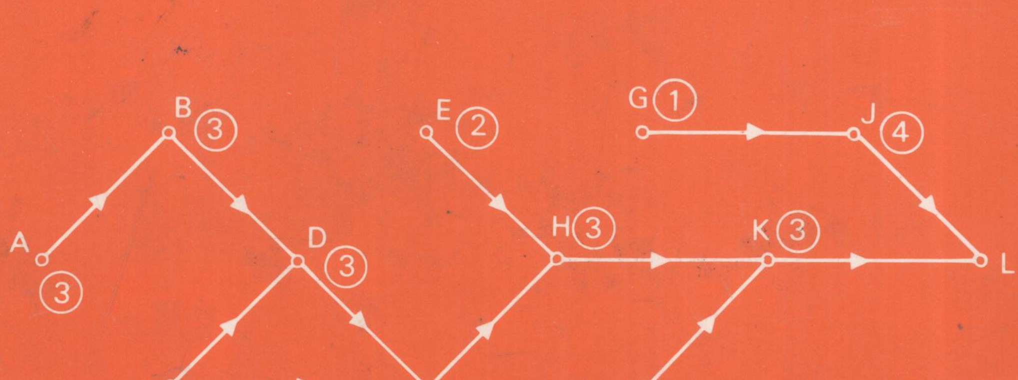
CONTROL AND SYSTEMS THEORY Volume 7

Principles of Dynamic Programming

Part I
Basic Analytic and Computational Methods

Robert E. Larson
John L. Casti



Principles of Dynamic Programming

Part I
Basic Analytic and Computational Methods

by **ROBERT E. LARSON**
Systems Control, Inc.
Palo Alto, California

JOHN L. CASTI
New York University
New York, New York

MARCEL DEKKER, INC. New York and Basel

Library of Congress Cataloging in Publication Data

Larson, Robert Edward.
Principles of dynamic programming.

(Control and systems theory ; v. 7)
Bibliography: v. 1, p.
CONTENTS: pt. 1. Basic analytic and computational methods.
1. Dynamic programming. I. Casti, John, joint author. II. Title.
T57.83.L37 519.7'03 78-15319
ISBN 0-8247-6589-3

MARCEL DEKKER, INC.

270 Madison Avenue, New York, New York 10016

Current printing (last digit):
10 9 8 7 6 5 4 3 2 1

PRINTED IN THE UNITED STATES OF AMERICA

Principles of Dynamic Programming

CONTROL AND SYSTEMS THEORY

A Series of Monographs and Textbooks

Volume 1 Discrete Techniques of Parameter … The Equation Error Formulation, *Jerry M. M*

Volume 2 Mathematical …escription o… …ns, *Wilson J. Rugh*

Volume 3 The Qualitative Theory of … …cesses, *R. Gabasov and F. Kirillova*

Volume 4 Self-Organizing Control … …astic Systems, *George N. Saridis*

Volume 5 An Introduction to Linear …ntrol Systems, *Thomas E. Fortmann and Konrad L. Hitz*

Volume 6 Distributed Parameter Systems: Identification, Estimation, and Control, *edited by W. Harmon Ray and Demetrios G. Lainiotis*

Volume 7 Principles of Dynamic Programming. Part I. Basic Analytic and Computational Methods, *Robert E. Larson and John L. Casti*

Additional Volumes in Preparation

To RICHARD BELLMAN

Teacher, Inspiration, and Friend to us both.

PREFACE

We are living in an age of ever-increasing complexity, instability, and uncertainty. Sociological, economic, and physical pressures in all areas of modern life have generated an accelerated demand for high-level decisionmaking based upon limited information about the processes being controlled. In light of such inherent uncertainties, decisionmaking procedures must remain flexible and possess the capabilities for adapting to the changing needs of the moment. In other words, regulation of a complex system is a multistage decisionmaking process carried out within the context of an environment whose features are only partially known.

Several years ago, a systematic and concerted mathematical study of such decisionmaking situations was initiated by Richard Bellman. This pioneering work was based upon the fundamental system-theoretic notion of feedback, i.e., that decision rules should be based upon the current (and perhaps past) states of the process under study. Living in the high intellectual culture of today, this basic idea seems rather obvious; nevertheless, at the time it represented a major conceptual advance, which enabled a decision-maker to deal at one stroke with processes unfolding over time in an uncertain manner. Bellman and his colleagues continued to develop the feedback decisionmaking concept under the name of "dynamic programming" and applied it to a wide variety of problems in economics, engineering, operations research, and mathematics, itself. It is probably true to say, though, that much of this pioneering work was in advance of its time since the majority of problems of true practical concern were computationally intractable due to the limited state of the computing art at that time (circa 1960).

In recent years, a combination of rapid progress in computer technology, coupled with the development of refined computational procedures, has extended the range of problems amenable to a dynamic programming treatment to such a degree that, for the first time, truly significant problems can be attacked with confidence. Recent crises in areas such as energy, the environment, industrial productivity, and economics have forced decisionmakers to examine their options more carefully, and this examination has produced renewed interest in utilizing methodological tools capable of providing optimal policies for complex processes. Thus, the need for dynamic programming is greater now than ever before.

The foregoing considerations make it clear that we are currently at a most appropriate time for a reexamination of dynamic programming as a practical tool for solving significant problems in a variety of fields. In response to this need, the current book has been written to give an account of the basic analytic and computational aspects of dynamic programming in a form accessible to undergraduates. This book is intended as the first of two volumes. The second volume will build on the material presented here and will also deal with advanced topics and applications.

The current volume is devoted to the topic "Theory and Basic Computational Procedures". The prerequisites for the material are a semester course in ordinary differential equations and elementary linear algebra. By keeping the background material to a minimum, it is hoped that the book will be accessible to undergraduates, possibly even in the sophomore year. Numerous solved problems are also included to illustrate each key point made in the text. These problems should be useful as models for more complex situations, as well as providing deeper insight into the basic dynamic programming procedures. Each chapter ends with a set of unsolved supplementary problems which illustrate aspects of the theory not covered in the earlier sections.

The current book is divided into four chapters. The first chapter describes the basic properties of multistage decision processes. The chapter begins with a review of the concept of a differential equation, adds vector and matrix notation to the extent needed in the volume, and develops the definition of a multistage decision process in a step-by-step fashion.

The second chapter presents the basic theory of dynamic programming. In particular, the fundamental iterative equation of dynamic programming is derived from basic principles. The significance of the equation is discussed in detail.

The third chapter discusses the basic computational procedure of dynamic programming. All steps of the procedure are described in detail, and a flow chart for a computer program to implement the procedure is given. The steps are clarified through the detailed working of an illustrative example. The computational implications of the procedure are also examined.

In chapter four we treat various extensions of the basic procedures of the earlier chapters. Problems without an explicit stage variable are discussed, and procedures for their solution are developed. In addition, a number of auxiliary topics such as infinite stage processes, forward dynamic programming, and modified computational procedures are also considered.

The second volume of the series will build on this basic material to cover all aspects of dynamic programming. Volume II describes the extension of the basic methods to problems where uncertainty is present and indicates many application areas where such effects are important. Both stochastic and adaptive methods are developed. This volume also describes the numerous improvements in the basic computational procedure that have been made over the past 20 years. Other topics include the extension of dynamic programming to continuous-time systems, some fundamental analytic results in the case of linear systems with quadratic criteria, and some case studies of applications to actual large-scale, complex problems.

The authors feel that these volumes will provide a comprehensive treatment of all the major results obtained in the field of dynamic programming. It is hoped that by compiling this material and illustrating it with numerous examples, dynamic programming will be brought to the attention of numerous researchers in a variety of fields, assuring its central position in the family of problem-solving approaches.

As in all undertakings of this type, heavy debts of gratitude have been incurred through imposition on colleagues and students. Much of the material has been tested, in one form or another, in a course given over the past few years at Stanford University by one of the authors (REL). Our grateful appreciation is extended to the students in this course for acting as willing, although captive, guinea pigs. In this same connection, we thank Professor S. Yakowitz for his comments based upon feedback from a similar course given at the University of Arizona. Finally, it is our great pleasure to express gratitude to Professor Richard Bellman for his constant encouragement and help during the course of this multi-year project.

PALO ALTO, CALIFORNIA
JULY 1977

ROBERT E. LARSON
JOHN L. CASTI

TABLE OF CONTENTS

Preface v

CHAPTER 1

Systems, Processes, and Decisions

Introduction 1

Vector-Matrix Notation 1

Dynamical Systems 13

Multistage Processes 14

Multistage Decision Processes 19

Summary 27

Solved Problems 28

Supplementary Problems 41

References 42

CHAPTER 2

The Principle of Optimality and Dynamic Programming Processes

Introduction 43

Imbedding and Recurrence Equations 43

Bellman's Principle of Optimality 54

The Optimum Decision Policy 56

Summary 61

Solved Problems 62

Supplementary Problems 96

References 102

CHAPTER 3

The Basic Dynamic Programming Computational Procedure

Introduction 103
Problem Formulation 104
Optimization Via Enumeration of Admissible Control Sequences 110
The Iterative Functional Equation vs. Direct Enumeration 114
Constraints and Quantization 119
Starting Procedure 122
Calculation of Optimal Decision 122
An Illustrative Example 125
Review of the Dynamic Programming Computational Procedure and a Flow Chart for a Computer Program to Implement It 135
Recovery of an Optimal Trajectory 137
Properties of the Dynamic Programming Computational Procedure 139
Interpolation Procedures 144
Implementation of Dynamic Programming Solution . . 148
Computational Requirements 150
Summary. 154
Solved Problems 156
Supplementary Problems 194
References 209

CHAPTER 4

Extensions of the Basic Procedures

Introduction 210
Problems With an Implicit Stage Variable 211
Infinite Stage Processes 221
Forward Dynamic Programming 233
Solved Problems. 256
Supplementary Problems 285
References 292

References 293
Subject Index 329

Chapter 1

SYSTEMS, PROCESSES, AND DECISIONS

INTRODUCTION

The theory of dynamic programming may be considered the fundamental theory for the optimization of multistage decision processes. Consequently, to lay the foundations for what follows, we begin our exposition by giving a precise description of the concepts of a multi-stage decision process. Through discussion of these fundamental topics, it will be seen that dynamic programming provides a mathematical framework suitable for the consideration of far-reaching generalizations of many classical problems, including significant problems in the domains of engineering, physics, biology, economics, and operations research, as will be amply illustrated in later chapters.

VECTOR-MATRIX NOTATION

A subject of major importance in this book is the analysis of systems described by several simultaneous differential equations. In treating these systems it is convenient to make use of vector-matrix notation. Let us now briefly summarize some of the definitions and results that we shall require.

A column of n numbers

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

will be called an n-dimensional <u>vector</u>. The elements x_1, x_2,..., x_n are called the components. Two vectors x and y will be considered equal when all of their respective components are equal.

We define addition of two n-dimensional vectors as the vector formed by adding componentwise, i.e.,

$$x+y = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_n + y_n \end{bmatrix}$$

Multiplication of a vector by a scalar (real or complex) α is defined by the relation.

$$\alpha x = \begin{bmatrix} \alpha x_1 \\ \alpha x_2 \\ \vdots \\ \alpha x_n \end{bmatrix}$$

EXAMPLE

1.1 Define the vectors x and y as

$$x = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \quad , \quad y = \begin{bmatrix} 2 \\ 3 \\ 0 \end{bmatrix}$$

Compute

(a) x + y

(b) 2x

(c) 2x + 3y

(a)

$$x+y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} + \begin{bmatrix} 2 \\ 3 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ -1 \end{bmatrix}$$

(b)

$$2x = 2\begin{bmatrix}1\\0\\-1\end{bmatrix} = \begin{bmatrix}2\\0\\-2\end{bmatrix}$$

(c)

$$2x+3y = 2\begin{bmatrix}1\\0\\-1\end{bmatrix} +3\begin{bmatrix}2\\3\\0\end{bmatrix} = \begin{bmatrix}2\\0\\-2\end{bmatrix} + \begin{bmatrix}6\\9\\0\end{bmatrix} = \begin{bmatrix}8\\9\\-2\end{bmatrix}$$

The transpose of a vector, denoted by a superscript T, is simply the quantities in the column array written as a row array. Thus, if a vector real x is defined as above, then

$$x^T = [x_1, x_2, \ldots, x_n]$$

Clearly, the transpose of the transpose of a vector is the vector itself.

The scalar product (or dot product) of two vectors is defined as the sum of the products of the components of the vectors. In other words, if x and y are two vectors of dimension n, then the scalar product, denoted as (x,y), takes the form

$$(x,y) = x_1y_1 + x_2y_2 + \ldots + x_ny_n$$

$$= \sum_{i=1}^{n} x_iy_i$$

Note from the definition that (x,y) = (y,x).

EXAMPLE

1.2 Using the vectors x and y defined in Example 1.1 and defining

$$z = \begin{bmatrix}0\\0\\1\end{bmatrix},$$

compute

(a) (x,y)
(b) (y,z)

(a) $(x,y) = 1\cdot 2 + 0\cdot 3 + (-1)\cdot 0 = 2$
(b) $(y,z) = 2\cdot 0 + 3\cdot 0 + 0\cdot 1 = 0$

A rectangular array of numbers having m rows and n columns

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = \{a_{ij}\};\ i=1,2,\ldots m,\ j=1,2,\ldots n$$

is called an mxn matrix. If m=n, A is termed a square matrix. Addition of two mxn matrices is defined as

$$A + B = (a_{ij} + b_{ij});\ i=1,2,\ldots m,\ j=1,2,\ldots,n$$

while scalar multiplication is given by

$$\alpha A = (\alpha a_{ij});\ i=1,2,\ldots,m,\ j=1,2,\ldots,n$$

The transpose of a matrix, denoted by a superscript T, is the nxm array obtained by exchanging the element in the i-th row, j-th column with the element in the j-th row, i-th column. Formally, if $A = \{a_{ij}\}$, then $A^T = \{a_{ji}\}$. Note that the definition for the transpose of a vector is just a special case of this equation.

Multiplication of an n-dimensional vector x by an mxn matrix A is defined so that the set of m linear algebraic equations