# Lessons from AlphaZero for Optimal, Model Predictive, and Adaptive Control

Dimitri P. Bertsekas

2nd Printing

# *Lessons from AlphaZero for Optimal, Model Predictive, and Adaptive Control*

by

Dimitri P. Bertsekas

**Arizona State University**
**and**
**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 03060**
**U.S.A.**

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



**Publisher's Cataloging-in-Publication Data**

Bertsekas, Dimitri P.
Lessons from AlphaZero for Optimal, Model Predictive, and Adaptive Control
Includes Bibliography and Index
1. Mathematical Optimization. 2. Dynamic Programming. I. Title.
QA402.5 .B465 2020 519.703 00-91281

ISBN-10: 1-886529-17-5, ISBN-13: 978-1-886529-17-5

2nd Printing: In this printing, minor typos were corrected, Section 6.5 was expanded to include material on incremental rollout, and small editorial changes were made.

# 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 Department, Stanford University, and the Electrical Engineering Department 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 McAfee Professor of Engineering. In 2019, he joined the School of Computing and Augmented Intelligence at the Arizona State University, Tempe, AZ, as Fulton Professor of Computational Decision Making.

Professor Bertsekas' teaching and research have spanned several fields, including deterministic optimization, dynamic programming and stochastic control, large-scale and distributed computation, artificial intelligence, and data communication networks. He has authored or coauthored numerous research papers and nineteen books, several of which are currently used as textbooks in MIT classes, including "Dynamic Programming and Optimal Control," "Data Networks," "Introduction to Probability," and "Nonlinear Programming." At ASU, he has been focusing in teaching and research in reinforcement learning, and he has developed several textbooks and research monographs in this field since 2019.

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 2001 AACC John R. Ragazzini Education Award, the 2009 INFORMS Expository Writing Award, the 2014 AACC Richard Bellman Heritage Award, the 2014 INFORMS Khachiyan Prize for Life-Time Accomplishments in Optimization, the 2015 MOS/SIAM George B. Dantzig Prize, and the 2022 IEEE Control Systems Award. In 2018 he shared with his coauthor, John Tsitsiklis, the 2018 INFORMS John von Neumann Theory Prize for the contributions of the research monographs "Parallel and Distributed Computation" and "Neuro-Dynamic Programming." Professor Bertsekas was elected in 2001 to the United States National Academy of Engineering for "pioneering contributions to fundamental research, practice and education of optimization/control theory."

**ATHENA SCIENTIFIC**

**OPTIMIZATION AND COMPUTATION SERIES**

1. Lessons from AlphaZero for Optimal, Model Predictive, and Adaptive Control by Dimitri P. Bertsekas, 2022, ISBN 978-1-886529-17-5, 245 pages
2. Abstract Dynamic Programming, 3rd Edition, by Dimitri P. Bertsekas, 2022, ISBN 978-1-886529-47-2, 420 pages
3. Rollout, Policy Iteration, and Distributed Reinforcement Learning, by Dimitri P. Bertsekas, 2020, ISBN 978-1-886529-07-6, 480 pages
4. Reinforcement Learning and Optimal Control, by Dimitri P. Bertsekas, 2019, ISBN 978-1-886529-39-7, 388 pages
5. Dynamic Programming and Optimal Control, Two-Volume Set, by Dimitri P. Bertsekas, 2017, ISBN 1-886529-08-6, 1270 pages
6. Nonlinear Programming, 3rd Edition, by Dimitri P. Bertsekas, 2016, ISBN 1-886529-05-1, 880 pages
7. Convex Optimization Algorithms, by Dimitri P. Bertsekas, 2015, ISBN 978-1-886529-28-1, 576 pages
8. Convex Optimization Theory, by Dimitri P. Bertsekas, 2009, ISBN 978-1-886529-31-1, 256 pages
9. Introduction to Probability, 2nd Edition, by Dimitri P. Bertsekas and John N. Tsitsiklis, 2008, ISBN 978-1-886529-23-6, 544 pages
10. Convex Analysis and Optimization, by Dimitri P. Bertsekas, Angelia Nedić, and Asuman E. Ozdaglar, 2003, ISBN 1-886529-45-0, 560 pages
11. Network Optimization: Continuous and Discrete Models, by Dimitri P. Bertsekas, 1998, ISBN 1-886529-02-7, 608 pages
12. Network Flows and Monotropic Optimization, by R. Tyrrell Rockafellar, 1998, ISBN 1-886529-06-X, 634 pages
13. Introduction to Linear Optimization, by Dimitris Bertsimas and John N. Tsitsiklis, 1997, ISBN 1-886529-19-1, 608 pages
14. Parallel and Distributed Computation: Numerical Methods, by Dimitri P. Bertsekas and John N. Tsitsiklis, 1997, ISBN 1-886529-01-9, 718 pages
15. Neuro-Dynamic Programming, by Dimitri P. Bertsekas and John N. Tsitsiklis, 1996, ISBN 1-886529-10-8, 512 pages
16. Constrained Optimization and Lagrange Multiplier Methods, by Dimitri P. Bertsekas, 1996, ISBN 1-886529-04-3, 410 pages
17. Stochastic Optimal Control: The Discrete-Time Case, by Dimitri P. Bertsekas and Steven E. Shreve, 1996, ISBN 1-886529-03-5, 330 pages

# *Contents*

# *Preface*

**With four parameters I can fit an elephant, and with five I can make him wiggle his trunk.†**

**John von Neumann**

The purpose of this monograph is to propose and develop a new conceptual framework for approximate Dynamic Programming (DP) and Reinforcement Learning (RL). This framework centers around two algorithms, which are designed largely independently of each other and operate in synergy through the powerful mechanism of Newton's method. We call these the *off-line training* and the *on-line play* algorithms; the names are borrowed from some of the major successes of RL involving games. Primary examples are the recent (2017) AlphaZero program (which plays chess), and the similarly structured and earlier (1990s) TD-Gammon program (which plays backgammon). In these game contexts, the off-line training algorithm is the method used to teach the program how to evaluate positions and to generate good moves at any given position, while the on-line play algorithm is the method used to play in real time against human or computer opponents.

† From the meeting of Freeman Dyson and Enrico Fermi (p. 273 of the Segre and Hoerlin biography of Fermi, The Pope of Physics, Picador, 2017): "When Dyson met with him in 1953, Fermi welcomed him politely, but he quickly put aside the graphs he was being shown indicating agreement between theory and experiment. His verdict, as Dyson remembered, was "There are two ways of doing calculations in theoretical physics. One way, and this is the way I prefer, is to have a clear physical picture of the process you are calculating. The other way is to have a precise and self-consistent mathematical formalism. You have neither." When a stunned Dyson tried to counter by emphasizing the agreement between experiment and the calculations, Fermi asked him how many free parameters he had used to obtain the fit. Smiling after being told "Four," Fermi remarked, "I remember my old friend Johnny von Neumann used to say, with four parameters I can fit an elephant, and with five I can make him wiggle his trunk." See also the paper by Mayer, Khairy, and Howard [MKH10], which provides a verification of the von Neumann quotation.

Both AlphaZero and TD-Gammon were trained off-line extensively using neural networks and an approximate version of the fundamental DP algorithm of policy iteration. Yet the AlphaZero player that was obtained off-line is not used directly during on-line play (it is too inaccurate due to approximation errors that are inherent in off-line neural network training). Instead a separate on-line player is used to select moves, based on multistep lookahead minimization and a terminal position evaluator that was trained using experience with the off-line player. The on-line player performs a form of policy improvement, which is not degraded by neural network approximations. As a result, it greatly improves the performance of the off-line player.

Similarly, TD-Gammon performs on-line a policy improvement step using one-step or two-step lookahead minimization, which is not degraded by neural network approximations. To this end it uses an off-line neural network-trained terminal position evaluator, and importantly it also extends its on-line lookahead by rollout (simulation with the one-step lookahead player that is based on the position evaluator).

Thus in summary:

(a) The on-line player of AlphaZero plays much better than its extensively trained off-line player. This is due to the beneficial effect of exact policy improvement with long lookahead minimization, which corrects for the inevitable imperfections of the neural network-trained off-line player, and position evaluator/terminal cost approximation.

(b) The TD-Gammon player that uses long rollout plays much better than TD-Gammon without rollout. This is due to the beneficial effect of the rollout, which serves as a substitute for long lookahead minimization.

An important lesson from AlphaZero and TD-Gammon is that the performance of an off-line trained policy can be greatly improved by on-line approximation in value space, with long lookahead (involving minimization or rollout with the off-line policy, or both), and terminal cost approximation that is obtained off-line. This performance enhancement is often dramatic and is due to a simple fact, which is couched on algorithmic mathematics and is the focal point of this work:

(a) *Approximation in value space with one-step lookahead minimization amounts to a step of Newton's method for solving Bellman's equation.*

(b) *The starting point for the Newton step is based on the results of off-line training, and may be enhanced by longer lookahead minimization and on-line rollout.*

Indeed the major determinant of the quality of the on-line policy is the Newton step that is performed on-line, while off-line training plays a secondary role by comparison.

Significantly, the synergy between off-line training and on-line play also underlies Model Predictive Control (MPC), a major control system design methodology that has been extensively developed since the 1980s. This synergy can be understood in terms of abstract models of infinite horizon DP and simple geometrical constructions, and helps to explain the all-important stability issues within the MPC context.

An additional benefit of policy improvement by approximation in value space, not observed in the context of games (which have stable rules and environment), is that it works well with changing problem parameters and on-line replanning, similar to indirect adaptive control. Here the Bellman equation is perturbed due to the parameter changes, but approximation in value space still operates as a Newton step. An essential requirement within this context is that a system model is estimated on-line through some identification method, and is used during the one-step or multistep lookahead minimization process.

In this monograph we will aim to provide insights (often based on visualization), which explain the beneficial effects of on-line decision making on top of off-line training. In the process, we will bring out the strong connections between the artificial intelligence view of RL, and the control theory views of MPC and adaptive control. Moreover, we will show that in addition to MPC and adaptive control, our conceptual framework can be effectively integrated with other important methodologies such as multiagent systems and decentralized control, discrete and Bayesian optimization, and heuristic algorithms for discrete optimization.

One of our principal aims is to show, through the algorithmic ideas of Newton's method and the unifying principles of abstract DP, that the AlphaZero/TD-Gammon methodology of approximation in value space and rollout applies very broadly to deterministic and stochastic optimal control problems. Newton's method here is used for the solution of Bellman's equation, an operator equation that applies universally within DP with both discrete and continuous state and control spaces, as well as finite and infinite horizon. In this connection, we note that the mathematical complications associated with the formalism of Newton's method for nondifferentiable operators have been dealt with in the literature, using sophisticated methods of nonsmooth analysis. We have provided in an appendix a convergence analysis for a finite-dimensional version of Newton's method, which applies to finite-state problems, but conveys clearly the underlying geometrical intuition and points to infinite-state extensions. We have also provided an analysis for the classical linear-quadratic optimal control problem, the associated Riccati equation, and the application of Newton's method for its solution.

While we will deemphasize mathematical proofs in this work, there is considerable related analysis, which supports our conclusions, and can be found in the author's recent RL books [Ber19a], [Ber20a], and the abstract DP monograph [Ber22a]. In particular, the present work may be viewed as

a more intuitive, less mathematical, visually oriented exposition of the core material of the research monograph [Ber20a], which deals with approximation in value space, rollout, policy iteration, and multiagent systems. The abstract DP monograph [Ber22a] develops the mathematics that support the visualization framework of the present work, and is a primary resource for followup mathematical research. The RL textbook [Ber19a] provides a more general presentation of RL topics, and includes mathematical proof-based accounts of some of the core material of exact infinite horizon DP, as well as approximate DP. Much of this material is also contained, in greater detail, in the author's DP textbook [Ber12]. A mix of material contained in these books forms the core of the author's web-based RL course at ASU.

This monograph, as well as my earlier RL books, were developed while teaching several versions of my course at ASU over the last four years. Videolectures and slides from this course are available from my website

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

and provide a good supplement and companion resource to the present book. The hospitable and stimulating environment at ASU contributed much to my productivity during this period, and for this I am very thankful to my colleagues and students for useful interactions. My teaching assistants, Sushmita Bhattacharya, Sahil Badyal, and Jamison Weber, during my courses at ASU have been very supportive. I have also appreciated fruitful discussions with colleagues and students outside ASU, particularly Moritz Diehl, who provided very useful comments on MPC, and Yuchao Li, who proofread carefully the entire book, collaborated with me on research and implementation of various methods, and tested out several algorithmic variants.

Dimitri P. Bertsekas, 2022
dimitrib@mit.edu

# 1

# *AlphaZero, Off-Line Training, and On-Line Play*

**Contents**

In this work we will aim to provide a new conceptual framework for reinforcement learning and approximate dynamic programming. These two fields, through the synergy of their ideas in the 1980s and 1990s, and in conjunction of the emergence of machine learning, gave rise to a far-reaching synthesis that would eventually have a major impact on the field of algorithmic optimization.

In this chapter we provide an outline of the motivation and the algorithmic justification of our framework, and its connection to AlphaZero and related game programs, as well as Newton's method for solving fixed point problems. In subsequent chapters, we will flesh out our framework, drawing on the theory of abstract DP, related visualizations, ideas of adaptive, model predictive, and linear quadratic control, as well as paradigms from discrete and combinatorial optimization.

The development of the AlphaZero program by DeepMind Inc, as described in the papers [SHS17], [SSS17], is perhaps the most impressive success story in reinforcement learning (RL) todate. AlphaZero plays Chess, Go, and other games, and is an improvement in terms of performance and generality over the earlier AlphaGo program [SHM16], which plays the game of Go only. AlphaZero, and other chess programs based on similar principles, play as well or better than all competitor computer programs available in 2021, and much better than all humans. These programs are remarkable in several other ways. In particular, they have learned how to play without human instruction, just data generated by playing against themselves. Moreover, they learned how to play very quickly. In fact, AlphaZero learned how to play chess better than all humans and computer programs within hours (with the help of awesome parallel computation power, it must be said).

We should note also that the principles of the AlphaZero design have much in common with the TD-Gammon programs of Tesauro [Tes94], [Tes95], [TeG96] that play backgammon (a game of substantial computational and strategical complexity, which involves a number of states estimated to be in excess of $10^{20}$). Tesauro's programs stimulated much interest in RL in the middle 1990s, and one of these programs exhibits similarly different and better play than human backgammon players. A related program for the (one-player) game of Tetris, based on similar principles, is described by Scherrer et al. [SGG15], together with several antecedents, including algorithmic schemes dating to the 1990s, by Tsitsiklis and Van-Roy [TsV96], and Bertsekas and Ioffe [BeI96]. The backgammon and Tetris programs, while dealing with less complex games than chess, are of special interest because they involve significant stochastic uncertainty, and are thus unsuitable for the use of long lookahead minimization, which is widely believed to be one of the major contributors to the success of AlphaZero, and chess programs in general.

Still, for all of their brilliant implementations, these impressive game programs are couched on well established methodology, from optimal and

suboptimal control, which is portable to far broader domains of engineering, economics, and other fields. This is the methodology of dynamic programming (DP), policy iteration, limited lookahead minimization, rollout, and related approximations in value space. The aim of this work is to propose a conceptual, somewhat abstract framework, which allows insight into the connections of AlphaZero and TD-Gammon with some of the core problems in decision and control, and suggests potentially far-reaching extensions.

To understand the overall structure of AlphaZero and related programs, and their connections to the DP/RL methodology, it is useful to divide their design into two parts:

(a) *Off-line training*, which is an algorithm that learns how to evaluate chess positions, and how to steer itself towards good positions with a default/base chess player.

(b) *On-line play*, which is an algorithm that generates good moves in real time against a human or computer opponent, using the training it went through off-line.

An important empirical fact is that *the on-line player of AlphaZero plays much better than its extensively trained off-line player*. This supports a conceptual idea that applies in great generality and is central in this book, namely that *the performance of an off-line trained policy can be greatly improved by on-line play*. We will next briefly describe the off-training and on-line play algorithms, and relate them to DP concepts and principles, focusing on AlphaZero for the most part.

## 1.1 OFF-LINE TRAINING AND POLICY ITERATION

An off-line training algorithm like the one used in AlphaZero is the part of the program that learns how to play through self-training that takes place before real-time play against any opponent. It is illustrated in Fig. 1.1.1, and it generates a sequence of *chess players* and *position evaluators*. A chess player assigns "probabilities" to all possible moves at any given chess position: these may be viewed as a measure of "effectiveness" of the corresponding moves. A position evaluator assigns a numerical score to any given chess position, and thus predicts quantitatively the performance of a player starting from any position. The chess player and the position evaluator are represented by neural networks, a *policy network* and a *value network*, which accept as input a chess position and generate a set of move probabilities and a position evaluation, respectively.†

† Here the neural networks play the role of *function approximators*. By viewing a player as a function that assigns move probabilities to a position, and a position evaluator as a function that assigns a numerical score to a position, the policy and value networks provide approximations to these functions based

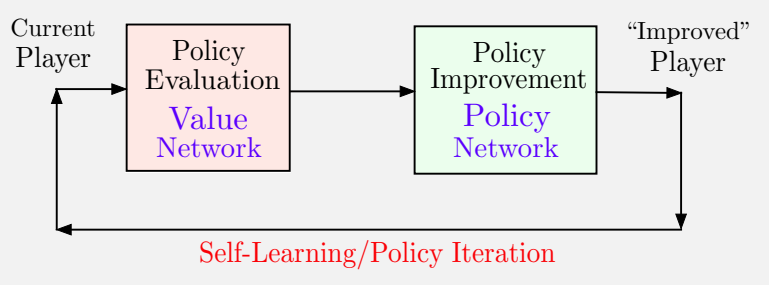


**Figure 1.1.1** Illustration of the AlphaZero off-line training algorithm. It generates a sequence of position evaluators and chess players. The position evaluator and the chess player are represented by two neural networks, a value network and a policy network, which accept a chess position and generate a position evaluation and a set of move probabilities, respectively.

In the more conventional DP-oriented terms of this work, a position is the state of the game, a position evaluator is a cost function that gives (an estimate of) the optimal cost-to-go at a given state, and the chess player is a randomized policy for selecting actions/controls at a given state.†

The overall training algorithm is a form of *policy iteration*, a DP algorithm that will be of primary interest to us in this work. Starting from a given player, it repeatedly generates (approximately) improved players, and settles on a final player that is judged empirically to be "best" out of all the players generated. Policy iteration may be separated conceptually into two stages (see Fig. 1.1.1).

(a) *Policy evaluation*: Given the current player and a chess position, the outcome of a game played out from the position provides a single data point. Many data points are thus collected, and are used to train a value network, whose output serves as the position evaluator for that player.

---

on training with data. Actually, AlphaZero uses the same neural network for training both value and policy. Thus there are two outputs of the neural net: value and policy. This is pretty much equivalent to having two separate neural nets and for the purposes of this work, we prefer to explain the structure as two separate networks. AlphaGo uses two separate value and policy networks. Tesauro's backgammon programs use a single value network, and generate moves when needed by one-step or two-step lookahead minimization, using the value network as terminal position evaluator.

† One more complication is that chess and Go are two-player games, while most of our development will involve single-player optimization. While DP theory and algorithms extend to two-player games, we will not discuss these extensions, except in a very limited way in Chapter 6. Alternatively, a chess program can be trained to play well against a fixed opponent, in which case the framework of single-player optimization applies.

(b) *Policy improvement*: Given the current player and its position evaluator, trial move sequences are selected and evaluated for the remainder of the game starting from many positions. An improved player is then generated by adjusting the move probabilities of the current player towards the trial moves that have yielded the best results.

In AlphaZero (as well as AlphaGo Zero, the version that plays the game of Go) the policy evaluation is done by using deep neural networks. The policy improvement uses a complicated algorithm called *Monte Carlo Tree Search* (MCTS for short), a form of randomized multistep lookahead minimization that enhances the efficiency of the multistep lookahead operation, by pruning intelligently the multistep lookahead graph.

We note, however, that deep neural networks and MCTS, while leading to some performance gains, are not of fundamental importance. The approximation quality that a deep neural network can achieve can also be achieved with a shallow neural network, perhaps with reduced sample efficiency. Similarly MCTS cannot achieve better lookahead accuracy than standard exhaustive search, although it may be more efficient computationally. Indeed, policy improvement can be done more simply without MCTS, as in Tesauro's TD-Gammon program: we try all possible move sequences from a given position, extend forward to some number of moves, and then evaluate the terminal position with the current player's position evaluator. The move evaluations obtained in this way are used to nudge the move probabilities of the current player towards more successful moves, thereby obtaining data that is used to train a policy network that represents the new player.†

Regardless of the use of deep neural networks and MCTS, it is important to note that *the final policy and the corresponding policy evaluation produced by approximate policy iteration and neural network training in AlphaZero involve serious inaccuracies, due to the approximations that are inherent in neural network representations*. The AlphaZero on-line player to be discussed next uses approximation in value space with multistep lookahead minimization, and does not involve any neural network, other than the one that has been trained off-line, so it is not subject to such inaccuracies. As a result, it plays much better than the off-line player.

† Quoting from the paper [SSS17] (p. 360): "The AlphaGo Zero selfplay algorithm can similarly be understood as an approximate policy iteration scheme in which MCTS is used for both policy improvement and policy evaluation. Policy improvement starts with a neural network policy, executes a MCTS based on that policy's recommendations, and then projects the (much stronger) search policy back into the function space of the neural network. Policy evaluation is applied to the (much stronger) search policy: the outcomes of selfplay games are also projected back into the function space of the neural network. These projection steps are achieved by training the neural network parameters to match the search probabilities and selfplay game outcome respectively."

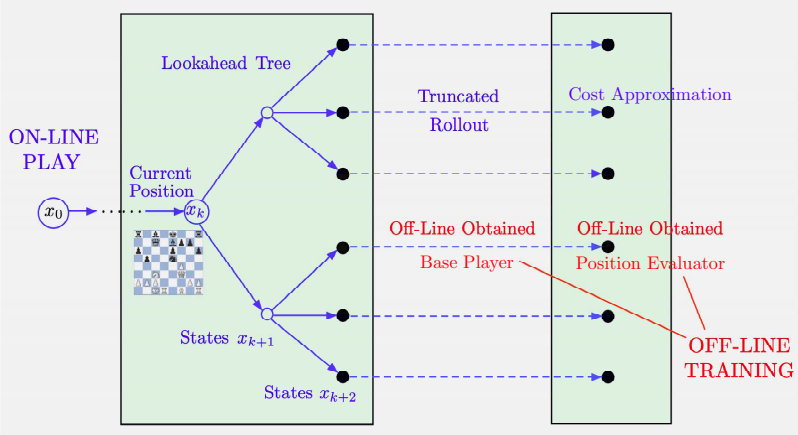


**Figure 1.2.1** Illustration of an on-line player such as the one used in AlphaGo, AlphaZero, and Tesauro's backgammon program [TeG96]. At a given position, it generates a lookahead graph of multiple moves up to some depth, then runs the off-line obtained player for some more moves, and evaluates the effect of the remaining moves by using the position evaluator of the off-line player.

## 1.2 ON-LINE PLAY AND APPROXIMATION IN VALUE SPACE - TRUNCATED ROLLOUT

Consider the "final" player obtained through the AlphaZero off-line training process. It can play against any opponent by generating move probabilities at any position using its off-line trained policy network, and then simply play the move of highest probability. This player would play very fast on-line, but it would not play good enough chess to beat strong human opponents. The extraordinary strength of AlphaZero is attained only after the player obtained from off-line training is embedded into another algorithm, which we refer to as the "on-line player."† In other words *AlphaZero plays on-line much better than the best player it has produced with sophisticated off-line training.* This phenomenon, *policy improvement through on-line play*, is centrally important for our purposes in this work.

Given the policy network/player obtained off-line and its value network/position evaluator, the on-line algorithm plays roughly as follows (see Fig. 1.2.1). At a given position, it generates a lookahead graph of all possi-

† Quoting from the paper [SSS17] (p. 354): "The MCTS search outputs probabilities of playing each move. These search probabilities usually select much stronger moves than the raw move probabilities of the neural network." To elaborate, this statement refers to the MCTS algorithm that is used on line to generate the move probabilities at each position encountered in the course of a given game. The neural network referred to is trained off-line, also using in part the MCTS algorithm.

ble multiple move and countermove sequences, up to a given depth. It then runs the off-line obtained player for some more moves, and evaluates the effect of the remaining moves by using the position evaluator of the value network.

The middle portion, called "truncated rollout," may be viewed as *an economical substitute for longer lookahead minimization*. Actually truncated rollout is not used in the published version of AlphaZero [SHS17]; the first portion (multistep lookahead minimization) is very long and implemented efficiently (partly through the use of MCTS), so that the rollout portion is not essential. Rollout has been used in AlphaGo, the AlphaZero predecessor [SHM16]. Moreover, chess and Go programs (including AlphaZero) typically use a well-known limited form of rollout, called "quiescence search," which aims to resolve imminent threats and highly dynamic positions through simulated multi-move piece exchanges, before invoking the position evaluator. Rollout is instrumental in achieving high performance in Tesauro's 1996 backgammon program [TeG96]. The reason is that backgammon involves stochastic uncertainty, so long lookahead minimization is not possible because of rapid expansion of the lookahead graph with every move.†

In control system design, similar architectures to the ones of AlphaZero and TD-Gammon are employed in model predictive control (MPC). There, the number of steps in lookahead minimization is called the *control interval*, while the total number of steps in lookahead minimization and truncated rollout is called the *prediction interval*; see e.g., Magni et al. [MDM01]. (The MATLAB toolbox for MPC design explicitly allows the user to choose these two intervals.) The benefit of truncated rollout in providing an economical substitute for longer lookahead minimization is well known within this context. We will discuss further the structure of MPC and its similarities with the AlphaZero architecture in Chapter 5.

Dynamic programming frameworks with cost function approximations that are similar to the on-line player illustrated in Fig. 1.2.1, are also known as *approximate dynamic programming*, or *neuro-dynamic pro-*

† Tesauro's rollout-based backgammon program [TeG96] uses only a value network, which was trained using an approximate policy iteration scheme developed several years earlier [Tes94]. This network is used to generate moves for the truncated rollout via a one-step or two-step lookahead minimization. Thus the value network also serves as a substitute for the policy network during the rollout operation. The position evaluation used at the end of the truncated rollout is also provided by the value network. The middle portion of Tesauro's scheme (truncated rollout) is important for achieving a very high quality of play, as it effectively extends the length of lookahead from the current position (the player with rollout [TeG96] plays much better than the player without rollout [Tes94]). In backgammon circles, Tesauro's program with truncated rollout is viewed as essentially "optimal."

*gramming*, and will be central for our purposes. They will be generically referred to as *approximation in value space* in this work.†

Note also that in general, off-line training and on-line policy implementation may be designed independently of each other. For example the off-line training portion may be very simple, such as using a known heuristic policy for rollout without truncation, or without terminal cost approximation. Conversely, a sophisticated process may be used for off-line training of a terminal cost function approximation, which is used following the lookahead minimization in a value space approximation scheme.

## 1.3 THE LESSONS OF ALPHAZERO

The AlphaZero and TD-Gammon experiences reinforce an important conclusion that applies more generally to decision and control problems: despite the extensive off-line effort that may have gone into the design of a policy, performance may be greatly improved by on-line approximation in value space, with extra lookahead involving minimization and/or with rollout using this policy, and terminal cost approximation.

In the following chapters, we will aim to amplify on this theme and to focus on the principal characteristics of AlphaZero-like architectures, within a broader context of optimal decision and control. We will make use of intuitive visualization, and the central role of Newton's method for solving Bellman's equation.‡ Briefly, our central point will be that *on-line approximation in value space amounts to a step of Newton's method for solving Bellman's equation, while the starting point for the Newton step is based on the results of off-line training*; see Fig. 1.3.1. Moreover, *this starting point may be enhanced by several types of on-line operations, including longer lookahead minimization, and on-line rollout with a policy obtained through off-line training, or heuristic approximations*.

---

† The names "approximate dynamic programming" and "neuro-dynamic programming" are often used as synonyms to RL. However, RL is often thought to also subsume the methodology of approximation in policy space, which involves search for optimal parameters within a parametrized set of policies. The search is done with methods that are largely unrelated to DP, such as for example stochastic gradient or random search methods (see the author's RL textbook [Ber19a]). Approximation in policy space may be used off-line to design a policy that can be used for on-line rollout. However, as a methodological subject, approximation in policy space has little connection to the ideas of the present work.

‡ Bellman's equation, the centerpiece of infinite horizon DP theory, is viewed here as a functional equation, whose solution is the cost of operating the system viewed as a function of the system's initial state. We will give examples of Bellman's equation in Chapter 2 for discounted and other problems, and we will also provide in Chapter 3 abstract forms of Bellman's equation that apply more generally.

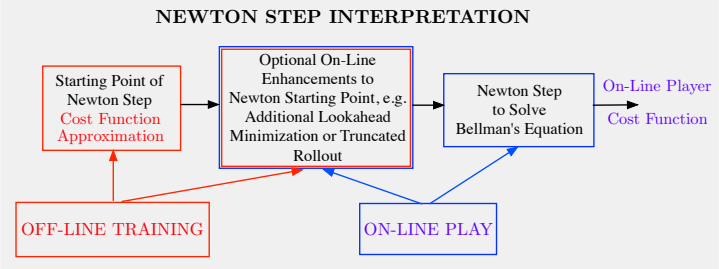


**Figure 1.3.1** Illustration of the connections between off-line training, on-line play, and Newton's method for solving Bellman's equation. On-line play is viewed as a Newton step, while off-line training provides the starting point for the Newton step. The Newton step starts with a cost approximation $\tilde{J}$, which may be enhanced on-line by additional lookahead minimization and/or rollout, and produces the cost function of the on-line player.

This interpretation will be the basis for powerful insights into issues of stability, performance, and robustness of the on-line generated policy. In particular, we will aim to show that feedback control, based on approximation in value space and the underlying off-line training/on-line play structure, offers benefits that go well beyond the conventional wisdom that "feedback corrects for uncertainty, and modeling errors." The reason is that by overlaying on-line play on top of off-line training, we gain significantly in performance, by correcting (through the Newton step) for the errors that are inherent in off-line training with approximation architectures such as neural networks.

Our mathematical framework is couched on unifying principles of abstract DP, including abstract forms of Bellman's equation, and the value and policy iteration algorithms (see the author's books [Ber12], [Ber22a]). However, in this work, we will deemphasize mathematical proofs. There is considerable related analysis, which supports our conclusions and can be found in the author's recent RL books [Ber19a], [Ber20a].

In summary, our discussion will aim to highlight the following points:

**Summary**

(a) Approximation in value space, with one-step lookahead minimization, is an exact step of Newton's method for solving Bellman's equation. This step may be preceded by on-line adjustments and/or value iterations, which enhance its starting point.

(b) The starting point for the Newton step of (a) is obtained by some unspecified off-line methodology, which may involve the solution of a related but simpler problem, and/or training with data that makes use of neural networks or feature-based architectures.

(c) The on-line play and off-line training parts of the AlphaZero/TD-Gammon design structure correspond to (a) and (b) above, respectively.

(d) The on-line player of AphaZero plays much better than its deep neural network-trained player for the same reason that the Newton step (a) improves substantially on its starting point (b), namely the underlying superlinear convergence property that is typical of Newton's method.

(e) $\ell$-step lookahead minimization can be viewed as one-step lookahead minimization where $\ell - 1$ value iterations are used to enhance the starting point of the Newton step of (a) above. It is important to perform the first step of the lookahead exactly, but for the last $\ell - 1$ steps, approximations may be tolerated.

(f) The algorithmic processes for (a) and (b) above can be designed by a variety of methods, and independently of each other. For example:

(1) The implementation of the Newton step (a) may or may not involve any of the following: truncated rollout, on-line Monte Carlo simulation, MCTS or other efficient graph search techniques, forms of continuous space optimization, on-line policy iteration, etc.

(2) The computation of the starting point (b) may or may not involve any of the following: Q-learning, approximate policy iteration based on temporal differences or aggregation, neural networks, feature-based function approximation, policies trained off-line by approximation in policy space, including policy gradient methods or policy random search, etc. Moreover, the details of this computation may vary broadly without affecting significantly the effectiveness of the overall scheme, which is primarily determined by the Newton step (a).

(g) An efficient implementation of the Newton step (a) is often critical in order to meet real-time constraints for generating controls, and to allow longer lookahead minimization, which enhances the starting point of the Newton step and its performance. By contrast, off-line training algorithms used for (b) have much less stringent real-time constraints, and the issues of sample efficiency and fine tuned performance, while important, are not critical.

(h) The efficient implementation of the Newton step may benefit from the use of distributed computation and other simplifications. A case in point is multiagent problems, which we will discuss later (see Chapter 3).

(i) Approximation in value space addresses effectively issues of robustness and on-line replanning for problems with changing parameters. The mechanism is similar to the one of indirect adaptive control: changing problem parameters are estimated on-line and a Newton step is used in place of an expensive full reoptimization of the controller. In the presence of changing parameters, the Bellman equation changes, but the Newton step itself remains powerful and aims at the optimal solution that corresponds to the estimated system parameters.

(j) Model predictive control (MPC) has a conceptually similar structure to the AlphaZero-like programs, and entails an on-line play component involving multistep lookahead minimization, forms of truncated rollout, and an off-line training component to construct terminal cost approximations, and "safe" state space regions or reachability tubes to deal with state constraints. The success of MPC may be attributed to these similarities and to its resilience to changing problem parameters as per (i) above.

(k) On-line rollout with a stable policy yields a good starting point for the Newton step (a): it improves the stability properties of the policy obtained by approximation in value space, and provides an economical substitute for long lookahead minimization.

(l) Because the ideas outlined above are couched on principles of DP that often hold for arbitrary state and control spaces, they are valid within very general contexts: continuous-spaces control systems, discrete-spaces Markov decision problems, control of hybrid systems, decision making in multiagent systems, and discrete and combinatorial optimization.

The preceding points are meant to highlight the essence of the connections between AlphaZero and TD-Gammon, approximation in value space, and decision and control. Naturally in practice there are exceptions and modifications, which need to be worked out in the context of particular applications, under appropriate assumptions. Moreover, while some results and elaborations are available through the research that has been done on approximate DP and on MPC, several of the results suggested by the analysis and insights of the present work remain to be rigorously established and enhanced within the context of specific problems.

## 1.4 A NEW CONCEPTUAL FRAMEWORK FOR REINFORCEMENT LEARNING

In this work we will emphasize the distinct roles of off-line training and on-line play algorithms within the structure of approximate sequential decision

making and approximation in value space schemes. In doing so, we will aim for a new conceptual framework for RL, which is based on the synergism and complementarity of off-line training and on-line play, and the analytical framework of Newton's method.

We will implicitly assume that *the time available for off-line training is very long (practically limitless)*, but that the problem at hand is such that exact DP algorithms, like policy iteration and Q-learning, are impossible for one (or both) of the following two reasons:

(a) There are too many states (either an infinite number as in continuous space problems, or very large as in chess). As a result a lookup table representation of policies, value functions, and/or Q-factors is impossible, and the only practical alternative is a compact representation, via a neural network or some other approximation architecture.

(b) The system model is changing over time as in adaptive control, so even if an exactly optimal policy is computed off-line under some nominal problem assumptions, it becomes suboptimal when the problem parameters change.

In this work, we will not discuss training algorithms and their associated sample efficiency issues, and we will refer to the many available sources, including the author's RL books [Ber19a], [Ber20a].

On the other hand, we will assume that there is limited time for on-line decision making, because of hard practical constraints on the real time that is available between decisions. These constraints are highly problem dependent: for some problems, following the observation of the state, we may need to produce the next decision within a fraction of a second, whereas for others we may have hours at our disposal. We will assume that *whatever time is available, it will be used to provide quite accurate (nearly exact) one-step or multistep lookahead minimization, and time permitting, to extend as much as possible the combined length of the lookahead minimization and the truncated rollout with an off-line computed policy*. We will thus implicitly take it as given that longer (as well as more accurate) lookahead minimization is better for the performance of the policy obtained,† although the division of effort between lookahead minimization and truncated rollout with a policy is a design decision that may depend on the circumstances. Note that parallel and distributed computation can play an important role in mitigating practical on-line time constraints.

The central fact in our conceptual framework is that approximation in value space with one-step lookahead minimization constitutes a single Newton step for solving Bellman's equation. Contrary to other Newton-like steps that may have been part of the off-line training process, this

† It is possible to construct artificial problems, where longer lookahead results in worse performance (see [Ber19a], Section 2.2), but such problems are rare in practice.

single Newton step is accurate: *all the approximation has been shifted to its starting point*. Moreover, the Newton step can be very powerful, and its starting point can be enhanced by multistep lookahead minimization or by truncated rollout. From an algorithmic point of view, the Newton step converges superlinearly without the need for differentiability of the Bellman operator $T$: it takes advantage of the monotonicity and concavity structure of $T$ (see the Appendix, where we will discuss Newton's method without differentiability assumptions).

To summarize, both off-line training and on-line play are subject to fundamental limits: the former's limit is the constrained power of the approximation architecture, while the latter's limit is the constrained on-line computation time. The former limit cannot be easily overcome, but the latter limit can be stretched a lot thanks to the power of the Newton step, supplemented by long lookahead minimization and truncated rollout, as well as through the use of parallel and distributed computation.

Our design philosophy in a nutshell is the following:

(1) The major determinant of the quality of the controller obtained by our schemes is the Newton step that is performed on-line. Stated somewhat bluntly, *off-line training is secondary by comparison*, in the sense that without on-line one-step or multistep lookahead minimization, the quality of the policy obtained by off-line training alone is often unacceptably poor. In particular, whether done by neural networks, feature-based linear architectures, temporal difference methods, aggregation, policy gradient, policy random search, or whatever other reasonable approach, off-line training principally serves the purpose of providing a good or reasonable starting point for the Newton step. This is the principal lesson from AlphaZero and TD-Gammon in our view. This philosophy also underlies MPC, where on-line lookahead minimization has traditionally been the principal focus, perhaps supplemented by truncated rollout, with off-line calculations playing a limited subsidiary role.†

(2) The Newton step is often powerful enough to smooth out differences in various off-line training methods. In particular, methods such as TD($\lambda$) with different values of $\lambda$, policy gradient, linear programming, etc, all give different, but more or less equally good starting points for the Newton step. The conclusion from this is that off-line training with a very large number of samples, and sophisticated ways to improve sample efficiency may not be very useful, beyond a certain point, because gains in efficiency and accuracy tend to be washed up by the Newton step.

† Incidentally, this is a major reason why there is an apparent disconnect between the MPC community, which is mostly focused on on-line play, and the RL community, which is mostly focused on off-line training.

(3) The on-line Newton step also works well in the context of adaptive control, as long as it is calculated on the basis of the currently correct model parameters (so this requires an on-line parameter identification algorithm). The reason is that when problem parameters change, the Bellman operator changes, but the Newton step is executed on the basis of the correct Bellman operator. This is also a principal reason why MPC schemes have been used with success in adaptive control contexts.

We will return to these points repeatedly in the course of our presentation.

## 1.5 NOTES AND SOURCES

The theory of DP dates to the late 40s and 50s, and provides the foundation for our subject. Indeed RL may be viewed as an approximate form of the exact DP methodology. The author's DP textbook [Ber17a] provides an extensive discussion of finite horizon DP, and its applications to discrete and continuous spaces problems, using a notation and style that is consistent with the present book. The books by Puterman [Put94] and by the author [Ber12] provide detailed treatments of infinite horizon finite-state Markovian decision problems.

Continuous spaces infinite horizon problems are covered in the author's book [Ber12], while some of the more complex mathematical aspects of exact DP are discussed in the monograph by Bertsekas and Shreve [BeS78] (particularly the probabilistic/measure-theoretic issues associated with stochastic optimal control).†

† The rigorous mathematical theory of stochastic optimal control, including the development of an appropriate measure-theoretic framework, dates to the 60s and 70s. It culminated in the monograph [BeS78], which provides the now "standard" framework, based on the formalism of Borel spaces, lower semianalytic functions, and universally measurable policies. This development involves daunting mathematical complications, which stem, among others, from the fact that when a Borel measurable function $F(x,u)$, of the two variables $x$ and $u$, is minimized with respect to $u$, the resulting function

$$G(x) = \min_u F(x,u)$$

need not be Borel measurable (it is lower semianalytic). Moreover, even if the minimum is attained by several functions/policies $\mu$, i.e., $G(x) = F\big(x,\mu(x)\big)$ for all $x$, it is possible that none of these $\mu$ is Borel measurable (however, there does exist a minimizing policy that belongs to the broader class of universally measurable policies). Thus, starting with a Borel measurability framework for cost functions and policies, we quickly get outside that framework when executing DP algorithms, such as value and policy iteration. The broader framework of universal measurability is required to correct this deficiency, in the absence of additional (fairly strong) assumptions.

The third edition of the author's abstract DP monograph [Ber22a], expands on the original 2013 first edition, and aims at a unified development of the core theory and algorithms of total cost sequential decision problems. It addresses simultaneously stochastic, minimax, game, risk-sensitive, and other DP problems, through the use of abstract DP operators (or Bellman operators as they are often called in RL). The abstract framework is important for some the visualization insights and the connections to Newton's method that are central for the purposes of this book.

The approximate DP and RL literature has expanded tremendously since the connections between DP and RL became apparent in the late 1980s and early 1990s. In what follows, we will provide a list of textbooks, research monographs, and broad surveys, which supplement our discussions, express related viewpoints, and collectively provide a guide to the literature.

**RL Textbooks**

Two books were written in the 1990s, setting the tone for subsequent developments in the field. One in 1996 by Bertsekas and Tsitsiklis [BeT96], which reflects a decision, control, and optimization viewpoint, and another in 1998 by Sutton and Barto, which reflects an artificial intelligence viewpoint (a 2nd edition, [SuB18], was published in 2018). We refer to the former book and also to the author's DP textbooks [Ber12], [Ber17a] for a broader discussion of some of the topics of this book, including algorithmic convergence issues and additional DP models, such as those based on average cost and semi-Markov problem optimization. Note that both of these books deal with finite-state Markov decision models and use a transition probability notation, as they do not address continuous spaces problems, which are also of major interest in this book.

More recent books are by Gosavi [Gos15] (a much expanded 2nd edition of his 2003 monograph), which emphasizes simulation-based optimization and RL algorithms, Cao [Cao07], which focuses on a sensitivity approach to simulation-based methods, Chang, Fu, Hu, and Mar-

The monograph [BeS78] provides an extensive treatment of these issues, while Appendix A of the DP textbook [Ber12] provides a tutorial introduction. The followup work by Huizhen Yu and the author [YuB15] resolves the special measurability issues that relate to policy iteration, and provides additional analysis relating to value iteration. In the RL literature, the mathematical difficulties around measurability are usually neglected (as they are in the present book), and this is fine because they do not play an important role in applications. Moreover, measurability issues do not arise for problems involving finite or countably infinite state and control spaces. We note, however, that there are quite a few published works in RL as well as exact DP, which purport to address measurability issues with a mathematical narrative that is either confusing or plain incorrect.

cus [CFH13] (a 2nd edition of their 2007 monograph), which emphasizes finite-horizon/multistep lookahead schemes and adaptive sampling, Busoniu, Babuska, De Schutter, and Ernst [BBD10a], which focuses on function approximation methods for continuous space systems and includes a discussion of random search methods, Szepesvari [Sze10], which is a short monograph that selectively treats some of the major RL algorithms such as temporal difference methods, armed bandit methods, and Q-learning, Powell [Pow11], which emphasizes resource allocation and operations research applications, Powell and Ryzhov [PoR12], which focuses on specialized topics in learning and Bayesian optimization, Vrabie, Vamvoudakis, and Lewis [VVL13], which discusses neural network-based methods and on-line adaptive control, Kochenderfer et al. [KAC15], which selectively discusses applications and approximations in DP, and the treatment of uncertainty, Jiang and Jiang [JiJ17], which addresses adaptive control and robustness issues within an approximate DP framework, Liu, Wei, Wang, Yang, and Li [LWW17], which deals with forms of adaptive dynamic programming, and topics in both RL and optimal control, and Zoppoli, Sanguineti, Gnecco, and Parisini [ZSG20], which addresses neural network approximations in optimal control as well as multiagent/team problems with nonclassical information patterns.

There are also several books that, while not exclusively focused on DP and/or RL, touch upon several of the topics of this book. The book by Borkar [Bor08] is an advanced monograph that addresses rigorously many of the convergence issues of iterative stochastic algorithms in approximate DP, mainly using the so called ODE approach. The book by Meyn [Mey07] is broader in its coverage, but discusses some of the popular approximate DP/RL algorithms. The book by Haykin [Hay08] discusses approximate DP in the broader context of neural network-related subjects. The book by Krishnamurthy [Kri16] focuses on partial state information problems, with discussion of both exact DP, and approximate DP/RL methods. The textbooks by Kouvaritakis and Cannon [KoC16], Borrelli, Bemporad, and Morari [BBM17], and Rawlings, Mayne, and Diehl [RMD17] collectively provide a comprehensive view of the MPC methodology. The book by Lattimore and Szepesvari [LaS20] is focused on multiarmed bandit methods. The book by Brandimarte [Bra21] is a tutorial introduction to DP/RL that emphasizes operations research applications and includes MATLAB codes. The book by Hardt and Recht [HaR21] focuses on broader subjects of machine learning, but covers selectively approximate DP and RL topics as well.

The present book is similar in style, terminology, and notation to the author's recent RL textbooks [Ber19a], [Ber20a], and the 3rd edition of the abstract DP monograph [Ber22a], which collectively provide a fairly comprehensive account of the subject. In particular, the 2019 RL textbook includes a broader coverage of approximation in value space methods, including certainty equivalent control and aggregation methods. It

also covers substantially policy gradient methods for approximation in policy space, which we will not address here. The 2020 book focuses more closely on rollout, policy iteration, and multiagent problems. The abstract DP monograph [Ber22a] is an advanced treatment of exact DP, which also connects with intuitive visualizations of Bellman's equation and related algorithms. The present book is less mathematical and more conceptual in character. It focuses on the connection of approximation in value space with Newton's method, relying on analysis first provided in the book [Ber20a] and the paper [Ber22b], as well as on visualizations of abstract DP ideas from the book [Ber22a].

**Surveys and Short Research Monographs**

In addition to textbooks, there are many surveys and short research monographs relating to our subject, which are rapidly multiplying in number. Influential early surveys were written, from an artificial intelligence viewpoint, by Barto, Bradtke, and Singh [BBS95] (which dealt with the methodologies of real-time DP and its antecedent, real-time heuristic search [Kor90], and the use of asynchronous DP ideas [Ber82], [Ber83], [BeT89] within their context), and by Kaelbling, Littman, and Moore [KLM96] (which focused on general principles of RL). The volume by White and Sofge [WhS92] also contains several surveys describing early work in the field.

Several overview papers in the volume by Si, Barto, Powell, and Wunsch [SBP04] describe some approximation methods that we will not be covering in much detail in this book: linear programming approaches (De Farias [DeF04]), large-scale resource allocation methods (Powell and Van Roy [PoV04]), and deterministic optimal control approaches (Ferrari and Stengel [FeS04], and Si, Yang, and Liu [SYL04]). Updated accounts of these and other related topics are given in the survey collections by Lewis, Liu, and Lendaris [LLL08], and Lewis and Liu [LeL13].

Recent extended surveys and short monographs are Borkar [Bor09] (a methodological point of view that explores connections with other Monte Carlo schemes), Lewis and Vrabie [LeV09] (a control theory point of view), Szepesvari [Sze10] (which discusses approximation in value space from a RL point of view), Deisenroth, Neumann, and Peters [DNP11], and Grondman et al. [GBL12] (which focus on policy gradient methods), Browne et al. [BPW12] (which focuses on Monte Carlo Tree Search), Mausam and Kolobov [MaK12] (which deals with Markov decision problems from an artificial intelligence viewpoint), Schmidhuber [Sch15], Arulkumaran et al. [ADB17], Li [Li17], Busoniu et al. [BDT18], the author's [Ber05] (which focuses on rollout algorithms and model predictive control), [Ber11] (which focuses on approximate policy iteration), and [Ber18b] (which focuses on aggregation methods), and Recht [Rec18] (which focuses on continuous spaces optimal control).

### Research Content of this Book

The research focus of this book is to propose and develop a new conceptual framework, which the author believes is fundamental within the context of DP-based RL methodology. This framework centers around the division of the design process of an RL scheme into the off-line training and the on-line play algorithms, and shows that these algorithms operate in synergy through the powerful mechanism of Newton's method.

The style of the book is different than the style of the author's more mathematically oriented RL books [Ber19a] and [Ber20a], and abstract DP book [Ber22a]. In particular, the present book emphasizes insights through visualization rather than rigorous proofs. At the same time, the book makes careful distinctions between provable and speculative claims. By highlighting the exceptional behavior that may occur, the book also aims to emphasize the need for serious mathematical research and experimentation into broad classes of problems, beyond the relatively well-behaved finite horizon and discounted/contractive problems.

### Book Organization

The book is structured as follows. In Chapter 2, we review the theory of classical infinite horizon optimal control problems, in order to provide some orientation and an analytical platform for what follows in subsequent chapters. In Chapter 3, we introduce an abstract DP framework that will set the stage for the conceptual and visual interpretations of approximation in value space in terms of Newton's method. In this chapter, we also present new research relating to on-line policy iteration, which aims to improve the on-line approximation in value space algorithm by using training data that is collected on-line. In Chapter 4, we illustrate our analysis within the simple and intuitive framework of linear quadratic problems, which admit visualizations through the Riccati equation operators. In Chapter 5, we discuss various issues of changing problem parameters, adaptive control, and MPC. In Chapter 6, we extend the ideas of earlier chapters to finite horizon problems and discrete optimization, with a special focus on rollout algorithms and their variations. This chapter also includes a section on approximation in value space schemes for deterministic continuous-time optimal control. Finally, in the Appendix, we outline the convergence theory of Newton's method, and explain how the theory applies to nondifferentiable fixed point problems, such as the solution of Bellman's equation in DP. We also describe how the connection with Newton's method can be used to derive new and more realistic error bounds for approximation in value space and approximate policy iteration.

# *References*


[ADB17] Arulkumaran, K., Deisenroth, M. P., Brundage, M., and Bharath, A. A., 2017. "A Brief Survey of Deep Reinforcement Learning," arXiv preprint arXiv:1708.05866.

[Arg08] Argyros, I. K., 2008. Convergence and Applications of Newton-Type Iterations, Springer, N. Y.

[AsH95] Aström, K. J., and Hagglund, T., 1995. PID Controllers: Theory, Design, and Tuning, Instrument Society of America, Research Triangle Park, NC.

[AsH06] Aström, K. J., and Hagglund, T., 2006. Advanced PID Control, Instrument Society of America, Research Triangle Park, N. C.

[AsW08] Aström, K. J., and Wittenmark, B., 2008. Adaptive Control, Dover Books; also Prentice-Hall, Englewood Cliffs, N. J, 1994.

[BBB22] Bhambri, S., Bhattacharjee, A., and Bertsekas, D. P., 2022. "Reinforcement Learning Methods for Wordle: A POMDP/Adaptive Control Approach," arXiv preprint arXiv:2211.10298.

[BBD10] Busoniu, L., Babuska, R., De Schutter, B., and Ernst, D., 2010. Reinforcement Learning and Dynamic Programming Using Function Approximators, CRC Press, N. Y.

[BBM17] Borrelli, F., Bemporad, A., and Morari, M., 2017. Predictive Control for Linear and Hybrid Systems, Cambridge Univ. Press, Cambridge, UK.

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

[BDL09] Bolte, J., Daniilidis, A., and Lewis, A., 2009. "Tame Functions are Semismooth," Math. Programming, Vol. 117, pp. 5-19.

[BDT18] Busoniu, L., de Bruin, T., Tolic, D., Kober, J., and Palunko, I., 2018. "Reinforcement Learning for Control: Performance, Stability, and Deep Approximators," Annual Reviews in Control, Vol. 46, pp. 8-28.

[BKB20] Bhattacharya, S., Kailas, S., Badyal, S., Gil, S., and Bertsekas, D. P., 2020. "Multiagent Rollout and Policy Iteration for POMDP with Application to Multi-Robot Repair Problems," in Proc. of Conference on Robot Learning (CoRL); also arXiv preprint, arXiv:2011.04222.

[BLW91] Bittanti, S., Laub, A. J., and Willems, J. C., eds., 2012. The Riccati Equation, Springer.

[BMZ09] Bokanowski, O., Maroso, S., and Zidani, H., 2009. "Some Convergence Results for Howard's Algorithm," SIAM J. on Numerical Analysis, Vol. 47, pp. 3001-3026.

[BPW12] Browne, C., Powley, E., Whitehouse, D., Lucas, L., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S., 2012. "A Survey of Monte Carlo Tree Search Methods," IEEE Trans. on Computational Intelligence and AI in Games, Vol. 4, pp. 1-43.

[BSW99] Beard, R. W., Saridis, G. N., and Wen, J. T., 1998. "Approximate Solutions to the Time-Invariant Hamilton-Jacobi-Bellman Equation," J. of Optimization Theory and Applications, Vol. 96, pp. 589-626.

[BTW97] Bertsekas, D. P., Tsitsiklis, J. N., and Wu, C., 1997. "Rollout Algorithms for Combinatorial Optimization," Heuristics, Vol. 3, pp. 245-262.

[BeC99] Bertsekas, D. P., and Castañon, D. A., 1999. "Rollout Algorithms for Stochastic Scheduling Problems," Heuristics, Vol. 5, pp. 89-108.

[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, MIT, Cambridge, MA.

[BeK65] Bellman, R., and Kalaba, R. E., 1965. Quasilinearization and Nonlinear Boundary-Value Problems, Elsevier, N.Y.

[BeR71] Bertsekas, D. P., and Rhodes, I. B., 1971. "On the Minimax Reachability of Target Sets and Target Tubes," Automatica, Vol. 7, pp. 233-247.

[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).

[BeT89] Bertsekas, D. P., and Tsitsiklis, J. N., 1989. Parallel and Distributed Computation: Numerical Methods, Prentice-Hall, Engl. Cliffs, N. J. (can be downloaded from the author's website).

[BeT96] Bertsekas, D. P., and Tsitsiklis, J. N., 1996. Neuro-Dynamic Programming, Athena Scientific, Belmont, MA.

[BeT08] Bertsekas, D. P., and Tsitsiklis, J. N., 2008. Introduction to Probability, 2nd Edition, Athena Scientific, Belmont, MA.

[BeY10] Bertsekas, D. P., and Yu, H., 2010. "Distributed Asynchronous Policy Iteration in Dynamic Programming," Proc. of Allerton Conf. on Communication, Control and Computing, Allerton Park, Ill, pp. 1368-1374.

[BeY12] Bertsekas, D. P., and Yu, H., 2012. "Q-Learning and Enhanced Policy Iteration in Discounted Dynamic Programming," Math. of Operations Research, Vol. 37, pp. 66-94.

[BeY16] Bertsekas, D. P., and Yu, H., 2016. "Stochastic Shortest Path Problems Under Weak Conditions," Lab. for Information and Decision Systems Report LIDS-2909, Massachusetts Institute of Technology.

[Bea95] Beard, R. W., 1995. Improving the Closed-Loop Performance of Nonlinear Systems, Ph.D. Thesis, Rensselaer Polytechnic Institute.

[Bel57] Bellman, R., 1957. Dynamic Programming, Princeton University Press, Princeton, N. J.

[Ber71] Bertsekas, D. P., 1971. "Control of Uncertain Systems With a Set-Membership Description of the Uncertainty," Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA (can be downloaded from the author's website).

[Ber72] 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.

[Ber82] Bertsekas, D. P., 1982. "Distributed Dynamic Programming," IEEE Trans. Aut. Control, Vol. AC-27, pp. 610-616.

[Ber83] Bertsekas, D. P., 1983. "Asynchronous Distributed Computation of Fixed Points," Math. Programming, Vol. 27, pp. 107-120.

[Ber97] Bertsekas, D. P., 1997. "Differential Training of Rollout Policies," Proc. of the 35th Allerton Conference on Communication, Control, and Computing, Allerton Park, Ill.

[Ber05] Bertsekas, D. P., 2005. "Dynamic Programming and Suboptimal Control: A Survey from ADP to MPC," European J. of Control, Vol. 11, pp. 310-334.

[Ber11] Bertsekas, D. P., 2011. "Approximate Policy Iteration: A Survey and Some New Methods," J. of Control Theory and Applications, Vol. 9, pp. 310-335.

[Ber12] Bertsekas, D. P., 2012. Dynamic Programming and Optimal Control, Vol. II, 4th Ed., Athena Scientific, Belmont, MA.

[Ber15] Bertsekas, D. P., 2015. "Lambda-Policy Iteration: A Review and a New Implementation," arXiv preprint arXiv:1507.01029.

[Ber16] Bertsekas, D. P., 2016. Nonlinear Programming, Athena Scientific, Belmont, MA.

[Ber17a] Bertsekas, D. P., 2017. Dynamic Programming and Optimal Control, Vol. I, 4th Ed., Athena Scientific, Belmont, MA.

[Ber17b] Bertsekas, D. P., 2017. "Value and Policy Iteration in Deterministic Optimal Control and Adaptive Dynamic Programming," IEEE Trans. on Neural Networks and Learning Systems, Vol. 28, pp. 500-509.

[Ber17c] Bertsekas, D. P., 2017. "Regular Policies in Abstract Dynamic Programming," SIAM J. on Optimization, Vol. 27, pp. 1694-1727.

[Ber18a] Bertsekas, D. P., 2018. Abstract Dynamic Programming, 2nd Ed., Athena Scientific, Belmont, MA (can be downloaded from the author's website).

[Ber18b] Bertsekas, D. P., 2018. "Feature-Based Aggregation and Deep Reinforcement Learning: A Survey and Some New Implementations," Lab. for Information and Decision Systems Report, MIT; arXiv preprint arXiv:1804.04577; IEEE/CAA Journal of Automatica Sinica, Vol. 6, 2019, pp. 1-31.

[Ber18c] Bertsekas, D. P., 2018. "Biased Aggregation, Rollout, and Enhanced Policy Improvement for Reinforcement Learning," Lab. for Information and Decision Systems Report, MIT; arXiv preprint arXiv:1910.02426.

[Ber18d] Bertsekas, D. P., 2018. "Proximal Algorithms and Temporal Difference Methods for Solving Fixed Point Problems," Computational Optimization and Applications, Vol. 70, pp. 709-736.

[Ber19a] Bertsekas, D. P., 2019. Reinforcement Learning and Optimal Control, Athena Scientific, Belmont, MA.

[Ber19b] Bertsekas, D. P., 2019. "Multiagent Rollout Algorithms and Reinforcement Learning," arXiv preprint arXiv:1910.00120.

[Ber19c] Bertsekas, D. P., 2019. "Constrained Multiagent Rollout and Multidimensional Assignment with the Auction Algorithm," arXiv preprint, arxiv.org/-abs/2002.07407.

[Ber19d] Bertsekas, D. P., 2019. "Robust Shortest Path Planning and Semicontractive Dynamic Programming," Naval Research Logistics, Vol. 66, pp. 15-37.

[Ber20a] Bertsekas, D. P., 2020. Rollout, Policy Iteration, and Distributed Reinforcement Learning, Athena Scientific, Belmont, MA.

[Ber20b] Bertsekas, D. P., 2020. "Multiagent Value Iteration Algorithms in Dynamic Programming and Reinforcement Learning," Results in Control and Optimization J., Vol. 1, 2020.

[Ber21a] Bertsekas, D. P., 2021. "On-Line Policy Iteration for Infinite Horizon Dynamic Programming," arXiv preprint arXiv:2106.00746.

[Ber21b] Bertsekas, D. P., 2021. "Multiagent Reinforcement Learning: Rollout and Policy Iteration," IEEE/ CAA J. of Automatica Sinica, Vol. 8, pp. 249-271.

[Ber21c] Bertsekas, D. P., 2021. "Distributed Asynchronous Policy Iteration for Sequential Zero-Sum Games and Minimax Control," arXiv preprint arXiv:2107.-10406.

[Ber22a] Bertsekas, D. P., 2022. Abstract Dynamic Programming, 3rd Edition, Athena Scientific, Belmont, MA (can be downloaded from the author's website).

[Ber22b] Bertsekas, D. P., 2022. "Newton's Method for Reinforcement Learning and Model Predictive Control," Results in Control and Optimization J., Vol. 7, 2022, pp. 100-121.

[Ber22c] Bertsekas, D. P., 2022. "Rollout Algorithms and Approximate Dynamic Programming for Bayesian Optimization and Sequential Estimation," arXiv preprint arXiv:2212.07998v3.

[Bit91] Bittanti, S., 1991. "Count Riccati and the Early Days of the Riccati Equation," in The Riccati Equation (pp. 1-10), Springer.

[Bla65] Blackwell, D., 1965. "Positive Dynamic Programming," Proc. Fifth Berkeley Symposium Math. Statistics and Probability, pp. 415-418.

[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.

[Bod20] Bodson, M., 2020. Adaptive Estimation and Control, Independently Published.

[Bor08] Borkar, V. S., 2008. Stochastic Approximation: A Dynamical Systems Viewpoint, Cambridge Univ. Press.

[Bor09] Borkar, V. S., 2009. "Reinforcement Learning: A Bridge Between Numerical Methods and Monte Carlo," in World Scientific Review, Vol. 9, Ch. 4.

[Bra21] Brandimarte, P., 2021. From Shortest Paths to Reinforcement Learning: A MATLAB-Based Tutorial on Dynamic Programming, Springer.

[CFH13] Chang, H. S., Hu, J., Fu, M. C., and Marcus, S. I., 2013. Simulation-Based Algorithms for Markov Decision Processes, 2nd Edition, Springer, N. Y.

[CaB07] Camacho, E. F., and Bordons, C., 2007. Model Predictive Control, 2nd Ed., Springer, New York, N. Y.

[Cao07] Cao, X. R., 2007. Stochastic Learning and Optimization: A Sensitivity-Based Approach, Springer, N. Y.

[DNP11] Deisenroth, M. P., Neumann, G., and Peters, J., 2011. "A Survey on Policy Search for Robotics," Foundations and Trends in Robotics, Vol. 2, pp. 1-142.

[DeF04] De Farias, D. P., 2004. "The Linear Programming Approach to Approxi-

mate Dynamic Programming," in Learning and Approximate Dynamic Programming, by J. Si, A. Barto, W. Powell, and D. Wunsch, (Eds.), IEEE Press, N. Y.

[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.

[DoR14] Dontchev, A. L., and Rockafellar, R. T., 2014. Implicit Functions and Solution Mappings, 2nd Edition, Springer, N. Y.

[FaP03] Facchinei, F., and Pang, J.-S., 2003. Finite-Dimensional Variational Inequalities and Complementarity Problems, Vols I and II, Springer, N. Y.

[FeS04] Ferrari, S., and Stengel, R. F., 2004. "Model-Based Adaptive Critic Designs," in Learning and Approximate Dynamic Programming, by J. Si, A. Barto, W. Powell, and D. Wunsch, (Eds.), IEEE Press, N. Y.

[GBL12] Grondman, I., Busoniu, L., Lopes, G. A. D., and Babuska, R., 2012. "A Survey of Actor-Critic Reinforcement Learning: Standard and Natural Policy Gradients," IEEE Trans. on Systems, Man, and Cybernetics, Part C, Vol. 42, pp. 1291-1307.

[GSD06] Goodwin, G., Seron, M. M., and De Dona, J. A., 2006. Constrained Control and Estimation: An Optimisation Approach, Springer, N. Y.

[GoS84] Goodwin, G. C., and Sin, K. S. S., 1984. Adaptive Filtering, Prediction, and Control, Prentice-Hall, Englewood Cliffs, N. J.

[Gos15] Gosavi, A., 2015. Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning, 2nd Edition, Springer, N. Y.

[HWL21] Ha, M., Wang, D., and Liu, D., 2021. "Offline and Online Adaptive Critic Control Designs With Stability Guarantee Through Value Iteration," IEEE Transactions on Cybernetics.

[HaR21] Hardt, M.. and Recht, B., 2021. Patterns, Predictions, and Actions: A Story About Machine Learning, arXiv preprint arXiv:2102.05242.

[Hay08] Haykin, S., 2008. Neural Networks and Learning Machines, 3rd Ed., Prentice-Hall, Englewood-Cliffs, N. J.

[Hew71] Hewer, G., 1971. "An Iterative Technique for the Computation of the Steady State Gains for the Discrete Optimal Regulator," IEEE Trans. on Automatic Control, Vol. 16, pp. 382-384.

[Hey17] Heydari, A., 2017. "Stability Analysis of Optimal Adaptive Control Under Value Iteration Using a Stabilizing Initial Policy," IEEE Trans. on Neural Networks and Learning Systems, Vol. 29, pp. 4522-4527.

[Hey18] Heydari, A., 2018. "Stability Analysis of Optimal Adaptive Control Using Value Iteration with Approximation Errors," IEEE Transactions on Automatic Control, Vol. 63, pp. 3119-3126.

[Hyl11] Hylla, T., 2011. Extension of Inexact Kleinman-Newton Methods to a General Monotonicity Preserving Convergence Theory, Ph.D. Thesis, Univ. of Trier.

[IoS96] Ioannou, P. A., and Sun, J., 1996. Robust Adaptive Control, Prentice-Hall, Englewood Cliffs, N. J.

[ItK03] Ito, K., and Kunisch, K., 2003. "Semi-Smooth Newton Methods for Variational Inequalities of the First Kind," Mathematical Modelling and Numerical Analysis, Vol. 37, pp. 41-62.

[JiJ17] Jiang, Y., and Jiang, Z. P., 2017. Robust Adaptive Dynamic Programming, J. Wiley, N. Y.

[Jos79] Josephy, N. H., 1979. "Newton's Method for Generalized Equations," Wisconsin Univ-Madison, Mathematics Research Center Report No. 1965.

[KAC15] Kochenderfer, M. J., with Amato, C., Chowdhary, G., How, J. P., Davison Reynolds, H. J., Thornton, J. R., Torres-Carrasquillo, P. A., Ore, N. K., Vian, J., 2015. Decision Making under Uncertainty: Theory and Application, MIT Press, Cambridge, MA.

[KKK95] Krstic, M., Kanellakopoulos, I., Kokotovic, P., 1995. Nonlinear and Adaptive Control Design, J. Wiley, N. Y.

[KKK20] Kalise, D., Kundu, S., and Kunisch, K., 2020. "Robust Feedback Control of Nonlinear PDEs by Numerical Approximation of High-Dimensional Hamilton-Jacobi-Isaacs Equations." SIAM J. on Applied Dynamical Systems, Vol. 19, pp. 1496-1524.

[KLM96] Kaelbling, L. P., Littman, M. L., and Moore, A. W., 1996. "Reinforcement Learning: A Survey," J. of Artificial Intelligence Res., Vol. 4, pp. 237-285.

[KeG88] Keerthi, S. S., and Gilbert, E. G., 1988. "Optimal Infinite-Horizon Feedback Laws for a General Class of Constrained Discrete-Time Systems: Stability and Moving-Horizon Approximations," J. Optimization Theory Appl., Vo. 57, pp. 265-293.

[Kle67] Kleinman, D. L., 1967. Suboptimal Design of Linear Regulator Systems Subject to Computer Storage Limitations, Doctoral dissertation, M.I.T., Electronic Systems Lab., Rept. 297.

[Kle68] Kleinman, D. L., 1968. "On an Iterative Technique for Riccati Equation Computations," IEEE Trans. Automatic Control, Vol. AC-13, pp. 114-115.

[KoC16] Kouvaritakis, B., and Cannon, M., 2016. Model Predictive Control: Classical, Robust and Stochastic, Springer, N. Y.

[KoS86] Kojima, M., and Shindo, S., 1986. "Extension of Newton and Quasi-Newton Methods to Systems of $PC^1$ Equations," J. of the Operations Research Society of Japan, Vol. 29, pp. 352-375.

[Kok91] Kokotovic, P. V., ed., 1991. Foundations of Adaptive Control, Springer.

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

[Kri16] Krishnamurthy, V., 2016. Partially Observed Markov Decision Processes, Cambridge Univ. Press.

[Kum88] Kummer, B., 1988. "Newton's Method for Non-Differentiable Functions," Mathematical Research, Vol. 45, pp. 114-125.

[Kum00] Kummer, B., 2000. "Generalized Newton and NCP-methods: Convergence, Regularity, Actions," Discussiones Mathematicae, Differential Inclusions, Control and Optimization, Vol. 2, pp. 209-244.

[KuK21] Kundu, S., and Kunisch, K., 2021. "Policy Iteration for Hamilton-Jacobi-Bellman Equations with Control Constraints," Computational Optimization and Applications, pp. 1-25.

[KuV86] Kumar, P. R., and Varaiya, P. P., 1986. Stochastic Systems: Estimation, Identification, and Adaptive Control, Prentice-Hall, Englewood Cliffs, N. J.

[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.

[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.

[LAM21] Lopez, V. G., Alsalti, M., and Muller, M. A., 2021. "Efficient Off-Policy Q-Learning for Data-Based Discrete-Time LQR Problems," arXiv preprint arXiv:2105.07761.

[LJM21] Li, Y., Johansson, K. H., Martensson, J., and Bertsekas, D. P., 2021. "Data-Driven Rollout for Deterministic Optimal Control," arXiv preprint arXiv:-2105.03116.

[LLL08] Lewis, F. L., Liu, D., and Lendaris, G. G., 2008. Special Issue on Adaptive Dynamic Programming and Reinforcement Learning in Feedback Control, IEEE Trans. on Systems, Man, and Cybernetics, Part B, Vol. 38, No. 4.

[LPS21] Liu, M., Pedrielli, G., Sulc, P., Poppleton, E., and Bertsekas, D. P., 2021. "ExpertRNA: A New Framework for RNA Structure Prediction," bioRxiv 2021.01.18.427087; to appear in INFORMS J. on Computing.

[LWW17] Liu, D., Wei, Q., Wang, D., Yang, X., and Li, H., 2017. Adaptive Dynamic Programming with Applications in Optimal Control, Springer, Berlin.

[LXZ21] Liu, D., Xue, S., Zhao, B., Luo, B., and Wei, Q., 2021. "Adaptive Dynamic Programming for Control: A Survey and Recent Advances," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 51, pp. 142-160.

[LaR95] Lancaster, P., and Rodman, L., 1995. Algebraic Riccati Equations, Clarendon Press.

[LaS20] Lattimore, T., and Szepesvari, C., 2020. Bandit Algorithms, Cambridge Univ. Press.

[LaW13] Lavretsky, E., and Wise, K., 2013. Robust and Adaptive Control with Aerospace Applications, Springer.

[LeL13] Lewis, F. L., and Liu, D., (Eds), 2013. Reinforcement Learning and Approximate Dynamic Programming for Feedback Control, Wiley, Hoboken, N. J.

[LeV09] Lewis, F. L., and Vrabie, D., 2009. "Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control," IEEE Circuits and Systems Magazine, 3rd Q. Issue.

[Li17] Li, Y., 2017. "Deep Reinforcement Learning: An Overview," arXiv preprint ArXiv: 1701.07274v5.

[MDM01] Magni, L., De Nicolao, G., Magnani, L., and Scattolini, R., 2001. "A Stabilizing Model-Based Predictive Control Algorithm for Nonlinear Systems," Automatica, Vol. 37, pp. 1351-1362.

[MKH10] Mayer, J., Khairy, K., and Howard, J., 2010. "Drawing an Elephant with Four Complex Parameters," American Journal of Physics, Vol. 78, pp. 648-649.

[MRR00] Mayne, D., 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.

[MVB20] Magirou, E. F., Vassalos, P., and Barakitis, N., 2020. "A Policy Iteration Algorithm for the American Put Option and Free Boundary Control

Problems," J. of Computational and Applied Mathematics, vol. 373, p. 112544.

[MaK12] Mausam, and Kolobov, A., 2012. "Planning with Markov Decision Processes: An AI Perspective," Synthesis Lectures on Artificial Intelligence and Machine Learning, Vol. 6, pp. 1-210.

[Mac02] Maciejowski, J. M., 2002. Predictive Control with Constraints, Addison-Wesley, Reading, MA.

[Man74] Mandl, P., 1974. "Estimation and Control in Markov Chains," Advances in Applied Probability, Vol. 6, pp. 40-60.

[May14] Mayne, D. Q., 2014. "Model Predictive Control: Recent Developments and Future Promise," Automatica, Vol. 50, pp. 2967-2986.

[Mes16] Mesbah, A., 2016. Stochastic Model Predictive Control: An Overview and Perspectives for Future Research," IEEE Control Systems Magazine, Vol. 36, pp. 30-44.

[Mey07] Meyn, S., 2007. Control Techniques for Complex Networks, Cambridge Univ. Press, N. Y.

[NaA12] Narendra, K. S., and Annaswamy, A. M., 2012. Stable Adaptive Systems, Courier Corporation.

[OrR67] Ortega, J. M., and Rheinboldt, W. C., 1967. "Monotone Iterations for Nonlinear Equations with Application to Gauss-Seidel Methods," SIAM J. on Numerical Analysis, Vol. 4, pp. 171-190.

[OrR70] Ortega, J. M., and Rheinboldt, W. C., 1970. Iterative Solution of Nonlinear Equations in Several Variables, Academic Press; republished in 2000 by the Society for Industrial and Applied Mathematics.

[PaJ21] Pang, B., and Jiang, Z. P., 2021. "Robust Reinforcement Learning: A Case Study in Linear Quadratic Regulation," arXiv preprint arXiv:2008.11592v3.

[Pan90] Pang, J. S., 1990. "Newton's Method for B-Differentiable Equations," Math. of Operations Research, Vol. 15, pp. 311-341.

[PoA69] Pollatschek, M., and Avi-Itzhak, B., 1969. "Algorithms for Stochastic Games with Geometrical Interpretation," Management Science, Vol. 15, pp. 399-413.

[PoR12] Powell, W. B., and Ryzhov, I. O., 2012. Optimal Learning, J. Wiley, N. Y.

[PoV04] Powell, W. B., and Van Roy, B., 2004. "Approximate Dynamic Programming for High-Dimensional Resource Allocation Problems," in Learning and Approximate Dynamic Programming, by J. Si, A. Barto, W. Powell, and D. Wunsch, (Eds.), IEEE Press, N. Y.

[Pow11] Powell, W. B., 2011. Approximate Dynamic Programming: Solving the Curses of Dimensionality, 2nd Edition, J. Wiley and Sons, Hoboken, N. J.

[PuB78] Puterman, M. L., and Brumelle, S. L., 1978. "The Analytic Theory of Policy Iteration," in Dynamic Programming and Its Applications, M. L. Puterman (ed.), Academic Press, N. Y.

[PuB79] Puterman, M. L., and Brumelle, S. L., 1979. "On the Convergence of Policy Iteration in Stationary Dynamic Programming," Math. of Operations Research, Vol. 4, pp. 60-69.

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

[Qi93] Qi, L., 1993. "Convergence Analysis of Some Algorithms for Solving Nonsmooth Equations," Math. of Operations Research, Vol. 18, pp. 227-244.

[QiS93] Qi, L., and Sun, J., 1993. "A Nonsmooth Version of Newton's Method," Math. Programming, Vol. 58, pp. 353-367.

[RMD17] Rawlings, J. B., Mayne, D. Q., and Diehl, M. M., 2017. Model Predictive Control: Theory, Computation, and Design, 2nd Ed., Nob Hill Publishing (updated in 2019 and 2020).

[Rec18] Recht, B., 2018. "A Tour of Reinforcement Learning: The View from Continuous Control," Annual Review of Control, Robotics, and Autonomous Systems.

[RoB17] Rosolia, U., and Borrelli, F., 2017. "Learning Model Predictive Control for Iterative Tasks. A Data-Driven Control Framework," IEEE Trans. on Automatic Control, Vol. 63, pp. 1883-1896.

[RoB19] Rosolia, U., and Borrelli, F., 2019. "Sample-Based Learning Model Predictive Control for Linear Uncertain Systems," 58th Conference on Decision and Control (CDC), pp. 2702-2707.

[Rob80] Robinson, S. M., 1980. "Strongly Regular Generalized Equations," Math. of Operations Research, Vol. 5, pp. 43-62.

[Rob88] Robinson, S. M., 1988. "Newton's Method for a Class of Nonsmooth Functions," Industrial Engineering Working Paper, University of Wisconsin; also in Set-Valued Analysis Vol. 2, 1994, pp. 291-305.

[Rob11] Robinson, S. M., 2011. "A Point-of-Attraction Result for Newton's Method with Point-Based Approximations," Optimization, Vol. 60, pp. 89-99.

[SGG15] Scherrer, B., Ghavamzadeh, M., Gabillon, V., Lesner, B., and Geist, M., 2015. "Approximate Modified Policy Iteration and its Application to the Game of Tetris," J. of Machine Learning Research, Vol. 16, pp. 1629-1676.

[SBP04] Si, J., Barto, A., Powell, W., and Wunsch, D., (Eds.) 2004. Learning and Approximate Dynamic Programming, IEEE Press, N. Y.

[SHM16] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., and Dieleman, S., 2016. "Mastering the Game of Go with Deep Neural Networks and Tree Search," Nature, Vol. 529, pp. 484-489.

[SHS17] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., and Lillicrap, T., 2017. "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," arXiv preprint arXiv:1712.01815.

[SSS17] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., and Chen, Y., 2017. "Mastering the Game of Go Without Human Knowledge," Nature, Vol. 550, pp. 354-359.

[SYL04] Si, J., Yang, L., and Liu, D., 2004. "Direct Neural Dynamic Programming," in Learning and Approximate Dynamic Programming, by J. Si, A. Barto, W. Powell, and D. Wunsch, (Eds.), IEEE Press, N. Y.

[SaB11] Sastry, S., and Bodson, M., 2011. Adaptive Control: Stability, Convergence and Robustness, Courier Corporation.

[SaL79] Saridis, G. N., and Lee, C.-S. G., 1979. "An Approximation Theory of Optimal Control for Trainable Manipulators," IEEE Trans. Syst., Man, Cybernetics, Vol. 9, pp. 152-159.

[SaR04] Santos, M. S., and Rust, J., 2004. "Convergence Properties of Policy Iteration," SIAM J. on Control and Optimization, Vol. 42, pp. 2094-2115.

[Sch15] Schmidhuber, J., 2015. "Deep Learning in Neural Networks: An Overview," Neural Networks, pp. 85-117.

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

[SlL91] Slotine, J.-J. E., and Li, W., Applied Nonlinear Control, Prentice-Hall, Englewood Cliffs, N. J.

[Str66] Strauch, R., 1966. "Negative Dynamic Programming," Ann. Math. Statist., Vol. 37, pp. 871-890.

[SuB18] Sutton, R., and Barto, A. G., 2018. Reinforcement Learning, 2nd Ed., MIT Press, Cambridge, MA.

[Sze10] Szepesvari, C., 2010. Algorithms for Reinforcement Learning, Morgan and Claypool Publishers, San Franscisco, CA.

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

[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.

[TsV96] Tsitsiklis, J. N., and Van Roy, B., 1996. "Feature-Based Methods for Large-Scale Dynamic Programming," Machine Learning, Vol. 22, pp. 59-94.

[VVL13] Vrabie, D., Vamvoudakis, K. G., and Lewis, F. L., 2013. Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles, The Institution of Engineering and Technology, London.

[Van67] Vandergraft, J. S., 1967. "Newton's Method for Convex Operators in Partially Ordered Spaces," SIAM J. on Numerical Analysis, Vol. 4, pp. 406-432.

[Van78] van der Wal, J., 1978. "Discounted Markov Games: Generalized Policy Iteration Method," J. of Optimization Theory and Applications, Vol. 25, pp. 125-138.

[WLL16] Wei, Q., Liu, D., and Lin, H., 2016. "Value Iteration Adaptive Dynamic Programming for Optimal Control of Discrete-Time Nonlinear Systems," IEEE Transactions on Cybernetics, Vol. 46, pp. 840-853.

[WLL21] Winnicki, A., Lubars, J., Livesay, M., and Srikant, R., 2021. "The Role of Lookahead and Approximate Policy Evaluation in Policy Iteration with Linear Value Function Approximation," arXiv preprint arXiv:2109.13419.

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

[YuB13] Yu, H., and Bertsekas, D. P., 2013. "Q-Learning and Policy Iteration Algorithms for Stochastic Shortest Path Problems," Annals of Operations Research, Vol. 208, pp. 95-132.

[YuB15] Yu, H., and Bertsekas, D. P., 2015. "A Mixed Value and Policy Iteration Method for Stochastic Control with Universally Measurable Policies," Math. of OR, Vol. 40, pp. 926-968.

[ZSG20] Zoppoli, R., Sanguineti, M., Gnecco, G., and Parisini, T., 2020. Neural Approximations for Optimal Control and Decision, Springer.

**Neuro-Dynamic Programming**
**Dimitri P. Bertsekas and John N. Tsitsiklis**

**Athena Scientific, 1996**
**512 pp., hardcover, ISBN 1-886529-10-8**

This is the first textbook that fully explains the neuro-dynamic programming/reinforcement learning methodology, a breakthrough in the practical application of neural networks and dynamic programming to complex problems of planning, optimal decision making, and intelligent control.

**From the review** by George Cybenko for IEEE Computational Science and Engineering, May 1998:

"Neuro-Dynamic Programming is a remarkable monograph that integrates a sweeping mathematical and computational landscape into a coherent body of rigorous knowledge. The topics are current, the writing is clear and to the point, the examples are comprehensive and the historical notes and comments are scholarly."

"In this monograph, Bertsekas and Tsitsiklis have performed a Herculean task that will be studied and appreciated by generations to come. I strongly recommend it to scientists and engineers eager to seriously understand the mathematics and computations behind modern behavioral machine learning."

Among its special features, the book:

- Describes and unifies a large number of NDP methods, including several that are new
- Describes new approaches to formulation and solution of important problems in stochastic optimal control, sequential decision making, and discrete optimization
- Rigorously explains the mathematical principles behind NDP
- Illustrates through examples and case studies the practical application of NDP to complex problems from optimal resource allocation, optimal feedback control, data communications, game playing, and combinatorial optimization
- Presents extensive background and new research material on dynamic programming and neural network training

**Neuro-Dynamic Programming is the winner of the 1997 INFORMS CSTS prize for research excellence in the interface between Operations Research and Computer Science**

**Reinforcement Learning and Optimal Control**
**Dimitri P. Bertsekas**

**Athena Scientific, 2019**
**388 pp., hardcover, ISBN 978-1-886529-39-7**

This book explores the common boundary between optimal control and artificial intelligence, as it relates to reinforcement learning and simulation-based neural network methods. These are popular fields with many applications, which can provide approximate solutions to challenging sequential decision problems and large-scale dynamic programming (DP). The aim of the book is to organize coherently the broad mosaic of methods in these fields, which have a solid analytical and logical foundation, and have also proved successful in practice.

The book discusses both approximation in value space and approximation in policy space. It adopts a gradual expository approach, which proceeds along four directions:

- From exact DP to approximate DP: We first discuss exact DP algorithms, explain why they may be difficult to implement, and then use them as the basis for approximations.
- From finite horizon to infinite horizon problems: We first discuss finite horizon exact and approximate DP methodologies, which are intuitive and mathematically simple, and then progress to infinite horizon problems.
- From model-based to model-free implementations: We first discuss model-based implementations, and then we identify schemes that can be appropriately modified to work with a simulator.

The mathematical style of this book is somewhat different from the one of the author's DP books, and the 1996 neuro-dynamic programming (NDP) research monograph, written jointly with John Tsitsiklis. While we provide a rigorous, albeit short, mathematical account of the theory of finite and infinite horizon DP, and some fundamental approximation methods, we rely more on intuitive explanations and less on proof-based insights. Moreover, our mathematical requirements are quite modest: calculus, a minimal use of matrix-vector algebra, and elementary probability (mathematically complicated arguments involving laws of large numbers and stochastic convergence are bypassed in favor of intuitive explanations).

The book is supported by on-line video lectures and slides, as well as new research material, some of which has been covered in the present monograph.

**Rollout, Policy Iteration, and Distributed Reinforcement Learning**

**Dimitri P. Bertsekas**

**Athena Scientific, 2020**
**480 pp., hardcover, ISBN 978-1-886529-07-6**

This book develops in greater depth some of the methods from the author's Reinforcement Learning and Optimal Control textbook (Athena Scientific, 2019). It presents new research, relating to rollout algorithms, policy iteration, multiagent systems, partitioned architectures, and distributed asynchronous computation.

The application of the methodology to challenging discrete optimization problems, such as routing, scheduling, assignment, and mixed integer programming, including the use of neural network approximations within these contexts, is also discussed.

Much of the new research is inspired by the remarkable AlphaZero chess program, where policy iteration, value and policy networks, approximate lookahead minimization, and parallel computation all play an important role.

Among its special features, the book:

- Presents new research relating to distributed asynchronous computation, partitioned architectures, and multiagent systems, with application to challenging large scale optimization problems, such as combinatorial/discrete optimization, as well as partially observed Markov decision problems.
- Describes variants of rollout and policy iteration for problems with a multiagent structure, which allow the dramatic reduction of the computational requirements for lookahead minimization.
- Establishes connections of rollout algorithms and model predictive control, one of the most prominent control system design methodology.
- Expands the coverage of some research areas discussed in the author's 2019 textbook Reinforcement Learning and Optimal Control.
- Provides the mathematical analysis that supports the Newton step interpretations and the conclusions of the present book.

The book is supported by on-line video lectures and slides, as well as new research material, some of which has been covered in the present monograph.