1)\big], \qquad i = 1, 2, \ldots.$$

As in Example 7.3.2, we can conclude that there exists a threshold i^* such that it is optimal to fill the orders if and only if their number i exceeds i^*.

Average Cost Problems

A natural cost function for the continuous-time average cost problem would be

$$\lim_{T \to \infty} \frac{1}{T} E\left\{ \int_0^T g\big(x(t), u(t)\big) dt \right\}. \tag{7.52}$$

However, we will use instead the cost function

$$\lim_{N \to \infty} \frac{1}{E\{t_N\}} E\left\{ \int_0^{t_N} g\big(x(t), u(t)\big) dt \right\}, \tag{7.53}$$

where t_N is the completion time of the Nth transition. This cost function is also reasonable and turns out to be analytically convenient. We note, however, that the cost functions (7.52) and (7.53) are equivalent under the conditions of the subsequent analysis, although a rigorous justification of

this is beyond our scope (see Ross [Ros70], p. 52 and p. 160 for related discussion).

For each pair (i, u), we denote by $G(i, u)$ the one-stage expected cost corresponding to state i and control u. We have

$$G(i, u) = g(i, u)\overline{\tau}_i(u),$$

where $\overline{\tau}_i(u)$ is the expected value of the transition time corresponding to (i, u). [If the cost per unit time g depends on the next state j, the expected transition cost $G(i, u)$ should be defined by

$$G(i, u) = \sum_{j=1}^{n} \int_0^\infty g(i, u, j)\, \tau\, dQ_{ij}(\tau, u),$$

and the following analysis and results go through without modification.] The cost function of an admissible policy $\pi = \{\mu_0, \mu_1, \ldots\}$ is given by

$$J_\pi(i) = \lim_{N\to\infty} \frac{1}{E\{t_N \mid x_0 = i,\ \pi\}} E\left\{ \sum_{k=0}^{N-1} \int_{t_k}^{t_{k+1}} g\big(x_k, \mu_k(x_k)\big)\, dt \;\middle|\; x_0 = i \right\}.$$

We will see that the character of the solution of the problem is determined by the structure of the *embedded Markov chain*, which is the controlled discrete-time Markov chain whose transition probabilities are

$$p_{ij}(u) = \lim_{\tau\to\infty} Q_{ij}(\tau, u).$$

In particular, assuming that the embedded Markov chain satisfies Assumption 7.4.1 of Section 7.4, we can show that the costs $J^*(i)$ are independent of i.

It turns out that Bellman' s equation for average cost semi-Markov problems takes the form

$$h(i) = \min_{u\in U(i)} \left[G(i, u) - \lambda\overline{\tau}_i(u) + \sum_{j=1}^{n} p_{ij}(u)h(j) \right].$$

As a special case, when $\overline{\tau}_i(u) = 1$ for all (i, u), we obtain the corresponding Bellman's equation for discrete-time problems, given in Section 7.4. We motivate the above form of Bellman's equation with the stochastic shortest path argument that we used in Section 7.4. We consider a sequence of generated states, and divide it into cycles marked by successive visits to the special state n. Each of the cycles can be viewed as a state trajectory of a corresponding stochastic shortest path problem with the termination state being essentially n, as in Section 7.4.

We next conjecture that the average cost problem is equivalent to the *minimum cycle cost problem* of finding a stationary policy μ that minimizes the average cycle cost

$$\frac{C_{nn}(\mu)}{T_{nn}(\mu)},$$

where for a fixed μ,

$C_{nn}(\mu)$: expected cost starting from n up to the first return to n,

$T_{nn}(\mu)$: expected time to return to n starting from n.

An intuitive conjecture is that the optimal average cost λ^* is equal to the optimal cycle cost, so it satisfies

$$C_{nn}(\mu) - N_{nn}(\mu)\lambda^* \geq 0, \qquad \text{for all } \mu, \tag{7.54}$$

with equality holding if μ is optimal. Thus, to attain an optimal μ, we must minimize over μ the expression $C_{nn}(\mu) - T_{nn}(\mu)\lambda^*$, which is the expected cost of μ starting from n in the associated stochastic shortest path problem with stage costs

$$G(i, u) - \lambda^*\overline{\tau}_i\big(\mu(i)\big), \qquad i = 1, \ldots, n.$$

Let us denote by $h^*(i)$ the optimal cost of this stochastic shortest path problem when starting at state i. Then $h^*(1), \ldots, h^*(n)$ solve uniquely the corresponding Bellman's equation, which has the form

$$h^*(i) = \min_{u\in U(i)} \left[G(i, u) - \lambda^*\overline{\tau}_i(u) + \sum_{j=1}^{n-1} p_{ij}(u)h^*(j) \right], \qquad i = 1, \ldots, n. \tag{7.55}$$

If μ^* is an optimal stationary policy, then this policy must satisfy

$$C_{nn}(\mu^*) - N_{nn}(\mu^*)\lambda^* = 0,$$

and from Eq. (7.54), this policy must also be optimal for the associated stochastic shortest path problem. Thus, we must have

$$h^*(n) = C_{nn}(\mu^*) - N_{nn}(\mu^*)\lambda^* = 0.$$

By using this equation, we can now write Bellman's equation (7.55) as

$$h^*(i) = \min_{u\in U(i)} \left[G(i, u) - \lambda^*\overline{\tau}_i(u) + \sum_{j=1}^{n} p_{ij}(u)h^*(j) \right], \quad i = 1, \ldots, n. \tag{7.56}$$

If there is an "instantaneous" one-stage cost $\hat{g}(i, u)$, the term $G(i, u)$ should be replaced by $\hat{g}(i, u) + G(i, u)$ in this equation.

Given the correct form of Bellman's equation and the connection with the associated stochastic shortest path problem, it is possible to essentially repeat the proof of Prop. 7.4.1 and to obtain analogous results to those for the discrete-time case.

Example 7.5.2

Consider the average cost version of the manufacturer's problem of Example 7.5.1. Here we have

$$\overline{\tau}_i(F) = \overline{\tau}_i(NF) = \frac{\overline{\tau}_{\max}}{2},$$

$$G(i,F) = K, \qquad G(i,NF) = \frac{ci\tau_{\max}}{2},$$

where F and NF denote the decisions to fill and not fill the orders, respectively. Bellman's equation (7.56) takes the form

$$h^*(i) = \min\left[K - \lambda^*\frac{\overline{\tau}_{\max}}{2} + h^*(1),\, ci\frac{\overline{\tau}_{\max}}{2} - \lambda^*\frac{\overline{\tau}_{\max}}{2} + h^*(i+1)\right].$$

We leave it as an exercise for the reader to show that there exists a threshold i^* such that it is optimal to fill the orders if and only if i exceeds i^*.

Example 7.5.3 [LiR71]

Consider a person providing a certain type of service to customers. Potential customers arrive according to a Poisson process with rate r; that is, the customer's interarrival times are independent and exponentially distributed with parameter r. Each customer offers one of n pairs (m_i, T_i), $i = 1, \ldots, n$, where m_i is the amount of money offered for the service and T_i is the average amount of time that will be required to perform the service. Successive offers are independent and offer (m_i, T_i) occurs with probability p_i, where $\sum_{i=1}^n p_i = 1$. An offer may be rejected, in which case the customer leaves, or may be accepted in which case all offers that arrive while the customer is being served are lost. The problem is to determine the acceptance-rejection policy that maximizes the service provider's average income per unit time.

Let us denote by i the state corresponding to the offer (m_i, T_i), and let A and R denote the accept and reject decision, respectively. We have

$$\overline{\tau}_i(A) = T_i + \frac{1}{r}, \qquad \overline{\tau}_i(R) = \frac{1}{r}, \qquad G(i,A) = -m_i, \qquad G(i,R) = 0,$$

$$p_{ij}(A) = p_{ij}(R) = p_j.$$

Bellman's equation is given by

$$h^*(i) = \min\left[-m_i - \lambda^*\left(T_i + \frac{1}{r}\right) + \sum_{j=1}^n p_j h^*(j),\, -\lambda^*\frac{1}{r} + \sum_{j=1}^n p_j h^*(j)\right].$$

It follows that an optimal policy is to

$$\text{accept offer } (i,T_i) \qquad \text{if and only if} \qquad \frac{m_i}{T_i} \geq -\lambda^*,$$

where $-\lambda^*$ is the optimal average income per unit time.

7.6 NOTES, SOURCES, AND EXERCISES

This chapter is only an introduction to infinite horizon problems. There is an extensive theory for these problems with interesting mathematical and computational content. Volume II provides a comprehensive treatment and gives many references to the literature.

The presentation in this chapter is original in that it uses the stochastic shortest path problem as the starting point for the analysis of the other problems. This line of development not only explains intuitively the connections between the various types of problems, but also leads to new solution methods. For example, an alternative value iteration algorithm for the average cost problem, based on the connection with the stochastic shortest path problem, is given in Bertsekas [Ber98b], and in Section 4.3 of Vol. II. On the other hand, there are also important results for undiscounted and average cost problems that cannot be obtained through the connection with the stochastic shortest path problem. Some of these alternative lines of analysis are pursued in Vol. II.

Semi-Markov problems were introduced by Jewell [Jew63] and were also discussed by Ross [Ros70]. Volume II contains a broader exposition of Semi-Markov problems, and applications to queueing and related systems.

EXERCISES

7.1

A tennis player has a Fast serve and a Slow serve, denoted F and S, respectively. The probability of F (or S) landing in bounds is p_F (or p_S, respectively). The probability of winning the point assuming the serve landed in bounds is q_F (or q_S, respectively). We assume that $p_F < p_S$ and $q_F > q_S$. The problem is to find the serve to be used at each possible scoring situation during a single game in order to maximize the probability of winning that game.

(a) Formulate this as a stochastic shortest path problem, argue that Assumption 7.2.1 of Section 7.2 holds, and write Bellman's equation.

(b) Computer assignment: Assume that $q_F = 0.6$, $q_S = 0.4$, and $p_S = 0.95$. Use value iteration to calculate and plot (in increments of 0.05) the probability of the server winning a game with optimal serve selection as a function of p_F.

7.2

A quarterback can choose between running and passing the ball on any given play. The number of yards gained by running is integer and is Poisson distributed with parameter λ_r. A pass is incomplete with probability p, is intercepted with probability q, and is completed with probability $1-p-q$. When completed, a pass gains an integer number of yards that is Poisson distributed with parameter λ_p. We assume that the probability of scoring a touchdown on a single play starting i yards from the goal is equal to the probability of gaining a number of yards greater than or equal to i. We assume also that yardage cannot be lost on any play and that there are no penalties. The ball is turned over to the other team on a fourth down or when an interception occurs.

(a) Formulate the problem as a stochastic shortest path problem, argue that Assumption 7.2.1 of Section 7.2 holds, and write Bellman's equation.

(b) Computer assignment: Use value iteration to compute the quarterback's play-selection policy that maximizes the probability of scoring a touchdown on any single drive for $\lambda_r = 3$, $\lambda_p = 10$, $p = 0.4$, and $q = 0.05$.

7.3

A computer manufacturer can be in one of two states. In state 1 his product sells well, while in state 2 his product sells poorly. While in state 1 he can advertise his product in which case the one-stage reward is 4 units, and the transition probabilities are $p_{11} = 0.8$ and $p_{12} = 0.2$. If in state 1, he does not advertise, the reward is 6 units and the transition probabilities are $p_{11} = p_{12} = 0.5$. While in state 2, he can do research to improve his product, in which case the one-stage reward is -5 units, and the transition probabilities are $p_{21} = 0.7$ and $p_{22} = 0.3$. If in state 2 he does not do research, the reward is -3, and the transition probabilities are $p_{21} = 0.4$ and $p_{22} = 0.6$. Consider the infinite horizon, discounted version of this problem.

(a) Show that when the discount factor α is sufficiently small, the computer manufacturer should follow the "shortsighted" policy of not advertising (not doing research) while in state 1 (state 2). By contrast, when α is sufficiently close to unity, he should follow the "farsighted" policy of advertising (doing research) while in state 1 (state 2).

(b) For $\alpha = 0.9$ calculate the optimal policy using policy iteration.

(c) For $\alpha = 0.99$, use a computer to solve the problem by value iteration, with and without the error bounds (7.23).

7.4

An energetic salesman works every day of the week. He can work in only one of two towns A and B on each day. For each day he works in town A (or B) his expected reward is r_A (or r_B, respectively). The cost for changing towns is c. Assume that $c > r_A > r_B$ and that there is a discount factor $\alpha < 1$.

(a) Show that for α sufficiently small, the optimal policy is to stay in the town he starts in, and that for α sufficiently close to 1, the optimal policy is to move to town A (if not starting there) and stay in A for all subsequent times.

(b) Solve the problem for $c = 3$, $r_A = 2$, $r_B = 1$, and $\alpha = 0.9$ using policy iteration.

(c) Use a computer to solve the problem of part (b) by value iteration, with and without the error bounds (7.23).

7.5

A person has an umbrella that she takes from home to office and vice versa. There is a probability p of rain at the time she leaves home or office independently of earlier weather. If the umbrella is in the place where she is and it rains, she takes the umbrella to go to the other place (this involves no cost). If there is no umbrella and it rains, there is a cost W for getting wet. If the umbrella is in the place where she is but it does not rain, she may take the umbrella to go to the other place (this involves an inconvenience cost V) or she may leave the umbrella behind (this involves no cost). Costs are discounted at a factor $\alpha < 1$.

(a) Formulate this as an infinite horizon total cost discounted problem. *Hint*: Try to use as few states as possible.

(b) Characterize the optimal policy as best as you can.

7.6

For the tennis player's problem (Exercise 7.1), show that it is optimal (regardless of score) to use F on both serves if

$$(p_F q_F)/(p_S q_S) > 1,$$

to use S on both serves if

$$(p_F q_F)/(p_S q_S) < 1 + p_F - p_S,$$

and to use F on the first serve and S on the second otherwise.

7.7

Consider the value iteration method for the Example 7.3.2:

$$J_{k+1}(i) = \min\big[K + \alpha(1-p)J_k(0) + \alpha p J_k(1),$$
$$ci + \alpha(1-p)J_k(i) + \alpha p J_k(i+1)\big], \qquad i = 0, 1, \ldots, n-1,$$
$$J_{k+1}(n) = K + \alpha(1-p)J_k(0) + \alpha p J_k(1),$$

where $J_0(i) = 0$ for all i. Show by induction that $J_k(i)$ is monotonically nondecreasing in i.

7.8 (WWW)

Consider the policy iteration algorithm for the problem of Example 7.3.2.

(a) Show that if we start the algorithm with a threshold policy, every subsequently generated policy will be a threshold policy. *Note*: This requires a careful argument.

(b) Carry out the algorithm for the case $c = 1$, $K = 5$, $n = 10$, $p = 0.5$, $\alpha = 0.9$, and an initial policy that always processes the unfilled orders.

7.9

Solve the average cost version ($\alpha = 1$) of the computer manufacturer's problem by using value iteration and by using policy iteration (Exercise 7.3).

7.10

An unemployed worker receives a job offer at each time period, which she may accept or reject. The offered salary takes one of n possible values $w^1, \ldots, w^n$ with given probabilities, independently of preceding offers. If she accepts the offer, she must keep the job for the rest of her life at the same salary level. If she rejects the offer, she receives unemployment compensation c for the current period and is eligible to accept future offers. Assume that income is discounted by a factor $\alpha < 1$.

(a) Show that there is a threshold $\overline{w}$ such that it is optimal to accept an offer if and only if its salary is larger than $\overline{w}$, and characterize $\overline{w}$.

(b) Consider the variant of the problem where there is a given probability p_i that the worker will be fired from her job at any one period if her salary is w^i. Show that the result of part (a) holds in the case where p_i is the same for all i. Analyze the case where p_i depends on i.

7.11

Do part (b) of Exercise 7.10 for the case where income is not discounted and the worker maximizes her average income per period.

7.12 (WWW)

Show that one can always take $m = n$ in Assumption 7.2.1. *Hint*: For any π and i, let $S_k(i)$ be the set of states that are reachable with positive probability from i under π in k stages or less. Show that under Assumption 7.2.1, we cannot have $S_k(i) = S_{k+1}(i)$ while $t \neq S_k(i)$.

7.13

Show the error bounds (7.17). These bounds constitute a generalization to the stochastic shortest path problem of the bounds (7.23) for the discounted problem, which have a long history, starting with the work of McQueen [McQ66]. *Hint*: Complete the details of the following argument. Let $\mu^k(i)$ attain the minimum in the value iteration (7.16) for all i. Then, in vector form, we have

$$J_{k+1} = g_k + P_k J_k,$$

where J_k and g_k are the vectors with components $J_k(i)$, $i = 1, \ldots, n$, and $g_k\big(i, \mu^k(i)\big)$, $i = 1, \ldots, n$, respectively, and P_k is the matrix whose components are the transition probabilities $p_{ij}\big(\mu^k(i)\big)$. Also from Bellman's equation, we have

$$J^* \leq g_k + P_k J^*,$$

where the vector inequality above is meant to hold separately for each component. Let $e = (1, \ldots, 1)'$. Using the above two relations, we have

$$J^* - J_k \leq J^* - J_{k+1} + \overline{c}_k e \leq P_k(J^* - J_k) + \overline{c}_k e. \tag{7.57}$$

Multiplying this relation with P_k and adding $\overline{c}_k e$, we obtain

$$P_k(J^* - J_k) + \overline{c}_k e \leq P_k^2(J^* - J_k) + \overline{c}_k(I + P_k)e.$$

Similarly continuing, we have for all $r \geq 1$

$$J^* - J_{k+1} + \overline{c}_k e \leq P_k^r(J^* - J_k) + \overline{c}_k(I + P_k + \cdots + P_k^{r-1})e.$$

For $s = 1, 2, \ldots$, the ith component of the vector $P_k^s e$ is equal to the probability $P\{x_s \neq t \mid x_0 = i, \mu^k\}$ that t has not been reached after s stages starting from i and using the stationary policy μ^k. Thus, Assumption 7.2.1 implies that $\lim_{r\to\infty} P_k^r = 0$, while we have

$$\lim_{r\to\infty} (I + P_k + \cdots + P_k^{r-1})e = N^k,$$

where N^k is the vector $\big(N^k(1), \ldots, N^k(n)\big)'$. Combining the above two relations, we obtain

$$J^* \leq J_{k+1} + \overline{c}_k(N^k - e),$$

proving the desired upper bound.

The lower bound is proved similarly, by using in place of μ^k, an optimal stationary policy μ^*. In particular, in place of Eq. (7.57), we can show that

$$J_k - J^* \leq J_{k+1} - J^* - \underline{c}_k e \leq P^*(J_k - J^*) - \underline{c}_k e,$$

where P^* is the matrix with elements $p_{ij}\big(\mu^*(i)\big)$. We similarly obtain for all $r \geq 1$

$$J_{k+1} - J^* - \underline{c}_k e \leq (P^*)^r(J_k - J^*) - \underline{c}_k\big(I + P^* + \cdots + (P^*)^{r-1}\big)e,$$

from which $J_{k+1} + \underline{c}_k(N^* - e) \leq J^*$, where N^* is the vector $\big(N^*(1), \ldots, N^*(n)\big)'$.

7.14

Apply the relative value iteration algorithm (7.41) for the case where there are two states and only one control per state. The transition probabilities are $p_{11} = \epsilon$, $p_{12} = 1 - \epsilon$, $p_{21} = 1 - \epsilon$, and $p_{22} = \epsilon$, where $0 \leq \epsilon < 1$. Show that if $0 < \epsilon$ the algorithm converges, but if $\epsilon = 0$, the algorithm may not converge. Show also that the variation (7.42) converges when $\epsilon = 0$.

7.15

Consider the average cost problem and its associated stochastic shortest path problem when the expected cost incurred at state i is $g(i,u) - \lambda^*$.

(a) Use Prop. 7.2.1(d) and Prop. 7.4.1(a) to show that if a stationary policy is optimal for the latter problem it is also optimal for the former.

(b) Show by example that the reverse of part (a) need not be true.

7.16

Consider a problem of operating a machine that can be in any one of n states, denoted $1, 2, \ldots, n$. We denote by $g(i)$ the operating cost per period when the machine is in state i, and we assume that

$$g(1) \leq g(2) \leq \cdots \leq g(n).$$

The implication here is that state i is better than state $i+1$, and state 1 corresponds to a machine being in the best condition. The transition probabilities during one period of operation satisfy

$$p_{i(i+1)} > 0 \qquad \text{if } i < n,$$

$$p_{ij} = 0 \qquad \text{if } j \neq i,\ j \neq i+1.$$

We assume that at the start of each period we know the state of the machine and we must choose one of the following two options:

(1) Let the machine operate one more period in the state it currently is.

(2) Repair the machine and bring it to the best state 1 at a cost R.

We assume that the machine, once repaired, is guaranteed to stay in state 1 for one period. In subsequent periods, it may deteriorate to states $j > 1$.

(a) Assume an infinite horizon and a discount factor $\alpha \in (0,1)$, and show that there is an optimal policy which is a threshold policy; that is, it takes the form

$$\text{replace if and only if } i \geq i^*,$$

where i^* is some integer.

(b) Show that the policy iteration method, when started with a threshold policy, generates a sequence of threshold policies.

7.17

Consider a person providing a certain type of service to customers. The person receives at the beginning of each time period with probability p_i a proposal by a customer of type i, where $i = 1, 2, \ldots, n$, who offers an amount of money M_i. We assume that $\sum_{i=1}^{n} p_i = 1$. The person may reject the offer, in which case the customer leaves and the person remains idle during that period, or the person may accept the offer in which case the person spends some random amount of time with that customer. In particular, we assume that the probability that the type i customer will leave after k periods ($k = 1, 2, \ldots$), given that the customer has already stayed with the person for $k-1$ periods is a given scalar $\beta_i \in (0,1)$. The problem is to determine an acceptance-rejection policy that maximizes

$$\lim_{N\to\infty} \frac{1}{N}\{\text{Expected payment over } N \text{ periods}\}.$$

(a) Formulate the person's problem as an average cost per stage problem, and show that the optimal cost is independent of the initial state.

(b) Show that there exists a scalar λ and an optimal policy that accepts the offer of a type i customer if and only if

$$\lambda \leq \frac{M_i}{T_i},$$

where T_i is the expected time spent with the type i customer.

7.18

A person has an asset to sell for which she receives offers that take one of n values s_j, $j = 1, \ldots, n$. The times between successive offers are random, identically distributed, and independent of preceding times. Let $Q_j(\tau)$ be the probability that the time between successive offers is less or equal to τ and the next offer is s_j. Find the offer acceptance policy that maximizes $E\{\alpha^T s\}$, where T is the time of sale, s is the sale price, and $\alpha \in (0,1)$ is a discount factor.

7.19

An unemployed worker receives job offers, which she may accept or reject. The times between successive offers are independent and exponentially distributed with parameter r. The offered salary (per unit time) takes one of n possible values w_i, $i = 1, \ldots, n$, with given probabilities p_i, independently of preceding offers. If she accepts an offer at salary w_i, she keeps the job for a random amount of time that has expected value t_i. If she rejects the offer, she receives unemployment compensation c (per unit time) and is eligible to accept future offers. Solve the problem of maximizing the worker's average income per unit time.

7.20

Consider a computing system where the interarrival times of the jobs are independent and exponentially distributed with parameter r. A job may be rejected, in which case the job is lost, or may be accepted in which case all jobs that arrive while the job is being processed are lost. There are n types of jobs. Each arriving job is of type i with probability p_i, independently of earlier jobs, and if processed, is worth a fixed positive benefit b_i $(i = 1, \ldots, n)$. Jobs of type i require an average amount of time T_i to complete processing. The problem is to determine the acceptance-rejection policy that maximizes the system's average benefit per unit time.

(a) Argue that the analysis of Example 7.5.3 applies for this problem.

(b) Calculate the optimal average benefit per unit time λ^* in terms of the given quantities for the case where there are only two job types.

(c) Suppose that the time to process a job of type i is exponentially distributed with mean T_i. Assume further that the system can process up to a given number $m > 1$ of jobs simultaneously (rather than just one). Formulate the problem as an average benefit per unit time semi-Markov problem, and write Bellman's equation for the case where $m = 2$. Why do we need the exponential distribution assumption?

7.21

Formulate a semi-Markov version of the stochastic shortest path problem of Section 7.2. The cost function has the form

$$\lim_{T\to\infty} E\left\{\int_0^T g\big(x(t),u(t)\big)dt\right\},$$

and there is a cost-free and absorbing state. Use the transition distributions Q_{ij} to formulate an assumption that is analogous to Assumption 7.2.1. Under this assumption, state and justify a result that parallels Prop. 7.2.1.

7.22

A treasure hunter has obtained a lease to search a site that contains n treasures, and wants to find a searching policy that maximizes his expected gain over an infinite number of days. At each day, knowing the current number of treasures not yet found, he may decide to continue searching for more treasures at a cost c per day, or to permanently stop searching. If he searches on a day when there are i treasures on the site, he finds $m \in [0, i]$ treasures with given probability $p(m \mid i)$, where we assume that $p(0 \mid i) < 1$ for all $i \geq 1$, and that the expected number of treasures found,

$$r(i) = \sum_{m=0}^{i} mp(m \mid i),$$

decreases monotonically with i. Each found treasure is worth 1 unit.

(a) Formulate the problem as an infinite horizon DP problem.

(b) Write Bellman's equation. How do you know that this equation holds and has a unique solution?

(c) Start policy iteration with the policy that never searches. How many policy iterations does it take to find an optimal policy, and what is that optimal policy?

7.23

The latest slot machine model has three arms, labeled 1, 2, and 3. A single play with arm i, where $i = 1, 2, 3$, costs c_i dollars, and has two possible outcomes: a "win," which occurs with probability p_i, and a "loss," which occurs with probability $1 - p_i$. The slot machine pays you m dollars each time you complete a sequence of three successive "wins," with each win obtained using a different arm.

(a) Consider the problem of finding the arm-playing order that minimizes the expected cost if you are restricted to stop at the first time the machine pays you. Formulate this problem as a stochastic shortest path problem where arm-playing orders are identified with stationary policies, and write Bellman's equation for each stationary policy.

(b) Show that the expected cost of the arm-playing order ABC is

$$\frac{c_A + p_A c_B + p_A p_B c_C - p_A p_B p_C m}{1 - p_A p_B p_C}.$$

Show that it is optimal to play the arms in order of decreasing $c_i/(1 - p_i)$.

(c) Consider the problem of finding the arm-playing order that minimizes the average expected cost per play, assuming you play infinitely many times. Formulate this problem as an average cost per stage problem, where arm-playing orders are identified with stationary policies, and write Bellman's equation for each stationary policy.

(d) Show that the expected cost per play of the arm-playing order ABC is

$$\frac{c_A + p_A c_B + p_A p_B c_C - p_A p_B p_C m}{1 + p_A p_B}.$$

Is it possible that the optimal playing order is different than the one of part (b)? If this is so, how do you explain it?

7.24

A person has a house that he rents at a fixed amount R per time period. At the beginning of each period k, the person receives an offer w_k to sell the house. The amount w_k takes one of m given values $w^1, \ldots, w^m$, with corresponding positive probabilities $q^1, \ldots, q^m$, independently of preceding offers. The person, at the

beginning of each period, must decide whether to accept the current offer or to decline the offer and continue to rent the house.

Upon selling the house, the sale amount, call it w, is immediately reinvested in some way so that it yields at time k a random amount $y_k w$, where y_k takes one of s given values $y^1, \ldots, y^s$. The value of y_k evolves according to a Markov chain with a single recurrent class and given transition probabilities

$$p_{ij} = P(y_{k+1} = y^j \mid y_k = y^i), \qquad i, j = 1, \ldots, s.$$

(a) Suppose that at time k the house is sold when y_k is equal to y^i. Let

$$\overline{y}(i) = \lim_{N \to \infty} \frac{1}{N} E\left\{ \sum_{l=k}^{k+N-1} y_l \;\middle|\; y_k = y^i \right\}$$

be the average future yield per unit time. Show that $\overline{y}(i)$ is equal to a common value $\overline{y}$, independent of i, and derive a Bellman-type equation for this value.

(b) Suppose that the person's objective is to maximize the average monetary benefit per time period. Argue that an optimal stationary policy is to wait until the maximum possible offer $\overline{w} = \max\{w^1, \ldots, w^m\}$ is received and then sell the house, assuming that $R/\overline{y} \leq \overline{w}$. Given this result, discuss whether the average cost formulation is satisfactory for this problem.

(c) Suppose that the person's objective is to maximize the total discounted monetary benefit over an infinite horizon, with a discount factor $\alpha < 1$. Show that for each $i = 1, \ldots, s$, there is a threshold $t(i)$ such that it is optimal to sell the house at period k when $y_k = y^i$ and the current offer is larger than $t(i)$.

7.25

You have just bought your first car, which raises the issue of where to park it. At the beginning of each day you may either park it in a garage, which costs G per day, or on the street for free. However, in the latter case, you run the risk of getting a parking ticket, which costs T, with probability p_j, where j is the number of consecutive days that the car has been parked on the street (e.g., on the first day you park on the street, you have probability p_1 of getting a ticket, on the second successive day you park on the street, you have probability p_2, etc). Assume that p_j is monotonically nondecreasing in j, and that you may receive at most one ticket per day when parked on the street. Assume also that there exists an integer m such that $p_m T > G$.

(a) Formulate this as an infinite horizon discounted cost problem with finite state space and write the corresponding Bellman's equation.

(b) Characterize as best as you can the optimal policy.

(c) Let n be the total number of states. Show how to use policy iteration so that it terminates after no more than n iterations. *Hint*: Use threshold policies as in Problem 7.8.

(d) Formulate the infinite horizon average cost version of this problem with finite state space and write the corresponding Bellman's equation. State an assumption under which Bellman's equation holds.

7.26

An engineer has invented a better mouse trap and is interested in selling it for the right price. At the beginning of each period, he receives a sale offer that takes one of the values $s_1, \ldots, s_n$ with corresponding probabilities $p_1, \ldots, p_n$, independently of prior offers. If he accepts the offer he retires from engineering. If he refuses the offer, he may accept subsequent offers but he also runs the risk that a competitor will invent an even better mouse trap, rendering his own unsaleable; this happens with probability $\beta > 0$ at each time period, independently of earlier time periods. While he is overtaken by the competitor, at each time period, he may choose to retire from engineering, or he may choose to invest an amount $v \geq 0$, in which case he has a probability γ to improve his mouse trap, overtake his competitor, and start receiving offers as earlier. The problem is to determine the engineer's strategy to maximize his discounted expected payoff (minus investment cost), assuming a discount factor $\alpha < 1$.

(a) Formulate the problem as an infinite horizon discounted cost problem and write the corresponding Bellman's equation.

(b) Characterize as best as you can an optimal policy.

(c) Assume that there is no discount factor. Does the problem make sense as an average cost per stage problem?

(d) Assume that there is no discount factor and that the investment cost v is equal to 0. Does the problem make sense as a stochastic shortest path problem, and what is then the optimal policy?

7.27 (Eliminating Self-Transitions)

Consider a stochastic shortest path problem (SSP) with termination state t, the nontermination states $1, \ldots, n$, transition probabilities $p_{ij}(u)$, and expected costs per stage $g(i, u)$. Let Assumption 7.2.1 hold.

(a) Modify the costs and transition probabilities as follows:

$$\tilde{g}(i, u) = \frac{g(i, u)}{1 - p_{ii}(u)}, \qquad i = 1, \ldots, n, \; u \in U(i),$$

$$\tilde{p}_{ij}(u) = \begin{cases} 0 & \text{if } j = i, \\ \dfrac{p_{ij}(u)}{1 - p_{ii}(u)} & \text{if } j \neq i, \end{cases} \qquad i = 1, \ldots, n, \; j = 1, \ldots, n, t, \; u \in U(i),$$

to obtain another SSP without self-transitions. Show that the modified SSP is equivalent to the original in the sense that its stationary policies and optimal policies have the same cost functions. What is the interpretation of the transitions of the modified SSP in terms of transitions of the original?

(b) Fix a policy μ. Let J_k and $\tilde{J}_k$ be the sequences of cost vectors generated by value iteration (for the fixed policy) in the original and the modified SSP, respectively, starting from the same initial vector J_0. Show that value iteration is faster for the modified SSP in the sense that if $J_0 \leq J_1$, then $J_k \leq \tilde{J}_k \leq J^*$ for all k, and if $J_0 \geq J_1$, then $J_k \geq \tilde{J}_k \geq J^*$ for all k.

7.28 (Total Cost Problems with Nonnegative Costs) WWW

This is a theoretical problem whose purpose is to provide some additional analysis for undiscounted cost problems, including an extension of the results of Section 7.2 for stochastic shortest path problems. The idea is to use the analysis of Section 7.3 for discounted problems to derive the basic results for total undiscounted cost problems under the assumption that the stage costs are nonnegative and the optimal costs are finite. These results apply, among others, to some stochastic shortest path problems where not all stationary policies are proper and Assumption 7.2.1 is violated.

Consider a controlled Markov chain with states $i = 1, \ldots, n$, controls u chosen from a finite constraint set $U(i)$ for each state i, and transition probabilities $p_{ij}(u)$. (The states may include a cost-free and absorbing termination state, but this is not relevant for the following analysis.) The cost of the kth stage at state i when control u is applied has the form

$$\alpha^k g(i,u), \qquad i = 1, \ldots, n, \ u \in U(i),$$

where α is a scalar from $(0, 1]$. Our key assumption is that

$$0 \leq g(i,u), \qquad i = 1, \ldots, n, \ u \in U(i).$$

For any policy π, let $J_{\pi,\alpha}$ be the cost function for the α-discounted problem ($\alpha < 1$), and let J_π be the cost function for the problem where $\alpha = 1$. Note that for $\alpha = 1$, we may have $J_\pi(i) = \infty$ for some π and i. However, the limit defining $J_\pi(i)$ exists either as a real number or ∞, thanks to the assumption $0 \leq g(i,u)$ for all i and u. Let $J^*_\alpha(i)$ and $J^*(i)$ be the optimal costs starting from i, when $\alpha < 1$ and $\alpha = 1$, respectively. We assume that

$$J^*(i) < \infty, \qquad i = 1, \ldots, n,$$

(this is true in particular for the case of a stochastic shortest path problem if there exists a proper stationary policy, i.e., a policy under which there is a positive transition probability path from every state to the termination state).

(a) Show that for all $\alpha < 1$, we have

$$0 \leq J^*_\alpha(i) \leq J^*(i), \qquad i = 1, \ldots, n.$$

(b) Show that for any admissible policy π, we have

$$\lim_{\alpha \uparrow 1} J_{\pi,\alpha}(i) = J_\pi(i), \qquad i = 1, \ldots, n.$$

Furthermore,

$$\lim_{\alpha \uparrow 1} J^*_\alpha(i) = J^*(i), \qquad i = 1, \ldots, n.$$

Hint: To show the first equality, note that for any $\alpha < 1$, N, and $\pi = \{\mu_0, \mu_1, \ldots\}$, we have

$$J_\pi(i) \geq J_{\pi,\alpha}(i) \geq \sum_{k=0}^{N-1} \alpha^k E\big\{g\big(i_k, \mu_k(i_k)\big) \mid i_0 = i, \pi\big\}.$$

Take the limit as $\alpha \to 1$ and then take the limit as $N \to \infty$. For the second equality, consider a stationary policy μ and a sequence $\{a_m\} \subset (0,1)$ with $\alpha_m \to 1$ such that $J_{\mu,\alpha_m} = J^*_{\alpha_m}$ for all m.

(c) Use Bellman's equation for $\alpha < 1$, to show that J^* satisfies Bellman's equation for $\alpha = 1$:

$$J^*(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^n p_{ij}(u) J^*(j) \right], \qquad i = 1, \ldots, n.$$

(d) Let $\tilde{J}$ be such that $0 \leq \tilde{J}(i) < \infty$ for all i. Show that if

$$\tilde{J}(i) \geq \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^n p_{ij}(u) \tilde{J}(j) \right], \qquad i = 1, \ldots, n,$$

then $\tilde{J}(i) \geq J^*(i)$ for all i. Show also that if for some stationary policy μ, we have

$$\tilde{J}(i) \geq g\big(i, \mu(i)\big) + \sum_{j=1}^n p_{ij}\big(\mu(i)\big) \tilde{J}(j), \qquad i = 1, \ldots, n,$$

then $\tilde{J}(i) \geq J_\mu(i)$ for all i. *Hint*: Argue that

$$\tilde{J}(i) \geq \min_{u \in U(i)} \left[g(i,u) + \alpha \sum_{j=1}^n p_{ij}(u) \tilde{J}(j) \right], \qquad i = 1, \ldots, n,$$

use value iteration to show that $\tilde{J} \geq J^*_\alpha$, and take the limit as $\alpha \to 1$.

(e) For $\alpha = 1$, show that if

$$\mu^*(i) = \arg \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^n p_{ij}(u) J^*(j) \right], \qquad i = 1, \ldots, n,$$

then μ^* is optimal. *Hint*: Use part (d) with $\tilde{J} = J^*$.

(f) For $\alpha = 1$, show that for the value iteration method, given by

$$J_{k+1}(i) = \min_{u \in U(i)} \left[g(i,u) + \sum_{j=1}^{n} p_{ij}(u) J_k(j) \right], \qquad i = 1, \ldots, n,$$

we have $J_k(i) \to J^*(i)$, $i = 1, \ldots, n$, assuming that

$$0 \leq J_0(i) \leq J^*(i), \qquad i = 1, \ldots, n.$$

Give examples showing what may happen when this last assumption is violated. *Hint*: Prove the result by first assuming that J_0 is the zero function.

(g) Show that the set of states $Z = \{ i \mid J^*(i) = 0 \}$ is nonempty. Furthermore, under an optimal stationary policy μ^*, the set of states Z is cost-free and absorbing, i.e., $g\big(i, \mu^*(i)\big) = 0$ and $p_{ij}\big(\mu^*(i)\big) = 0$ for all $i \in Z$ and $j \notin Z$. In addition, μ^* is proper in the sense that for every state $i \notin Z$, under μ^*, there is a positive probability path that starts at i and ends at a state of Z.

APPENDIX A: Mathematical Review

The purpose of this appendix is to provide a list of mathematical definitions, notations, and results that are used frequently in the text. For detailed expositions, the reader may consult textbooks such as Hoffman and Kunze [HoK71], Royden [Roy88], Rudin [Rud76], and Strang [Str76].

A.1 SETS

If x is a member of the set S, we write $x \in S$. We write $x \notin S$ if x is not a member of S. A set S may be specified by listing its elements within braces. For example, by writing $S = \{x_1, x_2, \ldots, x_n\}$ we mean that the set S consists of the elements $x_1, x_2, \ldots, x_n$. A set S may also be specified in the generic form

$$S = \{x \mid x \text{ satisfies } P\}$$

as the set of elements satisfying property P. For example,

$$S = \{x \mid x : \text{real}, \ 0 \leq x \leq 1\}$$

denotes the set of all real numbers x satisfying $0 \leq x \leq 1$.

The *union* of two sets S and T is denoted by $S \cup T$ and the *intersection* of S and T is denoted by $S \cap T$. The union and the intersection of a sequence of sets $S_1, S_2, \ldots, S_k, \ldots$ are denoted by $\cup_{k=1}^{\infty} S_k$ and $\cap_{k=1}^{\infty} S_k$, respectively. If S is a subset of T (i.e., if every element of S is also an element of T), we write $S \subset T$ or $T \supset S$.

Finite and Countable Sets

A set S is said to be *finite* if it consists of a finite number of elements. It is said to be *countable* if there exists a one-to-one function from S into the set of nonnegative integers. Thus, according to our definition, a finite set is also countable but not conversely. A countable set S that is not finite may be represented by listing its elements $x_0, x_1, x_2, \ldots$ (i.e., $S = \{x_0, x_1, x_2, \ldots\}$). A countable union of countable sets is countable, that is, if $A = \{a_0, a_1, \ldots\}$ is a countable set and $S_{a_0}, S_{a_1}, \ldots$ are each countable sets, then $\cup_{k=0}^{\infty} S_{a_k}$ is also a countable set.

Sets of Real Numbers

If a and b are real numbers or $+\infty$, $-\infty$, we denote by $[a, b]$ the set of numbers x satisfying $a \leq x \leq b$ (including the possibility $x = +\infty$ or $x = -\infty$). A rounded, instead of square, bracket denotes strict inequality in the definition. Thus $(a, b]$, $[a, b)$, and (a, b) denote the set of all x satisfying $a < x \leq b$, $a \leq x < b$, and $a < x < b$, respectively.

If S is a set of real numbers that is bounded above, then there is a smallest real number y such that $x \leq y$ for all $x \in S$. This number is called the *least upper bound* or *supremum* of S and is denoted by $\sup\{x \mid x \in S\}$ or $\max\{x \mid x \in S\}$. (This is somewhat inconsistent with normal mathematical usage, where the use of max in place of sup indicates that the supremum is attained by some element of S.) Similarly, the greatest real number z such that $z \leq x$ for all $x \in S$ is called the *greatest lower bound* or *infimum* of S and is denoted by $\inf\{x \mid x \in S\}$ or $\min\{x \mid x \in S\}$. If S is unbounded above, we write $\sup\{x \mid x \in S\} = +\infty$, and if it is unbounded below, we write $\inf\{x \mid x \in S\} = -\infty$. If S is the empty set, then by convention we write $\inf\{x \mid x \in S\} = +\infty$ and $\sup\{x \mid x \in S\} = -\infty$.

A.2 EUCLIDEAN SPACE

The set of all n-tuples $x = (x_1, \ldots, x_n)$ of real numbers constitutes the *n-dimensional Euclidean space*, denoted by $\Re^n$. The elements of $\Re^n$ are referred to as n-dimensional vectors or simply vectors when confusion cannot arise. The one-dimensional Euclidean space $\Re^1$ consists of all the real numbers and is denoted by $\Re$. Vectors in $\Re^n$ can be added by adding their corresponding components. They can be multiplied by a scalar by multiplication of each component by a scalar. The *inner product* of two vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ is denoted by $x'y$ and is equal to $\sum_{i=1}^n x_i y_i$. The *norm* of a vector $x = (x_1, \ldots, x_n) \in \Re^n$ is denoted by $\|x\|$ and is equal to $(x'x)^{1/2} = (\sum_{i=1}^n x_i^2)^{1/2}$.

A set of vectors $a_1, a_2, \ldots, a_k$ is said to be *linearly dependent* if there exist scalars $\lambda_1, \lambda_2, \ldots, \lambda_k$, not all zero, such that

$$\lambda_1 a_1 + \cdots + \lambda_k a_k = 0.$$

If no such set of scalars exists, the vectors are said to be *linearly independent*.

A.3 MATRICES

An $m \times n$ *matrix* is a rectangular array of numbers, referred to as elements or components, which are arranged in m rows and n columns. If $m = n$ the matrix is said to be *square*. The element in the ith row and jth column of a matrix A is denoted by a subscript ij, such as a_{ij}, in which case we write $A = [a_{ij}]$. The $n \times n$ *identity matrix*, denoted by I, is the matrix with elements $a_{ij} = 0$ for $i \neq j$ and $a_{ii} = 1$, for $i = 1, \ldots, n$. The *sum* of two $m \times n$ matrices A and B is written as $A + B$ and is the matrix whose elements are the sum of the corresponding elements in A and B. The *product of a matrix A and a scalar* λ, written as λA or $A\lambda$, is obtained by multiplying each element of A by λ. The *product* AB of an $m \times n$ matrix A and an $n \times p$ matrix B is the $m \times p$ matrix C with elements $c_{ij} = \sum_{k=1}^n a_{ik} b_{kj}$. If b is an n-dimensional column vector and A is an $m \times n$ matrix, then Ab is an m-dimensional column vector.

The *transpose* of an $m \times n$ matrix A is the $n \times m$ matrix A' with elements $a'_{ij} = a_{ji}$. The elements of a given row (or column) of A constitute a vector called a row vector (or column vector, respectively) of A. A square matrix A is *symmetric* if $A' = A$. An $n \times n$ matrix A is called *nonsingular* or *invertible* if there is an $n \times n$ matrix called the *inverse* of A and denoted by A^{-1}, such that $A^{-1}A = I = AA^{-1}$, where I is the $n \times n$ identity matrix. An $n \times n$ matrix is nonsingular if and only if its n row vectors are linearly independent or, equivalently, if its n column vectors are linearly independent. Thus, an $n \times n$ matrix A is nonsingular if and only if the relation $Av = 0$, where $v \in \Re^n$, implies that $v = 0$.

Rank of a Matrix

The *rank* of a matrix A is equal to the maximum number of linearly independent row vectors of A. It is also equal to the maximum number of linearly independent column vectors. Thus, the rank of an $m \times n$ matrix is at most equal to the minimum of the dimensions m and n. An $m \times n$ matrix is said to be of *full rank* if its rank is maximal, that is, if its rank is equal to the minimum of m and n. A square matrix is of full rank if and only if it is nonsingular.

Eigenvalues

Given a square $n \times n$ matrix A, the determinant of the matrix $\gamma I - A$, where I is the $n \times n$ identity matrix and γ is a scalar, is an nth degree polynomial. The n roots of this polynomial are called the *eigenvalues* of A. Thus, γ is an eigenvalue of A if and only if the matrix $\gamma I - A$ is singular, or equivalently, if and only if there exists a nonzero vector v such that $Av = \gamma v$. Such a vector v is called an *eigenvector* corresponding to γ. The eigenvalues and eigenvectors of A can be complex even if A is real. A matrix A is singular if and only if it has an eigenvalue that is equal to zero. If A is nonsingular, then the eigenvalues of A^{-1} are the reciprocals of the eigenvalues of A. The eigenvalues of A and A' coincide.

If $\gamma_1, \ldots, \gamma_n$ are the eigenvalues of A, then the eigenvalues of $cI + A$, where c is a scalar and I is the identity matrix, are $c + \gamma_1, \ldots, c + \gamma_n$. The eigenvalues of A^k, where k is any positive integer, are equal to $\gamma_1^k, \ldots, \gamma_n^k$. From this it follows that $\lim_{k \to 0} A^k = 0$ if and only if all the eigenvalues of A lie strictly within the unit circle of the complex plane. Furthermore, if the latter condition holds, the iteration

$$x_{k+1} = Ax_k + b,$$

where b is a given vector, converges to

$$\overline{x} = (I - A)^{-1} b,$$

which is the unique solution of the equation $x = Ax + b$.

If all the eigenvalues of A are distinct, then their number is exactly n, and there exists a set of corresponding linearly independent eigenvectors. In this case, if $\gamma_1, \ldots, \gamma_n$ are the eigenvalues and $v_1, \ldots, v_n$ are such eigenvectors, every vector $x \in \Re^n$ can be decomposed as

$$x = \sum_{i=1}^{n} \xi_i v_i,$$

where ξ_i are some unique (possibly complex) numbers. Furthermore, we have for all positive integers k,

$$A^k x = \sum_{i=1}^{n} \gamma_i^k \xi_i v_i.$$

If A is a transition probability matrix, that is, all the elements of A are nonnegative and the sum of the elements of each of its rows is equal to 1, then all the eigenvalues of A lie within the unit circle of the complex plane. Furthermore, 1 is an eigenvalue of A and the unit vector $(1, 1, \ldots, 1)$ is a corresponding eigenvector.

Positive Definite and Semidefinite Symmetric Matrices

A square symmetric $n \times n$ matrix A is said to be *positive semidefinite* if $x'Ax \geq 0$ for all $x \in \Re^n$. It is said to be *positive definite* if $x'Ax > 0$ for all nonzero $x \in \Re^n$. The matrix A is said to be *negative semidefinite (definite)* if $-A$ is *positive semidefinite (definite)*. In this book, the notions of positive definiteness and semidefiniteness will be used only in connection with symmetric matrices.

A positive definite symmetric matrix is invertible and its inverse is also positive definite symmetric. Also, an invertible positive semidefinite symmetric matrix is positive definite. Analogous results hold for negative definite and semidefinite symmetric matrices. If A and B are $n \times n$ positive semidefinite (definite) symmetric matrices, then the matrix $\lambda A + \mu B$ is also positive semidefinite (definite) symmetric for all $\lambda \geq 0$ and $\mu \geq 0$. If A is an $n \times n$ positive semidefinite symmetric matrix and C is an $m \times n$ matrix, then the matrix CAC' is positive semidefinite symmetric. If A is positive definite symmetric, and C has rank m (equivalently, $m \leq n$ and C has full rank), then CAC' is positive definite symmetric.

An $n \times n$ positive definite symmetric matrix A can be written as CC' where C is a square invertible matrix. If A is positive semidefinite symmetric and its rank is m, then it can be written as CC', where C is an $n \times m$ matrix of full rank.

A symmetric $n \times n$ matrix A has real eigenvalues and a set of n real linearly independent eigenvectors, which are orthogonal (the inner product of any pair is 0). If A is positive semidefinite (definite) symmetric, its eigenvalues are nonnegative (respectively, positive).

Partitioned Matrices

It is often convenient to partition a matrix into submatrices. For example, the matrix

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix}$$

may be partitioned into

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix},$$

where

$$A_{11} = \begin{pmatrix} a_{11} & a_{12} \end{pmatrix}, \qquad A_{12} = \begin{pmatrix} a_{13} & a_{14} \end{pmatrix},$$

$$A_{21} = \begin{pmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{pmatrix}, \qquad A_{22} = \begin{pmatrix} a_{23} & a_{24} \\ a_{33} & a_{34} \end{pmatrix}.$$

We separate the components of a partitioned matrix by a space, as in $(B\ C)$, or by a comma, as in (B, C). The transpose of the partitioned matrix A is

$$A' = \begin{pmatrix} A'_{11} & A'_{21} \\ A'_{12} & A'_{22} \end{pmatrix}.$$

Partitioned matrices may be multiplied just as nonpartitioned matrices, provided the dimensions involved in the partitions are compatible. Thus if

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \qquad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix},$$

then

$$AB = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix},$$

provided the dimensions of the submatrices are such that the preceding products $A_{ij}B_{jk}$, $i, j, k = 1, 2$ can be formed.

Matrix Inversion Formulas

Let A and B be square invertible matrices, and let C be a matrix of appropriate dimension. Then, if all the following inverses exist, we have

$$(A + CBC')^{-1} = A^{-1} - A^{-1}C(B^{-1} + C'A^{-1}C)^{-1}C'A^{-1}.$$

The equation can be verified by multiplying the right-hand side by

$$A + CBC'$$

and showing that the product is the identity matrix.

Consider a partitioned matrix M of the form

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}.$$

Then we have

$$M^{-1} = \begin{pmatrix} Q & -QBD^{-1} \\ -D^{-1}CQ & D^{-1} + D^{-1}CQBD^{-1} \end{pmatrix},$$

where

$$Q = (A - BD^{-1}C)^{-1},$$

provided all the inverses exist. The proof is obtained by multiplying M with the expression given for M^{-1} and verifying that the product yields the identity matrix.

A.4 ANALYSIS

Convergence of Sequences

A sequence of vectors $x_0, x_1, \ldots, x_k, \ldots$ in $\Re^n$, denoted by $\{x_k\}$, is said to converge to a *limit* x if $\|x_k - x\| \to 0$ as $k \to \infty$ (i.e., if, given any $\epsilon > 0$, there is an integer N such that for all $k \geq N$ we have $\|x_k - x\| < \epsilon$). If $\{x_k\}$ converges to x, we write $x_k \to x$ or $\lim_{k\to\infty} x_k = x$. We have $Ax_k + By_k \to Ax + By$ if $x_k \to x$, $y_k \to y$, and A, B are matrices of appropriate dimension.

A vector x is said to be a *limit point* of a sequence $\{x_k\}$ if there is a subsequence of $\{x_k\}$ that converges to x, that is, if there is an infinite subset $\mathcal{K}$ of the nonnegative integers such that for any $\epsilon > 0$, there is an integer N such that for all $k \in \mathcal{K}$ with $k \geq N$ we have $\|x_k - x\| < \epsilon$.

A sequence of real numbers $\{r_k\}$, which is monotonically nondecreasing (nonincreasing), that is, satisfies $r_k \leq r_{k+1}$ for all k, must either converge to a real number or be unbounded above (below). In the latter case we write $\lim_{k\to\infty} r_k = \infty$ $(-\infty)$. Given any bounded sequence of real numbers $\{r_k\}$, we may consider the sequence $\{s_k\}$, where $s_k = \sup\{r_i \mid i \geq k\}$. Since this sequence is monotonically nonincreasing and bounded, it must have a limit. This limit is called the *limit superior* of $\{r_k\}$ and is denoted by $\limsup_{k\to\infty} r_k$. The *limit inferior* of $\{r_k\}$ is similarly defined and is denoted by $\liminf_{k\to\infty} r_k$. If $\{r_k\}$ is unbounded above, we write $\limsup_{k\to\infty} r_k = \infty$, and if it is unbounded below, we write $\liminf_{k\to\infty} r_k = -\infty$. We also use this notation if $r_k \in [-\infty, \infty]$ for all k.

Open, Closed, and Compact Sets

A subset S of $\Re^n$ is said to be *open* if for every vector $x \in S$ one can find an $\epsilon > 0$ such that $\{z \mid \|z - x\| < \epsilon\} \subset S$. A set S is *closed* if and only if every convergent sequence $\{x_k\}$ with elements in S converges to a point that also belongs to S. A set S is said to be *compact* if and only if it is both closed and bounded (i.e., it is closed and for some $M > 0$ we have $\|x\| \leq M$ for all $x \in S$). A set S is compact if and only if every sequence $\{x_k\}$ with elements in S has at least one limit point that belongs to S. Another important fact is that if $S_0, S_1, \ldots, S_k, \ldots$ is a sequence of nonempty compact sets in $\Re^n$ such that $S_k \supset S_{k+1}$ for all k, then the intersection $\cap_{k=0}^{\infty} S_k$ is a nonempty and compact set.

Continuous Functions

A function f mapping a set S_1 into a set S_2 is denoted by $f : S_1 \to S_2$. A function $f : \Re^n \to \Re^m$ is said to be *continuous* if for all x, $f(x_k) \to f(x)$ whenever $x_k \to x$. Equivalently, f is continuous if, given $x \in \Re^n$ and $\epsilon > 0$,

there is a $\delta > 0$ such that whenever $\|y-x\| < \delta$, we have $\|f(y)-f(x)\| < \epsilon$. The function

$$(a_1 f_1 + a_2 f_2)(\cdot) = a_1 f_1(\cdot) + a_2 f_2(\cdot)$$

is continuous for any two scalars a_1, a_2 and any two continuous functions $f_1, f_2 : \Re^n \to \Re^m$. If S_1, S_2, S_3 are any sets and $f_1 : S_1 \to S_2$, $f_2 : S_2 \to S_3$ are functions, the function $f_2 \cdot f_1 : S_1 \to S_3$ defined by $(f_2 \cdot f_1)(x) = f_2\big(f_1(x)\big)$ is called the *composition* of f_1 and f_2. If $f_1 : \Re^n \to \Re^m$ and $f_2 : \Re^m \to \Re^p$ are continuous, then $f_2 \cdot f_1$ is also continuous.

Derivatives

Let $f : \Re^n \mapsto \Re$ be some function. For a fixed $x \in \Re^n$, the first partial derivative of f at the point x with respect to the ith coordinate is defined by

$$\frac{\partial f(x)}{\partial x_i} = \lim_{\alpha \to 0} \frac{f(x + \alpha e_i) - f(x)}{\alpha},$$

where e_i is the ith unit vector, and we assume that the above limit exists. If the partial derivatives with respect to all coordinates exist, f is called differentiable at x and its *gradient* at x is defined to be the column vector

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{pmatrix}.$$

The function f is called differentiable if it is differentiable at every $x \in \Re^n$. If $\nabla f(x)$ exists for every x and is a continuous function of x, f is said to be *continuously differentiable*. Such a function admits, for every fixed x, the first order expansion

$$f(x+y) = f(x) + y' \nabla f(x) + o(\|y\|),$$

where $o(\|y\|)$ is a function of y with the property $\lim_{\|y\| \to 0} o(\|y\|)/\|y\| = 0$.

A vector-valued function $f : \Re^n \mapsto \Re^m$ is called differentiable (respectively, continuously differentiable) if each component f_i of f is differentiable (respectively, continuously differentiable). The *gradient matrix* of f, denoted by $\nabla f(x)$, is the $n \times m$ matrix whose ith column is the gradient $\nabla f_i(x)$ of f_i. Thus,

$$\nabla f(x) = \Big[\nabla f_1(x) \cdots \nabla f_m(x)\Big].$$

The transpose of ∇f is the *Jacobian* of f; it is the matrix whose ijth entry is equal to the partial derivative $\partial f_i / \partial x_j$.

If the gradient $\nabla f(x)$ is itself a differentiable function, then f is said to be twice differentiable. We denote by $\nabla^2 f(x)$ the Hessian matrix of f at x, that is, the matrix

$$\nabla^2 f(x) = \left[\frac{\partial^2 f(x)}{\partial x^i \partial x^j}\right]$$

the elements of which are the second partial derivatives of f at x.

Let $f : \Re^k \mapsto \Re^m$ and $g : \Re^m \mapsto \Re^n$ be continuously differentiable functions, and let $h(x) = g\big(f(x)\big)$. The *chain rule* for differentiation states that

$$\nabla h(x) = \nabla f(x) \nabla g\big(f(x)\big), \qquad \text{for all } x \in \Re^k.$$

For example, if A and B are given matrices, then if $h(x) = Ax$, we have $\nabla h(x) = A'$ and if $h(x) = ABx$, we have $\nabla h(x) = B'A'$.

A.5 CONVEX SETS AND FUNCTIONS

A subset C of $\Re^n$ is said to be *convex* if for every $x, y \in C$ and every scalar α with $0 \le \alpha \le 1$, we have $\alpha x + (1-\alpha)y \in C$. In words, C is convex if the line segment connecting any two points in C belongs to C. A function $f : C \to \Re$, defined over a convex subset C of $\Re^n$, is said to be *convex* if for every $x, y \in C$ and every scalar α with $0 \le \alpha \le 1$ we have

$$f\big(\alpha x + (1-\alpha)y\big) \le \alpha f(x) + (1-\alpha) f(y).$$

The function f is said to be *concave* if $(-f)$ is convex, or equivalently if for every $x, y \in C$ and every scalar α with $0 \le \alpha \le 1$ we have

$$f\big(\alpha x + (1-\alpha)y\big) \ge \alpha f(x) + (1-\alpha) f(y).$$

If $f : C \to \Re$ is convex, then the sets $\Gamma_\lambda = \big\{x \mid x \in C,\ f(x) \le \lambda\big\}$ are convex for every scalar λ. An important property is that a real-valued convex function defined over $\Re^n$ is continuous.

If $f_1, f_2, \ldots, f_m$ are convex functions defined over a convex subset C of $\Re^n$ and $\alpha_1, \alpha_2, \ldots, \alpha_m$ are nonnegative scalars, then the function $\alpha_1 f_1 + \cdots + \alpha_m f_m$ is also convex over C. If $f : \Re^m \to \Re$ is convex, A is an $m \times n$ matrix, and b is a vector in $\Re^m$, the function $g : \Re^n \to \Re$ defined by $g(x) = f(Ax + b)$ is also convex. If $f : \Re^n \to \Re$ is convex, then the function $g(x) = E_w\big\{f(x+w)\big\}$, where w is a random vector in $\Re^n$, is a convex function provided the expected value is finite for every $x \in \Re^n$.

For functions $f : \Re^n \to \Re$ that are differentiable, there are alternative characterizations of convexity. Thus, f is convex if and only if

$$f(y) \ge f(x) + \nabla f(x)'(y-x), \qquad \text{for all } x, y \in \Re^n.$$

If f is twice continuously differentiable, then f is convex if and only if $\nabla^2 f(x)$ is a positive semidefinite symmetric matrix for every $x \in \Re^n$.

For accounts of convexity and its applications in optimization, see Bertsekas [BNO03] and Rockafellar [Roc70].

APPENDIX B:
On Optimization Theory

The purpose of this appendix is to provide a few definitions and results of deterministic optimization. For detailed expositions, which include both convex and nonconvex problems, see textbooks such as Bertsekas [Ber99], [BNO03], Luenberger [Lue84], and Rockafellar [Roc70].

B.1 OPTIMAL SOLUTIONS

Given a set S, a real-valued function $f : S \to \Re$, and a subset $X \subset S$, the optimization problem

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{B.1}$$

is to find an element $x^* \in X$ (called a *minimizing element* or an *optimal solution*) such that

$$f(x^*) \leq f(x), \qquad \text{for all } x \in X.$$

For any minimizing element x^*, we write

$$x^* = \arg\min_{x \in X} f(x).$$

Note that a minimizing element need not exist. For example, the scalar functions $f(x) = x$ and $f(x) = e^x$ have no minimizing elements over the set of real numbers. The first function decreases without bound to $-\infty$ as x tends toward $-\infty$, while the second decreases toward 0 as x tends toward $-\infty$ but always takes positive values. Given the range of values that $f(x)$ takes as x ranges over X, that is, the set of real numbers

$$\{f(x) \mid x \in X\},$$

there are two possibilities:

1. The set $\{f(x) \mid x \in X\}$ is unbounded below (i.e., contains arbitrarily small real numbers) in which case we write

$$\min\{f(x) \mid x \in X\} = -\infty \quad \text{or} \quad \min_{x \in X} f(x) = -\infty.$$

2. The set $\{f(x) \mid x \in X\}$ is bounded below; that is, there exists a scalar M such that $M \leq f(x)$ for all $x \in X$. The greatest lower bound of $\{f(x) \mid x \in X\}$ is also denoted by

$$\min\{f(x) \mid x \in X\} \quad \text{or} \quad \min_{x \in X} f(x).$$

In either case we call $\min_{x \in X} f(x)$ the *optimal value* of problem (B.1).

A maximization problem of the form

$$\begin{aligned} &\text{maximize} \quad f(x) \\ &\text{subject to} \quad x \in X \end{aligned}$$

may be converted to the minimization problem

$$\begin{aligned} &\text{minimize} \quad -f(x) \\ &\text{subject to} \quad x \in X, \end{aligned}$$

in the sense that both problems have the same optimal solutions, and the optimal value of one is equal to minus the optimal value of the other. The optimal value for the maximization problem is denoted by $\max_{x \in X} f(x)$.

Existence of at least one optimal solution in problem (B.1) is guaranteed if $f : \Re^n \to \Re$ is a continuous function and X is a compact subset of $\Re^n$. This is the *Weierstrass theorem*. By a related result, existence of an optimal solution is guaranteed if $f : \Re^n \to \Re$ is a continuous function, X is closed, and $f(x) \to \infty$ if $\|x\| \to \infty$.

B.2 OPTIMALITY CONDITIONS

Optimality conditions are available when f is a differentiable function on $\Re^n$ and X is a convex subset of $\Re^n$ (possibly $X = \Re^n$). In particular, if x^* is an optimal solution of problem (B.1), $f : \Re^n \to \Re$ is a continuously differentiable function on $\Re^n$, and X is convex, we have

$$\nabla f(x^*)'(x - x^*) \geq 0, \qquad \text{for all } x \in X, \tag{B.2}$$

where $\nabla f(x^*)$ denotes the gradient of f at x^*. When $X = \Re^n$ (i.e., the minimization is unconstrained), the necessary condition (B.2) is equivalent to

$$\nabla f(x^*) = 0. \tag{B.3}$$

When f is twice continuously differentiable and $X = \Re^n$, an additional necessary condition is that the *Hessian matrix* $\nabla^2 f(x^*)$ *be positive semidefinite at* x^*. An important fact is that *if* $f : \Re^n \to \Re$ *is a convex function and* X *is convex, then Eq. (B.2) is both a necessary and a sufficient condition for optimality of* x^*.

Other types of optimality conditions deal with the case where the constraint set X consists of equality and inequality constraints, i.e., problems of the form

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad h_1(x) = 0, \ldots, h_m(x) = 0, \ g_1(x) \leq 0, \ldots, g_r(x) \leq 0, \end{aligned}$$

where f, h_i, g_j are continuously differentiable functions from $\Re^n$ to $\Re$.

We say that the vectors $\lambda^* = (\lambda_1^*, \ldots, \lambda_m^*)$ and $\mu^* = (\mu_1^*, \ldots, \mu_r^*)$ are *Lagrange multiplier vectors* corresponding to a local minimum x^* if they satisfy the following conditions:

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla h_i(x^*) + \sum_{j=1}^{r} \mu_j^* \nabla g_j(x^*) = 0,$$

$$\mu_j^* \geq 0, \qquad \text{for all } j = 1, \ldots, r,$$

$$\mu_j^* = 0, \qquad \text{for all } j \notin A(x^*),$$

where $A(x^*)$ is the index set of inequality constraints that are active at x^*:

$$A(x^*) = \{j \mid g_j(x^*) = 0\}.$$

Lagrange multiplier theory revolves around conditions under which Lagrange multiplier vectors are guaranteed to exist for a given local minimum x^*. Such conditions are known as *constraint qualifications*. Some of the most useful ones are the following:

CQ1: The equality constraint gradients $\nabla h_i(x^*)$, $i = 1, \ldots, m$, and the active inequality constraint gradients $\nabla g_j(x^*)$, $j \in A(x^*)$, are linearly independent.

CQ2: The equality constraint gradients $\nabla h_i(x^*)$, $i = 1, \ldots, m$, are linearly independent, and there exists a $y \in \Re^n$ such that

$$\nabla h_i(x^*)'y = 0 \text{ for all } i = 1, \ldots, m, \quad \nabla g_j(x^*)'y < 0 \text{ for all } j \in A(x^*).$$

CQ3: The functions h_i are linear and the functions g_j are concave.

CQ4: The functions h_i are linear, the functions g_j are convex, and there exists a $y \in \Re^n$ such that

$$g_j(y) < 0, \qquad \text{for all } j = 1, \ldots, r.$$

Each of the above constraint qualifications implies the existence of at least one Lagrange multiplier vector associated with x^* (unique in the case of CQ1); see e.g., [Ber99] for a detailed account.

B.3 MINIMIZATION OF QUADRATIC FORMS

Let $f : \Re^n \to \Re$ be a quadratic form

$$f(x) = \tfrac{1}{2} x'Qx + b'x,$$

where Q is a symmetric $n \times n$ matrix and $b \in \Re^n$. Its gradient is given by

$$\nabla f(x) = Qx + b.$$

The function f is convex if and only if Q is positive semidefinite. If Q is positive definite, then f is convex and Q is invertible, so by Eq. (B.3), a vector x^* minimizes f if and only if

$$\nabla f(x^*) = Qx^* + b = 0,$$

or equivalently

$$x^* = -Q^{-1}b.$$

APPENDIX C:
On Probability Theory

This appendix lists selectively some of the basic probabilistic notions that we will be using. Its main purpose is to familiarize the reader with some of our terminology. It is not meant to be exhaustive, and the reader should consult textbooks such as Ash [Ash70], Feller [Fel68], Papoulis [Pap65], Ross [Ros85], Stirzaker [Sti94], and Bertsekas and Tsitsiklis [BeT02] for detailed accounts. For fairly accessible treatments of measure theoretic probability, see Adams and Guillemin [AdG86], and Ash [Ash72].

C.1 PROBABILITY SPACES

A *probability space* consists of

(a) A set Ω.

(b) A collection $\mathcal{F}$ of subsets of Ω, called *events*, which includes Ω and has the following properties:

(1) If A is an event, then the complement $\overline{A} = \{\omega \in \Omega \mid \omega \notin A\}$ is also an event. (The complement of Ω is the empty set and is considered to be an event.)

(2) If $A_1, A_2, \ldots, A_k, \ldots$ are events, then $\cup_{k=1}^{\infty} A_k$ is also an event.

(3) If $A_1, A_2, \ldots, A_k, \ldots$ are events, then $\cap_{k=1}^{\infty} A_k$ is also an event.

(c) A function $P(\cdot)$ assigning to each event A a real number $P(A)$, called the *probability of the event* A, and satisfying:

(1) $P(A) \geq 0$ for every event A.

(2) $P(\Omega) = 1$.

(3) $P(A_1 \cup A_2) = P(A_1) + P(A_2)$ for every pair of disjoint events A_1, A_2.

(4) $P(\cup_{k=1}^{\infty} A_k) = \sum_{k=1}^{\infty} P(A_k)$ for every sequence of mutually disjoint events $A_1, A_2, \ldots, A_k, \ldots$

The function P is referred to as a *probability measure*.

Convention for Finite and Countable Probability Spaces

The case of a probability space where the set Ω is a countable (possibly finite) set is encountered frequently in this book. When we specify that Ω is finite or countable, we implicitly assume that the associated collection of events is the collection of *all* subsets of Ω (including Ω and the empty set). Then, if Ω is a finite set, $\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\}$, the probability space is specified by the probabilities $p_1, p_2, \ldots, p_n$, where p_i denotes the probability of the event consisting of just ω_i. Similarly, if $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k, \ldots\}$, the probability space is specified by the corresponding probabilities $p_1, p_2, \ldots, p_k, \ldots$ In either case we refer to $(p_1, p_2, \ldots, p_n)$ or $(p_1, p_2, \ldots, p_k, \ldots)$ as a *probability distribution over* Ω.

C.2 RANDOM VARIABLES

A *random variable* on a probability space $(\Omega, \mathcal{F}, P)$ is a function $x : \Omega \to \Re$ such that for every scalar λ the set

$$\{\omega \in \Omega \mid x(\omega) \leq \lambda\}$$

is an event (i.e., belongs to the collection $\mathcal{F}$). An n-dimensional *random vector* $x = (x_1, x_2, \ldots, x_n)$ is an n-tuple of random variables $x_1, x_2, \ldots, x_n$, each defined on the same probability space.

We define the *distribution function* $F : \Re \to \Re$ [or *cumulative distribution function* (CDF for short)] of a random variable x by

$$F(z) = P\Big(\{\omega \in \Omega \mid x(\omega) \leq z\}\Big);$$

that is, $F(z)$ is the probability that the random variable takes a value less than or equal to z. We define the distribution function $F : \Re^n \to \Re$ of a random vector $x = (x_1, x_2, \ldots, x_n)$ by

$$F(z_1, z_2, \ldots, z_n) = P\Big(\{\omega \in \Omega \mid x_1(\omega) \leq z_1, x_2(\omega) \leq z_2, \ldots, x_n(\omega) \leq z_n\}\Big).$$

Given the distribution function of a random vector $x = (x_1, \ldots, x_n)$, the (marginal) distribution function of each random variable x_i is obtained from

$$F_i(z_i) = \lim_{z_j \to \infty, j \neq i} F(z_1, z_2, \ldots, z_n).$$

The random variables $x_1, \ldots, x_n$ are said to be *independent* if

$$F(z_1, z_2, \ldots, z_n) = F_1(z_1)F_2(z_2)\cdots F_n(z_n),$$

for all scalars $z_1, \ldots, z_n$.

The *expected value of a random variable* x with distribution function F is defined by

$$E\{x\} = \int_{-\infty}^{\infty} z dF(z)$$

provided the integral is well-defined. The *expected value of a random vector* $x = (x_1, \ldots, x_n)$ is the vector

$$E\{x\} = \big(E\{x_1\}, E\{x_2\}, \ldots, E\{x_n\}\big).$$

The *covariance matrix* of a random vector $x = (x_1, \ldots, x_n)$ with expected value $E\{x\} = (\overline{x}_1, \ldots, \overline{x}_n)$ is defined to be the $n \times n$ positive semidefinite symmetric matrix

$$\begin{pmatrix} E\big\{(x_1 - \overline{x}_1)^2\big\} & \cdots & E\big\{(x_1 - \overline{x}_1)(x_n - \overline{x}_n)\big\} \\ \vdots & \vdots & \vdots \\ E\big\{(x_n - \overline{x}_n)(x_1 - \overline{x}_1)\big\} & \cdots & E\big\{(x_n - \overline{x}_n)^2\big\} \end{pmatrix},$$

provided the expected values are well-defined.

Two random vectors x and y are said to be *uncorrelated* if

$$E\Big\{\big(x - E\{x\}\big)\big(y - E\{y\}\big)'\Big\} = 0,$$

where $\big(x - E\{x\}\big)$ is viewed as a column vector and $\big(y - E\{y\}\big)'$ is viewed as a row vector.

The random vector $x = (x_1, \ldots, x_n)$ is said to be characterized by a *probability density function* $f : \Re^n \to \Re$ if

$$F(z_1, z_2, \ldots, z_n) = \int_{-\infty}^{z_1} \int_{-\infty}^{z_2} \cdots \int_{-\infty}^{z_n} f(y_1, \ldots, y_n) dy_1 \ldots dy_n,$$

for every $z_1, \ldots, z_n$.

C.3 CONDITIONAL PROBABILITY

We restrict ourselves to the case where the underlying probability space Ω is a countable (possibly finite) set and the set of events is the set of all subsets of Ω.

Given two events A and B, we define the *conditional probability of B given A* by

$$P(B \mid A) = \begin{cases} \frac{P(A \cap B)}{P(A)} & \text{if } P(A) > 0, \\ 0 & \text{if } P(A) = 0. \end{cases}$$

We also use the notation $P\{B \mid A\}$ in place of $P(B \mid A)$. If $B_1, B_2, \ldots$ are a countable (possibly finite) collection of mutually exclusive and exhaustive events (i.e., the sets B_i are disjoint and their union is Ω) and A is an event, then we have

$$P(A) = \sum_i P(A \cap B_i).$$

From the two preceding relations, we obtain the *total probability theorem:*

$$P(A) = \sum_i P(B_i)P(A \mid B_i).$$

We thus obtain for every k,

$$P(B_k \mid A) = \frac{P(A \cap B_k)}{P(A)} = \frac{P(B_k)P(A \mid B_k)}{\sum_i P(B_i)P(A \mid B_i)},$$

assuming that $P(A) > 0$. This relation is referred to as *Bayes' rule.*

Consider now two random vectors x and y taking values in $\Re^n$ and $\Re^m$, respectively [i.e., $x(\omega) \in \Re^n$, $y(\omega) \in \Re^m$ for all $\omega \in \Omega$]. Given two subsets X and Y of $\Re^n$ and $\Re^m$, respectively, we denote

$$P(X \mid Y) = P\Big(\big\{\omega \mid x(\omega) \in X\big\} \mid \big\{\omega \mid y(\omega) \in Y\big\}\Big).$$

For a fixed vector $v \in \Re^n$, we define the *conditional distribution function* of x given v by

$$F(z \mid v) = P\Big(\big\{\omega \mid x(\omega) \leq z\big\} \mid \big\{\omega \mid y(\omega) = v\big\}\Big),$$

and the *conditional expectation* of x given v by

$$E\{x \mid v\} = \int_{\Re^n} z dF(z \mid v),$$

assuming that the integral is well-defined. Note that $E\{x \mid v\}$ is a function mapping v into $\Re^n$.

Finally, let us provide Bayes' rule for random vectors. If $\omega_1, \omega_2, \ldots$ are the elements of Ω, denote

$$z_i = x(\omega_i), \qquad v_i = y(\omega_i), \qquad i = 1, 2, \ldots$$

Also, for any vectors $z \in \Re^n$, $v \in \Re^m$, let us denote

$$P(z) = P\Big(\{\omega \mid x(\omega) = z\}\Big), \qquad P(v) = P\Big(\{\omega \mid y(\omega) = v\}\Big).$$

We have $P(z) = 0$ if $z \neq z_i$, $i = 1, 2, \ldots$, and $P(v) = 0$ if $v \neq v_i$, $i = 1, 2, \ldots$ Denote also

$$P(z \mid v) = P\Big(\{\omega \mid x(\omega) = z\} \mid \{\omega \mid y(\omega) = v\}\Big),$$

$$P(v \mid z) = P\Big(\{\omega \mid y(\omega) = v\} \mid \{\omega \mid x(\omega) = z\}\Big).$$

Then, for all $k = 1, 2, \ldots$, Bayes' rule yields

$$P(z_k \mid v) = \begin{cases} \frac{P(z_k)P(v|z_k)}{\sum_i P(z_i)P(v|z_i)} & \text{if } P(v) > 0, \\ 0 & \text{if } P(v) = 0. \end{cases}$$

APPENDIX D:
On Finite-State Markov Chains

This appendix provides some of the basic probabilistic notions related to stationary Markov chains with a finite number of states. For detailed presentations, see Ash [Ash70], Bertsekas and Tsitsiklis [BeT02], Chung [Chu60], Gallager [Gal99], Kemeny and Snell [KeS60], and Ross [Ros85].

D.1 STATIONARY MARKOV CHAINS

A square $n \times n$ matrix $[p_{ij}]$ is said to be a *stochastic* matrix if all its elements are nonnegative, that is, $p_{ij} \geq 0$, $i, j = 1, \ldots, n$, and the sum of the elements of each of its rows is equal to 1, that is, $\sum_{j=1}^n p_{ij} = 1$ for all $i = 1, \ldots, n$.

Suppose we are given a stochastic $n \times n$ matrix P together with a finite set of states $S = \{1, \ldots, n\}$. The pair (S, P) will be referred to as a *stationary finite-state Markov chain*. We associate with (S, P) a process whereby an initial state $x_0 \in S$ is chosen in accordance with some initial probability distribution

$$r_0 = (r_0^1, r_0^2, \ldots, r_0^n).$$

Subsequently, a transition is made from state x_0 to a new state $x_1 \in S$ in accordance with a probability distribution specified by P as follows. The

probability that the new state will be j is equal to p_{ij} whenever the initial state is i, i.e.,

$$P(x_1 = j \mid x_0 = i) = p_{ij}, \qquad i, j = 1, \ldots, n.$$

Similarly, subsequent transitions produce states $x_2, x_3, \ldots$ in accordance with

$$P(x_{k+1} = j \mid x_k = i) = p_{ij}, \qquad i, j = 1, \ldots, n. \tag{D.1}$$

The probability that after the kth transition the state x_k will be j, given that the initial state x_0 is i, is denoted by

$$p_{ij}^k = P(x_k = j \mid x_0 = i), \qquad i, j = 1, \ldots, n. \tag{D.2}$$

A straightforward calculation shows that these probabilities are equal to the elements of the matrix P^k (P raised to the kth power), in the sense that p_{ij}^k is the element in the ith row and jth column of P^k:

$$P^k = [p_{ij}^k]. \tag{D.3}$$

Given the initial probability distribution p_0 of the state x_0 (viewed as a row vector in $\Re^n$), the probability distribution of the state x_k after k transitions

$$r_k = (r_k^1, r_k^2, \ldots, r_k^n)$$

(viewed again as a row vector) is given by

$$r_k = r_0 P^k, \qquad k = 1, 2, \ldots \tag{D.4}$$

This relation follows from Eqs. (D.2) and (D.3) once we write

$$r_k^j = \sum_{i=1}^n P(x_k = j \mid x_0 = i) r_0^i = \sum_{i=1}^n p_{ij}^k r_0^i.$$

D.2 CLASSIFICATION OF STATES

Given a stationary finite-state Markov chain (S, P), we say that two states i and j *communicate* if there exist two positive integers k_1 and k_2 such that $p_{ij}^{k_1} > 0$ and $p_{ji}^{k_2} > 0$. In words, states i and j communicate if one can be reached from the other with positive probability.

Let $\tilde{S} \subset S$ be a subset of states such that:

1. All states in $\tilde{S}$ communicate.

2. If $i \in \tilde{S}$ and $j \notin \tilde{S}$, then $p_{ij}^k = 0$ for all k.

Then we say that $\tilde{S}$ forms a *recurrent class* of states.

If S forms by itself a recurrent class (i.e., all states communicate with each other), then we say that the Markov chain is *irreducible*. It is possible that there exist several recurrent classes. It can also be proved that at least one recurrent class must exist. A state that belongs to some recurrent class is called *recurrent*; otherwise it is called *transient*. We have

$$\lim_{k \to \infty} p_{ii}^k = 0 \quad \text{if and only if} \quad i \text{ is transient.}$$

In other words, if the process starts at a transient state, the probability of returning to the same state after k transitions diminishes to zero as k tends to infinity.

The definitions imply that if the process starts within a recurrent class, it stays within that class. If it starts at a transient state, it eventually (with probability one) enters a recurrent class after a number of transitions, and subsequently remains there.

D.3 LIMITING PROBABILITIES

An important property of any stochastic matrix P is that the matrix P^* defined by

$$P^* = \lim_{N \to \infty} \frac{1}{N} \sum_{k=0}^{N-1} P^k \tag{D.5}$$

exists [in the sense that the sequences of the elements of $(1/N) \sum_{k=0}^{N-1} P^k$ converge to the corresponding elements of P^*]. A proof of this is given in Prop. A.1 of Appendix A in Vol. II. The elements p_{ij}^* of P^* satisfy

$$p_{ij}^* \geq 0, \qquad \sum_{j=1}^n p_{ij}^* = 1, \qquad i, j = 1, \ldots, n.$$

Thus, P^* is a stochastic matrix.

Note that the (i, j)th element of the matrix P^k is the probability that the state will be j after k transitions starting from state i. With this in mind, it can be seen from the definition (D.5) that p_{ij}^* can be interpreted as the long term fraction of time that the state is j given that the initial state is i. This suggests that for any two states i and i' in the same recurrent class we have $p_{ij}^* = p_{i'j}^*$, and this can indeed be proved. In particular, if a Markov chain is irreducible, the matrix P^* has identical rows. Also, if j is a transient state, we have

$$p_{ij}^* = 0, \qquad \text{for all } i = 1, \ldots, n,$$

so the columns of the matrix P^* corresponding to transient states consist of zeroes.

D.4 FIRST PASSAGE TIMES

Let us denote by q_{ij}^k the probability that the state will be j for the first time after exactly $k \geq 1$ transitions given that the initial state is i, that is,

$$q_{ij}^k = P(x_k = j,\ x_m \neq j,\ 1 \leq m < k \mid x_0 = i).$$

Denote also, for fixed i and j,

$$K_{ij} = \min\{k \geq 1 \mid x_k = j,\ x_0 = i\}.$$

Then K_{ij}, called the *first passage time from i to j*, may be viewed as a random variable. We have, for every $k = 1, 2, \ldots$,

$$P(K_{ij} = k) = q_{ij}^k,$$

and we write

$$P(K_{ij} = \infty) = P(x_k \neq j, k = 1, 2, \ldots \mid x_0 = i) = 1 - \sum_{k=1}^{\infty} q_{ij}^k.$$

Note that it is possible that $\sum_{k=1}^{\infty} q_{ij}^k < 1$. This will occur, for example, if j cannot be reached from i, in which case $q_{ij}^k = 0$ for all $k = 1, 2, \ldots$ The *mean first passage time* from i to j is the expected value of K_{ij}:

$$E\{K_{ij}\} = \begin{cases} \sum_{k=1}^{\infty} k q_{ij}^k & \text{if } \sum_{k=1}^{k} q_{ij}^k = 1, \\ \infty & \text{if } \sum_{k=1}^{\infty} q_{ij}^k < 1. \end{cases}$$

It may be proved that if i and j belong to the same recurrent class then

$$E\{K_{ij}\} < \infty.$$

In fact if there is only one recurrent class and t is a state of that class, the mean first passage times $E\{K_{it}\}$ are the unique solution of the following linear system of equations

$$E\{K_{it}\} = 1 + \sum_{j=1,\, j\neq t}^{n} p_{ij} E\{K_{jt}\}, \qquad i = 1, \ldots, n,\ i \neq t;$$

see Example 7.2.1. If i and j belong to two different recurrent classes, then $E\{K_{ij}\} = E\{K_{ji}\} = \infty$. If i belongs to a recurrent class and j is transient, we have $E\{K_{ij}\} = \infty$.

APPENDIX E: Kalman Filtering

In this appendix we present the basic principles of least-squares estimation and their application in estimating the state of a linear discrete-time dynamic system using measurements that are linear in the state variables.

Fundamentally, the problem is the following. There are two random vectors x and y, which are related through their joint probability distribution so that the value of one provides information about the value of the other. We get to know the value of y, and we want to estimate the value of x so that the average squared error between x and its estimate is minimized. A related problem is to find the best estimate of x within the class of all estimates that are *linear* in the measured vector y. We will specialize these problems to a case where there is an underlying linear dynamic system. In particular, we will estimate the state of the system using measurements that are obtained sequentially in time. By exploiting the special structure of the problem, the computation of the state estimate can be organized conveniently in a recursive algorithm – the Kalman filter.

E.1 LEAST-SQUARES ESTIMATION

Consider two jointly distributed random vectors x and y taking values in $\Re^n$ and $\Re^m$, respectively. We view y as a measurement that provides some information about x. Thus, while prior to knowing y our estimate of x may

have been the expected value $E\{x\}$, once the value of y is known, we want to form an updated estimate $x(y)$ of the value x. This updated estimate depends, of course, on the value of y, so we are interested in a rule that gives us the estimate for each possible value of y, i.e., we are interested in a function $x(\cdot)$, where $x(y)$ is the estimate of x given y. Such a function $x(\cdot) : \Re^m \to \Re^m$ is called an *estimator*. We are seeking an estimator that is optimal in some sense and the criterion we shall employ is based on minimization of

$$\underset{x,y}{E}\{\|x - x(y)\|^2\}. \tag{E.1}$$

Here, $\|\cdot\|$ denotes the usual norm in $\Re^n$ ($\|z\|^2 = z'z$ for $z \in \Re^n$). Furthermore, throughout the appendix, we assume that all encountered expected values are finite.

An estimator that minimizes the expected squared error above over all $x(\cdot) : \Re^n \to \Re^m$ is called a *least-squares estimator* and is denoted by $x^*(\cdot)$. Since

$$\underset{x,y}{E}\{\|x - x(y)\|^2\} = \underset{y}{E}\Big\{\underset{x}{E}\{\|x - x(y)\|^2 \mid y\}\Big\},$$

it is clear that $x^*(\cdot)$ is a least-squares estimator if $x^*(y)$ minimizes the conditional expectation in the right-hand side above for every $y \in \Re^m$, that is,

$$\underset{x}{E}\{\|x - x^*(y)\|^2 \mid y\} = \min_{z \in \Re^m} \underset{x}{E}\{\|x - z\|^2 \mid y\}, \qquad \text{for all } y \in \Re^m. \tag{E.2}$$

By carrying out this minimization, we obtain the following proposition.

Proposition E.1: The least-squares estimator $x^*(\cdot)$ is given by

$$x^*(y) = \underset{x}{E}\{x \mid y\}, \qquad \text{for all } y \in \Re^m. \tag{E.3}$$

Proof: We have for every fixed $z \in \Re^n$

$$\underset{x}{E}\{\|x - z\|^2 \mid y\} = \underset{x}{E}\{\|x\|^2 \mid y\} - 2z' \underset{x}{E}\{x \mid y\} + \|z\|^2.$$

By setting to zero the derivative with respect to z, we see that the above expression is minimized by $z = E_x\{x \mid y\}$, and the result follows. **Q.E.D.**

E.2 LINEAR LEAST-SQUARES ESTIMATION

The least-squares estimator $E_x\{x \mid y\}$ may be a complicated nonlinear function of y. As a result its practical calculation may be difficult. This motivates finding optimal estimators within the restricted class of *linear* estimators, i.e., estimators of the form

$$x(y) = Ay + b, \tag{E.4}$$

where A is an $n \times m$ matrix and b is an n-dimensional vector. An estimator

$$\hat{x}(y) = \hat{A}y + \hat{b}$$

where $\hat{A}$ and $\hat{b}$ minimize

$$\underset{x,y}{E}\{\|x - Ay - b\|^2\}$$

over all $n \times m$ matrices A and vectors $b \in \Re^n$ is called a *linear least-squares estimator*.

In the special case where x and y are jointly Gaussian random vectors it turns out that the conditional expectation $E_x\{x \mid y\}$ is a linear function of y (plus a constant vector), and as a result, a linear least-squares estimator is also a least-squares estimator. This is shown in the next proposition.

Proposition E.2: If x, y are jointly Gaussian random vectors, then the least-squares estimate $E_x\{x \mid y\}$ of x given y is linear in y.

Proof: Consider the random vector $z \in \Re^{n+m}$

$$z = \begin{pmatrix} x \\ y \end{pmatrix}$$

and assume that z is Gaussian with mean

$$\overline{z} = E\{z\} = \begin{pmatrix} E\{x\} \\ E\{y\} \end{pmatrix} = \begin{pmatrix} \overline{x} \\ \overline{y} \end{pmatrix} \tag{E.5}$$

and covariance matrix

$$\Sigma = E\big\{(z - \overline{z})(z - \overline{z})'\big\} = \begin{pmatrix} E\big\{(x - \overline{x})(x - \overline{x})'\big\} & E\big\{(x - \overline{x})(y - \overline{y})'\big\} \\ E\big\{(y - \overline{y})(x - \overline{x})'\big\} & E\big\{(y - \overline{y})(y - \overline{y})'\big\} \end{pmatrix} = \begin{pmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{pmatrix}. \tag{E.6}$$

To simplify our proof we assume that Σ is a positive definite symmetric matrix so that it possesses an inverse; the result, however, holds without this assumption. We recall that if z is Gaussian, its probability density function is of the form

$$p(z) = p(x,y) = ce^{-\frac{1}{2}(z-\overline{z})'\Sigma^{-1}(z-\overline{z})},$$

where

$$c = (2\pi)^{-(n+m)/2}(\det\Sigma)^{-1/2}$$

and $\det\Sigma$ denotes the determinant of Σ. Similarly the probability density functions of x and y are of the form

$$p(x) = c_1 e^{-\frac{1}{2}(x-\overline{x})'\Sigma_{xx}^{-1}(x-\overline{x})},$$

$$p(y) = c_2 e^{-\frac{1}{2}(y-\overline{y})'\Sigma_{yy}^{-1}(y-\overline{y})},$$

where c_1 and c_2 are appropriate constants. By Bayes' rule the conditional probability density function of x given y is

$$p(x \mid y) = \frac{p(x,y)}{p(y)} = \frac{c}{c_2} e^{-\frac{1}{2}\left((z-\overline{z})'\Sigma^{-1}(z-\overline{z}) - (y-\overline{y})'\Sigma_{yy}^{-1}(y-\overline{y})\right)}. \tag{E.7}$$

It can now be seen that there exist a positive definite symmetric $n \times n$ matrix D, an $n \times m$ matrix A, a vector $b \in \Re^n$, and a scalar s such that

$$(z-\overline{z})'\Sigma^{-1}(z-\overline{z}) - (y-\overline{y})'\Sigma_{yy}^{-1}(y-\overline{y}) = (x-Ay-b)'D^{-1}(x-Ay-b)+s. \tag{E.8}$$

This is because by substitution of the expressions for $\overline{z}$ and Σ of Eqs. (E.5) and (E.6), the left-hand side of Eq. (E.8) becomes a quadratic form in x and y, which can be put in the form indicated in the right-hand side of Eq. (E.8). In fact, by computing the inverse of Σ using the partitioned matrix inversion formula (Appendix A) it can be verified that A, b, D, and s in Eq. (E.8) have the form

$$A = \Sigma_{xy}\Sigma_{yy}^{-1}, \qquad b = \overline{x} - \Sigma_{xy}\Sigma_{yy}^{-1}\overline{y}, \qquad D = \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}, \qquad s = 0.$$

Now it follows from Eqs. (E.8) and (E.7) that the conditional expectation $E_x\{x \mid y\}$ is of the form $Ay+b$, where A is some $n \times m$ matrix and $b \in \Re^n$. **Q.E.D.**

We now turn to the characterization of the linear least-squares estimator.

Proposition E.3: Let x, y be random vectors taking values in $\Re^n$ and $\Re^m$, respectively, with given joint probability distribution. The expected values and covariance matrices of x, y are denoted by

$$E\{x\} = \overline{x} \qquad E\{y\} = \overline{y}, \tag{E.9}$$

$$E\{(x-\overline{x})(x-\overline{x})'\} = \Sigma_{xx}, \qquad E\{(y-\overline{y})(y-\overline{y})'\} = \Sigma_{yy}, \tag{E.10}$$

$$E\{(x-\overline{x})(y-\overline{y})'\} = \Sigma_{xy}, \qquad E\{(y-\overline{y})(x-\overline{x})'\} = \Sigma_{xy}', \tag{E.11}$$

and we assume that Σ_{yy} is invertible. Then the linear least-squares estimator of x given y is

$$\hat{x}(y) = \overline{x} + \Sigma_{xy}\Sigma_{yy}^{-1}(y-\overline{y}). \tag{E.12}$$

The corresponding error covariance matrix is given by

$$\underset{x,y}{E}\left\{\big(x-\hat{x}(y)\big)\big(x-\hat{x}(y)\big)'\right\} = \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}. \tag{E.13}$$

Proof: The linear least-squares estimator is defined as

$$\hat{x}(y) = \hat{A}y + \hat{b},$$

where $\hat{A}, \hat{b}$ minimize the function $f(A,b) = E_{x,y}\{\|x-Ay-b\|^2\}$ over A and b. Taking the derivatives of $f(A,b)$ with respect to A and b and setting them to zero, we obtain the two conditions

$$0 = \frac{\partial f}{\partial A}\Big|_{\hat{A},\hat{b}} = 2\,\underset{x,y}{E}\{(\hat{b}+\hat{A}y-x)y'\}, \tag{E.14}$$

$$0 = \frac{\partial f}{\partial b}\Big|_{\hat{A},\hat{b}} = 2\,\underset{x,y}{E}\{\hat{b}+\hat{A}y-x\}. \tag{E.15}$$

The second condition yields

$$\hat{b} = \overline{x} - \hat{A}\overline{y}, \tag{E.16}$$

and by substitution in the first, we obtain

$$\underset{x,y}{E}\left\{y\big(\hat{A}(y-\overline{y}) - (x-\overline{x})\big)'\right\} = 0. \tag{E.17}$$

We have

$$\underset{x,y}{E}\{\hat{A}(y-\overline{y}) - (x-\overline{x})\}' = 0,$$

so that

$$\overline{y}\ \underset{x,y}{E}\left\{\hat{A}(y-\overline{y})-(x-\overline{x})\right\}'=0. \tag{E.18}$$

By subtracting Eq. (E.18) from Eq. (E.17), we obtain

$$\underset{x,y}{E}\left\{(y-\overline{y})\big(\hat{A}(y-\overline{y})-(x-\overline{x})\big)'\right\}=0.$$

Equivalently,

$$\Sigma_{yy}\hat{A}'-\Sigma_{yx}=0,$$

from which

$$\hat{A}=\Sigma_{yx}'\Sigma_{yy}^{-1}=\Sigma_{xy}\Sigma_{yy}^{-1}. \tag{E.19}$$

Using the expressions (E.16) and (E.19) for $\hat{b}$ and $\hat{A}$, respectively, we obtain

$$\hat{x}(y)=\hat{A}y+\hat{b}=\overline{x}+\Sigma_{xy}\Sigma_{yy}^{-1}(y-\overline{y}),$$

which was to be proved. The desired Eq. (E.13) for the error covariance follows upon substitution of the expression for $\hat{x}(y)$ obtained above. **Q.E.D.**

We list some of the properties of the least-squares estimator as corollaries.

Corollary E.3.1: The linear least-squares estimator is unbiased, i.e.,

$$\underset{y}{E}\{\hat{x}(y)\}=\overline{x}.$$

Proof: This follows from Eq. (E.12). **Q.E.D.**

Corollary E.3.2: The estimation error $x-\hat{x}(y)$ is uncorrelated with both y and $\hat{x}(y)$, i.e.,

$$\underset{x,y}{E}\left\{y\big(x-\hat{x}(y)\big)'\right\}=0,$$

$$\underset{x,y}{E}\left\{\hat{x}(y)\big(x-\hat{x}(y)\big)'\right\}=0.$$

Proof: The first equality is Eq. (E.14). The second equality can be written as

$$\underset{x,y}{E}\left\{(\hat{A}y+\hat{b})\big(x-\hat{x}(y)\big)'\right\}=0$$

and follows from the first equality and Cor. E.3.1. **Q.E.D.**

Corollary E.3.2 is known as the *orthogonal projection principle*. It states a property that characterizes the linear least-squares estimate and forms the basis for an alternative treatment of least-squares estimation as a problem of projection in a Hilbert space of random variables (see Luenberger [Lue69]).

Corollary E.3.3: Consider in addition to x and y, the random vector z defined by

$$z=Cx,$$

where C is a given $p\times m$ matrix. Then the linear least-squares estimate of z given y is

$$\hat{z}(y)=C\hat{x}(y),$$

and the corresponding error covariance matrix is given by

$$\underset{z,y}{E}\left\{\big(z-\hat{z}(y)\big)\big(z-\hat{z}(y)\big)'\right\}=C\ \underset{x,y}{E}\left\{\big(x-\hat{x}(y)\big)\big(x-\hat{x}(y)\big)'\right\}C'.$$

Proof: We have $E\{z\}=\overline{z}=C\,\overline{x}$ and

$$\begin{aligned}\Sigma_{zz}&=\underset{z}{E}\{(z-\overline{z})(z-\overline{z})'\}=C\Sigma_{xx}C',\\ \Sigma_{zy}&=\underset{z,y}{E}\{(z-\overline{z})(y-\overline{y})'\}=C\Sigma_{xy},\\ \Sigma_{yz}&=\Sigma_{zy}'=\Sigma_{yx}C'.\end{aligned}$$

By Prop. E.3 we have

$$\hat{z}(y)=\overline{z}+\Sigma_{zy}\Sigma_{yy}^{-1}(y-\overline{y})=C\,\overline{x}+C\Sigma_{xy}\Sigma_{yy}^{-1}(y-\overline{y})=C\hat{x}(y),$$

$$\begin{aligned}\underset{x,y}{E}\left\{\big(z-\hat{z}(y)\big)\big(z-\hat{z}(y)\big)'\right\}&=\Sigma_{zz}-\Sigma_{zy}\Sigma_{yy}^{-1}\Sigma_{yz}\\ &=C\big(\Sigma_{xx}-\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}\big)C'\\ &=C\ \underset{x,y}{E}\left\{\big(x-\hat{x}(y)\big)\big(x-\hat{x}(y)\big)'\right\}C'.\end{aligned}$$

Q.E.D.

Corollary E.3.4: Consider in addition to x and y, an additional random vector z of the form

$$z = Cy + u, \tag{E.20}$$

where C is a given $p \times m$ matrix of rank p and u is a given vector in $\Re^p$. Then the linear least-squares estimate $\hat{x}(z)$ of x given z is

$$\hat{x}(z) = \overline{x} + \Sigma_{xy} C'(C\Sigma_{yy}C')^{-1}(z - C\overline{y} - u), \tag{E.21}$$

and the corresponding error covariance matrix is

$$\underset{x,z}{E}\left\{\big(x - \hat{x}(z)\big)\big(x - \hat{x}(z)\big)'\right\} = \Sigma_{xx} - \Sigma_{xy}C'(C\Sigma_{yy}C')^{-1}C\Sigma_{yx}. \tag{E.22}$$

Proof: We have

$$\overline{z} = E\{z\} = C\overline{y} + u, \tag{E.23a}$$

$$\Sigma_{zz} = E\big\{(z - \overline{z})(z - \overline{z})'\big\} = C\Sigma_{yy}C', \tag{E.23b}$$

$$\Sigma_{zx} = E\big\{(z - \overline{z})(x - \overline{x})'\big\} = C\Sigma_{yx}, \tag{E.23c}$$

$$\Sigma_{xz} = E\big\{(x - \overline{x})(z - \overline{z})'\big\} = \Sigma_{xy}C'. \tag{E.23d}$$

From Prop. E.3 we have

$$\hat{x}(z) = \overline{x} + \Sigma_{xz}\Sigma_{zz}^{-1}(z - \overline{z}), \tag{E.24a}$$

$$\underset{x,z}{E}\left\{\big(x - \hat{x}(z)\big)\big(x - \hat{x}(z)\big)'\right\} = \Sigma_{xx} - \Sigma_{xz}\Sigma_{zz}^{-1}\Sigma_{zx}, \tag{E.24b}$$

where $\Sigma_{zz} = C\Sigma_{yy}C'$ has an inverse, since Σ_{yy} is invertible and C has rank p. By substituting the relations (E.23) into Eqs. (E.24a) and (E.24b) the result follows. **Q.E.D.**

Frequently we want to estimate a vector of parameters $x \in \Re^n$ given a measurement vector $z \in \Re^m$ of the form $z = Cx + v$, where C is a given $m \times n$ matrix, and $v \in \Re^m$ is a random measurement error vector. The following corollary gives the linear least-squares estimate $\hat{x}(z)$ and its error covariance.

Corollary E.3.5: Let

$$z = Cx + v,$$

where C is a given $m \times n$ matrix, and the random vectors $x \in \Re^n$ and $v \in \Re^m$ are uncorrelated. Denote

$$E\{x\} = \overline{x}, \qquad E\big\{(x - \overline{x})(x - \overline{x})'\big\} = \Sigma_{xx},$$

$$E\{v\} = \overline{v}, \qquad E\big\{(v - \overline{v})(v - \overline{v})'\big\} = \Sigma_{vv},$$

and assume further that Σ_{vv} is a positive definite matrix. Then

$$\hat{x}(z) = \overline{x} + \Sigma_{xx}C'(C\Sigma_{xx}C' + \Sigma_{vv})^{-1}(z - C\overline{x} - \overline{v}),$$

$$\underset{x,v}{E}\left\{\big(x - \overline{x}(z)\big)\big(x - \overline{x}(z)\big)'\right\} = \Sigma_{xx} - \Sigma_{xx}C'(C\Sigma_{xx}C' + \Sigma_{vv})^{-1}C\Sigma_{xx}.$$

Proof: Define

$$y = (x' \quad v')', \qquad \overline{y} = (\overline{x}' \quad \overline{v}')', \qquad \tilde{C} = (C \quad I).$$

Then we have $z = \tilde{C}y$, and by Cor. E.3.3,

$$\hat{x}(z) = (I \quad 0)\,\hat{y}(z),$$

$$E\Big\{\big(x - \hat{x}(z)\big)\big(x - \hat{x}(z)\big)'\Big\} = (I \quad 0)\,E\Big\{\big(y - \hat{y}(z)\big)\big(y - \hat{y}(z)\big)'\Big\}\begin{pmatrix} I \\ 0 \end{pmatrix},$$

where $\hat{y}(z)$ is the linear least-squares estimate of y given z. By applying Cor. E.3.4 with $u = 0$ and $x = y$ we obtain

$$\hat{y}(z) = \overline{y} + \Sigma_{yy}\tilde{C}'(\tilde{C}\Sigma_{yy}\tilde{C}')^{-1}(z - \tilde{C}\overline{y}),$$

$$E\Big\{\big(y - \hat{y}(z)\big)\big(y - \hat{y}(z)\big)'\Big\} = \Sigma_{yy} - \Sigma_{yy}\tilde{C}'(\tilde{C}\Sigma_{yy}\tilde{C}')^{-1}\tilde{C}\Sigma_{yy}.$$

By using the equations

$$\Sigma_{yy} = \begin{pmatrix} \Sigma_{xx} & 0 \\ 0 & \Sigma_{vv} \end{pmatrix}, \qquad \tilde{C} = (C \quad I),$$

and by carrying out the straightforward calculation the result follows. **Q.E.D.**

The next two corollaries deal with least-squares estimates involving multiple measurement vectors that are obtained sequentially. In particular,

the corollaries show how to modify an existing least-squares estimate $\hat{x}(y)$ to obtain $\hat{x}(y,z)$ once an additional vector z becomes known. This is a central operation in Kalman filtering.

Corollary E.3.6: Consider in addition to x and y, an additional random vector z taking values in $\Re^p$, which is uncorrelated with y. Then the linear least-squares estimate $\hat{x}(y,z)$ of x given y *and* z [i.e., given the composite vector (y,z)] has the form

$$\hat{x}(y,z) = \hat{x}(y) + \hat{x}(z) - \overline{x}, \tag{E.25}$$

where $\hat{x}(y)$ and $\hat{x}(z)$ are the linear least-squares estimates of x given y and given z, respectively. Furthermore,

$$\mathop{E}_{x,y,z}\left\{\big(x-\hat{x}(y,z)\big)\big(x-\hat{x}(y,z)\big)'\right\} = \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx} - \Sigma_{xz}\Sigma_{zz}^{-1}\Sigma_{zx}, \tag{E.26}$$

where

$$\Sigma_{xz} = \mathop{E}_{x,z}\big\{(x-\overline{x})(z-\overline{z})'\big\}, \qquad \Sigma_{zx} = \mathop{E}_{x,z}\big\{(z-\overline{z})(x-\overline{x})'\big\},$$

$$\Sigma_{zz} = \mathop{E}_{z}\big\{(z-\overline{z})(z-\overline{z})'\big\}, \qquad \overline{z} = \mathop{E}_{z}\{z\},$$

and it is assumed that Σ_{zz} is invertible.

Proof: Let

$$w = \begin{pmatrix} y \\ z \end{pmatrix}, \qquad \overline{w} = \begin{pmatrix} \overline{y} \\ \overline{z} \end{pmatrix}.$$

By Eq. (E.12) we have

$$\hat{x}(w) = \overline{x} + \Sigma_{xw}\Sigma_{ww}^{-1}(w-\overline{w}). \tag{E.27}$$

Furthermore

$$\Sigma_{xw} = [\Sigma_{xy},\ \Sigma_{xz}],$$

and since y and z are uncorrelated, we have

$$\Sigma_{ww} = \begin{pmatrix} \Sigma_{yy} & 0 \\ 0 & \Sigma_{zz} \end{pmatrix}.$$

Substituting the above expressions in Eq. (E.27), we obtain

$$\hat{x}(w) = \overline{x} + \Sigma_{xy}\Sigma_{yy}^{-1}(y-\overline{y}) + \Sigma_{xz}\Sigma_{zz}^{-1}(z-\overline{z}) = \hat{x}(y) + \hat{x}(z) - \overline{x},$$

and Eq. (E.25) is proved. The proof of Eq. (E.26) is similar by using the relations above and the covariance Eq. (E.13). **Q.E.D.**

Corollary E.3.7: Let z be as in the preceding corollary and assume that y and z are not necessarily uncorrelated, that is, we may have

$$\Sigma_{yz} = \Sigma_{zy}' = \mathop{E}_{y,z}\big\{(y-\overline{y})(z-\overline{z})'\big\} \neq 0.$$

Then

$$\hat{x}(y,z) = \hat{x}(y) + \hat{x}\big(z-\hat{z}(y)\big) - \overline{x}, \tag{E.28}$$

where $\hat{x}\big(z-\hat{z}(y)\big)$ denotes the linear least-squares estimate of x given the random vector $z-\hat{z}(y)$ and $\hat{z}(y)$ is the linear least-squares estimate of z given y. Furthermore,

$$\mathop{E}_{x,y,z}\left\{\big(x-\hat{x}(y,z)\big)\big(x-\hat{x}(y,z)\big)'\right\} = \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx} - \hat{\Sigma}_{xz}\hat{\Sigma}_{zz}^{-1}\hat{\Sigma}_{zx}, \tag{E.29}$$

where

$$\hat{\Sigma}_{xz} = \mathop{E}_{x,y,z}\left\{\big(x-\overline{x}\big)\big(z-\hat{z}(y)\big)'\right\},$$

$$\hat{\Sigma}_{zz} = \mathop{E}_{y,z}\left\{\big(z-\hat{z}(y)\big)\big(z-\hat{z}(y)\big)'\right\},$$

$$\hat{\Sigma}_{zx} = \mathop{E}_{x,y,z}\left\{\big(z-\hat{z}(y)\big)(x-\overline{x})'\right\}.$$

Proof: It can be seen that, since $\hat{z}(y)$ is a linear function of y, the linear least-squares estimate of x given y and z is the same as the linear least-squares estimate of x given y and $z-\hat{z}(y)$. By Cor. E.3.2 the random vectors y and $z-\hat{z}(y)$ are uncorrelated. Given this observation the result follows by application of the preceding corollary. **Q.E.D.**

E.3 STATE ESTIMATION – THE KALMAN FILTER

Consider now a linear dynamic system of the type considered in Section 5.2 but without a control vector ($u_k \equiv 0$)

$$x_{k+1} = A_k x_k + w_k, \qquad k = 0, 1, \ldots, N-1, \tag{E.30}$$

where $x_k \in \Re^n$ and $w_k \in \Re^n$ denote the state and random disturbance vectors, respectively, and the matrices A_k are known. Consider also the

measurement equation

$$z_k = C_k x_k + v_k, \qquad k = 0, 1, \ldots, N-1, \tag{E.31}$$

where $z_k \in \Re^s$ and $v_k \in \Re^s$ are the observation and observation noise vectors, respectively.

We assume that $x_0, w_0, \ldots, w_{N-1}, v_0, \ldots, v_{N-1}$ are independent random vectors with given probability distributions and that

$$E\{w_k\} = E\{v_k\} = 0, \qquad k = 0, 1, \ldots, N-1. \tag{E.32}$$

We use the notation

$$S = E\Big\{\big(x_0 - E\{x_0\}\big)\big(x_0 - E\{x_0\}\big)'\Big\}, \quad M_k = E\{w_k w_k'\}, \quad N_k = E\{v_k v_k'\}, \tag{E.33}$$

and we assume that N_k is positive definite for all k.

A Nonrecursive Least-Squares Estimate

We first give a straightforward but somewhat tedious method to derive the linear least-squares estimate of x_{k+1} or x_k given the values of $z_0, z_1, \ldots, z_k$. Let us denote

$$Z_k = (z_0', z_1', \ldots, z_k')', \qquad r_{k-1} = (x_0', w_0', w_1', \ldots, w_{k-1}')'.$$

In this method, we first find the linear least-squares estimate of r_{k-1} given Z_k, and we then obtain the linear least-squares estimate of x_k given Z_k after expressing x_k as a linear function of r_{k-1}.

For each i with $0 \le i \le k$ we have, by using the system equation,

$$x_{i+1} = L_i r_i,$$

where L_i is the $n \times \big(n(i+1)\big)$ matrix

$$L_i = \big(\, A_i \cdots A_0, \quad A_i \cdots A_1, \quad \ldots \quad , A_i, \quad I \,\big).$$

As a result we may write

$$Z_k = \Phi_{k-1} r_{k-1} + V_k,$$

where

$$V_k = (v_0', v_1', \ldots, v_k')',$$

and Φ_{k-1} is an $s(k+1) \times (nk)$ matrix of the form

$$\Phi_{k-1} = \begin{pmatrix} C_0 & 0 \\ C_1 L_0 & 0 \\ \vdots & \vdots \\ C_{k-1} L_{k-2} & 0 \\ C_k L_{k-1} & \end{pmatrix}.$$

We can thus use Cor. E.3.5, the equations above, and the data of the problem to compute

$$\hat{r}_{k-1}(Z_k) \quad \text{and} \quad E\Big\{\big(r_{k-1} - \hat{r}_{k-1}(Z_k)\big)\big(r_{k-1} - \hat{r}_{k-1}(Z_k)\big)'\Big\}.$$

Let us denote the linear least-squares estimates of x_{k+1} and x_k given Z_k by $\hat{x}_{k+1|k}$ and $\hat{x}_{k|k}$, respectively. We can now obtain $\hat{x}_{k|k} = \hat{x}_k(Z_k)$ and the corresponding error covariance matrix by using Cor. E.3.3, that is,

$$\hat{x}_{k|k} = L_{k-1} \hat{r}_{k-1}(Z_k),$$

$$E\big\{(x_k - \hat{x}_{k|k})(x_k - \hat{x}_{k|k})'\big\} = L_{k-1} E\Big\{\big(r_{k-1} - \hat{r}_{k-1}(Z_k)\big)\big(r_{k-1} - \hat{r}_{k-1}(Z_k)\big)'\Big\} L_{k-1}'.$$

These equations may in turn be used to yield $\hat{x}_{k+1|k}$ and the corresponding error covariance again via Cor. E.3.3.

The Kalman Filtering Algorithm

The preceding method for obtaining the least-squares estimate of x_k is cumbersome when the number of measurements is large. Fortunately, the sequential structure of the problem can be exploited and the computations can be organized conveniently, as first proposed by Kalman [Kal60]. The main attractive feature of the Kalman filtering algorithm is that the estimate $\hat{x}_{k+1|k}$ can be obtained by means of a simple equation that involves the previous estimate $\hat{x}_{k|k-1}$ and the new measurement z_k but *does not involve any of the past measurements* $z_0, z_1, \ldots, z_{k-1}$.

Suppose that we have computed the estimate $\hat{x}_{k|k-1}$ together with the covariance matrix

$$\Sigma_{k|k-1} = E\big\{(x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})'\big\}. \tag{E.34}$$

At time k we receive the additional measurement

$$z_k = C_k x_k + v_k.$$

We may use now Cor. E.3.7 to compute the linear least-squares estimate of x_k given $Z_{k-1} = (z_0', z_1', \ldots, z_{k-1}')'$ *and* z_k. This estimate is denoted by $\hat{x}_{k|k}$ and, by Cor. E.3.7, it is given by

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \hat{x}_k\big(z_k - \hat{z}_k(Z_{k-1})\big) - E\{x_k\}, \tag{E.35}$$

where $\hat{z}_k(Z_{k-1})$ denotes the linear least-squares estimate of z_k given Z_{k-1} and $\hat{x}_k\big(z_k - \hat{z}_k(Z_{k-1})\big)$ denotes the linear least-squares estimate of x_k given $\big(z_k - \hat{z}_k(Z_{k-1})\big)$.

We now calculate the term $\hat{x}_k\big(z_k - \hat{z}_k(Z_{k-1})\big)$ in Eq. (E.35). We have by Eqs. (E.31), (E.32), and Cor. E.3.3,

$$\hat{z}_k(Z_{k-1}) = C_k \hat{x}_{k|k-1}. \tag{E.36}$$

Also we use Cor. E.3.3 to obtain

$$E\Big\{\big(z_k - \hat{z}_k(Z_{k-1})\big)\big(z_k - \hat{z}_k(Z_{k-1})\big)'\Big\} = C_k \Sigma_{k|k-1} C_k' + N_k, \tag{E.37}$$

$$\begin{aligned} &E\Big\{x_k\big(z_k - \hat{z}_k(Z_{k-1})\big)'\Big\} \\ &\quad = E\Big\{x_k\big(C_k(x_k - \hat{x}_{k|k-1})\big)'\Big\} + E\{x_k v_k'\} \\ &\quad = E\Big\{(x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})'\Big\}C_k' + E\Big\{\hat{x}_{k|k-1}(x_k - \hat{x}_{k|k-1})'\Big\}C_k'. \end{aligned}$$

The last term in the right-hand side above is zero by Cor. E.3.2, so by using Eq. (E.34) we have

$$E\Big\{x_k\big(z_k - \hat{z}_k(Z_{k-1})\big)'\Big\} = \Sigma_{k|k-1} C_k'. \tag{E.38}$$

Using Eqs. (E.36)-(E.38) in Prop. E.3, we obtain

$$\hat{x}_k\big(z_k - \hat{z}_k(Z_{k-1})\big) = E\{x_k\} + \Sigma_{k|k-1} C_k' (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1}(z_k - C_k \hat{x}_{k|k-1}),$$

and Eq. (E.35) is written as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \Sigma_{k|k-1} C_k' (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1}(z_k - C_k \hat{x}_{k|k-1}). \tag{E.39}$$

By using Cor. E.3.3 we also have

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k}. \tag{E.40}$$

Concerning the covariance matrix $\Sigma_{k+1|k}$, we have from Eqs. (E.30), (E.32), (E.33), and Cor. E.3.3,

$$\Sigma_{k+1|k} = A_k \Sigma_{k|k} A_k' + M_k, \tag{E.41}$$

where

$$\Sigma_{k|k} = E\Big\{(x_k - \hat{x}_{k|k})(x_k - \hat{x}_{k|k})'\Big\}.$$

The error covariance matrix $\Sigma_{k|k}$ may be computed via Cor. E.3.7 similar to $\hat{x}_{k|k}$ [cf. Eq. (E.35)]. Thus, we have from Eqs. (E.29), (E.37), (E.38)

$$\Sigma_{k|k} = \Sigma_{k|k-1} - \Sigma_{k|k-1} C_k' (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1} C_k \Sigma_{k|k-1}. \tag{E.42}$$

Equations (E.39)-(E.42) with the initial conditions [cf. Eq. (E.33)]

$$\hat{x}_{0|-1} = E\{x_0\}, \qquad \Sigma_{0|-1} = S, \tag{E.43}$$

constitute the *Kalman filtering algorithm*. This algorithm recursively generates the linear least-squares estimates $\hat{x}_{k+1|k}$ or $\hat{x}_{k|k}$ together with the associated error covariance matrices $\Sigma_{k+1|k}$ or $\Sigma_{k|k}$. In particular, given $\Sigma_{k|k-1}$ and $\hat{x}_{k|k-1}$, Eqs. (E.39) and (E.42) yield $\Sigma_{k|k}$ and $\hat{x}_{k|k}$, and then Eqs. (E.41) and (E.40) yield $\Sigma_{k+1|k}$ and $\hat{x}_{k+1|k}$.

An alternative expression for Eq. (E.39) is

$$\hat{x}_{k|k} = A_{k-1}\hat{x}_{k-1|k-1} + \Sigma_{k|k} C_k' N_k^{-1}(z_k - C_k A_{k-1} \hat{x}_{k-1|k-1}), \tag{E.44}$$

which can be obtained from Eqs. (E.39) and (E.40) by using the following equality

$$\Sigma_{k|k} C_k' N_k^{-1} = \Sigma_{k|k-1} C_k' (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1}. \tag{E.45}$$

This equality may be verified by using Eq. (E.42) to write

$$\begin{aligned} \Sigma_{k|k} C_k' N_k^{-1} &= \big(\Sigma_{k|k-1} - \Sigma_{k|k-1} C_k' (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1} C_k \Sigma_{k|k-1}\big) C_k' N_k^{-1} \\ &= \Sigma_{k|k-1} C_k' \big(N_k^{-1} - (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1} C_k \Sigma_{k|k-1} C_k' N_k^{-1}\big), \end{aligned}$$

and then use in the above formula the following calculation

$$\begin{aligned} N_k^{-1} &= (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1}(C_k \Sigma_{k|k-1} C_k' + N_k) N_k^{-1} \\ &= (C_k \Sigma_{k|k-1} C_k' + N_k)^{-1}(C_k \Sigma_{k|k-1} C_k' N_k^{-1} + I). \end{aligned}$$

When the system equation contains a control vector u_k,

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

it is straightforward to show that Eq. (E.44) takes the form

$$\begin{aligned} \hat{x}_{k|k} = {} & A_{k-1}\hat{x}_{k-1|k-1} + B_{k-1}u_{k-1} \\ & + \Sigma_{k|k} C_k' N_k^{-1}(z_k - C_k A_{k-1} \hat{x}_{k-1|k-1} - C_k B_{k-1} u_{k-1}), \end{aligned} \tag{E.46}$$

where $\hat{x}_{k|k}$ is the linear least-squares estimate of x_k given $z_0, z_1, \ldots, z_k$ and $u_0, u_1, \ldots, u_{k-1}$. The equations (E.41)-(E.43) that generate $\Sigma_{k|k}$ remain unchanged.

Steady-State Kalman Filtering Algorithm

Finally we note that Eqs. (E.41) and (E.42) yield

$$\Sigma_{k+1|k} = A_k\big(\Sigma_{k|k-1} - \Sigma_{k|k-1}C_k'(C_k\Sigma_{k|k-1}C_k' + N_k)^{-1}C_k\Sigma_{k|k-1}\big)A_k' + M_k, \tag{E.47}$$

with the initial condition $\Sigma_{0|-1} = S$. This equation is a matrix Riccati equation of the type considered in Section 4.1. Thus when A_k, C_k, N_k, and M_k are constant matrices,

$$A_k = A, \quad C_k = C, \quad N_k = N, \quad M_k = M, \qquad k = 0, 1, \ldots, N-1,$$

we have by invoking the proposition proved there, that $\Sigma_{k+1|k}$ tends to a positive definite symmetric matrix Σ that solves the algebraic Riccati equation

$$\Sigma = A\big(\Sigma - \Sigma C'(C\Sigma C' + N)^{-1}C\Sigma\big)A' + M,$$

assuming observability of the pair (A, C) and controllability of the pair (A, D), where $M = DD'$. Under the same conditions, we have $\Sigma_{k|k} \to \overline{\Sigma}$, where from Eq. (E.42),

$$\overline{\Sigma} = \Sigma - \Sigma C'(C\Sigma C' + N)^{-1}C\Sigma.$$

We may then write the Kalman filter recursion [cf. Eq. (E.44)] in the asymptotic form

$$\hat{x}_{k|k} = A\hat{x}_{k-1|k-1} + \overline{\Sigma}C'N^{-1}(z_k - CA\hat{x}_{k-1|k-1}). \tag{E.48}$$

This estimator is simple and convenient for implementation.

E.4 STABILITY ASPECTS

Let us consider now the stability properties of the steady-state form of the Kalman filter. From Eqs. (E.39) and (E.40), we have

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k-1} + A\Sigma C'(C\Sigma C' + N)^{-1}(z_k - C\hat{x}_{k|k-1}). \tag{E.49}$$

Let e_k denote the "one-step prediction" error

$$e_k = x_k - \hat{x}_{k|k-1}.$$

By using Eq. (E.49), the system equation

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

and the measurement equation

$$z_k = Cx_k + v_k,$$

we obtain

$$e_{k+1} = \big(A - A\Sigma C'(C\Sigma C' + N)^{-1}C\big)e_k + w_k - A\Sigma C'(C\Sigma C' + N)^{-1}v_k. \tag{E.50}$$

From the practical point of view it is important that the error equation (E.50) represents a stable system, that is, the matrix

$$A - A\Sigma C'(C\Sigma C' + N)^{-1}C \tag{E.51}$$

has eigenvalues strictly within the unit circle. This, however, follows by Prop. 4.4.1 of Section 4.1 under the observability and controllability assumptions given earlier, since Σ is the unique positive semidefinite symmetric solution of the algebraic Riccati equation

$$\Sigma = A\big(\Sigma - \Sigma C'(C\Sigma C' + N)^{-1}C\Sigma\big)A' + M.$$

Actually this proposition yields that the transpose of the matrix (E.51) has eigenvalues strictly within the unit circle, but this is sufficient for our purposes since the eigenvalues of a matrix are the same as those of its transpose.

Let us consider also the stability properties of the equation governing the estimation error

$$\tilde{e}_k = x_k - \hat{x}_{k|k}.$$

We have by a straightforward calculation

$$\tilde{e}_k = \big(I - \Sigma C'(C\Sigma C' + N)^{-1}C\big)e_k - \Sigma C'(C\Sigma C' + N)^{-1}v_k. \tag{E.52}$$

By multiplying both sides of Eq. (E.50) by $I - \Sigma C'(C\Sigma C' + N)^{-1}C$ and by using Eq. (E.52), we obtain

$$\begin{aligned}\tilde{e}_{k+1} &+ \Sigma C'(C\Sigma C' + N)^{-1}v_{k+1} \\ &= \big(A - \Sigma C'(C\Sigma C' + N)^{-1}CA\big)\big(\tilde{e}_k + \Sigma C'(C\Sigma C' + N)^{-1}v_k\big) \\ &\quad + \big(I - \Sigma C'(C\Sigma C' + N)^{-1}C\big)\big(w_k - A\Sigma C'(C\Sigma C' + N)^{-1}v_k\big),\end{aligned}$$

or equivalently

$$\begin{aligned}\tilde{e}_{k+1} &= \big(A - \Sigma C'(C\Sigma C' + N)^{-1}CA\big)\tilde{e}_k \\ &\quad + \big(I - \Sigma C'(C\Sigma C' + N)^{-1}C\big)w_k - \Sigma C'(C\Sigma C' + N)^{-1}v_{k+1}.\end{aligned} \tag{E.53}$$

Since the matrix (E.51) has eigenvalues strictly within the unit circle, the sequence $\{e_k\}$ generated by Eq. (E.50) tends to zero whenever the vectors w_k and v_k are identically zero for all k. Hence, by Eq. (E.52), the same is true for the sequence $\{\tilde{e}_k\}$. It follows from Eq. (E.53) that the matrix

$$A - \Sigma C'(C\Sigma C' + N)^{-1}CA \tag{E.54}$$

has eigenvalues strictly within the unit circle, and the estimation error sequence $\{\tilde{e}_k\}$ is generated by a stable system.

Let us finally consider the stability properties of the $2n$-dimensional system of equations with state vector $(x_k', \hat{x}_k')$:

$$x_{k+1} = Ax_k + BL\hat{x}_k, \tag{E.55}$$

$$\hat{x}_{k+1} = \overline{\Sigma}C'N^{-1}CAx_k + (A + BL - \overline{\Sigma}C'N^{-1}CA)\hat{x}_k. \tag{E.56}$$

This is the steady-state, asymptotically optimal closed-loop system that was encountered at the end of Section 5.2.

We assume that the appropriate observability and controllability assumptions stated there are in effect. By using the equation

$$\overline{\Sigma}C'N^{-1} = \Sigma C'(C\Sigma C' + N)^{-1},$$

shown earlier, we obtain from Eqs. (E.55) and (E.56) that

$$(x_{k+1} - \hat{x}_{k+1}) = \big(A - \Sigma C'(C\Sigma C' + N)^{-1}CA\big)(x_k - \hat{x}_k).$$

Since we have proved that the matrix (E.54) has eigenvalues strictly within the unit circle, it follows that

$$\lim_{k\to\infty}(x_{k+1} - \hat{x}_{k+1}) = 0, \tag{E.57}$$

for arbitrary initial states x_0 and $\hat{x}_0$. From Eq. (E.55) we obtain

$$x_{k+1} = (A + BL)x_k + BL(\hat{x}_k - x_k). \tag{E.58}$$

Since in accordance with the theory of Section 4.1 the matrix $(A + BL)$ has eigenvalues strictly within the unit circle, it follows from Eqs. (E.57) and (E.58) that we have

$$\lim_{k\to\infty} x_k = 0 \tag{E.59}$$

and hence from Eq. (E.57),

$$\lim_{k\to\infty} \hat{x}_k = 0. \tag{E.60}$$

Since the equations above hold for arbitrary initial states x_0 and $\hat{x}_0$ it follows that the system defined by Eqs. (E.55) and (E.56) is stable.

E.5 GAUSS-MARKOV ESTIMATORS

Suppose that we want to estimate a vector $x \in \Re^n$ given a measurement vector $z \in \Re^m$ that is related to x by

$$z = Cx + v, \tag{E.61}$$

where C is a given $m \times n$ matrix with rank m, and v is a random measurement error vector. Let us assume that v is uncorrelated with x, and has a known mean and a positive definite covariance matrix

$$E\{v\} = \overline{v}, \qquad E\big\{(v - \overline{v})(v - \overline{v})'\big\} = \Sigma_{vv}. \tag{E.62}$$

If the a priori probability distribution of x is known, we can obtain a linear least-squares estimate of x given z by using the theory of Section E.2 (cf. Cor. E.3.5). In many cases, however, the probability distribution of x is unknown. In such cases we can use the Gauss-Markov estimator, which is optimal within the class of linear estimators that satisfy certain restrictions, as described below.

Let us consider an estimator of the form

$$\hat{x}(z) = \hat{A}(z - \overline{v}),$$

where $\hat{A}$ minimizes

$$f(A) = \underset{x,z}{E}\big\{\|x - A(z - \overline{v})\|^2\big\} \tag{E.63}$$

over all $n \times m$ matrices A. Since x and v are uncorrelated, we have using Eqs. (E.61)-(E.63)

$$\begin{aligned} f(A) &= \underset{x,v}{E}\big\{\|x - ACx - A(v - \overline{v})\|^2\big\} \\ &= \underset{x}{E}\big\{\|(I - AC)x\|^2\big\} + \underset{v}{E}\big\{\|A(v - \overline{v})\|^2\big\}, \end{aligned}$$

where I is the $n \times n$ identity matrix. Since $f(A)$ depends on the unknown statistics of x, we see that the optimal matrix $\hat{A}$ also depends on these statistics. We can circumvent this difficulty by requiring that

$$AC = I.$$

Then our problem becomes

$$\begin{aligned} &\text{minimize} \quad \underset{v}{E}\big\{\|A(v - \overline{v})\|^2\big\} \\ &\text{subject to} \quad AC = I. \end{aligned} \tag{E.64}$$

Note that the requirement $AC = I$ is not only convenient analytically, but also makes sense conceptually. In particular, it is equivalent to requiring that the estimator $x(z) = A(z - \overline{v})$ be *unbiased* in the sense that

$$E\{x(z)\} = E\{x\} = \overline{x}, \qquad \text{for all } \overline{x} \in \Re^n.$$

This can be seen by writing

$$E\{x(z)\} = E\{A(Cx + v - \overline{v})\} = ACE\{x\} = AC\,\overline{x} = \overline{x}.$$

To derive the optimal solution $\hat{A}$ of problem (E.64), let a_i' denote the ith row of A. We have

$$\|A(v - \overline{v})\|^2 = (v - \overline{v})' \begin{pmatrix} a_1 & \cdots & a_n \end{pmatrix} \begin{pmatrix} a_1' \\ \vdots \\ a_n' \end{pmatrix} (v - \overline{v})$$

$$= \sum_{i=1}^{n} (v - \overline{v})' a_i a_i' (v - \overline{v})$$

$$= \sum_{i=1}^{n} a_i' (v - \overline{v})(v - \overline{v})' a_i.$$

Hence, the minimization problem (E.64) can also be written as

$$\begin{aligned} &\text{minimize} && \sum_{i=1}^{n} a_i' \Sigma_{vv} a_i \\ &\text{subject to} && C' a_i = e_i, \qquad i = 1, \ldots, n, \end{aligned}$$

where e_i is the ith column of the identity matrix. The minimization can be carried out separately for each i, yielding

$$\hat{x}_i = \Sigma_{vv}^{-1} C (C' \Sigma_{vv} C)^{-1} e_i, \qquad i = 1, \ldots, n,$$

and finally

$$\hat{A} = (C' \Sigma_{vv}^{-1} C)^{-1} C' \Sigma_{vv}^{-1}.$$

Thus, the Gauss-Markov estimator is given by

$$\hat{x}(z) = (C' \Sigma_{vv}^{-1} C)^{-1} C' \Sigma_{vv}^{-1} (z - \overline{v}). \tag{E.65}$$

Let us also calculate the corresponding error covariance matrix. We have

$$\begin{aligned} E\Big\{ \big(x - \hat{x}(z)\big) \big(x - \hat{x}(z)\big)' \Big\} &= E\Big\{ \big(x - \hat{A}(z - \overline{v})\big) \big(x - \hat{A}(z - \overline{v})\big)' \Big\} \\ &= E\big\{ \hat{A}(v - \overline{v})(v - \overline{v})' \hat{A}' \big\} \\ &= \hat{A} \Sigma_{vv} \hat{A}' \\ &= (C' \Sigma_{vv}^{-1} C)^{-1} C' \Sigma_{vv}^{-1} \Sigma_{vv} \Sigma_{vv}^{-1} C (C' \Sigma_{vv}^{-1} C)^{-1}, \end{aligned}$$

and finally

$$E\Big\{ \big(x - \hat{x}(z)\big) \big(x - \hat{x}(z)\big)' \Big\} = (C' \Sigma_{vv}^{-1} C)^{-1}. \tag{E.66}$$

Finally, let us compare the Gauss-Markov estimator with the linear least-squares estimator of Cor. E.3.5. Assuming that Σ_{xx} is invertible, a straightforward calculation shows that the latter estimator can be written as

$$\hat{x}(z) = \overline{x} + \big(\Sigma_{xx}^{-1} + C' \Sigma_{vv}^{-1} C\big)^{-1} C' \Sigma_{vv}^{-1} (z - C\,\overline{x} - \overline{v}). \tag{E.67}$$

By comparing Eqs. (E.65) and (E.67), we see that the Gauss-Markov estimator is obtained from the linear least-squares estimator by setting $\overline{x} = 0$ and $\Sigma_{xx}^{-1} = 0$, i.e., a zero mean and infinite covariance for the unknown random variable x. Thus, the Gauss-Markov estimator may be viewed as a limiting form of the linear least-squares estimator. The error covariance matrix (E.66) of the Gauss-Markov estimator is similarly related with the error covariance matrix of the linear least-squares estimator.

E.6 DETERMINISTIC LEAST-SQUARES ESTIMATION

Suppose again that we want to estimate a vector $x \in \Re^n$ given a measurement vector $z \in \Re^m$ that is related to x by

$$z = Cx + v,$$

where C is a known $m \times n$ matrix of rank m. However, we know nothing about the probability distribution of x and v, and thus we can't use a statistically-based estimator. Then it is reasonable to select as our estimate the vector $\hat{x}$ that minimizes

$$f(x) = \|z - Cx\|^2,$$

that is, the estimate that fits best the data in a least-squares sense. We denote this estimate by $\hat{x}(z)$.

By setting to zero the gradient of f at $\hat{x}(z)$, we obtain

$$\nabla f|_{\hat{x}(z)} = 2C' \big(C\hat{x}(z) - z\big) = 0,$$

from which

$$\hat{x}(z) = (C'C)^{-1} C' z. \tag{E.68}$$

An interesting observation is that the estimate (E.68) is the same as the Gauss-Markov estimate given by Eq. (E.65), provided the measurement error has zero mean and covariance matrix equal to the identity, i.e., $\overline{v} = 0$, $\Sigma_{vv} = I$. In fact, if instead of $\|z - Cx\|^2$, we minimize

$$(z - \overline{v} - Cx)' \Sigma_{vv}^{-1} (z - \overline{v} - Cx),$$

then the deterministic least-squares estimate obtained is identical to the Gauss-Markov estimate. If instead of $\|z - Cx\|^2$ we minimize

$$(x - \overline{x})'\Sigma_{xx}^{-1}(z - \overline{x}) + (z - \overline{v} - Cx)'\Sigma_{vv}^{-1}(z - \overline{v} - Cx),$$

then the estimate obtained is identical to the linear least-squares estimate given by Eq. (E.67). Thus, we arrive at the interesting conclusion that the estimators obtained earlier on the basis of a stochastic optimization framework can also be obtained by minimization of a deterministic measure of fitness of estimated parameters to the data at hand.

APPENDIX F: Modeling of Stochastic Linear Systems

In this appendix we show how controlled linear time-invariant systems with stochastic inputs can be represented by the ARMAX model used in Section 5.3.

F.1 LINEAR SYSTEMS WITH STOCHASTIC INPUTS

Consider a linear system with output $\{y_k\}$, control input $\{u_k\}$, and an additional zero-mean random input $\{w_k\}$. We assume that $\{w_k\}$ is a stationary (up to second order) stochastic process. That is, $\{w_k\}$ is a sequence of random variables satisfying, for all $i, k = 0, \pm 1, \pm 2, \ldots$,

$$E\{w_k\} = 0, \qquad E\{w_0 w_i\} = E\{w_k w_{k+i}\} < \infty.$$

(All references to stationary processes in this section are meant in the limited sense just described.) By linearity, y_k is the sum of one sequence $\{y_k^1\}$ due to the presence of $\{u_k\}$ and another sequence $\{y_k^2\}$ due to the presence of $\{w_k\}$:

$$y_k = y_k^1 + y_k^2. \tag{F.1}$$

We assume that y_k^1 and y_k^2 are generated by some filters $B_1(s)/A_1(s)$ and $B_2(s)/A_2(s)$, respectively:

$$A_1(s)y_k^1 = B_1(s)u_k, \tag{F.2a}$$

$$A_2(s)y_k^2 = B_2(s)w_k. \tag{F.2b}$$

Operating on Eqs. (F.2a) and (F.2b) with $A_2(s)$ and $A_1(s)$, respectively, adding, and using Eq. (F.1), we obtain

$$\overline{A}(s)y_k = \overline{B}(s)u_k + v_k, \tag{F.3}$$

where $\overline{A}(s) = A_1(s)A_2(s)$, $\overline{B}(s) = A_2(s)B_1(s)$, and $\{v_k\}$, given by

$$v_k = A_1(s)B_2(s)w_k, \tag{F.4}$$

is a zero mean, generally correlated, stationary stochastic process.

We are interested in the case where u_k is a control input applied after y_k has occurred and has been observed, so that in Eq. (F.2a) we have $B_1(0) = 0$. Then, we may assume that the polynomials $\overline{A}(s)$ and $\overline{B}(s)$ have the form

$$\overline{A}(s) = 1 + \overline{a}_1 s + \cdots + \overline{a}_{m_0}s^{m_0}, \qquad \overline{B}(s) = \overline{b}_1 s + \cdots + \overline{b}_{m_0}s^{m_0}$$

for some scalars $\overline{a}_i$ and $\overline{b}_i$, and some positive integer m_0.

To summarize, we have constructed a model of the form

$$\overline{A}(s)y_k = \overline{B}(s)u_k + v_k,$$

where $\overline{A}(s)$ and $\overline{B}(s)$ are polynomials of the preceding form and $\{v_k\}$ is some zero-mean, correlated, stationary stochastic process. We now need to model further the sequence $\{v_k\}$.

F.2 PROCESSES WITH RATIONAL SPECTRUM

Given a zero-mean, stationary scalar process $\{v_k\}$, denote by $V(k)$ the autocorrelation function

$$V(k) = E\{v_i v_{i+k}\}, \qquad k = 0, \pm 1, \pm 2, \ldots$$

We say that $\{v_k\}$ has *rational spectrum* if the transform of $\{V(k)\}$ defined by

$$S_v(\lambda) = \sum_{k=-\infty}^{\infty} V(k)e^{-jk\lambda}$$

exists for $\lambda \in [-\pi, \pi]$ and can be expressed as

$$S_v(\lambda) = \sigma^2 \frac{|C(e^{i\lambda})|^2}{|D(e^{j\lambda})|^2}, \qquad \lambda \in [-\pi, \pi], \tag{F.5}$$

where σ is a scalar, $C(z)$ and $D(z)$ are some polynomials with real coefficients

$$C(z) = 1 + c_1 z + \cdots + c_m z^m, \tag{F.6a}$$

$$D(z) = 1 + d_1 z + \cdots + d_m z^m, \tag{F.6b}$$

and $D(z)$ has no roots on the unit circle $\{z \mid |z| = 1\}$.

The following facts are of interest:

(a) If $\{v_k\}$ is an uncorrelated process with $V(0) = \sigma^2$, $V(k) = 0$ for $k \neq 0$, then

$$S_v(\lambda) = \sigma^2, \qquad \lambda \in [-\pi, \pi],$$

and clearly $\{v_k\}$ has rational spectrum.

(b) If $\{v_k\}$ has rational spectrum S_v given by Eq. (F.5), then S_v can be written as

$$S_v(\lambda) = \tilde{\sigma}^2 \frac{|\tilde{C}(e^{j\lambda})|^2}{|\tilde{D}(e^{j\lambda})|^2}, \qquad \lambda \in [-\pi, \pi],$$

where $\tilde{\sigma}$ is a scalar and $\tilde{C}(z)$, $\tilde{D}(z)$ are unique real polynomials of the form

$$\tilde{C}(z) = 1 + \tilde{c}_1 z + \cdots + \tilde{c}_m z^m,$$

$$\tilde{D}(z) = 1 + \tilde{d}_1 z + \cdots + \tilde{d}_m z^m,$$

such that:

(1) $\tilde{C}(z)$ has all its roots outside or on the unit circle, and if $C(z)$ has no roots on the unit circle, then the same is true for $\tilde{C}(z)$.

(2) $\tilde{D}(z)$ has all roots strictly outside the unit circle.

These facts are seen by noting that if $\rho \neq 0$ is a root of $D(z)$, then $|D(e^{j\lambda})|^2 = D(e^{j\lambda})D(e^{-j\lambda})$ contains a factor

$$(1 - \rho^{-1}e^{j\lambda})(1 - \rho^{-1}e^{-j\lambda}) = \rho^{-2}(\rho - e^{j\lambda})(\rho - e^{-j\lambda}).$$

A little reflection shows that the roots of $\tilde{D}(z)$ should be ρ or ρ^{-1} depending on whether ρ is outside or inside the unit circle. Similarly, the roots of $\tilde{C}(z)$ are obtained from the roots of $C(z)$. Thus the polynomials $\tilde{C}(z)$ and $\tilde{D}(z)$ as well as $\tilde{\sigma}^2$ can be uniquely determined. We may thus assume without loss of generality that $C(z)$ and $D(z)$ in Eq. (F.5) have no roots inside the unit circle.

There is a fundamental result here that relates to the realization of processes with rational spectrum. The proof is hard; see for example, Ash and Gardner [AsG75, pp. 75-76].

Proposition F.1: If $\{v_k\}$ is a zero-mean, stationary stochastic process with rational spectrum

$$S_v(\lambda) = \sigma^2 \frac{|C(e^{j\lambda})|^2}{|D(e^{j\lambda})|^2}, \qquad \lambda \in [-\pi, \pi],$$

where the polynomials $C(s)$ and $D(s)$ are given by

$$C(s) = 1 + c_1 s + \cdots + c_m s^m, \qquad D(s) = 1 + d_1 s + \cdots + d_m s^m,$$

and are assumed (without loss of generality) to have no roots inside the unit circle, then there exists a zero-mean, uncorrelated stationary process $\{\epsilon_k\}$ with $E\{\epsilon_k^2\} = \sigma^2$ such that for all k

$$v_k + d_1 v_{k-1} + \cdots + d_m v_{k-m} = \epsilon_k + c_1 \epsilon_{k-1} + \cdots + c_m \epsilon_{k-m}.$$

F.3 THE ARMAX MODEL

Let us now return to the problem of representation of a linear system with stochastic inputs. We had arrived at the model

$$\overline{A}(s) y_k = \overline{B}(s) u_k + v_k. \tag{F.7}$$

If the zero-mean stationary process $\{v_k\}$ has rational spectrum, the preceding analysis and proposition show that there exists a zero-mean, uncorrelated stationary process $\{\epsilon_k\}$ satisfying

$$D(s) v_k = C(s) \epsilon_k,$$

where $C(s)$ and $D(s)$ are polynomials, and $C(s)$ has no roots inside the unit circle. Operating on both sides of Eq. (F.7) with $D(s)$ and using the relation $D(s)v_k = C(s)\epsilon_k$, we obtain

$$A(s) y_k = B(s) u_k + C(s) \epsilon_k, \tag{F.8}$$

where $A(s) = D(s)\overline{A}(s)$ and $B(s) = D(s)\overline{B}(s)$. Since $\overline{A}(0) = 1$, $\overline{B}(0) = 0$, we can write Eq. (F.8) as

$$y_k + \sum_{i=1}^{m} a_i y_{k-i} = \sum_{i=1}^{m} b_i u_{k-i} + \epsilon_k + \sum_{i=1}^{m} c_i \epsilon_{k-i},$$

for some integer m and scalars a_i, b_i, c_i, $i = 1, \ldots, m$. This is the ARMAX model that we have used in Section 5.3.

APPENDIX G: Formulating Problems of Decision Under Uncertainty

In this appendix we discuss various approaches for formulating problems of decision under uncertainty. After a brief discussion of the min-max approach, we focus on the expected utility approach, and we show how this approach can be theoretically justified even if the decision maker is sensitive to the "variability" or "risk" associated with the results of different decisions.

G.1 THE PROBLEM OF DECISION UNDER UNCERTAINTY

A decision problem in one of its simplest and most abstract forms consists of three nonempty sets $\mathcal{D}$, $\mathcal{N}$, and $\mathcal{O}$, a function $f : \mathcal{D} \times \mathcal{N} \mapsto \mathcal{O}$, and a complete and transitive relation $\preceq$ on $\mathcal{O}$. Here

$\mathcal{D}$ is the set of possible decisions,

$\mathcal{N}$ indexes the uncertainty in the problem and may be called the set of "states of nature,"

$\mathcal{O}$ is the set of outcomes of the decision problem,

f is the function that determines which outcome will result from a given decision and state of nature, i.e., if decision $d \in \mathcal{D}$ is selected and state of nature $n \in \mathcal{N}$ prevails, then the outcome $f(d, n) \in \mathcal{O}$ occurs,

$\preceq$ is a relation that determines our preference among the outcomes.† Thus, for $O_1, O_2 \in \mathcal{O}$, by $O_1 \preceq O_2$ we mean that outcome O_2 is at least as preferable as outcome O_1. By completeness of the relation, we mean that every two elements of $\mathcal{O}$ are related, i.e., given any $O_1, O_2 \in \mathcal{O}$, there are three possibilities: either $O_1 \preceq O_2$ but not $O_2 \preceq O_1$, or $O_2 \preceq O_1$ but not $O_1 \preceq O_2$, or both $O_1 \preceq O_2$ and $O_2 \preceq O_1$. By transitivity we mean that $O_1 \preceq O_2$ and $O_2 \preceq O_3$ implies $O_1 \preceq O_3$ for any three elements $O_1, O_2, O_3 \in \mathcal{O}$.

Example G.1

Consider an individual that may bet \$1 on the toss of a coin or not bet at all. If he bets and guesses correctly, he wins \$1 and if he does not guess correctly, he loses \$1. Here $\mathcal{D}$ consists of three elements

$$\mathcal{D} = \{\text{bet on heads, bet on tails, not bet}\},$$

$\mathcal{N}$ consists of two elements

$$\mathcal{N} = \{\text{heads, tails}\},$$

and $\mathcal{O}$ consists of three elements, the three possible final fortunes of the player

$$\mathcal{O} = \{\$0, \$1, \$2\}.$$

The preference relation on $\mathcal{O}$ is the natural one, i.e., $0 \preceq 1$, $0 \preceq 2$, $1 \preceq 2$, and the values of the function f are given by

$$f(H,H) = \$2, \quad f(T,H) = \$0, \quad f(\text{not bet}, H) = \$1,$$

$$f(H,T) = \$0, \quad f(T,T) = \$2, \quad f(\text{not bet}, T) = \$1.$$

Now the relative order by which we rank outcomes is usually clear in any given situation. On the other hand, for the decision problem to be completely formulated, we need a *ranking among decisions* that is consistent in a well-defined sense with our ranking of outcomes. Furthermore, to facilitate a mathematical or computational analysis, this ranking should be determined by a numerical function F that maps the set of decisions $\mathcal{D}$ to the set of real numbers $\Re$ and is such that

$$d_1 \preceq d_2 \quad \text{if and only if} \quad F(d_1) \leq F(d_2), \qquad \text{for all } d_1, d_2 \in \mathcal{D}, \quad \text{(G.1)}$$

† The symbol $\preceq$ in this appendix will be used (somewhat loosely) to denote a preference relation within either the set of outcomes or the set of decisions. The precise meaning should be clear from the context, and hopefully the use of the same symbol to denote different preference relations will create no confusion.

where the notation $d_1 \preceq d_2$ implies that the decision d_2 is at least as preferable as the decision d_1.

It is by no means clear how one should go about determining and characterizing a ranking among decisions. For example, in the gambling example above, different people will have different preferences as to accepting or refusing the gamble. In fact, the method by which one goes from a ranking of outcomes to a ranking of decisions is a central issue in decision theory. There are a number of approaches and viewpoints, and we now proceed to discuss some of these.

Payoff Functions, Dominant, and Noninferior Decisions

Let us consider the case where it is possible to assign to each element of $\mathcal{O}$ a real number in a way that the order between elements of $\mathcal{O}$ agrees with the usual order of the corresponding numbers. In particular, we assume that there exists a real-valued function $G : \mathcal{O} \mapsto \Re$ with the property

$$G(O_1) \leq G(O_2) \quad \text{if and only if} \quad O_1 \preceq O_2, \qquad \text{for all } O_1, O_2 \in \mathcal{O}. \quad \text{(G.2)}$$

Such a G does not always exist (see Exercise G.2). However, its existence can be guaranteed under quite general assumptions. In particular, one may show that it exists if $\mathcal{O}$ is a countable set. Also if G exists, it is far from unique, since if Φ is any monotonically increasing function $\Phi : \Re \mapsto \Re$, the composite function $\Phi \cdot G$ [defined by $(\Phi \cdot G)(O) = \Phi\big(G(O)\big)$] has the same property (G.2) as G. For instance, in the example given earlier, a function $G : \{0,1,2\} \mapsto \Re$ satisfies Eq. (G.2) if and only if $G(0) < G(1) < G(2)$ and there is an infinity of such functions.

For any choice of G satisfying Eq. (G.2), we define the function $J : \mathcal{D} \times \mathcal{N} \mapsto \Re$ by means of

$$J(d,n) = G\big(f(d,n)\big)$$

and call it a *payoff function*.

Given a payoff function J, it is possible to obtain a complete ranking of decisions by means of a numerical function in the *special case of certainty* (the case where the set $\mathcal{N}$ of states of nature consists of a single element $\overline{n}$). By defining

$$F(d) = J(d, \overline{n}),$$

we have

$$d_1 \preceq d_2 \quad \text{if and only if} \quad F(d_1) \leq F(d_2) \quad \text{if and only if} \quad f(d_1, \overline{n}) \preceq f(d_2, \overline{n}),$$

and the numerical function F defines a complete ranking of decisions.

In the case where there is uncertainty, i.e., when $\mathcal{N}$ contains more than one element, the order on $\mathcal{O}$ induces only a *partial order* on $\mathcal{D}$ by means of the relations

$$\begin{aligned} d_1 \preceq d_2 \quad &\text{if and only if} \quad F(d_1) \leq F(d_2), \quad \text{for all } n \in \mathcal{N} \\ &\text{if and only if} \quad f(d_1, n) \preceq f(d_2, n), \quad \text{for all } n \in \mathcal{N}. \end{aligned} \tag{G.3}$$

In this partial order, it is not necessary that every two elements of $\mathcal{D}$ be related, i.e., for some $d, d' \in \mathcal{D}$ we may have neither $d \preceq d'$ nor $d' \preceq d$. If, however, for two decisions $d_1, d_2 \in \mathcal{D}$ we have $d_1 \preceq d_2$ in the sense of Eq. (G.3), then we can conclude that d_2 is at least as preferable as d_1 since the resulting outcome $f(d_2, n)$ is at least as preferable as $f(d_1, n)$, *regardless of the state of nature* n *that will occur*.

A decision $d^* \in \mathcal{D}$ is called a *dominant decision* if

$$d \preceq d^*, \qquad \text{for all } d \in \mathcal{D},$$

where $\preceq$ is understood in the sense of the partial order defined by Eq. (G.3). Naturally such a decision need not exist, but if it does exist, then it may be viewed as optimal. Unfortunately, in most problems of interest to an analyst there exists no dominant decision. For instance, this is so in the gambling Example G.1, as the reader can easily verify. In fact no two decisions are related in the sense of Eq. (G.3) for this example.

In the absence of a dominant decision, one can consider the set $\mathcal{D}_m \subset \mathcal{D}$ of all *noninferior decisions*, where $d_m \in \mathcal{D}_m$ if for every $d \in \mathcal{D}$ the relation $d_m \preceq d$ implies $d \preceq d_m$ in the sense of the partial order defined by Eq. (G.3). In terms of a payoff function J, noninferior decisions may be characterized by

$$\begin{aligned} d_m \in \mathcal{D}_m \quad &\text{if and only if} \quad \text{there is no } d \in \mathcal{D} \text{ such that} \\ &J(d_m, n) \leq J(d, n) \text{ for all } n \in \mathcal{N} \text{ and} \\ &J(d_m, n) < J(d, n) \text{ for some } n \in \mathcal{N}. \end{aligned}$$

Clearly it makes sense to consider only the decisions in $\mathcal{D}_m$ as candidates for optimality since any decision that is not in $\mathcal{D}_m$ is dominated by one that belongs to $\mathcal{D}_m$. Furthermore, it may be proved that the set $\mathcal{D}_m$ is nonempty when the set $\mathcal{D}$ is a finite set, so that at least for this case there exists at least one noninferior decision. However, in practice the set $\mathcal{D}_m$ of noninferior decisions often is either difficult to determine or contains too many elements. For instance, in the gambling example given earlier, the reader may verify that every decision is noninferior.

Whenever the partial order of Eq. (G.3) fails to produce a satisfactory ranking among decisions, one must turn to other approaches to formulate the decision problem. The approaches that we will examine assume a notion of a *generalized outcome* of a decision and introduce a complete order on the set of these generalized outcomes that is consistent with the original order on the set of outcomes $\mathcal{O}$. The complete order on the set of generalized outcomes in turn induces a complete order on the set of decisions.

The Min-Max Approach

In the min-max (or max-min) approach we take the point of view that the generalized outcome of a decision d is the set of all possible outcomes resulting from d:

$$f(d, \mathcal{N}) = \big\{ O \in \mathcal{O} \mid \text{there exists } n \in \mathcal{N} \text{ with } f(d, n) = O \big\}.$$

In addition, we adopt a pessimistic attitude and rank the sets $f(d, \mathcal{N})$ on the basis of their worst possible element. In particular, we introduce a complete order on the set of all subsets of $\mathcal{O}$ by means of the relation

$$\mathcal{O}_1 \preceq \mathcal{O}_2 \quad \text{if and only if} \quad \inf_{O \in \mathcal{O}_1} G(O) \leq \inf_{O \in \mathcal{O}_2} G(O), \quad \text{for all } \mathcal{O}_1, \mathcal{O}_2 \subset \mathcal{O}, \tag{G.4}$$

where $\mathcal{O}_1, \mathcal{O}_2$ is any pair of subsets of $\mathcal{O}$, and G is a numerical function consistent with the order on $\mathcal{O}$ in accordance with Eq. (G.2). From Eq. (G.4) we have a complete order on the set of decisions $\mathcal{D}$ by means of

$$\begin{aligned} d_1 \preceq d_2 \quad &\text{if and only if} \quad f(d_1, \mathcal{N}) \preceq f(d_2, \mathcal{N}) \\ &\text{if and only if} \quad \inf_{n \in \mathcal{N}} G\big(f(d_1, n)\big) \leq \inf_{n \in \mathcal{N}} G\big(f(d_2, n)\big), \end{aligned}$$

or in terms of a payoff function J,

$$d_1 \preceq d_2 \quad \text{if and only if} \quad \inf_{n \in \mathcal{N}} J(d_1, n) \leq \inf_{n \in \mathcal{N}} J(d_2, n).$$

Thus, by using the min-max approach, the decision problem is formulated concretely in that it reduces to maximizing over $\mathcal{D}$ the numerical function

$$F(d) = \inf_{n \in \mathcal{N}} J(d, n).$$

Furthermore, it can be easily shown that the elements of $\mathcal{D}$ that maximize $F(d)$ above will not change if J is replaced by $\Phi \cdot J$, where $\Phi : \Re \mapsto \Re$ is any monotonically increasing function. Nonetheless, the min-max approach is pessimistic in nature and will often produce an unduly conservative decision. Characteristically, in the gambling example G.1, the optimal decision according to the min-max approach is to refuse the gamble.

We next discuss another approach for formulating decision problems. This approach is quantifying the likelihood of various states of nature through probabilities.

G.2 EXPECTED UTILITY THEORY AND RISK

In many decision problems under uncertainty we have additional information about the mechanism by which states of nature occur. In particular,

we are often in a position to know that these states occur in accordance with a given probabilistic mechanism, which may depend on the decision d adopted. To be specific, assume for convenience that the set of states of nature $\mathcal{N}$ is either a finite set or a countable set† and that for every decision $d \in \mathcal{D}$ we know that states of nature occur according to a given probability law $P(\cdot \mid d)$ defined on $\mathcal{N}$. Now each decision $d \in \mathcal{D}$ specifies the probability of each outcome via the function $f(d, \cdot)$ and the relation

$$P_d(O) = P\Big(\{n \mid f(d,n) = O\} \mid d\Big), \qquad \text{for all } O \in \mathcal{O}.$$

In this relation, $P_d(O)$ denotes the probability that the outcome O will occur when the decision d is adopted. One may view the probability law P_d associated with each $d \in \mathcal{D}$ as a "probabilistic outcome" (or "generalized outcome" to use the term of the preceding section) corresponding to d, since P_d specifies the probabilistic mechanism by which outcomes occur once d is selected. We shall also use the term *lottery*‡ for a probability law on the set of outcomes. In the gambling example G.1, the decision "bet on heads" has as a generalized outcome the probability law (or lottery) $(1/2, 0, 1/2)$ on the set of outcomes $\mathcal{O} = \{\$0, \$1, \$2\}$. The decision "bet on tails" has the same generalized outcome, while the decision "not bet" has as a generalized outcome the probability law $(0, 1, 0)$.

The basic idea of the expected utility approach is the following. We already have a complete ranking of the outcomes, i.e., the elements of $\mathcal{O}$. If we had a complete ranking of all *lotteries* on the set of outcomes (presumably consistent with the original ranking on $\mathcal{O}$ in the sense that if the outcome O_1 is preferable to the outcome O_2, then the lottery assigning probability 1 to O_1 is preferable to the lottery assigning probability 1 to outcome O_2), then we could in turn obtain a complete ranking of all decisions in $\mathcal{D}$. This is true simply because we could rank any two decisions $d_1, d_2 \in \mathcal{D}$ according to the relative order of their corresponding lotteries P_{d_1}, P_{d_2}, i.e., by means of the relation

$$d_1 \preceq d_2 \text{ if and only if } P_{d_1} \preceq P_{d_2}.$$

The fundamental premise of the expected utility approach is to assume at the outset that *the decision maker has a complete ranking of all lotteries on the set of outcomes*, i.e., the decision maker is in a position to express his preference between any two probability laws on the set of

† If $\mathcal{N}$ is not countable, it is necessary to introduce a probability space structure on $\mathcal{N}$ and $\mathcal{O}$ as in Appendix C. Furthermore, it is necessary that the function $f(d, \cdot)$ satisfy certain (measurability) assumptions.

‡ The term "lottery" is associated with conceptually convenient device of viewing outcomes as prizes of some sort and viewing a fixed probabilistic mechanism for winning a prize as a lottery.

outcomes. This in turn settles the question of ranking decisions in view of the preceding relation. Furthermore, if there exists a numerical function G by means of which preferences on the set of lotteries can be expressed,

$$P_{d_1} \preceq P_{d_2} \text{ if and only if } G(P_{d_1}) \le G(P_{d_2}),$$

then decisions can be ranked by means of a numerical function F,

$$d_1 \preceq d_2 \text{ if and only if } F(d_1) \le F(d_2),$$

where $F(d) = G(P_d)$ for all $d \in \mathcal{D}$.

The aspect of this formulation that is analytically very appealing, however, is that the ordering of decisions can be expressed not only by a function G as above, but also by means of an essentially unique numerical function called the *utility function*. This function, denoted U, maps the space of outcomes into the set of real numbers and satisfies

$$\begin{aligned} d_1 \preceq d_2 \text{ if and only if } & P_{d_1} \preceq P_{d_2} \\ \text{if and only if } & E\Big\{U\big(f(d_1,n)\big) \mid d_1\Big\} \le E\Big\{U\big(f(d_2,n)\big) \mid d_2\Big\}, \end{aligned} \tag{G.5}$$

where the expectations are taken with respect to the corresponding probability law $P(\cdot \mid d)$ on $\mathcal{N}$. The problem of selecting an optimal decision is thus reduced to the problem of maximizing over $\mathcal{D}$ the expected value of the numerical function U.

To clarify the problem formulation based on the approach of this section and to illustrate the advantages resulting from the introduction of a utility function, let us consider an example.

Example G.2

Consider a problem of allocating one unit of capital between two investment opportunities A and B. Opportunity A yields \$1.5 per dollar invested with certainty, while opportunity B yields \$1 per dollar invested with probability 1/2 and \$3 per dollar invested with probability 1/2. the problem is to decide on the fractions d and $(1 - d)$ of the capital to be invested in opportunities A and B, respectively, where $0 \le d \le 1$.

In terms of the framework of the decision problem of Section G.1, the set of decisions $\mathcal{D}$ consists of the interval $[0, 1]$, i.e., the set of values that the fraction d invested in A can take. The set of states of nature $\mathcal{N}$ consists of two elements n_1, n_2, where n_1: B yields \$1 per dollar invested, and n_2: B yields \$3 per dollar invested. The set of outcomes $\mathcal{O}$ may be taken to be the interval $[1, 3]$, which is the set of possible final fortunes of the investor resulting from all possible decisions and states of nature. The function f that determines the outcome corresponding to any decision d and state of nature n is given by

$$f(d,n) = \begin{cases} 1.5d + (1-d) & \text{if } n = n_1, \\ 1.5d + 3(1-d) & \text{if } n = n_2. \end{cases}$$

The preference relation on the set of outcomes is the natural one, i.e., a final fortune O_1 is at least as preferable as a final fortune O_2 if O_1 is numerically greater than or equal to O_2 (i.e., $O_2 \preceq O_1$ if $O_2 \leq O_1$).

Let us note that since B has a higher expected rate of return, the decision that maximizes expected value of profit is to invest exclusively in opportunity B ($d^* = 0$). On the other hand the optimal decision based on the max-min approach is to invest exclusively in A ($d^* = 1$) since in this approach one maximizes profit based on the assumption that the most unfavorable state of nature will occur. Mathematically this can be verified by noting that $d^* = 1$ maximizes over $[0, 1]$ the function $F(d)$ given by

$$F(d) = \min\{1.5d + (1 - d),\ 1.5d + 3(1 - d)\}.$$

Note that the approach of maximizing expected profit and the max-min approach lead to very different decisions. Yet it is safe to assume that many decision makers would settle on a decision that differs from both decisions mentioned above and that invests a positive fraction of the capital in both opportunities A and B.

Now in the expected utility approach, the fundamental assumption is that the decision maker has a complete ranking of all lotteries on the set of outcomes. In other words, given any two probability distributions on the interval of final fortunes $[0, 3]$, the decision maker can express his preference between the two, in the sense that he can point out the probability distribution in accordance with which he would rather have his final fortune selected. Now the probability distribution on the set of final fortunes corresponding to a decision d is the one that assigns probability $1/2$ to $\big(1.5d + (1 - d)\big)$ and probability $1/2$ to $\big(1.5d + 3(1 - d)\big)$. According to the expected utility approach, a decision d is optimal if its corresponding probability distribution is at least as preferable as all other probability distributions of the type described above. It should be clear, however, that a mathematical formulation of the corresponding optimization problem is very cumbersome since it is difficult to visualize or conjecture the form of a numerical function by means of which these probability distributions can be ranked. On the other hand, let us assume that a utility function U satisfying Eq. (G.5) exists (and it does exist under mild assumptions, as will be indicated shortly). Then an optimal decision is one that solves the problems

$$\begin{aligned} &\text{maximize} \quad E\Big\{U\big(f(d,n)\big)\Big\} \\ &\text{subject to} \quad 0 \leq d \leq 1. \end{aligned}$$

Substituting the problem data, we have

$$E\Big\{U\big(f(d,n)\big)\Big\} = \tfrac{1}{2}\Big(U\big(1.5d + (1 - d)\big) + U\big(1.5d + 3(1 - d)\big)\Big)$$

so the maximization problem is conveniently formulated.

As an example, let us assume that the decision maker's utility function is quadratic of the form

$$U(O) = \alpha O - O^2,$$

where α is some scalar. We require that $6 < \alpha$ so that $U(O)$ is increasing in the interval $[0, 3]$. This is necessary for the original preference relation on the set of outcomes to be consistent with the one specified by the utility function. Solution of the maximization problem above yields the optimal decision d^*, where

$$d^* = \begin{cases} 0 & \text{if } 8 \leq \alpha, \\ (8 - \alpha)/5 & \text{if } 6 < \alpha < 8. \end{cases}$$

Note that for $6 < \alpha < 8$, a positive fraction of the capital is invested in opportunity A even though it offers a return that is less than the average return of B.

It should be noted, of course, that different decision makers faced with the same decision problem may have different utility functions, so that before the problem can be numerically solved, the form of the utility function must be specified. This can be done experimentally if necessary (see Exercise G.3). However, the importance of the notion of a utility function satisfying Eq. (G.5) lies primarily with the fact that under relatively mild assumptions, it exists and can serve as the starting point of analysis of the decision problem. The reason is that important conclusions about optimal decisions can often be obtained based on either incomplete knowledge of the utility function or fairly general assumptions on its form.

We provide below the theorem of existence of a utility function for the case where the set of outcomes $\mathcal{O}$ is a finite set. For more general cases, see the book by Fishburn [Fis70].

Consider the set $\mathcal{O}$ of outcomes and assume that it is a finite set, $\mathcal{O} = \{O_1, O_2, \ldots, O_N\}$. Let $\mathcal{P}$ be the set of all probability laws $P = (p_1, p_2, \ldots, p_N)$ on $\mathcal{O}$, where p_i is the probability of outcome O_i, $i = 1, \ldots, N$. For any $P_1, P_2 \in \mathcal{P}$, $P_1 = (p_1^1, \ldots, p_N^1)$, $P_2 = (p_1^2, \ldots, p_N^2)$, and any $\alpha \in [0, 1]$, we use the notation

$$\alpha P_1 + (1 - \alpha)P_2 = \big(\alpha p_1^1 + (1 - \alpha)p_1^2, \ldots, \alpha p_N^1 + (1 - \alpha)p_N^2\big).$$

Let us make the following assumptions:

A.1 There exists a complete and transitive relation $\preceq$ on $\mathcal{P}$. (For any $P_1, P_2 \in \mathcal{P}$, we write $P_1 \sim P_2$ if $P_1 \preceq P_2$ and $P_2 \preceq P_1$, and we write $P_1 \prec P_2$ if $P_1 \preceq P_2$ but not $P_2 \preceq P_1$.)

A.2 If $P_1 \sim P_2$, then for all $\alpha \in [0, 1]$ and all $P \in \mathcal{P}$

$$\alpha P_1 + (1 - \alpha)P \sim \alpha P_2 + (1 - \alpha)P.$$

A.3 If $P_1 \prec P_2$, then for all $\alpha \in (0, 1]$ and all $P \in \mathcal{P}$

$$\alpha P_1 + (1 - \alpha)P \prec \alpha P_2 + (1 - \alpha)P.$$

A.4 If $P_1 \prec P_2 \prec P_3$, there exists an $\alpha \in (0, 1)$ such that

$$\alpha P_1 + (1 - \alpha)P_3 \sim P_2.$$

Before proving the expected utility theorem, let us briefly discuss the above assumptions. It is convenient for interpretation purposes to view each of the outcomes $O_1, O_2, \ldots, O_N$ as a monetary prize. Consider any probability law $(p_1, p_2, \ldots, p_N)$ on the set of outcomes. Imagine a pointer that spins in the center of a circle divided into N regions, and assume that it spins in a way that when it stops it is equally likely to be pointing in any direction. The region associated with each prize O_i, $i = 1, \ldots, N$, occupies a fraction p_i of the circumference of the circle. Then we associate with P the game (or lottery) whereby we spin the wheel and win the prize corresponding to the region within which the pointer stops. Now given any two probability laws P_1 and P_2 and a scalar $\alpha \in [0, 1]$, we can associate with the probability law

$$\alpha P_1 + (1 - \alpha)P_2$$

the following game. A pointer is spun in the center of a circle divided in two regions, say 1 and 2, occupying respective fractions α and $(1 - \alpha)$ of its circumference. Depending on whether the pointer stops in region 1 or region 2, the game corresponding to P_1 and P_2 is played and a prize is won accordingly.

Assumption A.1 requires that we are able to state our preference between games such as the above, which correspond to any two probability laws P_1 and P_2. Furthermore, our preference relation must be transitive, I.e., if $P_1 \preceq P_2$ and $P_2 \preceq P_3$, then $P_1 \preceq P_3$. This is the basic assumption, which forms the core of the expected utility approach. Assumptions A.2 and A.3 have obvious interpretations and both seem reasonable. Assumption A.4 is a continuity assumption requiring that if $P_1 \prec P_2 \prec P_3$, one is indifferent to the game associated with P_2 and a game whose outcome decides with respective probabilities α and $(1 - \alpha)$ whether the game associated with P_1 or P_3 will be played. This assumption is inconsistent with a worst-case viewpoint whereby one ranks lotteries according to the worst outcome that can occur with positive probability, and has been the subject of some controversy. For example, consider the extreme situation where there are three outcomes O_1 = death, O_2 = receive nothing, and O_1 = receive \$1. Then it appears reasonable that any probability law that assigns a positive probability to O_1 (death) cannot be preferable or equivalent to any probability law that assigns a zero probability to O_1. Yet Assumption A.4 requires that for some α with $0 < \alpha < 1$ we are indifferent between the status quo and a game whereby we receive \$1 with probability $(1 - \alpha)$ and die with probability α. On the other hand, it is possible to argue that if the probability of death α is extremely close to zero, then this might actually be the case.

The following theorem is the central result of the expected utility theory. It states that a preference relation on the set of all lotteries, which satisfies Assumptions A.1-A.4 can be characterized numerically by means of an essentially unique function, the utility function. Note that *this result concerns an arbitrary preference relation on lotteries on the set of outcomes and is thus completely decoupled from any decision problem that one may be considering.*

Proposition G.1: Under Assumptions A.1-A.4, there exists a real-valued function $U : \mathcal{O} \mapsto \Re$, called *utility function*, such that for all $P_1, P_2 \in \mathcal{P}$,

$$P_1 \preceq P_2 \text{ if and only if } E_{P_1}\{U(O)\} \leq E_{P_2}\{U(O)\},$$

where we denote by $E_P\{\cdot\}$ the expected value with respect to a probability law P. Furthermore, U is unique up to a positive linear transformation, i.e., if U^* is another function with the above property, there exists a positive scalar s_1 and a scalar s_2, such that

$$U^*(O) = s_1 U(O) + s_2, \qquad \text{for all } O \in \mathcal{O}.$$

Proof: We first show the following statement:

S If $P_1 \prec P_3$, and P_2 is such that $P_1 \preceq P_2 \preceq P_3$, then there exists a *unique* scalar $\alpha \in [0, 1]$ such that

$$\alpha P_1 + (1 - \alpha)P_3 \sim P_2. \tag{G.7}$$

Furthermore, if P_2' is such that $P_1 \preceq P_2 \preceq P_2' \preceq P_3$ and α' corresponds to P_2' as in Eq. (G.7), then $\alpha \geq \alpha'$.

Indeed if $P_1 \sim P_2 \prec P_3$, then $\alpha = 1$ is the unique scalar satisfying Eq. (G.7), since if for some $\alpha \in [0, 1)$ we had

$$\alpha P_1 + (1 - \alpha)P_3 \sim P_2 \sim \alpha P_1 + (1 - \alpha)P_2,$$

then Assumption A.3 would be contradicted. Similarly, if $P_1 \prec P_2 \sim P_3$, then $\alpha = 0$ is the unique scalar satisfying Eq. (G.7). Assume now that $P_1 \prec P_2 \prec P_3$. Then by Assumption A.4, there exists an $\alpha_1 \in (0, 1)$ satisfying Eq. (G.7). Assume that α_1 is not unique and there exists another scalar $\alpha_2 \in (0, 1)$ such that Eq. (G.7) is satisfied, i.e.,

$$\alpha_1 P_1 + (1 - \alpha_1)P_3 \sim P_2 \sim \alpha_2 P_1 + (1 - \alpha_2)P_3. \tag{G.8}$$

Let us assume that $0 < \alpha_1 < \alpha_2 < 1$. Then we have

$$P_3 = \frac{\alpha_2 - \alpha_1}{1 - \alpha_1} P_3 + \frac{1 - \alpha_2}{1 - \alpha_1} P_1, \tag{G.9}$$

$$\alpha_2 P_1 + (1-\alpha_2)P_3 = \alpha_1 P_1 + (1-\alpha)\left\{\frac{\alpha_2-\alpha_1}{1-\alpha_1}P_1 + \frac{1-\alpha_2}{1-\alpha_1}P_3\right\}. \quad \text{(G.10)}$$

Since $P_1 \prec P_3$, we have by Assumption A.3 and Eq. (G.9)

$$\frac{\alpha_2-\alpha_1}{1-\alpha_1}P_1 + \frac{1-\alpha_2}{1-\alpha_1}P_3 \prec \frac{\alpha_2-\alpha_1}{1-\alpha_1}P_1 + \frac{1-\alpha_2}{1-\alpha_1}P_3 = P_3.$$

Again, using Assumption A.3 and Eq. (G.10), we have

$$\alpha_2 P_1 + (1-\alpha_2)P_3 \prec \alpha_1 P_1 + (1-\alpha_1)P_3.$$

However, this contradicts Eq. (G.8) and hence the uniqueness of the scalar α in Eq. (G.7) is proved.

To show that $P_1 \preceq P_2 \preceq P_2' \preceq P_3$ implies $\alpha \geq \alpha'$, assume the contrary, i.e., $\alpha < \alpha'$. Then we have, using Assumption A.3,

$$\begin{aligned}
P_2' &\sim \alpha' P_1 + (1-\alpha')P_3 \\
&= (1-\alpha+\alpha')\left\{\frac{\alpha}{1-\alpha+\alpha'}P_1 + \frac{1-\alpha'}{1-\alpha+\alpha'}P_3\right\} + (\alpha'-\alpha)P_1 \\
&\prec (1-\alpha+\alpha')\left\{\frac{\alpha}{1-\alpha+\alpha'}P_1 + \frac{1-\alpha'}{1-\alpha+\alpha'}P_3\right\} + (\alpha'-\alpha)P_1 \\
&= \alpha P_1 + (1-\alpha)P_3 \\
&\sim P_2.
\end{aligned}$$

Hence $P_2' \prec P_2$, which contradicts the assumption $P_2 \preceq P_2'$. It follows that $\alpha \geq \alpha'$ and statement S is proved.

Now consider the probability laws

$$\overline{P}_1 = (1,0,\ldots,0), \quad \overline{P}_2 = (0,1,\ldots,0), \quad \ldots, \quad \overline{P}_N = (0,0,\ldots,1).$$

Assume without loss of generality that $\overline{P}_1 \preceq \overline{P}_2 \preceq \cdots \preceq \overline{P}_N$ and assume further that $\overline{P}_1 \prec \overline{P}_N$ (if $\overline{P}_1 \sim \overline{P}_2, \sim \cdots \sim \overline{P}_N$, the proof of the proposition is trivial). Let A_1, A_N be any scalars with $A_1 < A_N$ and define

$$U(O_1) = A_1, \qquad U(O_N) = A_N.$$

Let α_i, $i = 1,\ldots,n$, be the unique scalar $\alpha_i \in [0,1]$ such that

$$\alpha_i \overline{P}_1 + (1-\alpha_i)\overline{P}_N \sim \overline{P}_i, \qquad i = 1,\ldots,N, \quad \text{(G.11)}$$

and define

$$U(O_i) = A_i = \alpha_i A_1 + (1-\alpha_i)A_N, \qquad i = 1,\ldots,N. \quad \text{(G.12)}$$

We shall prove that the function $U : \mathcal{O} \mapsto \Re$ defined above has the desired property (G.6). Indeed for any probability law $P = (p_1,\ldots,p_N)$, it is easily

seen that $\overline{P}_1 \prec P \prec \overline{P}_N$, and thus we can define $\alpha(P)$ to be the unique scalar in $[0,1]$ such that

$$\alpha(P)\overline{P}_1 + \big(1-\alpha(P)\big)\overline{P}_N \sim P. \quad \text{(G.13)}$$

From statement S we obtain for all P, P'

$$P \preceq P' \text{ if and only if } \alpha(P) \geq \alpha(P'). \quad \text{(G.14)}$$

Now from Eq. (G.11), we have

$$\begin{aligned}
P &= \sum_{i=1}^N p_i \overline{P}_i \\
&\sim \sum_{i=1}^N p_i\big(\alpha_i \overline{P}_i + (1-\alpha_i)\overline{P}_N\big) \\
&\sim \sum_{i=1}^N p_i \alpha_i \overline{P}_1 + \left(1 - \sum_{i=1}^N p_i\alpha_i\right)\overline{P}_N.
\end{aligned} \quad \text{(G.15)}$$

Comparing Eqs. (G.13) and (G.15), we obtain

$$\alpha(P) = \sum_{i=1}^N p_i \alpha_i,$$

and from Eq. (G.14),

$$P_1 \preceq P_2 \text{ if and only if } \sum_{i=1}^N p_i^1 \alpha_i \geq \sum_{i=1}^N p_i^2 \alpha_i. \quad \text{(G.16)}$$

From Eq. (G.12), we have $\alpha_i = (A_N - A_i)/(A_N - A_1)$, and substituting in Eq. (G.16), we obtain

$$P_1 \preceq P_2 \text{ if and only if } \sum_{i=1}^N p_i^1 A_i \leq \sum_{i=1}^N p_i^2 A_i,$$

which is equivalent to the desired relation (G.6).

There remains to show that the function U defined by Eq. (G.12) is unique up to a positive linear transformation. Indeed if U^* were another utility function satisfying Eq. (G.6), then by denoting $U^*(O_i) = A_i^*$, $i = 1,\ldots,N$, we would have from Eqs. (G.11) and (G.6)

$$U^*(O_i) = \alpha_i U^*(O_1) + (1-\alpha_i)U^*(O_N).$$

It follows that

$$\alpha_i = \frac{A_N - A_i}{A_N - A_1} = \frac{A_N^* - A_i^*}{A_N^* - A_1^*},$$

from which

$$A_i^* = \frac{A_N^* - A_1^*}{A_N - A_1} A_i + A_N^* - \frac{A_N(A_N^* - A_1^*)}{A_N - A_1}.$$

This proves the theorem. **Q.E.D.**

Returning now to the decision problem, once we assume the existence of a preference relation on the set of lotteries that is characterized by a utility function, we can rank decisions as follows: Given the probability law $P(\cdot \mid d)$ on the set of states of nature $\mathcal{N}$, every decision $d \in \mathcal{D}$ induces a probability law (or lottery) P_d on the set of outcomes $\mathcal{O}$. Under the assumptions of the expected utility theorem, there exists a utility function $U : \mathcal{O} \mapsto \Re$ such that for any $d_1, d_2 \in \mathcal{D}$

$$P_{d_1} \preceq P_{d_2} \text{ if and only if } E_{P_{d_1}}\{U(O)\} \leq E_{P_{d_2}}\{U(O)\}.$$

We have, however,

$$E_{P_d}\{U(O)\} = E\Big\{U\big(f(d,n)\big) \mid d\Big\}, \qquad \text{for all } d \in \mathcal{D},$$

where the expectation on the left is taken with respect to P_d and the expectation on the right is taken with respect to the probability law $P(\cdot \mid d)$ on $\mathcal{N}$. Hence

$$P_{d_1} \preceq P_{d_2} \text{ if and only if } E\Big\{U\big(f(d_1,n)\big) \mid d_1\Big\} \leq E\Big\{U\big(f(d_2,n)\big) \mid d_2\Big\}.$$

By ranking decisions $d \in \mathcal{D}$ in accordance with the ranking of the corresponding P_d, i.e.,

$$\begin{aligned} d_1 \preceq d_2 &\text{ if and only if } P_{d_1} \preceq P_{d_2} \\ &\text{if and only if } E\Big\{U\big(f(d_1,n)\big) \mid d_1\Big\} \leq E\Big\{U\big(f(d_2,n)\big) \mid d_2\Big\}, \end{aligned}$$

we obtain a complete order on the set $\mathcal{D}$ induced by the utility function U. The optimal decision is found by maximization of the numerical function $F : \mathcal{D} \mapsto \Re$, where

$$F(d) = E\Big\{U\big(f(d,n)\big) \mid d\Big\}$$

and the decision problem is formulated in a way that is amenable to mathematical analysis.

The Notion of Risk

Consider a decision maker possessing a utility function U defined over an interval X of real numbers. We say that the decision maker is *risk averse* if

$$E_P\{U(x)\} \leq U\big(E_P\{x\}\big) \tag{G.17}$$

for every probability distribution P on X for which the expected value above is finite. In other words, a decision maker is risk averse if he always prefers the expected value of the lottery over the lottery itself. Such behavior characterizes most decision makers. One may show that risk aversion is equivalent to concavity of the utility function (see Appendix A for the definition and properties of concave and convex functions.) On the other hand, we say that the decision maker is *risk preferring* if the opposite inequality holds in Eq. (G.17), which is the case of a convex utility function. A gambler playing an unbiased roulette and receiving no reward or pleasure from gambling per se is a typical example of a risk preferring decision maker. Finally, a decision maker having a linear utility function is said to be *risk neutral*.

The notion of risk is important since it captures a basic attribute of the attitudes of the decision maker and often characterizes significant aspects of his behavior. An important and widely accepted measure of risk has been proposed by Pratt [Pra64]. He introduced the function

$$r(x) = -\frac{U''(x)}{U'(x)}, \tag{G.18}$$

where U' and U'' denote the first and second derivative of U, and it is assumed that $U'(x) \neq 0$ for all x. This function, called the *index of absolute risk aversion*, measures locally (at the point x) the risk aversion of the decision maker. It can be interpreted as follows.

Let x be a gamble over the set of real numbers (i.e., a random variable) with given distribution and expected value $\overline{x} = E\{x\}$. Let us denote by y the amount of insurance the decision maker is willing to pay in order to avoid the gamble x, and instead receive the expected value $\overline{x}$ of the gamble. In other words, y is such that

$$U(\overline{x} - y) = E\{U(x)\}. \tag{G.19}$$

Intuitively, y provides a natural measure of risk aversion. Using a Taylor series expansion around $\overline{x}$, we have

$$U(\overline{x} - y) = U(\overline{x}) - yU'(\overline{x}) + o(y), \tag{G.20}$$

where by $o(y)$ we denote a quantity that is negligible compared with the scalar α provided α is close to zero, i.e., $\lim_{\alpha \to 0}\big(o(\alpha)/\alpha\big) = 0$. Also we have

$$\begin{aligned} E\big\{U(x)\big\} &= E\Big\{U(\overline{x}) + (x-\overline{x})U'(\overline{x}) + \tfrac{1}{2}(x-\overline{x})^2U''(\overline{x}) + o\big((x-\overline{x})^2\big)\Big\} \\ &= U(\overline{x}) + \tfrac{1}{2}\sigma^2U''(\overline{x}) + E\Big\{o\big((x-\overline{x})^2\big)\Big\}, \end{aligned} \tag{G.21}$$

where σ^2 is the variance of x. From Eqs. (G.19)-(G.21), we have

$$yU'(\overline{x}) = -\tfrac{1}{2}\sigma^2U''(\overline{x}) + o(y) + E\Big\{o\big((x-\overline{x})^2\big)\Big\}.$$

From this equation and Eq. (G.18) it follows that the amount of insurance or risk premium y that the decision maker is willing to pay is proportional (up to first order) to the index of absolute risk aversion $r(\overline{x})$ at the mean $\overline{x}$ of the gamble, thus justifying the use of r as a measure of local risk aversion. Notice that in the investment Example G.2, we have $r(y) = 2/(\alpha - 2y)$, so $r(y)$ tends to decrease as α increases. This fact is reflected in the optimal investment, where an increasing fraction of the capital is invested in the risky asset as α is increased.

The index $r(x)$ often plays an important role in the analysis of behavior of decision makers. It is generally accepted that for most decision makers, $r(x)$ is a decreasing or at least nonincreasing function of x, i.e., the decision maker more readily accepts risk as his wealth is increased. On the other hand, for the quadratic utility function $U(x) = -\frac{1}{2}x^2 + bx + c$, the index $r(x)$ is equal to $(b-x)^{-1}$ and is an increasing function of x (for $x < b$). For this reason the quadratic utility function is often considered inappropriate or at least accepted with reservation in economics applications, despite the analytical simplifications resulting from its use.

Example G.3

An individual with given initial wealth α wishes to invest part of it in a risky asset offering a rate of return e, and the rest in a secure asset offering rate of return $s > 0$. We assume that s is known with certainty while e is a random variable with known probability distribution P. If x is the amount invested in the risky asset, then the final wealth of the decision maker is given by

$$y = s(\alpha - x) + ex = s\alpha + (e-s)x.$$

The decision to be made by the individual is to choose x so as to maximize

$$J(x) = E\big\{U(y)\big\} = E\Big\{U\big(s\alpha + (e-s)x\big)\Big\}$$

subject to the constraint $x \geq 0$. We assume that U is a concave, monotonically increasing, twice continuously differentiable function with negative second derivative, and with index of absolute risk aversion

$$r(y) = -\frac{U''(y)}{U'(y)}.$$

We also assume that the probability distribution of e is such that all expected values appearing below are finite, and furthermore we assume that the utility function U is such that the maximization problem has a solution (the necessary and sufficient conditions for this have an interesting economic interpretation, which is discussed in Bertsekas [Ber74]).

Now given α, the amount x^* to be invested in the risky asset is determined from the necessary conditions

$$\frac{dJ(x^*)}{dx} = E\Big\{(e-s)U'\big(s\alpha + (e-s)x^*\big)\Big\} = 0, \qquad \text{if } x^* > 0, \tag{G.22}$$

$$\frac{dJ(x^*)}{dx} \leq 0, \qquad \text{if } x^* = 0.$$

Now since U' is everywhere positive it follows that if $E\big\{(e-s)\big\} > 0$, then we cannot have $x^* = 0$ since

$$\frac{dJ(0)}{dx} = E\big\{(e-s)\big\}U'(s\alpha) > 0.$$

Hence $E\big\{(e-s)\big\} > 0$ implies $x^* > 0$, or in words, a positive amount will be invested in the risky asset if its expected rate of return is greater than the rate of return of the secure asset.

Assume now that $E\big\{(e-s)\big\} > 0$ and denote by $x^*(\alpha)$ the amount invested in the risky asset when the initial wealth is α. We would like to investigate the effects of changes in initial wealth α on the amount $x^*(\alpha)$ invested. By differentiating Eq. (G.22) with respect to α we obtain

$$E\Big\{(e-s)U''\big(s\alpha + (e-s)x^*(\alpha)\big)\big(s\alpha + (e-s)\big(dx^*(\alpha)/d\alpha\big)\big)\Big\} = 0,$$

from which

$$\frac{dx^*(\alpha)}{d\alpha} = -\frac{E\Big\{(e-s)U''\big(s\alpha + (e-s)x^*(\alpha)\big)\Big\}}{E\Big\{(e-s)^2U''\big(s\alpha + (e-s)x^*(\alpha)\big)\Big\}}.$$

Since the denominator is always negative and the constant s is positive, the sign of $dx^*(\alpha)/d\alpha$ is the same as the sign of

$$E\Big\{(e-s)U''\big(s\alpha + (e-s)x^*(\alpha)\big)\Big\},$$

which using the definition of the index of absolute risk aversion $r(y)$ is equal to

$$f(\alpha) = E\Big\{(e-s)U'\big(s\alpha + (e-s)x^*(\alpha)\big)r\big(s\alpha + (e-s)x^*(\alpha)\big)\Big\}.$$

Now assume that $r(y)$ is monotonically decreasing, i.e.,

$$r(y_1) > r(y_2) \qquad \text{if} \qquad y_1 < y_2.$$

Then we have

$$(e-s)r\big(s\alpha+(e-s)x^*(\alpha)\big) \le (e-r)r(s\alpha)$$

with strict inequality if $e \neq s$, and from the preceding relations, we obtain

$$\begin{aligned} f(\alpha) &> -r(s\alpha)E\Big\{(e-s)U'\big(s\alpha+(e-s)x^*(\alpha)\big)\Big\} \\ &= -r(s\alpha)\frac{dJ\big(x^*(\alpha)\big)}{dx} \\ &= 0. \end{aligned}$$

Thus we have $f(\alpha) > 0$ and hence $dx^*(\alpha)/d\alpha > 0$ if $r(y)$ is monotonically decreasing. Similarly, we obtain $dx^*(\alpha)/d\alpha < 0$ if $r(y)$ is monotonically increasing. In words, the individual, given more wealth, will invest more (less) in the risky asset if his utility function has decreasing (increasing) index of absolute risk aversion. Aside from its intrinsic value, this result illustrates the important role of the index of risk aversion in shaping significant aspects of a decision maker's behavior.

G.3 STOCHASTIC OPTIMAL CONTROL PROBLEMS

The class of decision problems considered so far in this appendix is very broad. In this book we are interested in a subclass of decision problems that involves a dynamic system. Such systems have an input-output description and furthermore in such systems, inputs are selected sequentially after observing past outputs. This allows the possibility of feedback. Let us first give an abstract description of these problems.

Let us consider a system characterized by three sets U, W, and Y, and a function $S : U \times W \mapsto Y$. We call U the *input set*, W the *uncertainty set*, Y the *output set*, and S the *system function*. Thus an input $u \in U$ and an uncertain quantity $w \in W$ produce an output $y = S(u, w)$ through the system function S (see Fig. G.1). Implicit here is the assumption that the choice of the input u is somehow controlled by a decision maker or device to be designed, while w is chosen by nature according to some mechanism, probabilistic or not.

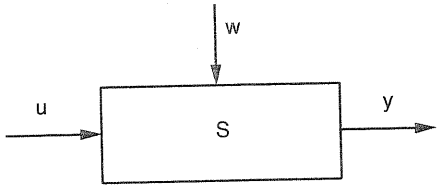


Figure G.1 Structure of an uncertain system: u is the input, w is the uncertain state of nature, y is the output, and S is the system function.

In many problems that evolve in time, the input is a time function or sequence, and there may be a possibility of observing the output y as it evolves in time. Naturally, this output may provide some information about the uncertain quantity w, which may be fruitfully taken into account in choosing the input u by means of a feedback mechanism.

Let us say that a function $\pi : Y \mapsto U$ is a *feedback controller* (otherwise called *policy* or *decision function*) for the system if for each $w \in W$ the equation

$$u = \pi\big(S(u, w)\big)$$

has a unique solution (dependent on w) for u. Thus for any fixed w, a feedback controller π generates a unique input u and hence a unique output y (see Fig. G.2). In any practical situation, the class of admissible feedback controllers is further restricted by causality (present inputs should not depend on future outputs), and possibly other constraints.

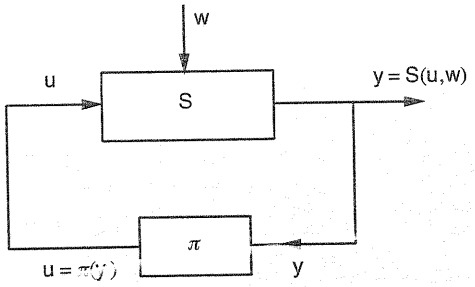


Figure G.2 Structure of a feedback controller π. We require that for each $w \in W$ the equation

$$u = \pi\big(S(u, w)\big)$$

has a unique solution in u.

Given the system (U, W, Y, S) and a set of admissible controllers Π, it is possible to formulate a decision problem in accordance with the theory of the previous section. We take Π as the decision set and W as the set of states of nature. We take as the set of outcomes the Cartesian product of U, W, and Y, i.e.,

$$\mathcal{O} = (U \times W \times Y).$$

Now a feedback controller $\pi \in \Pi$ and a state of nature $w \in W$ generate a unique outcome (u, w, y), where u is the unique solution of the equation $u = \pi\big(S(u, w)\big)$ and $y = S(u, w)$. Thus we may write $(u, w, y) = f(\pi, w)$, where f is some function determined by the system function S.

If G is a numerical function ordering our preferences on $\mathcal{O}$, J is the corresponding payoff function for the decision problem above, and a max-min viewpoint is adopted, then the problem becomes one of finding $\pi \in \Pi$ that maximizes

$$F(\pi) = \min_{w \in W} J(\pi, w) = \min_{w \in W} G(u, w, y),$$

where u and y are expressed in terms of π and w by means of $u = \pi\big(S(u, w)\big)$ and $y = S(u, w)$ (min here denotes least upper bound over the corresponding set W).

If w is selected in accordance with a known probabilistic mechanism, i.e., a given probability law that may depend on π, and the function S and the elements of Π satisfy suitable (measurability) assumptions, then it is possible to use a utility function U to formulate the decision problem as one of finding $\pi \in \Pi$ that maximizes

$$F(\pi) = E\{U(u, w, y)\},$$

where u and y are expressed in terms of π and w by means of $u = \pi\big(S(u, w)\big)$ and $y = S(u, w)$.

While in the formulations just given, we have reduced the problem to one of decision under certainty [the problem of maximizing over Π the numerical function $F(\pi)$], this is not an easy problem. The reason is that due to the feedback possibility, *the set* Π *is a set of functions* (of the system output). This renders inapplicable deterministic optimization techniques, such as those based on linear and nonlinear programming, or Pontryagin's minimum principle. Dynamic programming offers some possibility of analysis by decomposing the problem of minimizing $F(\pi)$ into a sequence of much simpler optimization problems that are solved backwards in time, as discussed in Chapter 1.

Finally, let us indicate how to convert the basic problem of Section 1.2 into the general form given in this section. Referring to the discrete time dynamic system

$$x_{k+1} = f_k(x_k, u_k, w_k), \qquad k = 0, 1, \ldots, N-1, \tag{G.23}$$

introduced in Section 1.2, the system input is the control sequence $u = \{u_0, u_1, \ldots, u_{N-1}\}$, the uncertainty is $w = \{w_0, w_1, \ldots, w_{N-1}\}$ (perhaps together with the initial state x_0, if x_0 is uncertain), the output is the state sequence $y = \{x_0, x_1, \ldots, x_N\}$, and the system function is determined in the obvious manner from the system equation (G.23). The class Π of admissible feedback controllers is the set of sequences of functions $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$, where μ_k is a function that depends on the output y exclusively through the state x_k. Furthermore, μ_k must satisfy constraints such as $\mu_k(x_k) \in U_k(x_k)$ for all x_k and k.

EXERCISES

G.1

Show that there exists a function $G : \mathcal{O} \mapsto \Re$ satisfying relation (G.2) provided the set $\mathcal{O}$ is countable. Show also that if the set of decisions is finite, there exists at least one noninferior decision.

G.2

Let $\mathcal{O} = [-1, 1]$. Define an order on $\mathcal{O}$ by means of

$$O_1 \prec O_2 \text{ if and only if } |O_1| < |O_2| \text{ or } O_1 < O_2 = |O_1|.$$

Show that there exists no real-valued function G on $\mathcal{O}$ such that

$$O_1 \prec O_2 \text{ if and only if } G(O_1) < G(O_2), \quad \text{for all } O_1, O_2 \in \mathcal{O}.$$

Hint: Assume the contrary and associate with every $O \in (0, 1)$ a rational number $r(O)$ such that

$$G(-O) < r(O) < G(O).$$

Show that if $O_1 \neq O_2$, then $r(O_1) \neq r(O_2)$.

G.3 (Experimental Measurement of Utility)

Consider an individual faced with a decision problem with a finite collection of outcomes $O_1, O_2, \ldots, O_N$. Assume that the individual has a preference relation over the set of lotteries on the set of outcomes satisfying Assumptions A.1-A.4 of the expected utility theorem, and hence a utility function over the set of outcomes exists. Suppose also that $O_1 \preceq O_2 \preceq \cdots \preceq O_N$ and furthermore that $O_1 \prec O_N$.

(a) Show that the following method will determine a utility function. Define $U(O_1) = 0, U(O_N) = 1$. Let p_i with $0 \leq p_i \leq 1$ be the probability for which one is indifferent between the lottery $\{(1 - p_i), 0, \ldots, 0, p_i\}$ and O_i occurring with certainty. Then let $U(O_i) = p_i$. Try the procedure on yourself for $O_i = 100i$ with $i = 0, 1, \ldots, 10$.

(b) Show that the following procedure will also yield a utility function. Determine $U(O_{N-1})$ as in (a), but set

$$U(O_{N-2}) = \tilde{p}_{N-2} U(O_{N-1}),$$

where $\tilde{p}_{N-2}$ is the probability for which one is indifferent between the lottery $\{(1 - \tilde{p}_{N-2}), 0, \ldots, 0, \tilde{p}_{N-2}, 0\}$ and O_{N-2} occurring with certainty.

Similarly, set $U(O_i) = \tilde{p}_i U(O_{i+1})$, where $\tilde{p}_i$ is the appropriate probability. Again try this procedure on yourself for $O_i = 100i$ with $i = 0, 1, \ldots, 10$, and compare the results with the ones obtained in part (a).

(c) Devise a third procedure whereby the utilities $U(O_1)$, $U(O_2)$ are specified initially and $U(O_i)$, $i = 3, \ldots, N$, is obtained from $U(O_{i-2})$, $U(O_{i-1})$ through a comparison of the type considered above. Again try this procedure on yourself for $O_i = 100i$ with $i = 0, 1, \ldots, 10$.

G.4

Suppose that two persons, A and B, want to make a bet. Person A will pay \$1 to person B if a certain event occurs and person B will pay x dollars to person A if the event does not occur. Person A believes that the probability of the event occurring is p_A with $0 < p_A < 1$, while person B believes that this probability is p_B with $0 < p_B < 1$. Suppose that the utility functions U_A and U_B of persons A and B are strictly increasing functions of monetary gain. Let α, β be such that

$$U_A(\alpha) = \frac{U_A(0) - p_A U_A(-1)}{1 - p_A}, \qquad U_B(-\beta) = \frac{U_B(0) - p_B U_B(1)}{1 - p_B}.$$

Show that if $\alpha < \beta$, then any value of x between α and β is a mutually satisfactory bet.

References

[ABC65] Atkinson, R. C., Bower, G. H., and Crothers, E. J., 1965. An Introduction to Mathematical Learning Theory, Wiley, N. Y.

[ABF93] Arapostathis, A., Borkar, V., Fernandez-Gaucherand, E., Ghosh, M., and Marcus, S., 1993. "Discrete-Time Controlled Markov Processes with Average Cost Criterion: A Survey," SIAM J. on Control and Optimization, Vol. 31, pp. 282-344.

[ABG49] Arrow, K. J., Blackwell, D., and Girshick, M. A., 1949. "Bayes and Minimax Solutions of Sequential Design Problems," Econometrica, Vol. 17, pp. 213-244.

[AGK77] Athans, M., Ku, R., and Gershwin, S. B., 1977. "The Uncertainty Threshold Principle," IEEE Trans. on Automatic Control, Vol. AC-22, pp. 491-495.

[AHM51] Arrow, K. J., Harris, T., and Marschack, J., 1951. "Optimal Inventory Policy," Econometrica, Vol. 19, pp. 250-272.

[AKS58] Arrow, K. J., Karlin, S., and Scarf, H., 1958. Studies in the Mathematical Theory of Inventory and Production, Stanford Univ. Press, Stanford, CA.

[Abr90] Abramson, B., 1990. "Expected-Outcome: A General Model of Static Evaluation," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, pp. 182-193.

[AdG86] Adams, M., and Guillemin, V., 1986. Measure Theory and Probability, Wadsworth and Brooks, Monterey, CA.

[AsG75] Ash, R. B., and Gardner, M. F., 1975. Topics in Stochastic Processes, Academic Press, N. Y.

[AnM79] Anderson, B. D. O., and Moore, J. B., 1979. Optimal Filtering, Prentice-Hall, Englewood Cliffs, N. J.

[AoL69] Aoki, M., and Li, M. T., 1969. "Optimal Discrete-Time Control Systems with Cost for Observation," IEEE Trans. Automatic Control, Vol. AC-14, pp. 165-175.

[AsW73] Aström, K. J., and Wittenmark, B., 1973. "On Self-Tuning Regulators," Automatica, Vol. 9, pp. 185-199.

[AsW84] Aström, K. J., and Wittenmark, B., 1984. Computer Controlled Systems, Prentice-Hall, Englewood Cliffs, N. J.

[AsW94] Aström, K. J., and Wittenmark, B., 1994. Adaptive Control, (2nd Ed.), Prentice-Hall, Englewood Cliffs, N. J.

[Ash70] Ash, R. B., 1970. Basic Probability Theory, Wiley, N. Y.

[Ash72] Ash, R. B., 1972. Real Analysis and Probability, Academic Press, N. Y.

[Ast83] Aström, K. J., 1983. "Theory and Applications of Adaptive Control – A Survey," Automatica, Vol. 19, pp. 471-486.

[AtF66] Athans, M., and Falb, P., 1966. Optimal Control, McGraw-Hill, N. Y.

[BGM95] Bertsekas, D. P., Guerriero, F., and Musmanno, R., 1995. "Parallel Shortest Path Methods for Globally Optimal Trajectories," High Performance Computing: Technology, Methods, and Applications, (J. Dongarra et al., Eds.), Elsevier.

[BGM96] Bertsekas, D. P., Guerriero, F., and Musmanno, R., 1996. "Parallel Label Corrrecting Methods for Shortest Paths," J. Optimization Theory Appl., Vol. 88, 1996, pp. 297-320.

[BMS99] Boltyanski, V., Martini, H., and Soltan, V., 1999. Geometric Methods and Optimization Problems, Kluwer, Boston.

[BNO03] Bertsekas, D. P., with A. Nedić, A., and A. E. Ozdaglar, 2003. Convex Analysis and Optimization, Athena Scientific, Belmont, MA.

[BTW97] Bertsekas, D. P., Tsitsiklis, J. N., and Wu, C., 1997. "Rollout Algorithms for Combinatorial Optimization," Heuristics, Vol. 3, pp. 245-262.

[BaB95] Basar, T., and Bernhard, P., 1995. H-∞ Optimal Control and Related Minimax Design Problems: A Dynamic Game Approach, Birkhuser, Boston, MA.

[Bar81] Bar-Shalom, Y., 1981. "Stochastic Dynamic Programming: Caution and Probing," IEEE Trans. on Automatic Control, Vol. AC-26, pp. 1184-1195.

[Bas91] Basar, T., 1991. "Optimum Performance Levels for Minimax Filters, Predictors, and Smoothers," Systems and Control Letters, Vol. 16, pp. 309-317.

[Bas00] Basar, T., 2000. "Risk-Averse Designs: From Exponential Cost to Stochastic Games," In T. E. Djaferis and I. C. Schick, (Eds.), System Theory: Modeling, Analysis and Control, Kluwer, Boston, pp. 131-144.

[BeC99] Bertsekas, D. P., and Castanon, D. A., 1999. "Rollout Algorithms for Stochastic Scheduling Problems," Heuristics, Vol. 5, pp. 89-108.

[BeC04] Bertsekas, D. P., and Castanon, D. A., 2004. Unpublished Collaboration.

[BeD62] Bellman, R., and Dreyfus, S., 1962. Applied Dynamic Programming, Princeton Univ. Press, Princeton, N. J.

[BeD02] Bertsimas, D., and Demir, R., 2002. "An Approximate Dynamic Programming Approach to Multi-Dimensional Knapsack Problems," Management Science, Vol. 4, pp. 550-565.

[BeG92] Bertsekas, D. P., and Gallager, R. G., 1992. Data Networks (2nd Edition), Prentice-Hall, Englewood Cliffs, N. J.

[BeN98] Ben-Tal, A., and Nemirovski, A., 1998. "Robust Convex Optimization," Math. of Operations Research, Vol. 23, pp. 769-805.

[BeN01] Ben-Tal, A., and Nemirovski, A., 2001. Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications, SIAM, Phila., PA

[BeP03] Bertsimas, D., and Popescu, I., 2003. "Revenue Management in a Dynamic Network Environment,"Transportation Science, Vol. 37, pp. 257-277.

[BeR71a] Bertsekas, D. P., and Rhodes, I. B., 1971. "Recursive State Estimation for a Set-Membership Description of the Uncertainty," IEEE Trans. Automatic Control, Vol. AC-16, pp. 117-128.

[BeR71b] Bertsekas, D. P., and Rhodes, I. B., 1971. "On the Minimax Reachability of Target Sets and Target Tubes," Automatica, Vol. 7, pp. 233-247.

[BeR73] Bertsekas, D. P., and Rhodes, I. B., 1973. "Sufficiently Informative Functions and the Minimax Feedback Control of Uncertain Dynamic Systems," IEEE Trans. Automatic Control, Vol. AC-18, pp. 117-124.

[BeS78] Bertsekas, D. P., and Shreve, S. E., 1978. Stochastic Optimal Control: The Discrete Time Case, Academic Press, N. Y.; republished by Athena Scientific, Belmont, MA, 1996; can be downloaded from the author's website.

[BeS03] Bertsimas, D., and Sim, M., 2003. "Robust Discrete Optimization and Network Flows," Math. Programming, Series B, Vol. 98, pp. 49-71.

[BeT89] Bertsekas, D. P., and Tsitsiklis, J. N., 1989. Parallel and Distributed Computation: Numerical Methods, Prentice-Hall, Englewood Cliffs, N. J.; republished by Athena Scientific, Belmont, MA, 1997.

[BeT91] Bertsekas, D. P., and Tsitsiklis, J. N., 1991. "An Analysis of Stochastic Shortest Path Problems," Math. Operations Res., Vol. 16, pp. 580-595.

[BeT96] Bertsekas, D. P., and Tsitsiklis, J. N., 1996. Neuro-Dynamic Programming, Athena Scientific, Belmont, MA.

[BeT97] Bertsimas, D., and Tsitsiklis, J. N., 1997. Introduction to Linear Optimization, Athena Scientific, Belmont, MA.

[Bel57] Bellman, R., 1957. Dynamic Programming, Princeton University Press, Princeton, N. J.

[Ber70] Bertsekas, D. P., 1970. "On the Separation Theorem for Linear Systems, Quadratic Criteria, and Correlated Noise," Unpublished Report, Electronic Systems Lab., Massachusetts Institute of Technology.

[Ber71] Bertsekas, D. P., 1971. "Control of Uncertain Systems With a Set-Membership Description of the Uncertainty," Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge, MA (available in scanned form from the author's www site).

[Ber72a] Bertsekas, D. P., 1972. "Infinite Time Reachability of State Space Regions by Using Feedback Control," IEEE Trans. Automatic Control, Vol. AC-17, pp. 604-613.

[Ber72b] Bertsekas, D. P., 1972. "On the Solution of Some Minimax Control Problems," Proc. 1972 IEEE Decision and Control Conf., New Orleans, LA.

[Ber74] Bertsekas, D. P., 1974. "Necessary and Sufficient Conditions for Existence of an Optimal Portfolio," J. Econ. Theory, Vol. 8, pp. 235-247.

[Ber75] Bertsekas, D. P., 1975. "Convergence of Discretization Procedures in Dynamic Programming," IEEE Trans. Automatic Control, Vol. AC-20, pp. 415-419.

[Ber76] Bertsekas, D. P., 1976. Dynamic Programming and Stochastic Control, Academic Press, N. Y.

[Ber82a] Bertsekas, D. P., 1982. "Distributed Dynamic Programming," IEEE Trans. Automatic Control, Vol. AC-27, pp. 610-616.

[Ber82b] Bertsekas, D. P., 1982. Constrained Optimization and Lagrange Multiplier Methods, Academic Press, N. Y.; republished by Athena Scientific, Belmont, MA, 1996.

[Ber93] Bertsekas, D. P., 1993. "A Simple and Fast Label Correcting Algorithm for Shortest Paths," Networks, Vol. 23, pp. 703-709.

[Ber97] Bertsekas, D. P., 1997. "Differential Training of Rollout Policies," Proc. of the

35th Allerton Conference on Communication, Control, and Computing, Allerton Park, Ill.

[Ber98a] Bertsekas, D. P., 1998. Network Optimization: Continuous and Discrete Models, Athena Scientific, Belmont, MA.

[Ber98b] Bertsekas, D. P., 1998. "A New Value Iteration Method for the Average Cost Dynamic Programmming Problem," SIAM J. on Control and Optimization, Vol. 36, pp. 742-759.

[Ber99] Bertsekas, D. P., 1999. Nonlinear Programming, (2nd Ed.), Athena Scientific, Belmont, MA.

[BiL97] Birge, J. R., and Louveaux, 1997. Introduction to Stochastic Programming, Springer-Verlag, New York, N. Y.

[Bis95] Bishop, C. M. 1995. Neural Networks for Pattern Recognition, Oxford University Press, N. Y.

[BlT00] Blondel, V. D., and Tsitsiklis, J. N., 2000. "A Survey of Computational Complexity Results in Systems and Control," Automatica, Vol. 36, pp. 1249-1274.

[Bla99] Blanchini, F., 1999. "Set Invariance in Control – A Survey," Automatica, Vol. 35, pp. 1747-1768.

[BoV79] Borkar, V., and Varaiya, P. P., 1979. "Adaptive Control of Markov Chains, I: Finite Parameter Set," IEEE Trans. Automatic Control, Vol. AC-24, pp. 953-958.

[CGC04] Chang, H. S., Givan, R. L., and Chong, E. K. P., 2004. "Parallel Rollout for Online Solution of Partially Observable Markov Decision Processes," Discrete Event Dynamic Systems, Vol. 14, pp. 309-341.

[CaB04] Camacho, E. F., and Bordons, C., 2004. Model Predictive Control, 2nd Edition, Springer-Verlag, New York, N. Y.

[ChT89] Chow, C.-S., and Tsitsiklis, J. N., 1989. "The Complexity of Dynamic Programming," Journal of Complexity, Vol. 5, pp. 466-488.

[ChT91] Chow, C.-S., and Tsitsiklis, J. N., 1991. "An Optimal One–Way Multigrid Algorithm for Discrete–Time Stochastic Control," IEEE Trans. on Automatic Control, Vol. AC-36, 1991, pp. 898-914.

[Che72] Chernoff, H., 1972. "Sequential Analysis and Optimal Design," Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, PA.

[Chr97] Christodouleas, J. D., 1997. "Solution Methods for Multiprocessor Network Scheduling Problems with Application to Railroad Operations," Ph.D. Thesis, Operations Research Center, Massachusetts Institute of Technology.

[Chu60] Chung, K. L., 1960. Markov Chains with Stationary Transition Probabilities, Springer-Verlag, N. Y.

[CoL55] Coddington, E. A., and Levinson, N., 1955. Theory of Ordinary Differential Equations, McGraw-Hill, N. Y.

[DeG70] DeGroot, M. H., 1970. Optimal Statistical Decisions, McGraw-Hill, N. Y.

[DeP84] Deo, N., and Pang, C., 1984. "Shortest Path Problems: Taxonomy and Annotation," Networks, Vol. 14, pp. 275-323.

[Del89] Deller, J. R., 1989. "Set Membership Identification in Digital Signal Processing," IEEE ASSP Magazine, Oct., pp. 4-20.

[DoS80] Doshi, B., and Shreve, S., 1980. "Strong Consistency of a Modified Maximum

Likelihood Estimator for Controlled Markov Chains," J. of Applied Probability, Vol. 17, pp. 726-734.

[Dre65] Dreyfus, S. D., 1965. Dynamic Programming and the Calculus of Variations, Academic Press, N. Y.

[Dre69] Dreyfus, S. D., 1969. "An Appraisal of Some Shortest-Path Algorithms," Operations Research, Vol. 17, pp. 395-412.

[Eck68] Eckles, J. E., 1968. "Optimum Maintenance with Incomplete Information," Operations Res., Vol. 16, pp. 1058-1067.

[Elm78] Elmaghraby, S. E., 1978. Activity Networks: Project Planning and Control by Network Models, Wiley-Interscience, N. Y.

[Fal87] Falcone, M., 1987. "A Numerical Approach to the Infinite Horizon Problem of Deterministic Control Theory," Appl. Math. Opt., Vol. 15, pp. 1-13.

[FeM94] Fernandez-Gaucherand, E., and Markus, S. I., 1994. "Risk Sensitive Optimal Control of Hidden Markov Models," Proc. 33rd IEEE Conf. Dec. Control, Lake Buena Vista, Fla.

[FeV02] Ferris, M. C., and Voelker, M. M., 2002. "Neuro-Dynamic Programming for Radiation Treatment Planning," Numerical Analysis Group Research Report NA-02/06, Oxford University Computing Laboratory, Oxford University.

[FeV04] Ferris, M. C., and Voelker, M. M., 2004. "Fractionation in Radiation Treatment Planning," Mathematical Programming B, Vol. 102, pp. 387-413.

[Fel68] Feller, W., 1968. An Introduction to Probability Theory and its Applications, Wiley, N. Y.

[Fis70] Fishburn, P. C., 1970. Utility Theory for Decision Making, Wiley, N. Y.

[For56] Ford, L. R., Jr., 1956. "Network Flow Theory," Report P-923, The Rand Corporation, Santa Monica, CA.

[For73] Forney, G. D., 1973. "The Viterbi Algorithm," Proc. IEEE, Vol. 61, pp. 268-278.

[Fox71] Fox, B. L., 1971. "Finite State Approximations to Denumerable State Dynamic Progams," J. Math. Anal. Appl., Vol. 34, pp. 665-670.

[GaP88] Gallo, G., and Pallottino, S., 1988. "Shortest Path Algorithms," Annals of Operations Research, Vol. 7, pp. 3-79.

[Gal99] Gallager, R. G., 1999. Discrete Stochastic Processes, Kluwer, Boston.

[GoR85] Gonzalez, R., and Rofman, E., 1985. "On Deterministic Control Problems: An Approximation Procedure for the Optimal Cost, Parts I, II," SIAM J. Control Optimization, Vol. 23, pp. 242-285.

[GoS84] Goodwin, G. C., and Sin, K. S. S., 1984. Adaptive Filtering, Prediction, and Control, Prentice-Hall, Englewood Cliffs, N. J.

[GrA66] Groen, G. J., and Atkinson, R. C., 1966. "Models for Optimizing the Learning Process," Psychol. Bull., Vol. 66, pp. 309-320.

[GuF63] Gunckel, T. L., and Franklin, G. R., 1963. "A General Solution for Linear Sampled-Data Control," Trans. ASME Ser. D. J. Basic Engrg., Vol. 85, pp. 197-201.

[GuM01] Guerriero, F., and Musmanno, R., 2001. "Label Correcting Methods to Solve Multicriteria Shortest Path Problems," J. Optimization Theory Appl., Vol. 111, pp. 589-613.

[GuM03] Guerriero, F., and Mancini, M., 2003. "A Cooperative Parallel Rollout Algorithm for the Sequential Ordering Problem," Parallel Computing, Vol. 29, pp. 663-677.

[HMS55] Holt, C. C., Modigliani, F., and Simon, H. A., 1955. "A Linear Decision Rule for Production and Employment Scheduling," Management Sci., Vol. 2, pp. 1-30.

[HPC96] Helmsen, J., Puckett, E. G., Colella, P., and Dorr, M., 1996. "Two New Methods for Simulating Photolithography Development," SPIE, Vol. 2726, pp. 253-261.

[HaL82] Hajek, B., and van Loon, T., 1982. "Decentralized Dynamic Control of a Multiaccess Broadcast Channel," IEEE Trans. Automatic Control, Vol. AC-27, pp. 559-569.

[Hak70] Hakansson, N. H., 1970. "Optimal Investment and Consumption Strategies under Risk for a Class of Utility Functions," Econometrica, Vol. 38, pp. 587-607.

[Hak71] Hakansson, N. H., 1971. "On Myopic Portfolio Policies, With and Without Serial Correlation of Yields," The Journal of Business of the University of Chicago, Vol. 44, pp. 324-334.

[Han80] Hansen, P., 1980. "Bicriterion Path Problems," in Multiple-Criteria Decision Making: Theory and Applications, Edited by G. Fandel and T. Gal, Springer Verlag, Heidelberg, Germany, pp. 109-127.

[Hay98] Haykin, S., 1998. Neural Networks: A Comprehensive Foundation, (2nd Ed.), McMillan, N. Y.

[Hes66] Hestenes, M. R., 1966. Calculus of Variations and Optimal Control Theory, Wiley, N. Y.

[Her89] Hernandez-Lerma, O., 1989. Adaptive Markov Control Processes, Springer-Verlag, N. Y.

[HoK71] Hoffman, K., and Kunze, R., 1971. Linear Algebra, Prentice-Hall, Englewood Cliffs, N. J.

[IEE71] IEEE Trans. Automatic Control, 1971. Special Issue on Linear-Quadratic Gaussian Problem, Vol. AC-16.

[IoS96] Ioannou, P. A., and Sun, J., 1996. Robust Adaptive Control, Prentice-Hall, Englewood Cliffs, N. J.

[JBE94] James, M. R., Baras, J. S., and Elliott, R. J., 1994. "Risk-Sensitive Control and Dynamic Games for Partially Observed Discrete-Time Nonlinear Systems," IEEE Trans. on Automatic Control, Vol. AC-39, pp. 780-792.

[Jac73] Jacobson, D. H., 1973. "Optimal Stochastic Linear Systems With Exponential Performance Criteria and their Relation to Deterministic Differential Games," IEEE Trans. Automatic Control, Vol. AC-18, pp. 124-131.

[Jaf84] Jaffe, J. M., 1984. "Algorithms for Finding Paths with Multiple Constraints," Networks, Vol. 14, pp. 95-116.

[Jew63] Jewell, W., 1963. "Markov Renewal Programming I and II," Operations Research, Vol. 2, pp. 938-971.

[JoT61] Joseph, P. D., and Tou, J. T., 1961. "On Linear Control Theory," AIEE Trans., Vol. 80 (II), pp. 193-196.

[KKK95] Krstic, M., Kanellakopoulos, I., Kokotovic, P., 1995. Nonlinear and Adaptive Control Design, J. Wiley, N. Y.

[KGB82] Kimemia, J., Gershwin, S. B., and Bertsekas, D. P., 1982. "Computation of Production Control Policies by a Dynamic Programming Technique," in Analysis and

Optimization of Systems, A. Bensoussan and J. L. Lions (eds.), Springer-Verlag, N. Y., pp. 243-269.

[KLB92] Kosut, R. L., Lau, M. K., and Boyd, S. P., 1992. "Set-Membership Identification of Systems with Parametric and Nonparametric Uncertainty," IEEE Trans. on Automatic Control, Vol. AC-37, pp. 929-941.

[KaD66] Karush, W., and Dear, E. E., 1966. "Optimal Stimulus Presentation Strategy for a Stimulus Sampling Model of Learning," J. Math. Psychology, Vol. 3, pp. 15-47.

[KaK58] Kalman, R. E., and Koepcke, R. W., 1958. "Optimal Synthesis of Linear Sampling Control Systems Using Generalized Performance Indexes," Trans. ASME, Vol. 80, pp. 1820-1826.

[KaW94] Kall, P., and Wallace, S. W., 1994. Stochastic Programming, Wiley, Chichester, UK.

[Kal60] Kalman, R. E., 1960. "A New Approach to Linear Filtering and Prediction Problems," Trans. ASME Ser. D. J. Basic Engrg., Vol. 82, pp. 35-45.

[KeS60] Kemeny, J. G., and Snell, J. L., 1960. Finite Markov Chains, Van Nostrand-Reinhold, N. Y.

[KeG88] Keerthi, S. S., and Gilbert, E. G., 1988. "Optimal, Infinite Horizon Feedback Laws for a General Class of Constrained Discete Time Systems: Stability and Moving-Horizon Approximations," J. Optimization Theory Appl., Vo. 57, pp. 265-293.

[Kim82] Kimemia, J., 1982. "Hierarchical Control of Production in Flexible Manufacturing Systems," Ph.D. Thesis, Dep. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

[KuA77] Ku, R., and Athans, M., 1977. "Further Results on the Uncertainty Threshold Principle," IEEE Trans. on Automatic Control, Vol. AC-22, pp. 866-868.

[KuD92] Kushner, H. J., and Dupuis, P. G., 1992. Numerical Methods for Stochastic Control Problems in Continuous Time, Springer-Verlag, N. Y.

[KuL82] Kumar, P. R., and Lin, W., 1982. "Optimal Adaptive Controllers for Unknown Markov Chains," IEEE Trans. Automatic Control, Vol. AC-27, pp. 765-774.

[KuV86] Kumar, P. R., and Varaiya, P. P., 1986. Stochastic Systems: Estimation, Identification, and Adaptive Control, Prentice-Hall, Englewood Cliffs, N. J.

[KuV97] Kurzhanski, A., and Valyi, I., 1997. Ellipsoidal Calculus for Estimation and Control, Birkhauser, Boston, MA.

[Kum83] Kumar, P. R., 1983. "Optimal Adaptive Control of Linear-Quadratic-Gaussian Systems," SIAM J. on Control and Optimization, Vol. 21, pp. 163-178.

[Kum85] Kumar, P. R., 1985. "A Survey of Some Results in Stochastic Adaptive Control," SIAM J. on Control and Optimization, Vol. 23, pp. 329-380.

[Kus90] Kushner, H. J., 1990. "Numerical Methods for Continuous Control Problems in Continuous Time," SIAM J. on Control and Optimization, Vol. 28, pp. 999-1048.

[Las85] Lasserre, J. B., 1985. "A Mixed Forward-Backward Dynamic Programming Method Using Parallel Computation," J. Optimization Theory Appl., Vol. 45, pp. 165-168.

[Lev84] Levy, D., 1984. The Chess Computer Handbook, B. T. Batsford Ltd., London.

[LiR71] Lippman, S. A., and Ross, S. M., 1971. "The Streetwalker's Dilemma: A Job-Shop Model," SIAM J. of Appl. Math., Vol. 20, pp. 336-342.

[LjS83] Ljung, L., and Soderstrom, T., 1983. Theory and Practice of Recursive Identification, MIT Press, Cambridge, MA.

[Lju86] Ljung, L., 1986. System Identification: Theory for the User, Prentice-Hall, Englewood Cliffs, N. J.

[Lov91a] Lovejoy, W. S., 1991. "Computationally Feasible Bounds for Partially Observed Markov Decision Processes," Operations Research, Vol. 39, pp. 162175.

[Lov91b] Lovejoy, W. S., 1991. "A Survey of Algorithmic Methods for Partially Observed Markov Decision Processes," Annals of Operations Research, Vol. 18, pp. 47-66.

[Lue69] Luenberger, D. G., 1969. Optimization by Vector Space Methods, Wiley, N. Y.

[Lue84] Luenberger, D. G., 1984. Linear and Nonlinear Programming, Addi-son-Wesley, Reading, MA.

[MHK98] Meuleau, N., Hauskrecht, M., Kim, K.-E., Peshkin, L., Kaelbling, L. K., and Dean, T., 1998. "Solving Very Large Weakly Coupled Markov Decision Processes," Proc. of the Fifteenth National Conference on Artificial Intelligence, Madison, WI, pp. 165-172.

[MMB02] McGovern, A., Moss, E., and Barto, A., 2002. "Building a Basic Building Block Scheduler Using Reinforcement Learning and Rollouts," Machine Learning, Vol. 49, pp. 141-160.

[MPP04] Meloni, C., Pacciarelli, D., and Pranzo, M., 2004. "A Rollout Metaheuristic for Job Shop Scheduling Problems," Annals of Operations Research, Vol. 131, pp. 215-235.

[MRR00] Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M., 2000. "Constrained Model Predictive Control: Stability and Optimality," Automatica, Vol. 36, pp. 789-814.

[Mac02] Maciejowski, J. M., 2002. Predictive Control with Constraints, Addison-Wesley, Reading, MA.

[Mar84] Martins, E. Q. V., 1984. "On a Multicriteria Shortest Path Problem," European J. of Operational Research, Vol. 16, pp. 236-245.

[May01] Mayne, D. Q., 2001. "Control of Constrained Dynamic Systems," European Journal of Control, Vol. 7, pp. 87-99.

[McQ66] MacQueen, J., 1966. "A Modified Dynamic Programming Method for Markovian Decision Problems," J. Math. Anal. Appl., Vol. 14, pp. 38-43.

[Mik79] Mikhailov, V. A., 1979. Methods of Random Multiple Access, Candidate Engineering Thesis, Moscow Institute of Physics and Technology, Moscow.

[MoL99] Morari, M., and Lee, J. H., 1999. "Model Predictive Control: Past, Present, and Future," Computers and Chemical Engineering, Vol. 23, pp. 667-682.

[Mos68] Mossin, J., 1968. "Optimal Multi-Period Portfolio Policies," J. Business, Vol. 41, pp. 215-229.

[NeW88] Nemhauser, G. L., and Wolsey, L. A., 1988. Integer and Combinatorial Optimization, Wiley, N. Y.

[New75] Newborn, M., 1975. Computer Chess, Academic Press, N. Y.

[Nic66] Nicholson, T., 1966. "Finding the Shortest Route Between Two Points in a Network," The Computer Journal, Vol. 9, pp. 275-280.

[Nil71] Nilsson, N. J., 1971. Problem-Solving Methods in Artificial Intelligence, McGraw-Hill, N. Y.

[Nil80] Nilsson, N. J., 1971. Principles of Artificial Intelligence, Morgan-Kaufmann, San Mateo, Ca.

[PBG65] Pontryagin, L. S., Boltyanski, V., Gamkrelidze, R., and Mishchenko, E., 1965. The Mathematical Theory of Optimal Processes, Interscience Publishers, Inc., N. Y.

[PBT98] Polymenakos, L. C., Bertsekas, D. P., and Tsitsiklis, J. N., 1998. "Efficient Algorithms for Continuous-Space Shortest Path Problems," IEEE Trans. on Automatic Control, Vol. 43, pp. 278-283.

[PaS82] Papadimitriou, C. H., and Steiglitz, K., 1982. Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Englewood Cliffs, N. J.

[PaT87] Papadimitriou, C. H., and Tsitsiklis, J. N., 1987. "The Complexity of Markov Decision Processes," Math. Operations Res., Vol. 12, pp. 441-450.

[Pap74] Pape, V., 1974. "Implementation and Efficiency of Moore Algorithms for the Shortest Path Problem," Math. Progr., Vol. 7, pp. 212-222.

[Pap65] Papoulis, A., 1965. Probability, Random Variables and Stochastic Processes, McGraw-Hill, N. Y.

[Pea84] Pearl, J., 1984. Heuristics, Addison-Wesley, Reading, MA.

[Pic90] Picone, J., 1990. "Continuous Speech Recognition Using Hidden Markov Models," IEEE ASSP Magazine, July Issue, pp. 26-41.

[Pin95] Pinedo, M., 1995. Scheduling: Theory, Algorithms, and Systems, Prentice-Hall, Englewood Cliffs, N. J.

[Pra64] Pratt, J. W., 1964. "Risk Aversion in the Small and in the Large," Econometrica, Vol. 32, pp. 300-307.

[Pre95] Prekopa, A., 1995. Stochastic Programming, Kluwer, Boston.

[PrS94] Proakis, J. G., and Salehi, M., 1994. Communication Systems Engineering, Prentice-Hall, Englewood Cliffs, N. J.

[Rab89] Rabiner, L. R., 1989. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proc. of the IEEE, Vol. 77, pp. 257-286.

[Roc70] Rockafellar, R. T., 1970. Convex Analysis, Princeton University Press, Princeton, N. J.

[Ros70] Ross, S. M., 1970. Applied Probability Models with Optimization Applications, Holden-Day, San Francisco, CA.

[Ros83] Ross, S. M., 1983. Introduction to Stochastic Dynamic Programming, Academic Press, N. Y.

[Ros85] Ross, S. M., 1985. Probability Models, Academic Press, Orlando, Fla.

[Roy88] Royden, H. L., 1988. Principles of Mathematical Analysis, (3rd Ed.), McGraw-Hill, N. Y.

[Rud76] Rudin, W., 1976. Real Analysis, (3rd Ed.), McGraw-Hill, N. Y.

[Rus97] Rust, J., 1997. "Using Randomization to Break the Curse of Dimensionality," Econometrica, Vol. 65, pp. 487-516.

[SBB89] Sastry, S., Bodson, M., and Bartram, J. F., 1989. Adaptive Control: Stability, Convergence, and Robustness, Prentice-Hall, Englewood Cliffs, N. J.

[SGC02] Savagaonkar, U., Givan, R., and Chong, E. K. P., 2002. "Sampling Techniques for Zero-Sum, Discounted Markov Games," in Proc. 40th Allerton Conference on Com-

munication, Control and Computing, Monticello, Ill.

[Sam69] Samuelson, P. A., 1969. "Lifetime Portfolio Selection by Dynamic Stochastic Programming," Review of Economics and Statistics, Vol. 51, pp. 239-246.

[Sar87] Sargent, T. J., 1987. Dynamic Macroeconomic Theory, Harvard Univ. Press, Cambridge, MA.

[Sca60] Scarf, H., 1960. "The Optimality of (s, S) Policies for the Dynamic Inventory Problem," Proceedings of the 1st Stanford Symposium on Mathematical Methods in the Social Sciences, Stanford University Press, Stanford, CA.

[Sch68] Schweppe, F. C., 1968. "Recursive State Estimation; Unknown but Bounded Errors and System Inputs," IEEE Trans. Automatic Control, Vol. AC-13.

[Sch74] Schweppe, F. C., 1974. Uncertain Dynamic Systems, Academic Press, N. Y.

[Sch97] Schaeffer, J., 1997. One Jump Ahead, Springer-Verlag, N. Y.

[Sec00] Secomandi, N., 2000. "Comparing Neuro-Dynamic Programming Algorithms for the Vehicle Routing Problem with Stochastic Demands," Computers and Operations Research, Vol. 27, pp. 1201-1225.

[Sec01] Secomandi, N., 2001. "A Rollout Policy for the Vehicle Routing Problem with Stochastic Demands," Operations Research, Vol. 49, pp. 796-802.

[Sec03] Secomandi, N., 2003. "Analysis of a Rollout Approach to Sequencing Problems with Stochastic Routing Applications," J. of Heuristics, Vol. 9, pp. 321-352.

[Set99a] Sethian, J. A., 1999. Level Set Methods and Fast Marching Methods Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, Cambridge University Press, N. Y.

[Set99b] Sethian, J. A., 1999. "Fast Marching Methods," SIAM Review, Vol. 41, pp. 199-235.

[Sha50] Shannon, C., 1950. "Programming a Digital Computer for Playing Chess," Phil. Mag., Vol. 41, pp. 356-375.

[Shi64] Shiryaev, A. N., 1964. "On Markov Sufficient Statistics in Non-Additive Bayes Problems of Sequential Analysis," Theory of Probability and Applications, Vol. 9, pp. 604-618.

[Shi66] Shiryaev, A. N., 1966. "On the Theory of Decision Functions and Control by an Observation Process with Incomplete Data," Selected Translations in Math. Statistics and Probability, Vol. 6, pp. 162-188.

[Shr81] Shreve, S. E., 1981. "A Note on Optimal Switching Between Two Activities," Naval Research Logistics Quarterly, Vol. 28, pp. 185-190.

[Sim56] Simon, H. A., 1956. "Dynamic Programming Under Uncertainty with a Quadratic Criterion Function," Econometrica, Vol. 24, pp. 74-81.

[Skl88] Sklar, B., 1988. Digital Communications: Fundamentals and Applications, Prentice-Hall, Englewood Cliffs, N. J.

[SlL91] Slotine, J.-J. E., and Li, W., Applied Nonlinear Control, Prentice-Hall, Englewood Cliffs, N. J.

[SmS73] Smallwood, R. D., and Sondik, E. J., 1973. "The Optimal Control of Partially Observable Markov Processes Over a Finite Horizon," Operations Res., Vol. 11, pp. 1071-1088.

[Sma71] Smallwood, R. D., 1971. "The Analysis of Economic Teaching Strategies for a Simple Learning Model," J. Math. Psychology, Vol. 8, pp. 285-301.

[Son71] Sondik, E. J., 1971. "The Optimal Control of Partially Observable Markov Processes," Ph.D. Dissertation, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.

[StW91] Stewart, B. S., and White, C. C., 1991. "Multiobjective A^*," J. ACM, Vol. 38, pp. 775-814.

[Sti94] Stirzaker, D., 1994. Elementary Probability, Cambridge University Press, Cambridge.

[StL89] Stokey, N. L., and Lucas, R. E., 1989. Recursive Methods in Economic Dynamics, Harvard University Press, Cambridge, MA.

[Str65] Striebel, C. T., 1965. "Sufficient Statistics in the Optimal Control of Stochastic Systems," J. Math. Anal. Appl., Vol. 12, pp. 576-592.

[Str76] Strang, G., 1976. Linear Algebra and its Applications, Academic Press, N. Y.

[SuB98] Sutton, R., and Barto, A. G., 1998. Reinforcement Learning, MIT Press, Cambridge, MA.

[TeG96] Tesauro, G., and Galperin, G. R., 1996. "On-Line Policy Improvement Using Monte Carlo Search," presented at the 1996 Neural Information Processing Systems Conference, Denver, CO; also in M. Mozer et al. (eds.), Advances in Neural Information Processing Systems 9, MIT Press (1997).

[ThW66] Thau, F. E., and Witsenhausen, H. S., 1966. "A Comparison of Closed-Loop and Open-Loop Optimum Systems," IEEE Trans. Automatic Control, Vol. AC-11, pp. 619-621.

[The54] Theil, H., 1954. "Econometric Models and Welfare Maximization," Weltwirtsch. Arch., Vol. 72, pp. 60-83.

[Tsi84a] Tsitsiklis, J. N., 1984. "Convexity and Characterization of Optimal Policies in a Dynamic Routing Problem," J. Optimization Theory Appl., Vol. 44, pp. 105-136.

[Tsi84b] Tsitsiklis, J. N., 1984. "Periodic Review Inventory Systems with Continuous Demand and Discrete Order Sizes," Management Sci., Vol. 30, pp. 1250-1254.

[Tsi87] Tsitsiklis, J. N., 1987. "Analysis of a Multiaccess Control Scheme," IEEE Trans. Automatic Control, Vol. AC-32, pp. 1017-1020.

[Tsi95] Tsitsiklis, J. N., 1995. "Efficient Algorithms for Globally Optimal Trajectories," IEEE Trans. Automatic Control, Vol. AC-40, pp. 1528-1538.

[TuP03] Tu, F., and Pattipati, K. R., 2003. "Rollout Strategies for Sequential Fault Diagnosis," IEEE Trans. on Systems, Man and Cybernetics, Part A, pp. 86-99.

[YuB04] Yu, H., and Bertsekas, D. P., 2004. "Discretized Approximations for POMDP with Average Cost," Proc. of 20th Conference on Uncertainty in Artificial Intelligence, Banff, Canada.

[VaW89] Varaiya, P., and Wets, R. J-B., 1989. "Stochastic Dynamic Optimization Approaches and Computation," Mathematical Programming: State of the Art, M. Iri and K. Tanabe (eds.), Kluwer, Boston, pp. 309-332.

[Vei65] Veinott, A. F., Jr., 1965. "The Optimal Inventory Policy for Batch Ordering," Operations Res., Vol. 13, pp. 424-432.

[Vei66] Veinott, A. F., Jr., 1966. "The Status of Mathematical Inventory Theory," Management Sci., Vol. 12, pp. 745-777.

[Vin74] Vincke, P., 1974. "Problemes Multicriteres," Cahiers du Centre d' Etudes de Recherche Operationnelle, Vol. 16, pp. 425-439.

[Vit67] Viterbi, A. J., 1967. "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," IEEE Trans. on Info. Theory, Vol. IT-13, pp. 260-269.

[WCG03] Wu, G., Chong, E. K. P., and Givan, R. L., 2003. "Congestion Control Using Policy Rollout," Proc. 2nd IEEE CDC, Maui, Hawaii, pp. 4825-4830.

[Wal47] Wald, A., 1947. Sequential Analysis, Wiley, N. Y.

[WeP80] Weiss, G., and Pinedo, M., 1980. "Scheduling Tasks with Exponential Service Times on Nonidentical Processors to Minimize Various Cost Functions," J. Appl. Prob., Vol. 17, pp. l87-202.

[WhH80] White, C. C., and Harrington, D. P., 1980. "Application of Jensen's Inequality to Adaptive Suboptimal Design," J. Optimization Theory Appl., Vol. 32, pp. 89-99.

[WhS89] White, C. C., and Scherer, W. T., 1989. "Solution Procedures for Partially Observed Markov Decision Processes," Operations Res., Vol. 30, pp. 791-797.

[Whi63] Whittle, P., 1963. Prediction and Regulation by Linear Least-Square Methods, English Universities Press, London.

[Whi69] White, D. J., 1969. Dynamic Programming, Holden-Day, San Francisco, CA.

[Whi78] Whitt, W., 1978. "Approximations of Dynamic Programs I," Math. Operations Res., Vol. 3, pp. 231-243.

[Whi79] Whitt, W., 1979. "Approximations of Dynamic Programs II," Math. Operations Res., Vol. 4, pp. 179-185.

[Whi82] Whittle, P., 1982. Optimization Over Time, Wiley, N. Y., Vol. 1, 1982, Vol. 2, 1983.

[Whi90] Whittle, P., 1990. Risk-Sensitive Optimal Control, Wiley, N. Y.

[Wit66] Witsenhausen, H. S., 1966. "Minimax Control of Uncertain Systems," Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge, MA.

[Wit68] Witsenhausen, H. S., 1968. "Sets of Possible States of Linear Systems Given Perturbed Observations," IEEE Trans. Automatic Control, Vol. AC-13.

[Wit69] Witsenhausen, H. S., 1969. "Inequalities for the Performance of Suboptimal Uncertain Systems," Automatica, Vol. 5, pp. 507-512.

[Wit70] Witsenhausen, H. S., 1970. "On Performance Bounds for Uncertain Systems," SIAM J. on Control, Vol. 8, pp. 55-89.

[Wit71] Witsenhausen, H. S., 1971. "Separation of Estimation and Control for Discrete-Time Systems," Proc. IEEE, Vol. 59, pp. 1557-1566.

[Wol98] Wolsey, L. A., 1998. Integer Programming, Wiley, N. Y.

[WuB99] Wu, C. C., and Bertsekas, D. P., 1999. "Distributed Power Control Algorithms for Wireless Networks," unpublished report, available from the author's www site.

[YDR05] Yan, X., Diaconis, P., Rusmevichientong, P., and Van Roy, B., 2005. "Solitaire: Man Versus Machine," Advances in Neural Information Processing Systems, Vol. 17, to appear.

INDEX