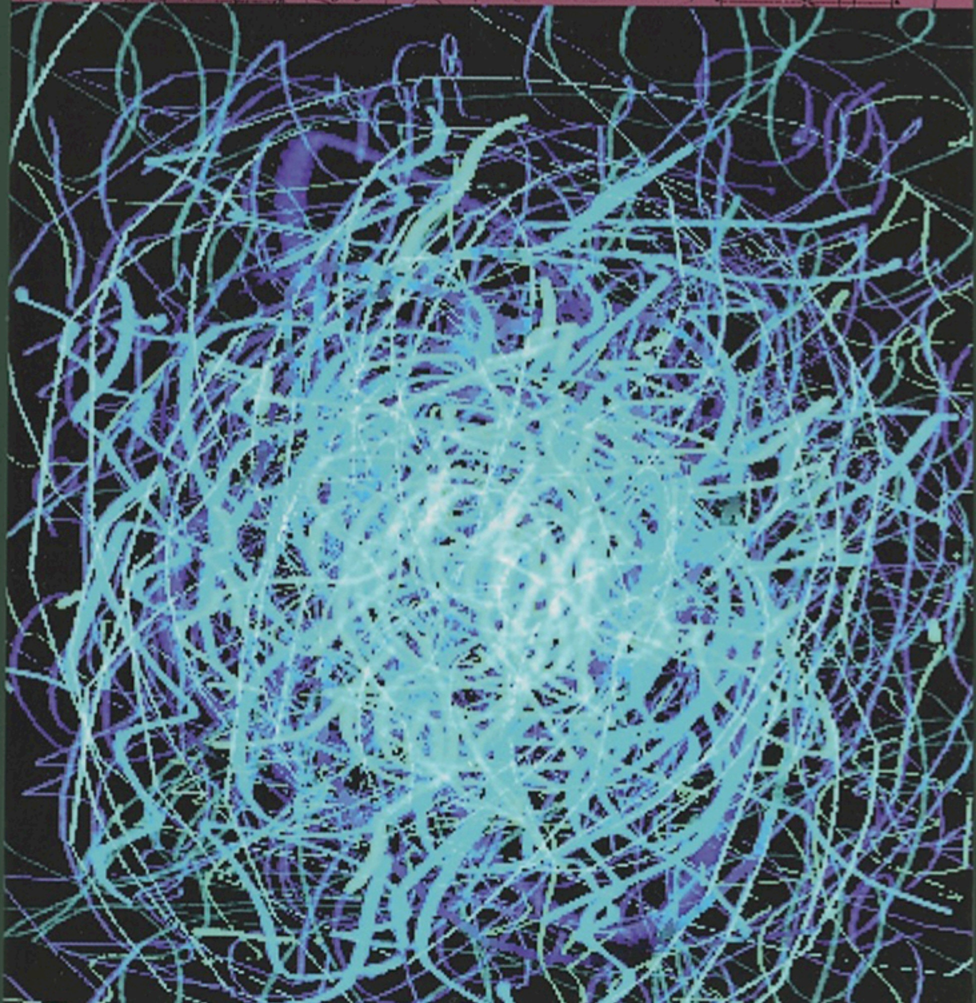
NEURO-DYNAMIC PROGRAMMING

DIMITRI P. BERTSEKAS
JOHN N. TSITSIKLIS

Athena Scientific

NEURO-DYNAMIC PROGRAMMING

DIMITRI P. BERTSEKAS
JOHN N. TSITSIKLIS



Neuro-Dynamic Programming

Dimitri P. Bertsekas and John N. Tsitsiklis

Massachusetts Institute of Technology

Athena Scientific, Belmont, Massachusetts

Athena Scientific
Belmont, Mass.
U.S.A.

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

Cover Design: *Ann Gallager*

Publisher's Cataloging-in-Publication Data

Bertsekas, Dimitri P.
Neuro-Dynamic Programming
Includes bibliographical references and index
1. Neural Networks. 2. Mathematical Optimization. 3. Dynamic Programming. I. Title.
QA76.87.B47 1996 519.703 96-085338

ISBN-10: 1-886529-10-8, ISBN-13: 978-1-886529-10-6

4th Printing (2021)

To the memory of Dimitri's parents
To John's parents

Contents

Ὦ Χαιρεφῶν, πολλαὶ τέχναι ἐν ἀνθρώποις εἰσὶν ἐκ τῶν ἐμπειριῶν ἐμπείρως ηὑρημέναι· ἐμπειρία μὲν γὰρ ποιεῖ τὸν αἰῶνα ἡμῶν πορεύεσθαι κατὰ τέχνην, ἀπειρία δὲ κατὰ τύχην.

O Chaerephon, many arts among men have been
discovered through practice, empirically;
for experience makes our life proceed deliberately,
but inexperience unpredictably.

(Plato, Gorgias 448c)

Preface

A few years ago our curiosity was aroused by reports on new methods in reinforcement learning, a field that was developed primarily within the artificial intelligence community, starting a few decades ago. These methods were aiming to provide effective suboptimal solutions to complex problems of planning and sequential decision making under uncertainty, that for a long time were thought to be intractable. Our first impression was that the new methods were ambitious, overly optimistic, and lacked firm foundation. Yet there were claims of impressive successes and indications of a solid core to the modern developments in reinforcement learning, suggesting that the correct approach to their understanding was through dynamic programming.

Three years later, after a lot of study, analysis, and experimentation, we believe that our initial impressions were largely correct. This is indeed an ambitious, often ad hoc, methodology, but for reasons that we now understand much better, it does have the potential of success with important and challenging problems. With a good deal of justification, it claims to deal effectively with the dual curses of dynamic programming and stochastic optimal control: Bellman's *curse of dimensionality* (the exponential computational explosion with the problem dimension is averted through the use of parametric approximate representations of the cost-to-go function), and the *curse of modeling* (an explicit system model is not needed, and a simulator can be used instead). Furthermore, the methodology has a logical structure and a mathematical foundation, which we systematically develop in this book. It draws on the theory of function approximation, the theory of iterative optimization and neural network training, and the theory of dynamic programming. In view of the close connection with both neural networks and dynamic programming, we settled on the name "neuro-dynamic programming" (NDP), which describes better in our opinion the nature of the subject than the older and more broadly applicable name "reinforcement learning."

Our objective in this book is to explain with mathematical analysis, examples, speculative insight, and case studies, a number of computational ideas and phenomena that collectively can provide the foundation for understanding and applying the NDP methodology. We have organized the book in three major parts.

(a) The first part consists of Chapters 2-4 and provides background. It includes a detailed introduction to dynamic programming (Chapter 2), a discussion of neural network architectures and methods for training them (Chapter 3), and the development of general convergence theorems for stochastic approximation methods (Chapter 4), which will provide the foundation for the analysis of various NDP algorithms later.

(b) The second part consists of the next three chapters and provides the core NDP methodology, including many mathematical results and methodological insights that were developed as this book was written and which are not available elsewhere. Chapter 5 covers methods involving a lookup table representation. Chapter 6 discusses the more practical methods that make use of function approximation. Chapter 7 develops various extensions of the theory in the preceding two chapters.

(c) The third part consists of Chapter 8 and discusses the practical aspects of NDP through case studies.

Inevitably, some choices had to be made regarding the material to be covered. Given that the reinforcement learning literature often involves a mixture of heuristic arguments and incomplete analysis, we decided to pay special attention to the distinction between factually correct and incorrect statements, and to rely on rigorous mathematical proofs. Because some of these proofs are long and tedious, we have made an effort to organize the material so that most proofs can be omitted without loss of continuity on the part of the reader. For example, during a first reading, a reader could omit all of the proofs in Chapters 2-5, and proceed to subsequent chapters.

However, we wish to emphasize our strong belief in the beneficial interplay between mathematical analysis and practical algorithmic insight. Indeed, it is primarily through an effort to develop a mathematical structure for the NDP methodology that we will ever be able to identify promising or solid algorithms from the bewildering array of speculative proposals and claims that can be found in the literature.

The fields of neural networks, reinforcement learning, and approximate dynamic programming have been very active in the last few years and the corresponding literature has greatly expanded. A comprehensive survey of this literature is thus beyond our scope, and we wish to apologize in advance to researchers in the field for not citing their works. We have confined ourselves to citing the sources that we have used and that contain results related to those presented in this book. We have also cited a few sources for their historical significance, but our references are far from complete in this regard.

Finally, we would like to express our thanks to a number of individuals. Andy Barto and Michael Jordan first gave us pointers to the research and the state of the art in reinforcement learning. Our understanding of the reinforcement learning literature and viewpoint gained significantly from interactions with Andy Barto, Satinder Singh, and Rich Sutton. The first author collaborated with Vivek Borkar on the average cost Q-learning research discussed in Chapter 7, and with Satinder Singh on the dynamic channel allocation research discussed in Chapter 8. The first author also benefited a lot through participation in an extensive NDP project at Alphatech, Inc., where he interacted with David Logan and Nils Sandell, Jr. Our students contributed substantially to our understanding through discussion, computational experimentation, and individual research. In particular, they assisted with some of the case studies in Chapter 8, on parking (Keith Rogers), football (Steve Patek), tetris (Sergey Ioffe and Dimitris Papaioannou), and maintenance and combinatorial optimization (Cynara Wu). The joint researches of the first author with Jinane Abounadi and with Steve Patek are summarized in Sections 7.1 and 7.2, respectively. Steve Patek also offered tireless and invaluable assistance with the experimental implementation, validation, and interpretation of a large variety of untested algorithmic ideas. The second author has enjoyed a fruitful collaboration with Ben Van Roy that led to many results, including those in Sections 6.3, 6.7, 6.8, and 6.9. We were fortunate to work at the Laboratory for Information and Decision Systems at M.I.T., which provided us with a stimulating research environment. Funding for our research that is reported in this book was provided by the National Science Foundation, the Army Research Office through the Center for Intelligent Control Systems, the Electric Power Research Institute, and Siemens. We are thankful to Prof. Charles Segal of Harvard's Department of Classics for suggesting the original quotation that appears at the beginning of this preface. Finally, we are grateful to our families for their love, encouragement, and support while this book was being written.

Dimitri P. Bertsekas
John N. Tsitsiklis
Cambridge, August 1996

Learning without thought is labour lost;
thought without learning is perilous.
(Confucian Analects)

1

Introduction

Contents

This book considers systems where decisions are made in stages. The outcome of each decision is not fully predictable but can be anticipated to some extent before the next decision is made. Each decision results in some immediate cost but also affects the context in which future decisions are to be made and therefore affects the cost incurred in future stages. We are interested in decision making policies that minimize the total cost over a number of stages. Such problems are challenging primarily because of the tradeoff between immediate and future costs. Dynamic programming (DP for short) provides a mathematical formalization of this tradeoff.

Generally, in DP formulations we have a discrete-time dynamic system whose state evolves according to given transition probabilities that depend on a decision/control u. In particular, if we are in state i and we choose control u, we move to state j with given probability $p_{ij}(u)$. The control u depends on the state i and the rule by which we select the controls is called a *policy* or *feedback control policy* (see Fig. 1.1). Simultaneously with a transition from i to j under control u, we incur a cost $g(i,u,j)$. In comparing, however, the available controls u, it is not enough to look at the magnitude of the cost $g(i,u,j)$; we must also take into account the desirability of the next state j. We thus need a way to rank or rate states j. This is done by using the optimal cost (over all remaining stages) starting from state j, which is denoted by $J^*(j)$ and is referred to as the optimal *cost-to-go* of state j. These costs-to-go can be shown to satisfy some form of *Bellman's equation*

$$J^*(i) = \min_u E\big[g(i,u,j) + J^*(j) \mid i,u\big], \qquad \text{for all } i,$$

where j is the state subsequent to i, and $E[\cdot \mid i,u]$ denotes expected value with respect to j, given i and u. Generally, at each state i, it is optimal to use a control u that attains the minimum above. Thus, controls are ranked based on the sum of the expected cost of the present period and the optimal expected cost of all subsequent periods.

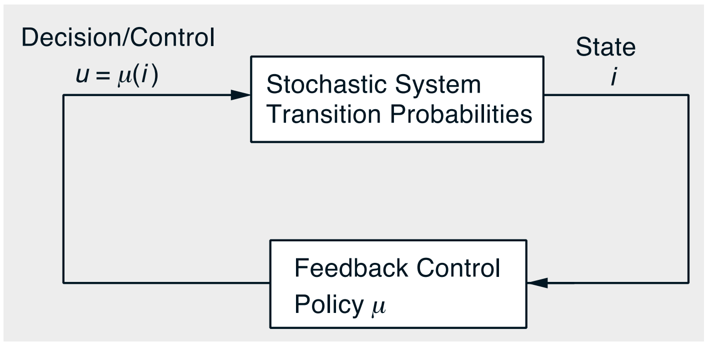


Figure 1.1: Structure of a discrete-time dynamic system under feedback control.

The objective of DP is to calculate numerically the optimal cost-to-go function J^*. This computation can be done off-line, i.e., before the real system starts operating. An optimal policy, that is, an optimal choice of u for each i, is computed either simultaneously with J^*, or in real time by minimizing in the right-hand side of Bellman's equation. It is well known, however, that for many important problems the computational requirements of DP are overwhelming, because the number of states and controls is very large (Bellman's "curse of dimensionality"). In such situations a suboptimal solution is required.

1.1 COST-TO-GO APPROXIMATIONS IN DYNAMIC PROGRAMMING

In this book, we primarily focus on suboptimal methods that center around the evaluation and approximation of the optimal cost-to-go function J^*, possibly through the use of neural networks and/or simulation. In particular, we replace the optimal cost-to-go $J^*(j)$ with a suitable approximation $\tilde{J}(j,r)$, where r is a vector of parameters, and we use at state i the (suboptimal) control $\tilde{\mu}(i)$ that attains the minimum in the (approximate) right-hand side of Bellman's equation, that is,

$$\tilde{\mu}(i) = \arg\min_u E\big[g(i,u,j) + \tilde{J}(j,r) \mid i,u\big].$$

The function $\tilde{J}$ will be called the *scoring function* or *approximate cost-to-go function*, and the value $\tilde{J}(j,r)$ will be called the *score* or *approximate cost-to-go* of state j (see Fig. 1.2). The general form of $\tilde{J}$ is known and is such that once the parameter vector r is fixed, the evaluation of $\tilde{J}(j,r)$ for any state j is fairly simple.

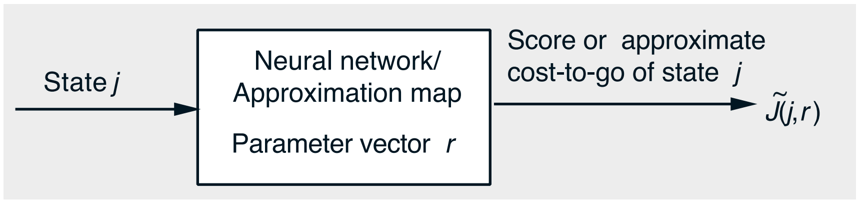


Figure 1.2: Structure of cost-to-go approximation.

We are interested in problems with a large number of states and in scoring functions $\tilde{J}$ that can be described with relatively few numbers (a vector r of small dimension). Scoring functions involving few parameters will be called *compact representations*, while the tabular description of J^* will be called the *lookup table representation*. In a lookup table representation, the values $J^*(j)$ for all states j are stored in a table. In a typical compact representation, only the vector r and the general structure of the scoring function $\tilde{J}(\cdot,r)$ are stored; the scores $\tilde{J}(j,r)$ are generated only when needed. For example, if $\tilde{J}(j,r)$ is the output of some neural network in response to the input j, then r is the associated vector of weights or parameters of the neural network; or if $\tilde{J}(j,r)$ involves a lower dimensional description of the state j in terms of its "significant features," then r could be a vector of relative weights of the features. Naturally, we would like to choose r algorithmically so that $\tilde{J}(\cdot,r)$ approximates well $J^*(\cdot)$. Thus, determining the scoring function $\tilde{J}(j,r)$ involves two complementary issues: (1) deciding on the general structure of the function $\tilde{J}(j,r)$, and (2) calculating the parameter vector r so as to minimize in some sense the error between the functions $J^*(\cdot)$ and $\tilde{J}(\cdot,r)$.

We note that in some problems the evaluation of the expression

$$E\big[g(i,u,j) + \tilde{J}(j,r) \mid i,u\big],$$

for each u, may be too complicated or too time-consuming for making decisions in real-time, even if the scores $\tilde{J}(j,r)$ are simply calculated. There are a number of ways to deal with this difficulty (see Section 6.1). An important possibility is to approximate the expression minimized in Bellman's equation,

$$Q^*(i,u) = E\big[g(i,u,j) + J^*(j) \mid i,u\big],$$

which is known as the *Q-factor corresponding to* (i,u). In particular, we can replace $Q^*(i,u)$ with a suitable approximation $\tilde{Q}(i,u,r)$, where r is a vector of parameters. We can then use at state i the (suboptimal) control that minimizes the approximate Q-factor corresponding to i:

$$\tilde{\mu}(i) = \arg\min_u \tilde{Q}(i,u,r).$$

Much of what will be said about the approximation of the optimal costs-to-go also applies to the approximation of Q-factors. In fact, we will see later that the Q-factors can be viewed as optimal costs-to-go of a related problem. We thus focus primarily on approximation of the optimal costs-to-go.

Approximations of the optimal costs-to-go have been used in the past in a variety of DP contexts. Chess playing programs represent an interesting example. A key idea in these programs is to use a *position evaluator* to rank different chess positions and to select at each turn a move that results in the position with the best rank. The position evaluator assigns a numerical value to each position according to a heuristic formula that includes weights for the various features of the position (material balance, piece mobility, king safety, and other factors); see Fig. 1.3. Thus, the position evaluator corresponds to the scoring function $\tilde{J}(j,r)$ above, while the weights of the features correspond to the parameter vector r. Usually, some general structure is selected for the position evaluator (this is largely an art that has evolved over many years, based on experimentation and human knowledge about chess), and the numerical weights are chosen by trial and error or (as in the case of the champion program Deep Thought) by "training" using a large number of sample grandmaster games. It should be mentioned that in addition to the use of sophisticated position evaluators, much of the success of chess programs can be attributed to the use of multimove lookahead, which has become deeper and more effective with the use of increasingly fast hardware.

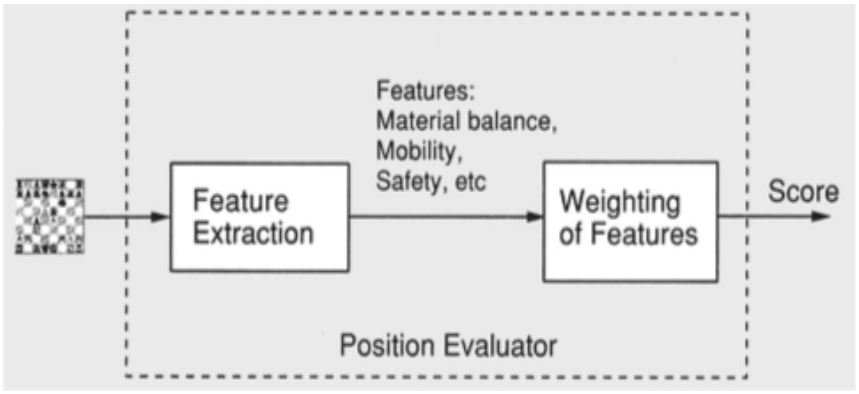


Figure 1.3: Structure of the position evaluator of a chess program.

As the chess program paradigm suggests, intuition about the problem, heuristics, and trial and error are all important ingredients for constructing cost-to-go approximations in DP. However, it is important to supplement heuristics and intuition with more systematic techniques that are broadly applicable and retain as much as possible of the nonheuristic characteristics of DP. This book will focus on several recent efforts to develop a methodological foundation for a rational approach to complex stochastic decision problems, which combines dynamic programming, function approximation, and simulation.

1.2 APPROXIMATION ARCHITECTURES

An important issue in function approximation is the *selection of an architecture*, that is, the choice of a parametric class of functions $\tilde{J}(\cdot,r)$ or

$\tilde{Q}(\cdot,\cdot,r)$ that suits the problem at hand. One possibility is to use a neural network architecture of some type. We should emphasize here that in this book we use the term "neural network" in a very broad sense, essentially as a synonym to "approximating architecture." In particular, we do not restrict ourselves to the classical multilayer perceptron structure with sigmoidal nonlinearities. Any type of universal approximator of nonlinear mappings could be used in our context. The nature of the approximating structure is left open in our discussion, and it could involve, for example, radial basis functions, wavelets, polynomials, splines, aggregation, etc.

Cost-to-go approximation can often be significantly enhanced through the use of *feature extraction*, a process that maps the state i into some vector $f(i)$, called the *feature vector* associated with i. Feature vectors summarize, in a heuristic sense, what are considered to be important characteristics of the state, and they are very useful in incorporating the designer's prior knowledge or intuition about the problem and about the structure of the optimal controller. For example, in a queueing system involving several queues, a feature vector may involve for each queue a three-valued indicator that specifies whether the queue is "nearly empty," "moderately busy," or "nearly full." In many cases, analysis can complement intuition to suggest the right features for the problem at hand.

Feature vectors are particularly useful when they can capture the "dominant nonlinearities" in the optimal cost-to-go function J^*. By this we mean that $J^*(i)$ can be approximated well by a "relatively smooth" function $\tilde{J}\big(f(i)\big)$; this happens for example, if through a change of variables from states to features, J^* becomes a (nearly) linear or low-order polynomial function of the features. When a feature vector can be chosen to have this property, it is appropriate to use approximation architectures where features and (relatively simple) neural networks are used together. In particular, the state is mapped to a feature vector, which is then used as input to a neural network that produces the score of the state (see Fig. 1.4). More generally, it is possible that both the state and the feature vector are provided as inputs to the neural network (see the second diagram in Fig. 1.4).

1.3 SIMULATION AND TRAINING

Some of the most successful applications of neural networks are in the areas of pattern recognition, nonlinear regression, and nonlinear system identification. In these applications the neural network is used as a universal approximator: the input-output mapping of the neural network is matched to an unknown nonlinear mapping F of interest using a least-squares optimization, known as *training the network*. To perform training, one must

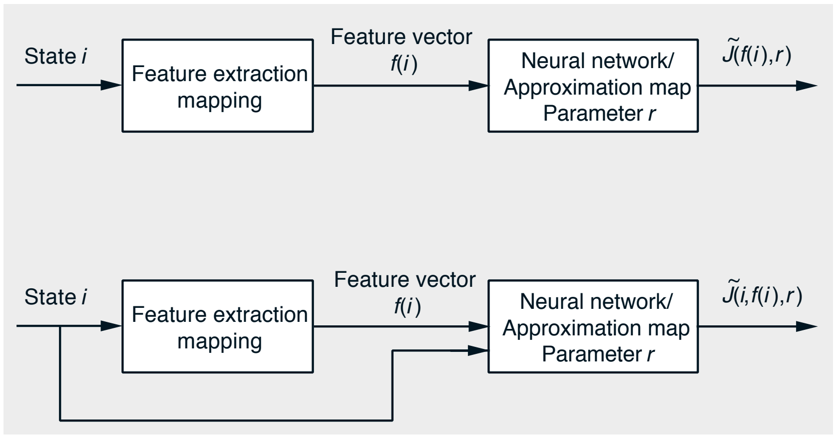

State i
Feature extraction mapping
Feature vector $f(i)$
Neural network/ Approximation map Parameter r
$\tilde{J}(f(i),r)$
State i
Feature extraction mapping
Feature vector $f(i)$
Neural network/ Approximation map Parameter r
$\tilde{J}(i,f(i),r)$

Figure 1.4: Approximation architectures involving feature extraction and neural networks.

have some training data, that is, a set of pairs $\big(i, F(i)\big)$, which is representative of the mapping F that is approximated.

It is important to note that in contrast with these neural network applications, in the DP context there is no readily available training set of input-output pairs $\big(i, J^*(i)\big)$ that could be used to approximate J^* with a least squares fit. The only possibility is to evaluate (exactly or approximately) by simulation the cost-to-go functions of given (suboptimal) policies, and to try to iteratively improve these policies based on the simulation outcomes. This creates analytical and computational difficulties that do not arise in classical neural network training contexts. Indeed the use of simulation to evaluate approximately the optimal cost-to-go function is a key new idea that distinguishes the methodology of this book from earlier approximation methods in DP.

Simulation offers another major advantage: it allows the methods of this book to be used for systems that are hard to model but easy to simulate, i.e., problems where a convenient explicit model is not available, and the system can only be observed, either through a software simulator or as it operates in real time. For such problems, the traditional DP techniques are inapplicable, and estimation of the transition probabilities to construct a detailed mathematical model is often cumbersome or impossible.

There is a third potential advantage of simulation: it can implicitly identify the "most important" or "most representative" states of the system. It appears plausible that these states are the ones most often visited during the simulation, and for this reason the scoring function will tend to approximate better the optimal cost-to-go for these states, and the suboptimal policy obtained will on the average perform better.

1.4 NEURO-DYNAMIC PROGRAMMING

In view of the reliance on both DP and neural network concepts, we use the name *neuro-dynamic programming* (NDP for short) to describe collectively the methods of this book. In the artificial intelligence community, where the methods originated, the name *reinforcement learning* is also used. In common artificial intelligence terms, the methods of this book allow systems to "learn how to make good decisions by observing their own behavior, and use built-in mechanisms for improving their actions through a reinforcement mechanism." In the less anthropomorphic DP terms used in this book, "observing their own behavior" relates to simulation, and "improving their actions through a reinforcement mechanism" relates to iterative schemes for improving the quality of approximation of the optimal costs-to-go, or the Q-factors, or the optimal policy. There has been a gradual realization that reinforcement learning techniques can be fruitfully motivated and interpreted in terms of classical DP concepts such as value and policy iteration.

In this book, we attempt to clarify some aspects of the current NDP methodology, we suggest some new algorithmic approaches, and we identify some open questions. Despite the great interest in NDP, the theory of the subject is only now beginning to take shape, and the corresponding literature is often confusing. Yet, there have been many reports of successes with problems too large and complex to be treated in any other way. A particularly impressive success that greatly motivated subsequent research, was the development of a backgammon playing program by Tesauro [Tes92] (see Section 8.6). Here a neural network was trained to approximate the optimal cost-to-go function of the game of backgammon by using simulation, that is, by letting the program play against itself. After training for several months, the program nearly defeated the human world champion. Unlike chess programs, this program did not use lookahead of many stages, so its success can be attributed primarily to the use of a properly trained approximation of the optimal cost-to-go function.

Our own experience has been that NDP methods can be impressively effective in problems where traditional DP methods would be hardly applicable and other heuristic methods would have limited potential. In this book, we outline some engineering applications, and we use a few computational studies for illustrating the methodology and some of the art that is often essential for success.

We note, however, that the practical application of NDP is computationally very intensive, and often requires a considerable amount of trial and error. Furthermore, success is often obtained using methods whose properties are not fully understood. Fortunately, all of the computation and experimentation with different approaches can be done off-line. Once the approximation is obtained off-line, it can be used to generate decisions fast enough for use in real time. In this context, we mention that in the artificial intelligence literature, reinforcement learning is often viewed as an "on-line" method, whereby the cost-to-go approximation is improved as the system operates in real time. This is reminiscent of the methods of traditional adaptive control. We will not discuss this viewpoint in this book, as we prefer to focus on applications involving a large and complex system. A lot of training data are required for such systems. These data often cannot be obtained in sufficient volume as the system is operating; even if they can, the corresponding processing requirements are often too large for effective use in real time.

We finally mention an alternative approach to NDP, known as *approximation in policy space*, which, however, we will not consider in this book. In this approach, in place of an overall optimal policy, we look for an optimal policy within some restricted class that is parametrized by some vector s of relatively low dimension. In particular, we consider policies of a given form $\tilde{\mu}(i, s)$. We then minimize over s the expected cost $E_i\big[J^{\tilde{\mu}(\cdot,s)}(i)\big]$, where the expectation is with respect to some suitable probability distribution of the initial state i. This approach applies to complex problems where there is no explicit model for the system and the cost, as long as the cost corresponding to a given policy can be calculated by simulation. Furthermore, insight and analysis can sometimes be used to select simple and effective parametrizations of the policies. On the other hand, there are many problems where such parametrizations are not easily obtained. Furthermore, the minimization of $E_i\big[J^{\tilde{\mu}(\cdot,s)}(i)\big]$ can be very difficult because the gradient of the cost with respect to s may not be easily calculated; while methods that require cost values (and not gradients) may be used, they tend to require many cost function evaluations and to be slow in practice.

The general organizational plan of the book is to first develop some essential background material on DP, and on deterministic and stochastic iterative optimization algorithms (Chs. 2-4), and then to develop the main algorithmic methods of NDP in Chs. 5 and 6. Various extensions of the methodology are discussed in Ch. 7. Finally, we present case studies in Ch. 8. Many of the ideas of the book extend naturally to continuous-state systems, although the NDP theory is far from complete for such systems. To keep the exposition simple, we have restricted ourselves to the case where the number of states is finite and the number of available controls at each state is also finite. This is consistent with the computational orientation of the book.

1.5 NOTES AND SOURCES

1.1. The origins of our subject can be traced to the early works on DP by Bellman, who used the term "approximation in value space," and to the works by Shannon [Sha50] on computer chess and by Samuel [Sam59], [Sam67] on computer checkers.

1.4. The works by Barto, Sutton, and Anderson [BSA83] on adaptive critic systems, by Sutton [Sut88] on temporal difference methods, and by Watkins [Wat89] on Q-learning initiated the modern developments which brought together the ideas of function approximation, simulation, and DP. The work of Tesauro [Tes92], [Tes94], [Tes95] on backgammon was the first to demonstrate impressive success on a very complex and challenging problem. Much research followed these seminal works. The extensive survey by Barto, Bradtke, and Singh [BBS95], and the overviews by Werbös [Wer92a], [Wer92b], and other papers in the edited volume by White and Sofge [WhS92] point out the connections between the artificial intelligence/reinforcement learning viewpoint and the control theory/DP viewpoint, and give many references. The DP textbook by Bertsekas [Ber95a] describes a broad variety of suboptimal control methods, including some of the NDP approaches that are treated in much greater depth in the present book.

References

[ABB96] Abounadi, J., Bertsekas, D. P., and Borkar, V. S., 1996. "ODE Analysis for Q-Learning Algorithms," Lab. for Info. and Decision Systems Draft Report, Massachusetts Institute of Technology, Cambridge, MA.

[AdG86] Adams, M., and Guillemin, V., 1986. Measure Theory and Probability, Wadsworth and Brooks, Monterey, CA.

[Ash70] Ash, R. B., 1970. Basic Probability Theory, Wiley, N. Y.

[Ash72] Ash, R. B., 1972. Real Analysis and Probability, Wiley, N. Y.

[AtF66] Athans, M., and Falb, P., 1966. Optimal Control, Mc-Graw Hill, N. Y.

[BBS95] Barto, A. G., Bradtke, S. J., and Singh, S. P., 1995. "Learning to Act Using Real-Time Dynamic Programming," Artificial Intelligence, Vol. 72, pp. 81-138.

[BKK73] Bellman, R., Kalaba, R., and Kotkin, B., 1973. "Polynomial Approximation – A New Computational Technique in Dynamic Programming: Allocation Processes," Mathematical Computation, Vol. 17, pp. 155-161.

[BMP90] Benveniste, A., Metivier, M., and Priouret, P., 1990. Adaptive Algorithms and Stochastic Approximations, Springer-Verlag, N. Y.

[BSA83] Barto, A. G., Sutton, R. S., and Anderson, C. W., 1983. "Neuronlike Elements that Can Solve Difficult Learning Control Problems," IEEE Trans. on Systems, Man, and Cybernetics, Vol. 13, pp. 835-846.

[BSS93] Bazaraa, M. S., Sherali, H. D., and Shetty, C. M., 1993. Nonlinear Programming Theory and Algorithms (2nd Ed.), Wiley, N. Y.

[Bai93] Baird, L. C., 1993. "Advantage Updating," Report WL-TR-93-1146, Wright Patterson AFB, OH.

[Bai95] Baird, L. C., 1995. "Residual Algorithms: Reinforcement Learning with Function Approximation," in Machine Learning: Proceedings of the Twelfth International Conference, Morgan Kaufmann, San Francisco, CA.

[Bar93] Barnard, E., 1993. "Temporal Difference Methods and Markov Models," IEEE Trans. on Systems, Man, and Cybernetics, Vol. 23, pp. 357-365.

[BeD59] Bellman, R. E., and Dreyfus, S. E., 1959. "Functional Approximations and Dynamic Programming," Mathematical Tables and Other Aids to Computation, Vol. 13, pp. 247-251.

[BeI96] Bertsekas, D. P., and Ioffe, S., 1996. "Temporal Differences-Based Policy Iteration and Applications in Neuro-Dynamic Programming," Lab. for Info. and Decision Systems Report LIDS-P-2349, Massachusetts Institute of Technology, Cambridge, MA.

[BeT89] Bertsekas, D. P., and Tsitsiklis, J. N., 1989. Parallel and Distributed Computation: Numerical Methods, Prentice-Hall, Englewood Cliffs, N. J.

[BeT91a] Bertsekas, D. P., and Tsitsiklis, J. N., 1991. "An Analysis of Stochastic Shortest Path Problems," Mathematics of Operations Research, Vol. 16, pp. 580-595.

[BeT91b] Bertsekas, D. P., and Tsitsiklis, J. N., 1991. "Some Aspects of Parallel and Distributed Iterative Algorithms – A Survey," Automatica, Vol. 27, pp. 3-21.

[Ber82a] Bertsekas, D. P., 1982. "Distributed Dynamic Programming," IEEE Trans. on Automatic Control, Vol. AC-27, pp. 610-616.

[Ber82b] Bertsekas, D. P., 1982. Constrained Optimization and Lagrange Multiplier Methods, Academic Press, N. Y.

[Ber95a] Bertsekas, D. P., 1995. Dynamic Programming and Optimal Control, Vols. I and II, Athena Scientific, Belmont, MA.

[Ber95b] Bertsekas, D. P., 1995. Nonlinear Programming, Athena Scientific, Belmont, MA.

[Ber95c] Bertsekas, D. P., 1995. "A Counterexample to Temporal Differences Learning," Neural Computation, Vol. 7, pp. 270-279.

[Ber95d] Bertsekas, D. P., 1995. "Incremental Least Squares Methods and the Extended Kalman Filter," Lab. for Info. and Decision Systems Report LIDS-P-2237, Massachusetts Institute of Technology, Cambridge, MA; to appear in SIAM J. on Optimization.

[Ber95e] Bertsekas, D. P., 1995. "A Hybrid Incremental Gradient Method for Least Squares Problems," Lab. for Info. and Decision Systems Report LIDS-P-2257, Massachusetts Institute of Technology, Cambridge, MA; to appear in SIAM J. on Optimization.

[Ber95f] Bertsekas, D. P., 1995. "A New Value Iteration Method for the Average Cost Dynamic Programming Problem," Lab. for Info. and Decision Systems Report LIDS-P-2307, Massachusetts Institute of Technology, Cambridge, MA; to appear in SIAM J. on Control and Optimization.

[Bis95] Bishop, C. M, 1995. Neural Networks for Pattern Recognition, Oxford University Press, N. Y.

[BoM95] Boyan, J. A., and Moore, A. W., 1995. "Generalization in Reinforcement Learning: Safely Approximating the Value Function," in Advances in Neural Information Processing Systems 7, MIT Press, Cambridge, MA.

[BoS93] Borkar, V. S., and Soumyanath, K., 1993. "A New Parallel Scheme for Fixed Point Computation Part I: Theory," unpublished report.

[Bor95] Borkar, V. S., 1995. "Asynchronous Stochastic Approximations," unpublished report.

[BrB96] Bradtke, S. J., and Barto, A. G., 1996. "Linear Least-Squares Algorithms for Temporal Difference Learning," Machine Learning, Vol. 22, pp. 33-57.

[BrH75] Bryson, A. E., and Ho, Y.-C., 1975. Applied Optimal Control, Hemisphere Publishing Corp., Washington, D.C.

[Bra94] Bradtke, S. J., 1994. Incremental Dynamic Programming for On-Line Adaptive Optimal Control, Ph. D. thesis, University of Massachusetts, Amherst, MA.

[CGM95] Cybenko, G., Gray, R., and Moizumi, K., 1995. "Q-Learning: A Tutorial and Extensions," unpublished report, presented at Mathematics of Artificial Neural Networks, Oxford University, England, July 1995.

[ChK86] Christensen, J., and Korf, R. E., 1986. "A Unified Theory of Heuristic Evaluation Functions and its Application to Learning," in Proceedings AAAI-86, pp. 148-152.

[ChR92] Chong, E. K. P., and Ramadge, P. J., 1992. "Convergence of Recursive Optimization Algorithms Using Infinitesimal Perturbation Analysis Estimates," Discrete Event Dynamic Systems: Theory and Applications, Vol. 1, pp. 339-372.

[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. 36, pp. 898-914.

[ChT94] Cheng, B., and Titterington D. M., 1994. "Neural Networks: A Review from a Statistical Perspective," Statistical Science, Vol. 9, pp. 2-54.

[Chu60] Chung, K. L., 1960. Markov Chains with Stationary Transition Probabilities, Springer-Verlag, Berlin and N. Y.

[CrB96] Crites, R. H., and Barto, A. G., 1996. "Improving Elevator Performance using Reinforcement Learning," in Advances in Neural Information

Processing Systems 8, MIT Press, Cambridge, MA, pp. 1017-1023.

[Cyb89] Cybenko, 1989. "Approximation by Superpositions of a Sigmoidal Function," Math. of Control, Signals, and Systems, Vol. 2, pp. 303-314.

[DaS94] Dayan, P., and Sejnowski, T. J., 1994. "TD(λ) Converges with Probability 1," Machine Learning, Vol. 14, pp. 295-301.

[DaS96] Dayan, P., and Singh, S., 1996. "Improving Policies without Measuring Merits," in Advances in Neural Information Processing Systems 8, MIT Press, Cambridge, MA, pp. 1059-1065.

[Dan76] Daniel, J. W., 1976. "Splines and Efficiency in Dynamic Programming," Journal of Mathematical Analysis and Applications, Vol. 54, pp. 402-407.

[Dav76] Davidon, W. C., 1976. "New Least Squares Algorithms," J. Optimization Theory and Applications, Vol. 18, pp. 187-197.

[Day92] Dayan, P., 1992. "The Convergence of TD(λ) for General λ," Machine Learning, Vol. 8, pp. 341-362.

[Fel68] Feller, W., 1968. An Introduction to Probability Theory and Its Applications, Wiley, N. Y.

[FiT91] Filar, J. A., and Tolwinski, B., 1991. "On the Algorithm of Pollatschek and Avi-Itzhak," in Raghavan, T. et al. (eds.), Stochastic Games and Related Topics, In Honor of Professor L. S. Shapley, Kluwer Academic Publishers, Dordrecht, The Netherlands.

[FlR75] Fleming, W. H., and Rishel, R. W., 1975. Deterministic and Stochastic Optimal Control, Springer-Verlag, N. Y.

[Fun89] Funahashi, K., 1989. " On the Approximate Realization of Continuous Mappings by Neural Networks," Neural Networks, Vol. 2, pp. 183-192.

[GLH94] Gurvits, L., Lin, L.-J., and Hanson, S. J., 1994. "Incremental Learning of Evaluation Functions for Absorbing Markov Chains: New Methods and Theorems," unpublished report.

[Gai94] Gaivoronski, A. A., 1994. "Convergence Analysis of Parallel Backpropagation Algorithm for Neural Networks," Optimization Methods and Software, Vol. 4, pp. 117-134.

[Gla91] Glasserman, P., 1991. Gradient Estimation via Perturbation Analysis, Kluwer Academic Publishers, Norwell, MA.

[Gor95] Gordon, G. J., 1995. "Stable Function Approximation in Dynamic Programming," in Machine Learning: Proceedings of the Twelfth International Conference, Morgan Kaufmann, San Francisco, CA.

[Gri94] Grippo, L., 1994. "A Class of Unconstrained Minimization Methods for Neural Network Training," Optimization Methods and Software, Vol.

4, pp. 135-150.

[HBK94] Harmon, M. E., Baird, L. C., and Klopf, A. H., 1994. "Advantage Updating Applied to a Differential Game," unpublished report, presented at the 1994 Neural Information Processing Systems Conference, Denver, CO.

[HSW89] Hornick, K., Stinchcombe, M., and White, H., 1989. "Multilayer Feedforward Networks are Universal Approximators," Neural Networks, Vol. 2, pp. 359-159.

[Hag88] Hager, W. W., 1988. Applied Numerical Linear Algebra, Prentice-Hall, Englewood Cliffs, N. J.

[Hay94] Haykin, S., 1994. Neural Networks: A Comprehensive Foundation, McMillan, N. Y.

[Hes66] Hestenes, M. R., 1966. Calculus of Variations and Optimal Control Theory, Wiley, N. Y.

[HoC91] Ho, Y.-C., and Cao, X.-R., 1991. Perturbation Analysis of Discrete Event Dynamic Systems, Kluwer Academic Publishers, Norwell, MA.

[HoK61] Hoffman, K., and Kunze, R., 1961. Linear Algebra, Prentice-Hall, Englewood Cliffs, N. J.

[Hol86] Holland, J. H., 1986. "Escaping Brittleness: the Possibility of General-Purpose Learning Algorithms Applied to Rule-Based Systems," in Machine Learning: An Artificial Intelligence Approach, Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., (eds.), Morgan Kaufmann, San Mateo, CA, pp. 593-623.

[JJS94] Jaakkola, T., Jordan, M. I., and Singh, S. P., 1994. "On the Convergence of Stochastic Iterative Dynamic Programming Algorithms," Neural Computation, Vol. 6, pp. 1185-1201.

[JaF92] Jalali, A., and Ferguson, M. J., 1992. "On Distributed Dynamic Programming," IEEE Trans. on Automatic Control, Vol. AC-37, pp. 685-689.

[JaM70] Jacobson, D. H., and Mayne, D. Q., 1970. Differential Dynamic Programming, Elsevier, N. Y.

[Jon90] Jones, L. K., 1990. "Constructive Approximations for Neural Networks by Sigmoidal Functions," Proceedings of the IEEE, Vol. 78, pp. 1586-1589.

[KeS60] Kemeny, J. G., and Snell, J. L., 1960. Finite Markov Chains, Van Nostrand-Reinhold, N. Y.

[Koh74] Kohonen, T., 1974. "An Adaptive Associative Memory Principle," IEEE Trans. on Computers, Vol. C-23, pp. 444-445.

[Kor90] Korf, R. E., 1990. “Real-Time Heuristic Search,” Artificial Intelligence, Vol. 42, pp. 189-211.

[KuC78] Kushner, H. J., and Clark, D. S., 1978. Stochastic Approximation Methods for Constrained and Unconstrained Systems, Springer-Verlag, Berlin.

[KuD92] Kushner, H. J., and Dupuis, P., 1992. Numerical Methods for Stochastic Control in Continuous Time, Springer-Verlag, N. Y.

[KuS81] Kumar, P. R, and Shiau, T. H., 1981. “Zero-Sum Dynamic Games,” in C. T. Leondes, (ed.), Academic Press, N. Y., pp. 1345-1378.

[KuY93] Kushner, H. J., and Yang, J., “Stochastic Approximation with Averaging: Optimal Asymptotic Rates of Convergence for General Processes,” SIAM J. Control and Optimization, Vol. 31, pp. 1045-1062.

[LBH96] Logan, D. A., Bertsekas, D. P., Homer, M. L., Looze, D. P., Patek, S. D., Pepyne, D., Sandell, N. R., 1996. “Application of Neural Networks to Command Centers,” Report TR-757, Alphatech, Inc., Burlington, MA.

[LeC85] Le Cun, Y., 1985. “Une Procédure d’ Apprentissage pour Réseau à Seuil Assymétrique,” in Cognitiva 85, à la Frontière de l’Intelligence Artificielle des Sciences de la Connaissance des Neurosciences, CESTA, Paris, pp. 599-604.

[Lit96] Littman, M. L., 1996. “Algorithms for Sequential Decision Making,” Ph. D. thesis, Brown University, Providence, R. I.

[Lju77] Ljung, L., 1977. “Analysis of Recursive Stochastic Algorithms,” IEEE Trans. on Automatic Control, Vol. 22, pp. 551-575.

[Lju79] Ljung, L., 1979. “Asymptotic Behavior of the Extended Kalman Filter as a Parameter Estimator for Linear Systems,” IEEE Trans. on Automatic Control, Vol. 24, pp. 36-50.

[Lju94] Ljung, L., 1994. “Aspects on Accelerated Convergence in Stochastic Approximation Schemes,” Proceedings of the 33d IEEE Conference on Decision and Control, Lake Buena Vista, FL.

[LuT94] Luo, Z. Q., and Tseng, P., 1994. “Analysis of an Approximate Gradient Projection Method with Applications to the Backpropagation Algorithm,” Optimization Methods and Software, Vol. 4, pp. 85-101.

[Lue69] Luenberger, D. G., 1969. Optimization by Vector Space Methods, Wiley, N. Y.

[Lue84] Luenberger, D. G., 1984. Introduction to Linear and Nonlinear Programming, (2nd Ed.), Addison-Wesley, Reading, MA.

[Luo91] Luo, Z. Q., 1991. “On the Convergence of the LMS Algorithm with Adaptive Learning Rate for Linear Feedforward Networks,” Neural Computation, Vol. 3, pp. 226-245.

[MaS94] Mangasarian, O. L., and Solodov, M. V., 1994. "Serial and Parallel Backpropagation Convergence Via Nonmonotone Perturbed Minimization," Optimization Methods and Software, Vol. 4, pp. 103-116.

[Mah96] Mahadevan, S., 1996. "Average Reward Reinforcement Learning: Foundations, Algorithms, and Empirical Results," Machine Learning, Vol. 22, pp. 1-38.

[Nev75] Neveu, J., 1975. Discrete Parameter Martingales, North-Holland, Amsterdam.

[Odo69] Odoni, A. R., 1969. "On Finding the Maximal Gain for Markov Decision Processes," Operations Research, Vol. 17, pp. 857-860.

[PLC96] Pepyne, D. L., Looze, D. P., Cassandras, C. G., and Djaferis, T. E., 1996. "Application of Q-Learning to Elevator Dispatching," unpublished report.

[PaB96a] Patek, S., and Bertsekas, D. P., 1996. "Stochastic Shortest Path Games," Lab. for Info. and Decision Systems Report LIDS-P-2319, Massachusetts Institute of Technology, Cambridge, MA.

[PaB96b] Patek, S., and Bertsekas, D. P., 1996. "Play Selection in Football: a Case Study in Neuro-Dynamic Programming," Lab. for Info. and Decision Systems Report LIDS-P-2350, Massachusetts Institute of Technology, Cambridge, MA.

[Pap65] Papoulis, A., 1965. Probability, Random Variables and Stochastic Processes, McGraw-Hill, N. Y.

[PeW94] Peng, J., and Williams, R. J., 1994. "Incremental Multi-Step Q-Learning," Proceedings of the Eleventh International Conference on Machine Learning, Morgan Kaufmann, San Franscisco, CA, pp. 226-232.

[PoA69] Pollatschek, M., and Avi-Itzhak, B., 1969. "Algorithms for Stochastic Games with Geometrical Interpretation," Management Science, Vol. 15, pp. 399-415.

[PoJ92] Polyak, B. T., and Juditsky, A. B., 1992. "Acceleration of Stochastic Approximation by Averaging," SIAM J. Control and Optimization, Vol. 30, pp. 838-855.

[PoT73] Poljak, B. T., and Tsypkin, Y. Z., 1973. "Pseudogradient Adaptation and Training Algorithms," Automation and Remote Control, Vol. 12, pp. 83-94.

[Pol64] Poljak, B. T., 1964. "Some Methods of Speeding up the Convergence of Iteration Methods," Z. VyČisl. Mat. i Mat. Fiz., Vol. 4, pp. 1-17.

[Pol87] Poljak, B. T., 1987. Introduction to Optimization, Optimization Software Inc., N. Y.

[PuS78] Puterman, M. L., and Shin, M. C., 1978. "Modified Policy Iteration Algorithms for Discounted Markov Decision Problems," Management Science, Vol. 24, pp. 1127-1137.

[Put94] Puterman, M. L., 1994. Markovian Decision Problems, Wiley, N. Y.

[RHW86] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., 1986. "Learning Representations by Back-Propagating Errors," Nature, Vol. 323, pp. 533-536.

[RaF91] Raghavan, T. E. S., and Filar, J. A., 1991. "Algorithms for Stochastic Games – A Survey," ZOR – Methods and Models of Operations Research, Vol. 35, pp. 437-472.

[Roc70] Rockafellar, R. T., 1970. Convex Analysis, Princeton University Press, Princeton, N. J.

[Ros83] Ross, S. M., 1983. Introduction to Stochastic Dynamic Programming, Academic Press, N. Y.

[Ros85] Ross, S. M., 1985. Probability Models, Academic Press, Orlando, FL.

[Roy68] Royden, H. L., 1968. Principles of Mathematical Analysis, McGraw-Hill, N. Y.

[Rud64] Rudin, W., 1964. Real Analysis, McGraw-Hill, N. Y.

[Rus96] Rust, J., 1996. "Numerical Dynamic Programming in Economics," in Amman, H., Kendrick, D., and Rust, J., (eds.), Handbook of Computational Economics, Elsevier, Amsterdam, Chapter 14, pp. 614-722.

[SBC93] Saarinen, S., Bramley, R., and Cybenko, G., 1993. "Ill-Conditioning in Neural Network Training Problems," SIAM J. Scientific Computation, Vol. 14, pp. 693-714.

[SJJ94] Singh, S. P., Jaakkola, T., and Jordan, M. I., 1994. "Learning without State-Estimation in Partially Observable Markovian Decision Processes," Proceedings of the Eleventh Machine Learning Conference, pp. 284-292.

[SJJ95] Singh, S. P., Jaakkola, T., and Jordan, M. I., 1995. "Reinforcement Learning with Soft State Aggregation," in Advances in Neural Information Processing Systems 7, MIT Press, Cambridge, MA.

[Sam59] Samuel, A. L., 1959. "Some Studies in Machine Learning Using the Game of Checkers," IBM Journal of Research and Development, pp. 210-229.

[Sam67] Samuel, A. L., 1967. "Some Studies in Machine Learning Using the Game of Checkers. II – Recent Progress," IBM Journal of Research and Development, pp. 601-617.

[ScS85] Schweitzer, P. J., and Seidman, A., 1985. "Generalized Polynomial Approximations in Markovian Decision Processes," Journal of Mathematical Analysis and Applications, Vol. 110, pp. 568-582.

[ScW96] Schapire, R. E., and Warmuth, M. K., 1996. "On the Worst-Case Analysis of Temporal-Difference Learning Algorithms," Machine Learning, Vol. 22.

[Sch93] Schwartz, A., 1993. "A Reinforcement Learning Method for Maximizing Undiscounted Rewards," Proceedings of the Tenth Machine Learning Conference, pp. 298-305.

[Sha50] Shannon, C., 1950. "Programming a Digital Computer for Playing Chess," Phil. Mag., Vol. 41, pp. 356-375.

[Sha53] Shapley, L. S., 1953. "Stochastic Games," Proceedings of the National Academy of Sciences, Mathematics, Vol. 39, pp. 1095-1100.

[SiB96] Singh, S. P., and Bertsekas, D. P., 1996. "Reinforcement Learning for Dynamic Channel Allocation in Cellular Telephone Systems," submitted to the 1996 Neural Information Processing Systems Conference.

[SiD96] Singh, S. P., and Dayan, P., 1996. "Analytical Mean Squared Error Curves in Temporal Difference Learning," unpublished report.

[SiS96] Singh, S. P., and Sutton, R. S., 1996. "Reinforcement Learning with Replacing Eligibility Traces," Machine Learning, Vol. 22, pp. 123-158.

[SiY94] Singh, S. P., and Yee, R. C., 1994. "An Upper Bound on the Loss from Approximate Optimal Value Functions," Machine Learning, Vol. 16, pp. 227-233.

[Sin94] Singh, S. P., 1994. "Reinforcement Learning Algorithms for Average-Payoff Markovian Decision Processes," Proceedings of the 12th National Conference on Artificial Intelligence, pp. 202-207.

[Str76] Strang, G., 1976. Linear Algebra and its Applications, Academic Press, N. Y.

[Sut84] Sutton, R. S., 1984. Temporal Credit Assignment in Reinforcement Learning, Ph. D. thesis, University of Massachusetts, Amherst, MA.

[Sut88] Sutton, R. S., 1988. "Learning to Predict by the Methods of Temporal Differences," Machine Learning, Vol. 3, pp. 9-44.

[Sut95] Sutton, R. S., 1995. "On the Virtues of Linear Learning and Trajectory Distributions," Proceedings of the Workshop on Value Function Approximation, Boyan, J. A., Moore, A. W., and Sutton, R. S., (eds.), Report CMU-CS-95-206, Carnegie Mellon University, Pittsburgh, PA.

[TeG96] Tesauro, G., and Galperin, G. R., 1996. "On-Line Policy Improvement Using Monte Carlo Search," unpublished report.

[Tes89] Tesauro, G. J., 1989. “Neurogammon Wins Computer Olympiad,” Neural Computation, Vol. 1, pp. 321-323.

[Tes92] Tesauro, G. J., 1992. “Practical Issues in Temporal Difference Learning,” Machine Learning, Vol. 8, pp. 257-277.

[Tes94] Tesauro, G. J., 1994. “TD-Gammon, a Self-Teaching Backgammon Program, Achieves Master-Level Play,” Neural Computation, Vol. 6, pp. 215-219.

[Tes95] Tesauro, G. J., 1995. “Temporal Difference Learning and TD-Gammon,” Communications of the ACM, Vol. 38, pp. 58-68.

[TrZ93] Trick, M. A., and Zin, S. E., 1993. “A Linear Programming Approach to Solving Stochastic Dynamic Programs,” preprint.

[TsV96a] Tsitsiklis, J. N., and Van Roy, B., 1996. “Feature-Based Methods for Large-Scale Dynamic Programming,” Machine Learning, Vol. 22, pp. 59-94.

[TsV96b] Tsitsiklis, J. N., and Van Roy, B., 1996. “An Analysis of Temporal-Difference Learning with Function Approximation,” Lab. for Info. and Decision Systems Report LIDS-P-2322, Massachusetts Institute of Technology, Cambridge, MA.

[Tsi89] Tsitsiklis, J. N., 1989. “A Comparison of Jacobi and Gauss-Seidel Parallel Iterations,” Applied Mathematics Letters, Vol. 2, pp. 167-170.

[Tsi94] Tsitsiklis, J. N., 1994. “Asynchronous Stochastic Approximation and Q-Learning,” Machine Learning, Vol. 16, pp. 185-202.

[Tsi96] Tsitsiklis, J. N., 1996. “On the Convergence of Optimistic Policy Iteration,” unpublished report.

[VaT96] Van Roy, B., and Tsitsiklis, J. N., 1996. “Stable Linear Approximations to Dynamic Programming for Stochastic Control Problems with Local Transitions,” in Advances in Neural Information Processing Systems 8, MIT Press, Cambridge, MA, pp. 1045-1051.

[Van76] van Nunen, J. A. E. E., 1976. “A Set of Successive Approximation Methods for Discounted Markovian Decision Problems,” Z. Oper. Res., Vol. 20, pp. 203-208.

[Van78] van der Wal, J., 1978. “Discounted Markov Games: Successive Approximation and Stopping Times,” International J. of Game Theory, Vol. 6, pp. 11-22.

[Van95] Van Roy, B., 1995. “Feature-Based Methods for Large Scale Dynamic Programming,” Lab. for Info. and Decision Systems Report LIDS-TH-2289, Massachusetts Institute of Technology, Cambridge, MA.

[WGM73] Widrow, B., Gupta, N. K., and Maitra, S., 1973. “Punish/Reward: Learning with a Critic in Adaptive Threshold Systems,” IEEE Trans. on

Systems, Man, and Cybernetics, Vol. 3, pp. 455-465.

[WaD92] Watkins, C. J. C. H., and Dayan, P., 1992. "Q-Learning," Machine Learning, Vol. 8, pp. 279-292.

[Wat89] Watkins, C. J. C. H., 1989. "Learning from Delayed Rewards," Ph.D. Thesis, Cambridge University, Cambridge, England.

[Wer74] Werbös, P. J, 1974. "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences," Ph.D. Thesis, Harvard University, Cambridge, MA.

[Wer77] Werbös, P. J., 1977. "Advanced Forecasting Methods for Global Crisis Warning and Models of Intelligence," General Systems Yearbook, Vol. 22, pp. 25-38.

[Wer90] Werbös, P. J., 1990. "Consistency of HDP Applied to a Simple Reinforcement Learning Problem," Neural Networks, Vol. 3, pp. 179-189.

[Wer92a] Werbös, P. J, 1992. "Approximate Dynamic Programming for Real-Time Control and Neural Modeling," in D. A. White and D. A. Sofge, (eds.), Handbook of Intelligent Control, Van Nostrand, N. Y.

[Wer92b] Werbös, P. J, 1992. "Neurocontrol and Supervised Learning: an Overview and Valuation," in D. A. White and D. A. Sofge, (eds.), Handbook of Intelligent Control, Van Nostrand, N. Y.

[WhS92] White, D. A., and Sofge, D. A., (eds.), 1992. Handbook of Intelligent Control, Van Nostrand, N. Y.

[Whi63] White, D. J., 1963. "Dynamic Programming, Markov Chains, and the Method of Successive Approximations," Journal of Mathematical Analysis and Applications, Vol. 6, pp. 373-376.

[Whi78] Whitt, W., 1978. "Approximations of Dynamic Programs I," Mathematics of Operations Research, Vol. 3, pp. 231-243.

[Whi79] Whitt, W., 1979. "Approximations of Dynamic Programs II," Mathematics of Operations Research, Vol. 4, pp. 179-185.

[Whi89] White, H., 1989. "Some Asymptotic Results for Learning in Single Hidden-Layer Feedforward Network Models," Journal of the American Statistical Association, Vol. 84, pp. 1003-1013.

[WiB93] Williams, R. J., and Baird, L. C., 1993. "Analysis of Some Incremental Variants of Policy Iteration: First Steps Toward Understanding Actor-Critic Learning Systems," Report NU-CCS-93-11, College of Computer Science, Northeastern University, Boston, MA.

[WiH60] Widrow, B., and Hoff, M. E., 1960. "Adaptive Switching Circuits," Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, part 4, pp. 96-104.

[Yin92] Yin, G., 1992. "On Extensions of Polyak's Averaging Approach to Stochastic Approximation," Stochastics, Vol. 36, pp. 245-264.

[ZhD95] Zhang, W., and Dietterich, T. G., 1995. "A Reinforcement Learning Approach to Job-Shop Scheduling," Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pp. 1114-1120.

[ZhD96] Zhang, W., and Dietterich, T. G., 1996. "High-Performance Job-Shop Scheduling with a Time-Delay TD(λ) Network," in Advances in Neural Information Processing Systems 8, MIT Press, Cambridge, MA, pp. 1024-1030.

[ZhY89] Zhang, M., and Yum, T.-S. P., 1989. "Comparisons of Channel-Assignment Strategies in Cellular Mobile Telephone Systems," IEEE Trans. Vehic. Technol., Vol. 38, pp. 211-215.