ANNUAL REVIEWS

Annual Review of Economics

Has Dynamic Programming Improved Decision Making?

John Rust

Department of Economics, Georgetown University, Washington, DC 20057, USA;
email: jr1393@georgetown.edu

Annu. Rev. Econ. 2019. 11:833–58

The *Annual Review of Economics* is online at economics.annualreviews.org

https://doi.org/10.1146/annurev-economics-080218-025721

Keywords

actor-critic algorithms, dynamic programming, artificial intelligence, behavioral economics, bounded rationality, curse of dimensionality, computational complexity, neural networks, revenue management, reinforcement learning

Abstract

Dynamic programming (DP) is a powerful tool for solving a wide class of sequential decision-making problems under uncertainty. In principle, it enables us to compute optimal decision rules that specify the best possible decision in any situation. This article reviews developments in DP and contrasts its revolutionary impact on economics, operations research, engineering, and artificial intelligence with the comparative paucity of its real-world applications to improve the decision making of individuals and firms. The fuzziness of many real-world decision problems and the difficulty in mathematically modeling them are key obstacles to a wider application of DP in real-world settings. Nevertheless, I discuss several success stories, and I conclude that DP offers substantial promise for improving decision making if we let go of the empirically untenable assumption of unbounded rationality and confront the challenging decision problems faced every day by individuals and firms.

> As new mathematical tools for computing optimal and satisfactory decisions are discovered, and as computers become more and more powerful, the recommendations of normative decision theory will change. But as the new recommendations are diffused, the actual, observed, practice of decision making in business firms will change also. And these changes may have macroeconomic consequences. For example, there is some agreement that average inventory holdings of American firms have been reduced significantly by the introduction of formal procedures for calculating reorder points and quantities.
>
> —Herbert Simon (1978, p. 351)

1. INTRODUCTION

The term dynamic programming (DP) was coined by Richard Bellman in 1950 to denote the recursive process of backward induction for finding optimal policies or decision rules for a wide class of dynamic, sequential decision-making problems under uncertainty (for details, history, and background on DP, see Rust 2008). Bellman (1984, p. 159) claimed he invented the term to hide "the fact that I was really doing mathematics inside the RAND Corporation," but the actual motivation for the development of DP was eminently practical. According to Bellman's coauthor and colleague, Stuart Dreyfus (2002, p. 48), Bellman had

> cast his lot instead with the kind of applied mathematics later to be known as operations research. In those days applied practitioners were regarded as distinctly second-class citizens of the mathematical fraternity. Always one to enjoy controversy, when invited to speak at various university mathematics department seminars, Bellman delighted in justifying his choice of applied over pure mathematics as being motivated by the real world's greater challenges and mathematical demands.

The earliest applications of DP included work by Massé (1944) (management of hydroelectric reservoirs), Arrow et al. (1951) (optimal inventory policy), and Wald (1947) (development of optimal sequential statistical decision rules). DP emerged as a fundamental tool of applied mathematics, and it revolutionized the way we do economics. It is probably the single most fundamental tool underlying game theory, macroeconomics, and microeconomics (see, e.g., Adda & Cooper 2003, Maskin & Tirole 2001, Stokey & Lucas 1989). As a result, the majority of the modern economics literature can be regarded as a type of applied DP. However, economists use DP primarily as an academic modeling tool, and compared to their colleagues in operations research and engineering, they have contributed less to the practical application of DP to improve decision and policy making. Why?

I believe this is largely due to the predominant orientation of economics as a positive (i.e., descriptive) theory of the behavior of individuals and firms under the assumption of unbounded rationality. Most economists are perhaps too comfortable with the assumption that firms maximize expected discounted profits, individuals maximize expected discounted utility, and governments behave as benevolent social planners and maximize social welfare. Most applied policy making by economists concerns fostering macroeconomic stabilization, counteracting market power, or correcting market failures such as externalities or incomplete markets. However, the prevailing view that individuals and firms are able to solve their own decision problems perfectly well leaves little motivation for economists to use DP as a normative tool to improve decision making in practice.

Simon (1978, p. 357) questioned the relevance of economists' assumption of unbounded rationality and noted that

> by the middle 1950s, a theory of bounded rationality had been proposed as an alternative to classical omniscient rationality [and] a significant number of empirical studies had been carried out that

showed actual business decision making to conform reasonably well with the assumptions of bounded rationality but not with the assumptions of perfect rationality.

Note that bounded rationality does not imply that individuals or firms behave stupidly. The opening quote from Simon suggests that he recognized the strong competitive and evolutionary advantages to making better decisions. However, my impression is that formal DP has not been widely adopted to improve decision making by individuals and firms. Why?

I believe the key explanation is the curse of dimensionality: The complexity of the real world makes it impossible for anyone to formulate and solve DP problems that can sufficiently approximate many or most of the problems individuals and firms actually confront. Bellman coined the concept of curse of dimensionality to refer to the exponential growth in computer power required to solve bigger, more realistic DP problems. The size of a DP problem can often be indexed by its dimension d, that is, the number of state and decision variables in the DP problem. A problem is subject to a curse of dimensionality if the computer time/power required to even approximately solve it increases exponentially in d. For many years it was not clear whether some amazing algorithm would be discovered that could break the curse of dimensionality, but subsequent research by computer scientists proved that the curse of dimensionality is unavoidable and cannot be broken by any algorithm.[1]

There are subclasses of DP problems that can be shown to be tractable (i.e., not subject to a curse of dimensionality) and thus solvable in polynomial time, or that have complexity bounds that do not increase exponentially in the problem dimension d. For example, linear-quadratic DP problems (see, e.g., Chow 1976) can be solved in polynomial time. Rust (1997) proved that it is possible to break the curse of dimensionality for discrete choice DP problems (i.e., DP problems with only a finite number of possible choices but with d continuous state variables). These problems can be solved using a randomized algorithm (i.e., using Monte Carlo integration), and Rust (1997) showed that the complexity of computing an approximate solution to a discrete choice DP problem with an expected maximum error of ε is bounded above by $(d/\varepsilon)^4$, so this class of problems can also be solved in polynomial time. However, Chow & Tsitsiklis (1989) proved that DP problems with continuous state and decision variables are subject to the curse of dimensionality, at least in the worst case. This implies that if individuals and firms have finite computational capacity, there is no algorithm, no matter how clever, that can a find a good approximation to the optimal DP decision rule that solves the high-dimensional decision problems that individuals and firms confront each day. Simon coined the term satisficing to refer to a range of suboptimal decision rules that individuals and firms actually adopt; some of these include rules of thumb, or what computer scientists call heuristics, that provide decision rules that are regarded as good enough in situations where nobody is capable of calculating or even closely approximating an optimal DP-based decision rule.

What room does this leave for the normative application of DP to real-world problems? As the opening quote from Simon noted, the power of digital computers has been growing exponentially. There has also been a tremendous amount of research on newer, better algorithms for solving DP problems. This implies that the set of problems that can be approximately solved using DP is steadily growing. Finally, although the decision problems that most individuals and firms confront are incredibly complex, difficult to formalize mathematically, and perhaps impossible to

[1]Traub & Werschulz (1998) provide an accessible introduction to computational complexity for continuous mathematical problems. Computer scientists define the complexity of a mathematical problem by a function comp(ε), which is the minimal time required to solve the problem with an error of at most ε in the worst case. When the lower bound on comp(ε) increases proportionately to $(1/\varepsilon)^d$, there is an inherent curse of dimensionality, and computer scientists refer to such problems as intractable.

solve numerically, there are many cases in which the principle of decomposition can be applied to identify subproblems of an overall decision problem that can be sufficiently well formalized, approximated, and solved using DP. One example is the use of GPS navigation systems to find shortest or fastest routes to a desired destination. In another example, Hall & Rust (2019) analyze the inventory investment and management decisions of a steel company that trades over 9,000 products. They apply the principle of decomposition to reduce the firm's problem to the solution of 9,000 two-dimensional DP problems (which is feasible) rather than the solution of a single $d = 18{,}000$ dimensional problem (which is essentially impossible). Even if the overall decision problem is not exactly decomposable into simpler subproblems, this method may provide a reasonable approximation and form the basis for a nearly optimal overall decision rule.

Simon (1978, p. 350) noted that "decision makers can satisfice either by finding optimum solutions for a simplified world, or by finding satisfactory solutions for a more realistic world. Neither approach, in general, dominates the other, and both have continued to coexist in the world of management science." The goal of this article is to provide a number of examples where DP has been successfully implemented and adopted in practice, as well as a number of examples where it appears that DP could help individuals and firms do better. These are all examples of Simon's first approach to satisficing, namely, the academic approach of finding optimal solutions for a simplified world.

I believe that we are on the verge of a more rapid and widespread practical application of DP due to a number of key recent developments. The most important of them is the exponential growth in power of digital computers, which combined with vastly improved electronic communications has radically changed our daily lives. After a long gestation lag (digital computers and DP both emerged in the early 1950s), and after a fair amount of early hype and a number of unsuccessful efforts to develop artificial intelligence (AI), we are now witnessing the power of DP to improve decision making, especially when it is combined with related tools from statistics and computer science such as machine learning (ML) and reinforcement learning (RL). The most clear-cut successes for DP/AI-inspired algorithms are quite recent and are limited to relatively narrow domains such as board games (chess, Go, shogi, etc.) in which DP-inspired algorithms have been able to achieve superhuman levels of performance.

For example, a team at DeepMind (Silver et al. 2017, p. 358) developed the AlphaGo Zero algorithm for playing the game of Go and concluded,

> Our results comprehensively demonstrate that a pure RL approach is fully feasible, even in the most challenging of domains: it is possible to train to superhuman level, without human examples or guidance, given no knowledge of the domain beyond basic rules. Furthermore, a pure RL approach requires just a few more hours to train, and achieves much better asymptotic performance, compared to training on human expert data.

Silver (2017) reports similar successes for the AlphaZero algorithm for chess and shogi in addition to Go, which are remarkable due to the vast number of possible states (board positions) in chess (4.5×10^{46}) and Go (2.08×10^{170}). These successes and rapid progress in other areas such as robotics may be the reason the late Stephen Hawking warned that "the development of full artificial intelligence could spell the end of the human race" (quoted in Cellan-Jones 2014).

Hawking's concern may be overblown, at least in the short run. When we look at broader, less well-defined domains for decision making, such as running a company or common life decisions such as choosing a career or a spouse, it is currently hard to imagine any algorithm—even one that is based on DP—that we would entrust to make better decisions than we would make ourselves. For the foreseeable future, the human neural network seems much better than any artificial neural network in fuzzy situations that require weighing many hard-to-quantify subjective factors

and considerations. Following Simon and the literatures on bounded rationality and behavioral economics, I find little evidence to support the standard assumption of unbounded rationality, i.e., that all individuals and firms have unbounded levels of knowledge, rationality, and computational ability and can be modeled as perfect utility or profit maximizers. The preponderance of evidence suggests that individuals and firms behave suboptimally and sometimes make bad decisions that can have serious long-term consequences, such as panicking and liquidating one's stock portfolio at the bottom of a temporary stock market crash. [Nutt (2002) provides a catalog of decision debacles by firms and government organizations.] Insufficient knowledge, emotion, and cognitive bias (see, e.g., Kahneman 2011) as well as limited reasoning/computational capacity can cause us to make suboptimal choices, creating a role for decision support tools such as DP that can provide recommended decisions that serve as intelligent defaults or suggestions that can be ignored, similar to the idea of a nudge by Thaler & Sunstein (2008).

The most promising domain for application of DP in the short run is to improve firm decision making, particularly in *pricing and revenue management*, where there is evidence that computer-generated recommended prices have significantly increased revenues and profits. As a result, there has been rapid entry and growth in this industry: A report by MarketsandMarkets (2016) forecasts it will grow at nearly 19% per year, from $9.7 billion in 2015 to $21.9 billion in 2020. I provide other examples where DP appears to have great promise to help firms solve some of their easier, more tractable subproblems and become more profitable. Thus, although there is indeed a curse of dimensionality and many, if not most, problems confronting individuals and firms are still beyond our ability to formulate and solve, the rapid improvement in computer hardware, software, and algorithms makes it reasonable to predict a steady growth in the practical applications of DP in the future. This is especially true in view of the rapid growth of *business analytics* software and services, which according to the INFORMS publication Analytics is growing at nearly 10% per year (MarketsandMarkets 2016).

In Section 2 I summarize the main numerical methods for solving DP problems, focusing on algorithms that are less familiar to economists but are well known in the AI and RL literatures. Knowledge of the strengths and weaknesses of different numerical methods and different methods for learning and approximating the objective function of the decision maker (DM) and the law of motion for the state variables affecting payoffs is key to understanding the different ways DP has been implemented in practice. In Section 3 I discuss several success stories concerning the application of DP by firms, as well as a number of additional academic studies that suggest that DP-based strategies for some key decision subproblems may help firms increase profits. These results call into question the standard economic assumption of discounted expected profit maximization by unboundedly rational firms. In Section 4 I present some conclusions and suggest directions and questions for future research in this area. My overall conclusion is that although DP does not appear to be widely applied in practice so far, we are likely to see rapid growth in its application in future years, which is likely to propel a more rational, science-based approach to decision making by firms. DP also has the potential to help improve individual decision making, and hopefully government decision making as well.

2. MACHINE- VERSUS HUMAN-LEARNING APPROACHES TO SOLVING DP PROBLEMS

Virtually all realistic DP problems that are used in practical applications need to be solved numerically. In this section I provide an overview of the main approaches for solving DP problems: standard approaches that require a high degree of human input and programming, and ML/RL algorithms that can learn optimal solutions through experience and trial and error

experimentation. (Readers are referred to Rust 2017 for a recent survey on numerical solution of DPs with further references and examples.)

There are two senses of the term learning used in the AI and DP literatures: learning (i.e., approximating) the optimal decision rule, and learning the problem structure—that is, the DM's reward/objective function and the laws of motion governing state variables in the DP problem as well as how rewards and states are affected (often probabilistically) by decisions. In strategic situations such as a dynamic game, the DP solution typically also requires learning the decision rules used by other DMs, which may take the form of probability distributions over possible moves (as in the case of board games such as chess or Go). The DP and control literatures use the term adaptive control to refer to situations where there is incomplete knowledge of the structure of the DP problem, and the DM attempts to learn this structure while also trying to make optimal decisions, leading to the classic trade-off between experimentation and exploitation that is captured in problems such as the multiarmed bandit problem (Gittins 1989).

Although the literatures on DP and AI originated around the same time,[2] and both depend on the exponential growth in speed of digital computers, the earliest work on the modeling and design of algorithms to solve a DP was fundamentally an outcome of human learning that required human programming. The more recent ML, by contrast, focuses on developing algorithms "to give computers the ability to 'learn' (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed" (Wikipedia 2018). A key distinction in the literature is whether learning takes place off-line (i.e., the underlying structure and approximate solutions are computed before the decision rule is used to make or recommend decisions) or in real time (i.e., the algorithm continuously updates its estimate of the problem structure and the optimal decision rule while decisions are being taken). Some of the real-time learning algorithms also have the advantage of being model free—that is, they only require the DM to observe realizations of rewards (which can be obtained from the real-time application of the decision rule or via sufficiently realistic computer simulations) rather than requiring specific functional forms for the underlying preferences, beliefs, and laws of motion that constitute the structure of the underlying decision problem. Another key distinction is whether the learning is supervised or unsupervised. Supervised learning presumes the existence of an expert DM whose behavior can be observed and used to train a decision rule, either by simple extrapolation or by structural estimation methods that infer the preferences and law of motion for state variables (beliefs) from the observed behavior of expert (presumably human) DMs. Unsupervised learning such as RL "comes into play when examples of desired behavior are not available but where it is possible to score examples of behavior according to some performance criterion" (Barto & Dietterich 2004, p. 50).

2.1. Standard DP Algorithms: Off-Line Solution Using Human Learning

Though DP can solve problems that are not directly intertemporal decision problems (e.g., shortest-path problems) as well as intertemporal decision problems with objective functions that are nonseparable functions of the state and decision variables, the most common applications of DP are for intertemporal decision problems that can be formulated as an expectation of discounted sum of payoffs, as in the example below:

$$V_0(s_0) = \max_{d_0,\ldots,d_T} E\left\{\sum_{t=0}^{T} \beta^t u_t(s_t, d_t)|s_0\right\}, \quad 1.$$

[2]The work by Wiener (1948) on cybernetics is considered to have provided the foundations for the subsequent developments in AI.

where t indexes time, $(d_0, d_1, \ldots, d_T)$ are the decision variables, and $(s_0, s_1, \ldots, s_T)$ are state variables.[3] Thus, the application of DP in practice is heavily dependent on the assumptions of expected utility and discounted utility maximization (or maximization of expected discounted profits in the case of firms), though there is substantial laboratory evidence that questions the relevance of these assumptions for modeling individual behavior; other evidence suggests the stock values of publicly traded firms do not equal the expected present value of their dividend streams, and this opens up important questions about what objective function firm managers are trying to maximize, if any.[4]

This serves as a warning that practical application of DP to intertemporal problems that maximize the expected value of the time-additive objective function, as in Equation 1, may not be maximizing the right objective. To apply standard DP, we need to assume that if the actual objective is not exactly intertemporally separable, it can at least be sufficiently well approximated by the expectation of a time-additive sum of payoffs. A further simplification, also in the interest of mathematical tractability, is that the stochastic law of motion for state variables is a controlled Markov process, resulting in the most commonly analyzed class of DP problems, known as Markovian decision problems (MDPs). In the absence of the Markovian assumption, an optimal decision rule is potentially a function of the entire history of previous states, which substantially increases the dimensionality of the decision problem. Since the well-known trick of expanding the state space can capture higher-order stochastic dependencies, the Markovian assumption is relatively innocuous in most applications, though it comes at a high computational cost due to the curse of dimensionality.

Under the Markovian assumption, the Bellman equation provides the standard recursive way of expressing the solution to the optimization problem in Equation 1 via DP, using the principle of backward induction that is the standard method for solving finite horizon, nonstationary problems (i.e., problems where u and p may depend on t). For stationary, infinite horizon MDPs (where $T = \infty$ and u_t and p_t are independent of t), the Bellman equation is

$$V(s) = \Gamma(V)(s) \equiv \max_{d \in D(s)} \left[u(s, d) + \beta \int V(s') p(s' \mid s, d) \right], \qquad 2.$$

[3]Though there is a theory of intertemporal decision making for DMs with some types of non-time-separable and non-expected utility preferences, mathematical tractability has limited this theory primarily to the case of recursive utility with particular types of certainty equivalent operator that is a generalization of the conditional expectation operator in standard time and state separable expected discounted utility theory. However, the theory underlying the validity of DP for such problems is not well developed, and even basic questions—such as existence and uniqueness to the generalized version of the Bellman equation and its relationship to optimal dynamic decision rules for such problems—are not fully understood. Bloise & Vailakis (2018) discuss recent results on the existence and uniqueness of the solution to the Bellman equation for deterministic DP problems with certain types of recursive utility. As they note, "In spite of a growing interest for recursive utility…concomitant progress in dynamic programming methods has not occurred in recent years…. [W]hen some form of nonexpected utility is introduced…certainty equivalent might not satisfy the additivity property required to establish Blackwell discounting, rendering the Contraction Mapping Theorem unavailable even when utility is otherwise time additive" (Bloise & Vailakis 2018, p. 119).

[4]The experimental violations of expected utility such as via the famous Allais and Ellsberg paradoxes are well known. In a survey of laboratory tests of the discounted utility model, Frederick et al. (2002, p. 393) concludes that "the DU [discounted utility] model, which continues to be widely used by economists, has little empirical support. Even its developers—Samuelson, who originally proposed the model, and Koopmans, who provided the first axiomatic derivation—had concerns about its descriptive realism, and it was never empirically validated as the appropriate model for intertemporal choice. Indeed, virtually every core and ancillary assumption of the DU model has been called into question by empirical evidence collected in the past two decades." Research by Campbell & Shiller (1988, p. 675) and others has found that stock prices and returns are "much too volatile to accord with a simple present value model."

and the corresponding optimal stationary decision rule $\delta(s)$ is given by

$$\delta(s) = \underset{d \in D(s)}{\operatorname{argmax}} \left[u(s, d) + \beta \int V(s') p(s' \mid s, d) \right]. \quad 3.$$

We define the structure of the DM's decision problem as the objects $[\beta, u, p, D]$. A huge theoretical and numerical literature provides many different ways to solve the Bellman equation (Equation 2) for V and the optimal decision rule δ in Equation 3. We can think of V as providing the appropriate shadow price for evaluating the effect of current choices on future payoffs, so that the decision to a complicated, infinite horizon sequential decision problem is reduced to the optimization problem that determines δ in Equation 3.

The Bellman equation can be written abstractly as a fixed point $V = \Gamma(V)$, where Γ is the Bellman operator that maps value functions into value functions as defined in Equation 2. Blackwell (1965) established the existence and uniqueness of a solution V to the Bellman equation for stationary infinite horizon MDPs by showing that Γ is a contraction mapping. This implies that the most commonly used algorithm for solving MDPs, value iteration, converges to V starting from any initial guess V_0: $V_{j+1} = \Gamma(V_j)$.[5]

There is a huge literature on different numerical strategies for approximating V and δ for MDPs with continuous state and decision variables, including different strategies for doing the maximization and numerical integration operations in Equations 2 and 3, and strategies for approximating (V, δ), which are infinite dimensional objects (i.e., functions of the state variable s that can assume a continuum of possible values) (see Rust 1996, 2017 for surveys of this literature). The basic approaches are the same: Either successive approximations or policy iterations are used as the outer algorithm to find the fixed point $V = \Gamma(V)$ to the Bellman equation. However, problems with continuous state variables are generally solved over a finite grid in the state space $\{s_1, \ldots, s_N\}$, and the resulting solution (V, δ) to the finite state problem at the N grid points is interpolated to provide values at state points s that are not on this grid. Other researchers, including Bellman, advocated the use of various forms of parametric approximation to V, such as approximating $V \simeq V_\omega$, where V_ω is an element of a class of functions that depend on a finite vector of coefficients ω such as neural networks (see, e.g., Bertsekas & Tsitsiklis 1996) or Chebyshev polynomials (Judd 1998). Then ω can be determined by methods such as nonlinear least squares by, for example, finding a value $\hat{\omega}$ that minimizes the mean squared Bellman equation residual,

$$\hat{\omega} = \underset{\omega}{\operatorname{argmin}} \int |V_\omega(s) - \Gamma(V_\omega)(s)|^2 \mu(s), \quad 4.$$

for some probability distribution μ.[6]

In the absence of special structure (e.g., the MDP is linear quadratic, there are only a finite number of possible decisions, etc.), there will be an inherent curse of dimensionality that no algorithm can circumvent, as we noted in the introduction. Though the software and algorithms are constantly improving and enabling us to solve increasingly high-dimensional problems (see, e.g., Brumm & Scheidegger 2017), there are still many interesting applied problems that are well beyond our ability to solve, and in my opinion this constitutes a major reason why DP has not

[5]Iterations continue until $||V_{j+1} - V_j|| < \varepsilon$ for some desired convergence tolerance ε. Proposition 2.2.1 by Bertsekas (2017) provides computable error bounds on the distance between the final iterate V_{j+1} and the true solution $V = \Gamma(V)$.

[6]Although often more difficult to implement than least squares, the sup norm can alternatively be used, so that $\hat{\omega} = \operatorname{argmin}_\omega ||V_\omega - \Gamma(V_\omega)||$, where $||f|| = \sup_s |f(s)|$. However, issues arise over the equivalence of the L_2 and sup norm in continuous state problems, as minimizing the L_2 norm does not necessarily insure the sup norm error is small, so $V_{\hat{\omega}}$ may not be close to the true fixed point V that satisfies $||V - \Gamma(V)|| = 0$.

been widely used in practice. Ideas from the ML literature, including neural networks, Gaussian process regression, and related methods, offer considerable promise to continue to extend the size of DP problems that can be solved (see, e.g., Bertsekas & Tsitsiklis 1996, Bilonis & Scheidegger 2017, Powell 2010), but as a mathematical matter the application of ML methods cannot break the underlying curse of dimensionality, so these methods should not be regarded as a panacea. Further, most of the successful applications of DP still rely critically on human insight and intervention to recognize the special structure in a problem (including the existence of simpler, decomposable subproblems of an overall decision problem), choose the functional forms representing the DM's preferences and the laws of motion for the problem, and design a numerical algorithm from the many choices available that can best exploit the structure of the problem (for example, the choices of optimization algorithms, approximation methods and architectures, methods for numerical integration, etc.).

Of course, even if the solution to a DP problem can be computed relatively accurately, the solution may not be useful for improving decision making if the assumed structure of the problem, say $[\beta, u, p, D]$, differs greatly from the actual structure. This is the other critical domain in which learning is required: How do the external advisors to a DM (who are the ones that formulate and solve the DP problem) learn about the underlying problem structure that the DM (a human, firm, or organization) actually faces? Though the notion of actor-critic algorithms has a more specific meaning in the RL literature, which I discuss below, I refer to it here in a more generalized sense to capture how DP is used in practice to improve decision making. The actor is the DM who seeks advice in the form of recommended decisions from a critic (or policy advisor) who is presumed to have a comparative advantage in solving the actor's decision problem.

How does the critic learn the structure of the decision problem that the actor confronts? The literature on structural estimation provides one way this can be done, essentially by finding a decision structure $[\beta, u, p, D]$ that enables the predicted behavior from the DP solution to best approximate the behavior we actually observe. This can be regarded as an inversion operation allowing us to uncover the underlying structure of the decision problem from the observed behavior, which is also known as inverse optimal control in the engineering literature. [Aguirregabiria & Mira (2010), Eckstein & Wolpin (1989), and Rust (1994) provide surveys of this literature.] However, structural estimation has a number of inherent limitations and depends on a key assumption that limits its usefulness for learning the underlying structure, including the curse of dimensionality as discussed in Rust (2014). A key assumption underlying structural estimation is the standard as-if assumption of optimization and unbounded rationality that underlies most of economics, as I discussed in the introduction. Therefore, most structural estimation methods learn the underlying structure by presuming the actor in question has in fact already solved their decision problem. If this is the case, the actor really has no need for the critic because is already behaving optimally! Another key problem is the identification problem: In the absence of fairly strong functional form restrictions, Rust (1994) and Magnac & Thesmar (2002) have shown that dynamic discrete choice models are nonparametrically unidentified: There are infinitely many different structures $[\beta, u, p, D]$ that can rationalize observed behavior, that is, the mapping of $[\beta, u, p, D] \rightarrow \delta$ is many to one and thus not invertible without strong additional restrictions such as parametric restrictions on (u, p) (i.e., their functional form is known up to a finite number of unknown parameters). Thus other sources of information, possibly including direct elicitation of preferences and beliefs of the DM, may be necessary in order for the critic to be able to learn the structure of the actor's decision problem.

In Section 4 I discuss what econometricians might call a semiparametric learning algorithm that relaxes the key as-if optimality and unbounded rationality assumption underlying most of structural estimation; such algorithms may be applicable to problems where there is sufficient

prior information to identify the structure of the DM's decision problem.[7] If, given sufficient data, we can identify the structure $[\beta, u, p, D]$ and solve the DP problem, then the implied optimal decision rule δ can help the DM behave optimally.

Specifically, we can use data from the actor in question (a firm), as well as other sources, and use nonparametric estimation methods to uncover the actor's existing or status quo decision rule, which may not be optimal. Using this estimated decision rule while dealing with econometric problems (such as missing or mismeasured data as well as endogeneity) that are also commonly present in firm data, it is sometimes possible to obtain consistent econometric estimates of the remaining structure of the firm's decision problem, which includes consumer demand and the firm's technological possibilities (e.g., production and cost functions). For publicly traded firms the discount factor can be estimated from stock return data (assuming the capital asset pricing model is a good approximation), or it can be directly elicited from the firm. Thus, there may be enough information for the critic (advisor) to learn about the structure of the firm's decision problem to be in a position to solve it via DP and provide the firm with optimal recommended decisions. Because the econometric model has already estimated the (potentially suboptimal) decision rule used by the firm (i.e., its status quo decision rule), it is possible to conduct stochastic simulations to compare the optimal and status quo policies and quantify the gains from switching to the calculated optimal DP policy. If these computer simulations are sufficiently promising, the actor is more likely to follow the recommended decisions in practice, and further validations of the critic's recommendations can be carried out via controlled field experiments as well as before/after comparisons to validate whether the simulated gain in performance from the DP decision rule is borne out in reality.

In situations where the nonparametric two-step learning algorithm discussed above can be used to uncover the underlying structure of the actor's decision problem, some limitations of the supervised learning approach are ameliorated. This algorithm no longer depends on the assumption that we can observe the behavior of an expert who is already behaving optimally in order to train the DP algorithm. Instead, we can train the DP algorithm using potentially suboptimal decisions by a nonexpert actor. However, the algorithm must be run offline, that is, before it can be used to make recommended decisions we need to gather data from the actor and conduct an econometric analysis to learn the structure of the actor's decision problem and then solve the DP problem. Thus, this approach contrasts with the online or real-time learning algorithms discussed below that can continuously monitor states and decisions made by the actor and iteratively improve their performance concurrent with actual decision making. The real-time learning algorithms can potentially adapt to changes in the structure of the actor's decision problem. Offline approaches can be adaptive if they are run in a sort of batch mode whereby data sets on states and decisions observed from the actor are continuously collected and periodically reanalyzed to update the econometric estimates of the underlying structure of the actor's decision problem and recalculate DP decision rules based on the latest available data. For this reason, it is important not to overemphasize the difference between online and offline approaches to DP.

I use the term supervised learning to refer to the high level of human intervention required to carry out the offline approach to solving DP problems discussed above. It should not be confused with supervised learning as it is used in the literature on ML, where it refers to training an algorithm to extrapolate and generalize observations of the decisions of an expert, who is typically a human being. This latter type of learning would be akin to nonparametric regression to uncover

[7] DellaVigna (2018) discusses a new literature on structural behavioral economics whose goal is to relax some of the strong assumptions, including optimization and unbounded rationality, that underlie standard structural econometrics.

an optimal decision rule $d = \delta(s)$ from a data set of pairs (d_i, s_i) of decisions and states of an expert human DM who might be considered to be a role model for some other DM who is not following an optimal policy. If the structure of the decision problem is the same for the role-model DM and the DM in question, then this simpler type of supervised ML might be reasonable and might completely avoid the need to actually solve the DP problem (since the role model is assumed to have already done it).

A deeper form of supervised learning (that includes actually solving the DP problem) is required if (*a*) we do not believe there is another DM who is behaving optimally and can serve as a role model, or (*b*) the structure of the decision problem of the DM in question is different from the structure of the decision problem faced by any potential role model we might consider. In such cases, supervised learning refers to the need to gather data and use econometric methods to uncover the structure of the DM's decision problem, as well as to explicitly solve the DP problem to provide recommended decisions to the particular DM in question. Both of these tasks take place offline and require a high degree of human intervention to conduct the econometric analysis of the data and solve the DP problem.

In the next section I discuss DP algorithms from the literature on RL that can operate in real time, in an unsupervised manner, and are model free. These algorithms achieve the ideal of ML methods by avoiding human intervention to program the algorithm, update it, or estimate econometric models to learn the structure of the DM's decision problem. The AlphaZero algorithm is an example of this type, and it attained superhuman performance in chess and Go after being trained in an unsupervised manner, entirely through self-play.

2.2. Real-Time Solution of DP Problems Versus Reinforcement Learning

The DP literature proved highly influential to researchers in AI because "DP provides the appropriate basis for compiling planning results into reactive strategies for real-time control, as well as for learning such strategies when the system being controlled is incompletely known" (Barto et al. 1995, p. 82). The value iteration and policy iteration algorithms for offline solution of DP problems inspired AI researchers interested in RL to develop stochastic versions of these algorithms for solving DP problems that can converge asymptotically to optimal decision rules and value functions, even for systems that are being controlled by these algorithms in real time and without the requirement for an explicit model of the structure of the DM's decision problem. Examples of RL algorithms include temporal difference learning (Sutton 1988) and *Q*-learning (Watkins 1989), as well as the Real-Time DP (RTDP) algorithm proposed by Barto et al. (1995). Not all versions of these algorithms are model free, and not all of them are run only in real time. In fact, many of them are trained offline, and to do this the analyst needs to have some type of model of the underlying structure of the problem to be able to simulate transitions and payoffs. However, after initial training most of these algorithms can operate in real time, and their performance tends to improve over time with experience; therefore, they qualify as learning algorithms that improve as a result of unsupervised learning in the sense that "DP-based learning algorithms are examples of reinforcement learning methods by which autonomous agents can improve skills in environments that do not contain explicit teachers" (Barto et al. 1995, p. 82).

The original work in this field focused on establishing the convergence of these algorithms in finite state stationary MDPs. This literature has developed in various directions and is now known by several terms, including neurodynamic programming (ND) (Bertsekas & Tsitsiklis 1996) and approximate dynamic programming (ADP) (Powell 2010). In these models, neural networks are used instead of table lookups of values for problems with continuous state and decision variables or finite MDPs such as Go, where there is a huge number of possible states and decisions. For

example, Mnih et al. (2015, p. 529) showed that by combining *Q*-learning with multilayer or deep neural networks, the resulting deep Q-network

> can learn successful policies directly from high-dimensional sensory inputs using end-to-end reinforcement learning…. [R]eceiving only the pixels and the game score as inputs, [the deep Q-network] was able to surpass the performance of all previous algorithms and achieve a level comparable to that of a professional human games tester across a set of 49 games, using the same algorithm, network architecture and hyperparameters.

Barto et al. (1995) proved the convergence of RTDP using previous work on asynchronous DP algorithms by Bertsekas (1982) and Bertsekas & Tsitsiklis (1989), which was developed to prove the convergence of the traditional value function iteration and policy iteration algorithms in parallel computing environments where not all processors are equally fast or perfectly synchronized. This implies that values for some states are updated or backed up (in the terminology of Barto and colleagues) at different times; yet, as long as these update or backups of the value function at all possible states occur infinitely often, the sequence of value functions and decision rules $\{V_k, \delta_k\}$ produced by asynchronous DP algorithms still converge to the optimal values given by the Bellman equation for stationary infinite horizon MDPs (see Equations 2 and 3). Recall that the standard value function iteration can be written as $V_{k+1} = \Gamma(V_k)$, where the optimization problem to determine the optimal decision rule $d = \delta(s)$ is done for all states s—that is, the current value $V_k(s)$ is updated and backed up as $V_{k+1}(s)$ for all states s. RTDP is simply a natural extension of the idea of asynchronous DP to a case where only the currently occupied state s_k at time k is backed up, but the values for other states that did not occur are left alone. That is, RTDP implies an updating rule for V_{k+1} of the form

$$V_{k+1}(s) = \begin{cases} \max_{d \in D(s)} \left[u(s, d) + \beta \int V_k(s') p(s'|s, d) \right] & \text{if } s = s_k \\ V_k(s) & \text{if } s \neq s_k \end{cases}, \qquad 5.$$

where s_k is the realized state of the decision process at time $t = k$, which is also the index for the iterative updating of the value function. We let $\delta_{k+1}(s)$ be the corresponding updated decision rule, the value of d that solves the maximization problem in Equation 5 at state s_k. Then, the DM takes the decision $\delta_{k+1}(s_k)$ resulting in a new realized state $\tilde{s}_{k+1}$ at time $t = k + 1$, a draw from the transition probability $p[s'|s_k, \delta_{k+1}(s_k)]$, and the process continues this way forever. Because of the stochastic nature of the updates, the sequence of value functions and decision rules from RTDP, $\{V_k, \delta_k\}$, is also stochastic. Under certain conditions, this sequence will converge with probability 1 to the true solution (V, δ), generalizing the standard convergence theorem on successive approximations for contraction mappings.

A key assumption is that all states must be visited infinitely often, but it can be difficult to provide high-level assumptions to guarantee this happens in real time, especially if the process has transient states that are visited infrequently. Thus, in practice, Barto et al. (1995, p. 103) discuss trial-based RTDP, which is "RTDP used with trials initiated so that every state will, with probability one, be a start state infinitely often in an infinite series of trials." However, these extra trials can only be done offline, and so the algorithm can no longer be described as real time. Further, note that neither variant of RTDP is model free since the updating formula (Equation 5) requires knowledge of the problem structure $[\beta, u, p, D]$ and necessitates explicit optimization and numerical integration.

Q-learning is a true real-time and model-free learning algorithm, which is one of the reasons it has been used in the above-cited work by DeepMind to train algorithms to achieve human and even superhuman levels of skill in a variety of two-player games such as chess and Go. Watkins

(1989) defined the $Q(s, d)$ function as the right-hand side of the Bellman equation, i.e.,

$$Q(s, d) = u(s, d) + \beta \int V(s')p(s' \mid s, d). \quad 6.$$

Economists refer to $Q(s, d)$ as the decision-specific value function; this function plays an important role in the literature on structural estimation of dynamic discrete choice models, since knowledge of the Q function is sufficient to describe the choices of a DM. Specifically, we have $V(s) = \max_{d \in D(s)} Q(s, d)$ and $\delta(s) = \operatorname{argmax}_{d \in D(s)} Q(s, d)$, and Q itself can be regarded as the unique fixed point of a contraction mapping, $Q = \Lambda(Q)$, given by

$$Q(s, d) = \Lambda(Q)(s, d) \equiv u(s, d) + \beta \int \left[\max_{d' \in D(s')} Q(s', d') \right] p(s' \mid s, d). \quad 7.$$

Let Q_k denote the estimate of the Q function at time $t = k$ when the current state is s_k. The DM is assumed to act in what computer scientists call a greedy manner, i.e., by taking a decision $d_k = \operatorname{argmax}_{d \in D(s_k)} Q_k(s_k, d)$ that has the highest value. Let $\tilde{s}_{k+1}$ be the new state resulting from this choice, i.e., it is a draw from the transition probability $p(s' \mid s_k, d_k)$. Given $(\tilde{s}_{k+1}, s_k, d_k)$, the DM updates Q_k as follows:

$$Q_{k+1}(s, d) = \begin{cases} Q_k(s_k, d_k) + \alpha_k \left[u(s_k, d_k) + \beta \max_{d' \in D(\tilde{s}_{k+1})} Q_k(\tilde{s}_{k+1}, d') - Q_k(s_k, d_k) \right] & \text{if } (s, d) = (s_k, d_k) \\ Q_k(s, d) & \text{otherwise} \end{cases}, \quad 8.$$

where $\alpha_k > 0$ is a step-size parameter that should decrease to zero as $k \to \infty$ but not too fast, as I discuss below. Thus, the updating rule in Q-learning is similar to the updating rule in RTDP (Equation 5): It only updates Q_{k+1} at the state/decision pair (s_k, d_k) that actually happens at $t = k$, and it is stochastic because it involves the stochastic successor state $\tilde{s}_{k+1}$ that results from the decision d_k in state s_k. However, unlike RTDP, Q-learning is indeed model free since the updating rule (Equation 8) only depends on the realized payoff $u(s_k, d_k)$, but it does not require an explicit functional form for the DM's reward function $u(s, d)$ or the transition probability $p(s' \mid s, d)$. Tsitsiklis (1995) proved that Q-learning converges with probability 1 to the true Q function, the unique fixed point to Equation 7 in finite MDPs. He showed that the iteration (Equation 8) constitutes a form of asynchronous stochastic approximation for finding the fixed point of Λ.[8] The key assumptions underlying the asynchronous stochastic approximation algorithm are that (*a*) Λ is a contraction mapping with a unique fixed point Q; (*b*) each possible state/decision pair (s, d) is visited, and thus updated, infinitely often; and (*c*) α_k obeys the standard rate conditions for stochastic approximation, i.e., $\sum_{k=0}^{\infty} \alpha_k = \infty$ and $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$.[9]

Q-learning suffers from the same problem as RTDP, namely that it can be hard to establish conditions that guarantee that each point (s, d) is visited/updated infinitely often as $k \to \infty$. One way to achieve this is to impose a degree of random experimentation on the DM, for example, by

[8]The adjustment term inside the brackets and multiplied by α_k in Equation 8 is stochastic but has conditional mean zero when $Q_k = Q$, the true fixed point, since from Equation 7 we see that $E\{u(s, d) + \beta \max_{d' \in D(s')} Q(s', d') - Q(s, d) \mid s, d\} = 0$. Thus, the term in the brackets multiplied by the step-size α_k in Equation 8 is the sum of a deterministic part, $\Lambda(Q_k)(s, d) - Q_k(s, d)$, plus another term that constitutes random noise. Since Λ is a contraction mapping, the deterministic part tends to zero as $k \to \infty$ and the stochastic noise tends to zero due to the decreasing step sizes α_k. The main difference of this algorithm and the well-known method of stochastic approximation by Robbins & Munro (1951) for finding zeros or fixed points of functions defined by conditional expectations is the asynchronous nature of the updates: Only the current state/decision component (s_k, d_k) of Q_k is updated at iteration k instead of being updated at all possible values under standard stochastic approximation.

[9]If the step sizes are random variables, they must be positive and obey these conditions with probability 1.

adding stochastic shocks $\varepsilon(d)$ to the calculated Q_k values, so decisions are given by the value of d_k that maximizes $Q_k(s_k, d) + \sigma_k \varepsilon(d)$, where σ_k is a sequence that converges to zero at an appropriate rate. These random shocks that affect the DM's choice of decision coupled with sufficient ergodicity in $p(s' \mid s, d)$ can lead to sufficient random experimentation to guarantee that every (s, d) point is updated infinitely often, and thus ensure the probability 1 convergence of the stochastic sequence $\{Q_k\}$ to the true value Q.

The stochastic convergence results discussed above are only for finite MDPs, and the algorithms only work well for problems with a relatively small number of states and decisions. Otherwise, if there are too many s or (s, d) pairs that are not backed up, the algorithms may fail to discover optimal strategies, especially if initialized from V or Q values that are far from optimal.[10] Thus, for relatively small problems where the structure of the decision problem $[\beta, u, p, D]$ is known, the standard value function iteration and policy iteration methods described in the previous section are generally faster and produce far more accurate approximate solutions compared to RL algorithms. However, for moderate- to large-scale MDPs, the curse of dimensionality starts to become evident, and it becomes increasingly difficult to employ the standard types of algorithms and guarantee a sufficiently precise solution to the problem. For sufficiently large-scale problems with multiple continuous state variables, RL algorithms start to be competitive for the simple reason that the traditional types of nonstochastic algorithms are generally no longer even computationally feasible. However, the convergence results for Q-learning and other types of RL are no longer applicable either. Even for finite MDPs such as board games like chess or Go, the number of possible states and decisions becomes far too large for the standard RL algorithms based on table lookup of values to be effective, so some sort of interpolation/extrapolation of values becomes necessary in order to implement these algorithms.

Just as the standard numerical algorithms discussed in the previous section have relied on parametric approximations to V or δ using a relatively small number of unknown parameters ω (for example, by approximating V via various linear combinations of basis functions or via neural networks), these same sorts of parametric approximation have been applied in the RL literature; here, however, the parameters ω are sequentially updated via various types of iterative stochastic gradient methods that avoid direct solution of the least squares approximation problem (Equation 4). Bertsekas & Tsitsiklis (1996) and Powell (2010) provide a comprehensive discussion of the various ways that RL algorithms such as temporal difference learning, RTDP, and Q-learning can be combined with various types of function approximation methods, but they caution that "convergence to [V or Q] cannot be expected, for the simple reason that [V or Q] may not be within the set of functions that can be represented exactly within the chosen architecture" (Bertsekas & Tsitsiklis 1996, p. 256).

Neural networks have proved attractive as an approximating class of functions due to their flexibility and the fact that they can approximate well a wide range of functions using relatively few parameters (see Barron 1994). However, the Achilles heel of neural networks is that the implied least squares problem (Equation 4) typically has a profusion of local optima, making it quite computationally intensive to search for parameter values ω that globally minimize the value function residuals, which is a necessary (though not sufficient) condition to find a good approximation to the true V or Q function. As a result, a great deal of art (i.e., practical experience and perhaps luck) is required to design a neural network architecture (i.e., choose the number of hidden layers and specify where and how the different state variables enter these layers) and to train it so that the outcome is satisfactory. Thus, neural networks should not be regarded as a panacea or cure

[10]There are cases where values of V or Q that are not close to being fixed points of their respective operators can nevertheless imply decision rules that are close to optimal (see, for example, Puterman 2005, chapter 6).

for the curse of dimensionality: They merely deal with one form of the curse (i.e., approximation of functions of d variables using a vector of parameters ω that increases only linearly in d) at the expense of another (i.e., the curse of dimensionality involved in globally minimizing the least squares criterion in ω). It is quite easy for training to find a local but not global minimum, and a poorly trained neural network approximation can perform poorly. For example, a neural network trader performed poorly in the double auction tournament run at the Santa Fe Institute (Miller et al. 1993).

However, there have been significant advances in the literature on RL and training of deep neural networks, as evidenced by the impressive human and superhuman level of play achieved in a wide range of games by DeepMind's deep Q-networks cited above, which demonstrate "that a general-purpose reinforcement learning algorithm can achieve, *tabula rasa*, superhuman performance across many challenging domains" (Silver 2017, p. 2). The fact that the AlphaZero algorithm was trained from scratch and entirely via self-play and was able to beat the leading competing chess program Stockfish after just four wall-clock hours attracted considerable publicity, but consideration should be given to the huge number of processors required to do this and the huge number (over 300,000) of games that must be played to train the algorithm to the point where it is capable of a high level of performance.

I am not aware of other real-world applications where RL (combined with deep nets, ordinary nets, or other approximation architectures) has achieved comparable success. I think this reflects a huge advantage associated with the use of RL to train algorithms to play well in ordinary board games: The structure of the decision problem is mostly known and the problem can be easily simulated. The payoffs of the players in a game such as chess is quite simple: for example, $+1$ for a win, -1 for a loss, and 0 for a draw. Further, the rules of chess and other board games such as Go are fully known and can be rapidly simulated on a computer, making it possible to play out millions of moves in offline training sessions on thousands of processors in a relatively small amount of wall-clock time. The main aspect of the game that is not known (and must be learned) is the response probability of the opponent. However, the training pits a copy of the algorithm to play against another copy of itself, and this facilitates the algorithm's ability to learn the probability distribution over the moves of its opponent in different states.[11]

It is less clear that real-time DP and RL algorithms are appropriate for approximating DP solutions in the much fuzzier and more complex real-world environments that consumers and firms ordinarily operate in. In those contexts the structure of the decision problem is not so easily specified, and for firms it may include the need to learn about consumer preferences and demand, including how demand is affected by decisions of other competitors and business cycle fluctuations. Though the idea of model-free learning is appealing, it seems unlikely that individuals or firms would have the willingness to follow recommendations of a poorly trained and initialized deep Q-network if its recommended actions do not seem sensible. Unless real DMs follow the recommendations of a RL algorithm, it is unclear how the algorithm can be trained to improve its performance over time. Most real-world DMs do not make enough decisions to provide the

[11]Chess is an example of a dynamic directional game (see Iskhakov et al. 2015 for a definition) and may have multiple equilibria. It is not clear that, despite the superhuman performance of AlphaZero, the training converges to Nash equilibrium strategies, because it could cycle as in the well-known fictitious play algorithm for finding Nash equilibria (Brown 1951). Iskhakov et al. (2018), using the recursive lexicographical search (RLS) algorithm of Iskhakov et al. (2015), characterize all equilibria to a class of dynamic directional games with Bertrand price competition and leapfrogging investments that have many fewer states than chess, yet have billions of equilibria. It is not clear that RL algorithms trained on copies of themselves will also result in strategies that are approximate best responses to other strategies, and it may be possible to train new generations of algorithms to exploit weaknesses in AlphaZero, similar to what was done in the evolutionary tournament of competing trading strategies in the double auction market, as reported by Miller et al. (1993).

thousands to millions of training instances needed by RL algorithms to learn in real time, and few have the patience to train the algorithms in a simulated decision-making environment.

Thus, it is clear that if RL is to be successful, it must have a capability to either (*a*) aggregate experience learned from advising similar DMs or (*b*) conduct offline simulations to train itself to be more effective form the much smaller number of real-world opportunities to advise individuals or firms on actual decisions. However, it is not clear how ML approaches by themselves can recognize when individuals or firms are sufficiently similar in the specific sense that the structure of their decision problems are nearly the same, a recommended decision that represents good advice for one will be equally good for others, and the collective data and training experience can be pooled and shared. However, it may be possible to effectively aggregate experience and knowledge across different individuals, as examples from something as seemingly subjective as the choice of clothing fashions suggest.[12]

Thus, it seems that both of the approaches that I have outlined in this section for solving DP problems depend critically on the ability to learn the structure of the decision problem. To my knowledge, ML is not able to acquire this type of knowledge from scratch: Its acquisition is related to a fundamental and unsolved puzzle of abduction, i.e., the process humans use to form models that provide efficient though imperfect representations of the world that may be key to human intelligence (see Griffiths & Tenenbaum 2009, Heckman & Singer 2017). Without good knowledge of the structure of the decision problem, it is not clear that DP algorithms of any type will be useful to real-world DMs if they fail to grasp the actual problem and the constraints, opportunities, and challenges that real-world decisions actually confront.

3. APPLICATIONS OF DP TO IMPROVE FIRM DECISION MAKING

As I discussed in the introduction, I am aware of only a few real-world situations in which DP is actually used. It is reasonable to presume that the stakes for adopting profit-maximizing policies are quite high, so successful, innovative firms that can muster the resources necessary to solve difficult decision problems should be among the early adopters of DP. Yet, though I have not done a systematic survey, some firms that are likely adopters of DP appear reluctant or unable to speak about it. For example, Patrick Bajari, chief economist at Amazon, reports "I'd love to help out here, but I can't really talk about the IP of our company" (P. Bajari, personal communication).

On the other hand, I can confirm that for whatever reason, DP is not used by some of the largest, most successful, and more sophisticated firms that we might expect would be among the most eager early adopters of this technology. For example, Benjamin Golub, a PhD in economics from MIT and chief risk officer of Blackrock, reports, "Regarding DP, I am not familiar with any uses of it on my side of Blackrock. People use recursive algorithms for various purposes, but I don't believe that is a form of DP. Similarly, there is some use of multi-period optimization, but again I don't think that counts as DP" (B. Golub, personal communication). Blackrock does invest heavily in other forms of high technology, including ML for default prediction and other applications. One of my PhD students and an expert in ML, Dongbo Guo, was hired by Blackrock to help develop advanced algorithms to predict mortgage default. Though I am advising another

[12]For example, the company Stitch Fix uses ML algorithms to recommend new clothing designs to its customers and learns from repeated feedback from individual customers, but also across customers with apparent similar tastes. As its chief algorithms officer notes, "As time goes by, algorithms learn from actual customers, both individual and in aggregate, how to think about clothes. This is possible thanks to the feedback data collected from customers, which is transmitted back to the algorithms so that they can see how their decisions worked in real life—and use this information to constantly improve their decision-making formulas (machine learning)" (Eric Colson, cited in Forbes Insights 2018).

PhD student, Yangfan Sun, who is developing promising structural models of the mortgage default decision using DP, to my knowledge Blackrock does not use DP and structural models to predict default.

DP is almost certainly used in some places on Wall Street for option pricing, including choosing the optimal time to exercise an American option. Two important contributors to the field of numerical option pricing are Mark Broadie and Paul Glasserman at Columbia University. A particularly complex type of option is a swaption, which gives an investor the right to enter into an interest rate swap or other type of swap. Bermudan swaptions give the investor this option at multiple predetermined dates. Glasserman noted:

> I believe hundreds of billions in Bermudan swaptions trade on DP (optimal stopping) algorithms. Most practitioners would describe what they do as following the least squares Monte Carlo algorithm of Longstaff and Schwartz. More interesting DP problems come up with swing options for commodities, where you have to decide how much to take in each period, not just whether to stop. (P. Glasserman, personal communication)

Broadie & Glasserman (2004, p. 35) developed a stochastic mesh algorithm "that uses a dynamic-programming style backward recursion for approximating the price and optimal exercise policy." Broadie notes that "the DP method described in the paper was (and I believe is) used extensively in practice" (M. Broadie, personal communication). Nevertheless, Andersen & Broadie (2004, p. 1222) note that "many numerical methods for pricing American options have been proposed, and although tremendous progress has been made, the pricing of these options in multifactor models with possibly path-dependent payouts has remained a formidable challenge."

Most of the applications of DP that I am aware of are in engineering, such as network routing (see Mak et al. 2011), power management in cloud computing centers (see Zhang et al. 2017), and several other examples. Powell (2010, chapter 1.3) discusses several real-world applications of DP and notes,

> Our experiences using approximate dynamic programming have been driven by problems in transportation with decision vectors with thousands or tens of thousands of dimensions, and state variables with thousands to millions of dimensions. We have solved energy problems with 175,000 time periods. Our applications have spanned applications in air force, finance, and health.

Below I discuss a specific application of DP to dynamic assignment of locomotives at a major US railroad.

The limited number of known applications of DP by firms may simply reflect the limitations of DP, including the inability to even mathematically formulate some of the most challenging problems that firms confront. Much of the complexity of running a firm stems from the fact that a firm is a multiperson team, and some of the most difficult challenges involve how to properly incentivize managers and employees to maximize the objective of the firm as a whole. However, the design of incentives is one of the area economists have the most expertise in, and indeed one of my few DP success stories is a case where DP helped a firm design a more incentive-efficient compensation plan.

Another problem limiting the application of DP is that for many firms, the objective function that they are maximizing is rather fuzzy and not entirely clear. The default assumption in economics is that public firms seek to maximize their market value, and their market value equals the expected stream of future dividends, discounted at an appropriate risk-adjusted discount rate. However, a substantial amount of empirical work in finance on the excess volatility of stock market valuations raises substantial doubt as to whether the market value of the firm equals its

fundamental value, i.e., the expected present value of its future dividend stream. To the extent that there is a large and systematic noise component to stock market valuations, it is no longer clear if a manager's objective function should be to maximize expected discounted dividends, even if all the other incentive problems involved in operating a firm could be solved.

Further, the theory of the firm ignores potential differences in the behavior of public and private firms. Gupta & Rust (2018) show that if the owner of a private firm is a single individual whose main source of income is the dividend of the firm itself, the appropriate objective is expected discounted utility maximization, not expected discounted profit maximization. The standard consumption smoothing motive implied by discounted utility maximization in risk-averse consumers means that the owners of private firms should engage in dividend smoothing; this, however, is inconsistent with the policy of maximizing discounted expected dividends, which generally results in highly volatile dividend policies, including paying zero dividends for long stretches of time in order to finance investment and accelerate the growth of the firm. Yet, paradoxically, there is empirical evidence that public firms tend to smooth dividends more than private firms (see, e.g., Leary & Michaely 2011). This is one of the numerous unsolved puzzles about firm behavior that suggest that the objective of many firms is not necessarily to maximize the present discounted value of dividends, and that our understanding of firm behavior (including exactly what objective the firms are trying to maximize) is still quite rudimentary.

A final barrier to the implementation of DP is the lack of tools and methods to validate whether the investment to develop or acquire a DP-based decision tool really generates sufficient incremental revenue or profit to justify its adoption. I provide a few examples of validations that I am aware of, but this remains an area where better scientific methods may help increase the credibility of DP-based decision tools and extend the range of real-world applications.

3.1. Application of DP to Employee Compensation

The work by Misra & Nair (2011) is the most successful demonstrated real-world application of DP and structural estimation that I am aware of. It arose from a study of the sales force of a company that produces contact lenses. The authors obtained monthly data on 87 sales agents who were assigned to sell large quantities of contact lenses to eye doctors in different regions of the United States. The agents were paid a salary plus a commission that was recalculated each quarter. The salary was paid monthly and the commission each quarter. Commissions were earned on sales exceeding a quota but lower than a ceiling. Each quarter the sales state variable used to determine commissions was reset to zero, and the firm had a complicated ratcheting formula that adjusted the sales quota and ceiling based on past sales performance.

An initial reduced-form empirical analysis of the data revealed spikes in sales when agents were close to making their quarterly sales quota (after which they would start receiving commissions), but sales were lower early in each quarter, suggesting the possibility that agents would shirk early in the quarter when they were far from reaching their quota. Further, for some agents sales would also fall at the end of the quarter, "perhaps because the agent realized a very large negative shock to demand early in the quarter and reduced effort, or because he 'made quota' early enough, and hence decided to postpone effort to the next sales-cycle" (Misra & Nair 2011, p. 219). The authors also provide evidence of more sophisticated forms of shirking in response to the firm's ratcheting policy for quotas and commission ceilings: "Ratcheting can help the firm fine-tune quotas to the agent's true productivity or territory potential; but can result in harmful distortions to behavior if forward-looking agents shade current effort in order to induce favorable, future quotas" (Misra & Nair 2011, p. 214).

This first-stage analysis led the authors to formulate a DP model of the dynamic allocation of sales agents' efforts over time, as a best response to the incentives of the firm's commission scheme.

The reduced-form analysis suggested distortions in sales effort in response to the peculiarities of the firm's incentive plan. Using a structural analysis and a DP model of the dynamic allocation of effort, the authors were able to make counterfactual predictions of sales effort in relation to different types of incentive plans that the firm had not considered. A brilliant feature of Misra & Nair's (2011) structural DP analysis is that they were able to estimate structural parameters characterizing agent-specific preferences for earnings versus sales effort, even though sales effort was unobserved. They were able to do this by making the modeling assumption that monthly sales (which they did observe for each agent) was a linear function of effort, and the coefficient on effort was normalized to 1.

Their structural analysis shows that their model provided a good fit to the patterns they observed in the sales data revealed by their reduced-form analysis, and it provided strong evidence that a poorly designed commission scheme was the cause of the distortions they documented. Using their structural model, the authors then solved for an approximately optimal commission scheme: They used DP to maximize the discounted profits of the contact lens division by optimizing over different parameters indexing different possible compensation plans, taking the agent's best responses to each possible compensation plan explicitly into account.[13] The authors were able to use DP to solve for agent-specific compensation plans since they had estimated separate preference and productivity parameters for each of the firm's 87 sales agents.

However, the firm had concerns over possible morale/discrimination problems if it implemented agent-specific compensation plans, and these morale concerns were not explicitly accounted for in the authors' structural model. The authors responded to the firm's concern by proposing a single, simplified, and uniform compensation plan that applied to all of the agents. This simplified plan eliminated the quotas, ceilings, and periodic ratcheting of the parameters of the plan. Even though this simplified plan was not fully optimal, the firm decided to adopt it starting in January 2009. A before/after comparison of the firm's profits revealed that "under the new plan, revenues to the firm increased by about $12 million incremental per year (a 9% improvement overall), indicating the success of the field implementation" (Misra & Nair 2011, p. 213). Employee compensation also increased under the new plan, which according to the authors "suggests the old plan was inefficient" (Misra & Nair 2011, p. 249). Not surprisingly, therefore, "the firm also reports that employee satisfaction with the new plan is high, arising primarily from the reduced emphasis on quotas, and the associated subjective evaluation induced via ratcheting" (Misra & Nair 2011, p. 253). Though it is possible to criticize a simple before/after comparison as not adequately controlling for possible macro shocks, it is worth noting that the new plan earned more even though it was implemented during the depths of the Great Recession. Thus it is likely that their estimates of the gains are underestimated and provide a conservative assessment of the actual improvements in profitability in a steady-state scenario.

3.2. Application of DP to Locomotive Allocation and Fleet Sizing

Powell et al. (2014) report another successful application of DP. The authors used RL/ADP to determine the optimal size of the locomotive fleet for the Norfolk Southern Railway and to provide optimal dynamic scheduling of locomotives to trains, accounting for dynamic schedule changes

[13]The use of parameterized compensation plans may result in schemes that are not fully optimal because they are parametric, and the authors did not invoke the revelation principle of Myerson (1981) to search over the set of all possible incentive-compatible compensation plans. There is a literature on dynamic incentive problems that is even more difficult, since the recursive methods for solving these problems involve the use of multidimensional promised utilities in the agents' DP problems. Renner & Scheidegger (2018) present recent progress in this area using methods from the ML literature.

and locomotive maintenance, including shop delays and unexpected locomotive failures. They note,

> To our knowledge, it is the first optimization-based model of locomotives for a North American freight railroad that calibrates accurately against historical data, making it useful as a tool for fleet sizing, one of the most demanding strategic planning problems. The technology allows locomotives and trains to be modeled at an extremely high level of detail; train delays can be modeled down to the minute. The model can simultaneously handle consist breakups and shop routing while also planning the repositioning of locomotives to other terminals. In addition, it can handle uncertainties in transit times, yard delays, and equipment failures in a simple and intuitive way. The methodology is based on a formal mathematical model, which guided the design of rigorous algorithms; as a result, it avoids the need for heuristic rules that have to be retuned as the data change. (Powell et al. 2014, p. 11)

Prior to the development of their model, which they called locomotive assignment and routing system (LARS), Norfolk Southern relied on a simple linear regression model to estimate the size of its locomotive fleet and noted that "although senior management acknowledged the limitations of the linear regression model for forecasting, it had no better alternatives" (Powell et al. 2014, p. 577). The gains from adopting the DP-based LARS model were potentially huge, since they noted,

> We cannot underestimate the importance of determining the correct fleet size. Norfolk Southern currently runs a road fleet of over 2,000 locomotives and a new locomotive costs well over $2 million. An insufficient number of locomotives results in deteriorating customer service as train delays increase and leads to a loss of competitiveness in key markets. In contrast, an excessive number of locomotives is a large capital expenditure and is expensive to maintain or store. (Powell et al. 2014, p. 9)

In terms of the algorithms, though the authors used RL/ADP due to the curse of dimensionality in allocating 2,000 locomotives on a minute-by-minute basis, they did not train the LARS system in real time, and their algorithm is not model free. Instead, LARS is an example of model-based ADP where value function iterations are used based on stochastic simulations of a detailed simulation model of system-wide demand for locomotives, including the time locomotives spend in the shop for maintenance (either planned or unplanned). This latter model, called PLASMA, required a separate multiyear development effort in which the team "carefully calibrated the model against several years of historical performance. This required us to painstakingly examine detailed assignments and compare high-level performance metrics. The process required that we iteratively identify and correct data errors, enhance the model, and make occasional improvements to the basic algorithm" (Powell et al. 2014, p. 10). The benefit of this substantial investment is that "LARS has been used for strategic fleet sizing for several years and has become an integral part of the company's network and resource planning processes" (Powell et al. 2014, p. 2).

Though the article does not provide an evaluation of how much LARS improved profitability at Norfolk Southern, the authors note that LARS was implemented just before the Great Recession and the company found it effective in helping downsize the fleet during the downturn and then increase it again in 2010 as the economy started to recover. The authors acknowledge that "these benefits have a cost. LARS requires substantial effort to prepare the data and calibrate the model. However, the impact on operating costs, performance, and network robustness is dramatically large relative to the cost of developing and maintaining the system" (Powell et al. 2014, p. 10).

3.3. Potential DP Application: Perishable Inventory Pricing and Revenue Management

Revenue management (also known as yield management in the airline industry) focuses on strategies to maximize revenue by allocating available seats (or rooms) at different prices via price

segmentation strategies (e.g., charging different prices for business versus leisure travelers) based on differential willingness to pay. Some revenue strategies are quantity based (i.e., protecting certain seats or rooms from early booking so they are available for last-minute booking by customers with high willingness to pay), and others are price-based dynamic pricing strategies that adjust prices and do not directly control quantities.

Phillips (2005) provides a comprehensive introduction to the literature on revenue management. He notes that despite the fact that pricing decisions "are usually critical determinants of profitability," they are "often badly managed (or even unmanaged)" (Phillips 2005, p. 38). Computerized revenue management systems (RMS) originated in the 1980s following airline deregulation when American Airlines was threatened by the entry of the low-cost carrier PeopleExpress:

> In response, American developed a management program based on differentiating prices between leisure and business travelers. A key element of this program was a "yield management" system that used optimization algorithms to determine the right number of seats to protect for later-booking full-fare passengers on each flight while still accepting early-booking low-fare passengers. This approach was a resounding success for American, resulting in the ultimate demise of PeopleExpress. (Phillips 2005, p. 78)

Surprisingly, there is no mention of DP in Phillips' book. Though he offers a great deal of insight on revenue management and provides intuitively appealing heuristic principles such as Littlewood's rule (a pricing rule that arose from academic work on static revenue management models in the 1970s), Phillips does not discuss the proprietary algorithms revenue management companies use to advise on pricing. Pricing decisions are challenging because they must be done in real time. A small hotel may need to adjust 1,000 prices per day, and a small airline 500,000 prices per day. The high dimensionality of these pricing decisions comes from the fact that hotels and airlines are constantly adjusting future prices for departures or occupancies on various flights and different classes of rooms as much as one year into the future. Further prices are being adjusted more and more frequently, and as Phillips (2005, p. 55) notes,

> The Internet increases the velocity of pricing decisions. Many companies that changed list prices once a quarter or less now find they face the daily challenge of determining which prices to display on their website or to transmit to e-commerce intermediaries. Many companies are beginning to struggle with this increased price velocity now—and things will only get worse.

There is a large academic literature that uses DP to solve the revenue management problem, including the pioneering work of Gallego & van Ryzin (1994, p. 1000) who note that "the benefits of yield management are often staggering; American Airlines reports a five-percent increase in revenue worth approximately $1.4 billion dollars over a three-year period, attributable to effective yield management." Though the size of the revenue management industry has mushroomed to over $10 billion per year, I am not aware of specific commercial revenue management firms that are actually using DP to calculate optimal price or quantity allocation recommendations to their customers. As McAfee & te Velde (2008, p. 437) note,

> At this point, the mechanism determining airline prices is mysterious and merits continuing investigation because airlines engage in the most computationally intensive pricing of any industry. The techniques used by airlines are likely to be copied elsewhere as the ability of companies to condition price offers on other considerations grows with larger databases and enhanced processing power.

Given this high degree of sophistication, what additional value might be provided by DP? Cho et al. (2018) use DP to evaluate the pricing decisions of a specific hotel that uses the IDeaS RMS, a

subsidiary of SAS. The hotel's revenue manager reports that she usually follows the recommended prices from IDeaS, though she is able to override these prices whenever she wants. The recommended prices are usually intuitively reasonable, but occasionally the system appears to produce unreasonable recommendations (such as unusually high prices for bookings in August, which is a slow month for this hotel) and the revenue manager overrides them and sets her own value. The IDeaS RMS does not appear to be adaptive in real time to these human-initiated price updates: The system tends to keep making the same mistakes until the revenue manager calls the company and asks them to fine-tune the pricing algorithm. Cho et al. (2018) find that the predicted prices from their DP model closely match the prices set by the hotel; unfortunately, the data they analyzed did not record situations where recommended prices from IDeaS were overridden by the human revenue manager, so they were unable to determine whether IDeaS's or the human revenue manager's prices more closely conformed to the predictions of the DP model.

However, the analysis of Cho et al. (2018) assumed optimal pricing on the part of the hotel. If the pricing is not actually optimal, it is possible that their structural estimation algorithm could have distorted its estimated demand parameters in order to rationalize the observed pricing of the hotel as optimal dynamic pricing. It is possible to relax the assumption of optimality and still obtain consistent estimates of the stochastic demand/arrival process for hotel bookings using a semiparametric estimator that relies on nonparametric estimation of the pricing rules used by Hotel X and its competitors in the first step, followed by a method of simulated moments (MSM) estimator that finds values of the parameters for the stochastic demand/arrival process for hotel bookings in the second step. The MSM estimator finds the parameters of consumer preferences and customer arrivals that best match a set of observed moments for pricing and bookings for these hotels. Given the estimated demand parameters, it is possible to formulate and solve the hotel's DP problem to calculate optimal dynamic price strategies. By relaxing the optimality assumption, this two-step estimation approach enables us to test for optimality rather than to assume it. Cho et al. (2019) present estimates of hotel demand without imposing the assumption of optimality and show that when this assumption is relaxed, the DP-based pricing strategy is significantly different from Hotel X's existing pricing strategy. Simulations comparing the two strategies confirm that the pricing strategy from the DP model results in significantly higher profits and raises revenues and occupancy by 10% and 9%, respectively. In principle, these predictions are testable via either before/after field tests or controlled field tests similar to the ones in Cho & Rust (2010).

4. CONCLUSION

DP is an extremely powerful tool for solving a huge class of sequential decision problems under uncertainty. In fact, it is hard to think of problems that cannot be formulated and solved using DP. The flexibility and power of DP has revolutionized the way we do economics, and it has provided a key principle that has produced some of the most impressive achievements in the field of AI. Despite this, I have only been able to come up with a few examples where DP is used and has resulted in demonstrable improvements in real-world decision making. What is the explanation for this paradox?

Of the three possible explanations I listed in the introduction, I think the first one—that individuals and firms have unbounded rationality and thus behave optimally and have no need for formal DP—can be immediately dismissed. As Herbert Simon noted in his 1978 Nobel Prize lecture, sufficient evidence against the assumption of unbounded rationality had already been accumulated during the 1950s. Since then, the literature on behavioral economics has provided ever more evidence of suboptimal decision making by individuals and firms, and additional examples have been provided in this article. Yet we cannot conclude that individuals and firms behave stupidly, or

that they will necessarily benefit from the advice of computerized algorithms solving DP problems. Hutchinson & Meyer (1994, p. 379) provide the right perspective on the nature of human suboptimality:

> When compared to normative sequential choice models, humans are likely to perform less well because the processes of forward planning, learning, and evaluation have a number of inherent biases. From a broader perspective, however, one can argue that optimal solutions are known for a relatively small number of similar, well-specified problems whereas humans evolved to survive in a world filled with a large and diverse set of ill-specified problems. Our "suboptimality" may be a small price to pay for the flexibility and adaptiveness of our intuitive decision processes.

The key reason for the limited number of real-world applications of DP is the difficulty in applying it to formulate and solve the highly complex, fuzzy, and poorly defined decision problems that individuals and firms typically face on a daily basis, an explanation that Herbert Simon emphasized in his work on AI and bounded rationality over four decades ago. DP has proved extremely successful as an academic modeling tool, but the formulation and solution of real-world decision problems as mathematical DP problems have proved far more challenging.

Simon recognized that computer power and algorithms would improve steadily over time and thus steadily expand the range of real-world problems that could be solved by DP. Yet even today the range of known real-world applications of DP seems disappointingly small, given the immense computer power and the decades of research that have produced a myriad of alternative solution methods for DP problems. I believe the biggest constraint on progress is not limited computer power, but instead the difficulty of learning the underlying structure of the decision problem. Individuals and firms are likely to have difficulty understanding their own decision problems and may be unable to communicate to an outside advisor exactly what objective they are trying to maximize. Perhaps the most painstaking task for an academic or a commercial service trying to play the role of the critic and recommend better decisions to the actor (i.e., the individual or firm making the actual decisions) is to understand the structure of the actor's decision problem. Calculating an optimal solution to the wrong objective, or misspecifying the constraints and opportunities the actor actually confronts, may not result in helpful advice to the actor. It is like providing the right answer to the wrong question.

Thus, I do not share Stephen Hawking's fear that AI has grown sufficiently powerful to constitute an imminent threat to the future of human race. Perhaps this will be true in a few decades or centuries, once algorithms have evolved to equal or surpass the level of human brilliance in constructing, revising, and improving mental models of reality. Humans are learning to replicate the type of subconscious model building that goes on inside their brains and bring it to the conscious, formal level—but they are doing this modeling themselves, since it is not clear how to teach computers how to model. The development of digital computers, statistics, mathematical and numerical methods, and DP are all contributing to a confluence that is starting to bring more intelligent types of algorithms, but I do not think we are yet at the threshold of true AI.

In the meantime, there are tremendous opportunities for researchers in economics, operations research, engineering, computer science, neuroscience, and AI to work together to produce increasingly intelligent DP-based algorithms that will have significant practical value to individuals and firms by helping them to improve particular parts of their decision making that are the easiest to model mathematically. I expect to see rapid growth of DP-based actor-critic systems (i.e., decision advisory services) in the coming years, as well as an increasing amount of research devoted to validating and measuring the impact of these systems on the welfare of individuals and the profitability of firms.

The credibility and influence of economists are at a historically low point. Some of this situation is self-inflicted, a consequence of the unbounded rationality assumption that individuals and

firms do just fine without any help from us. Our path to redemption may start by trying to be more in touch with reality and facing up to the sorts of normative decision problems that Herbert Simon challenged us to confront in his 1978 Nobel Prize lecture. We need to show that our arsenal of mathematical tools and skills, including DP and econometrics, have numerous real-world applications, and we need to provide credible demonstrations that these tools can really improve individual and firm decision making.

DISCLOSURE STATEMENT

The author is not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

I am grateful for financial support from the Gallagher Family Chair in Economics at Georgetown University and helpful comments from Sharat Ganapati, Paul Glasserman, Mark Broadie, Larry Kotlikoff, David Romer, Stephen Rassenti, Kevin James, Dustin Tracy, Preston McAfee, Benjamin Golub, Warren Powell, Amit Gandhi, Simon Scheidegger, Hal Varian, and Patrick Bajari.

LITERATURE CITED

Adda J, Cooper R. 2003. *Dynamic Economics: Quantitative Methods and Applications*. Cambridge, MA: MIT Press

Aguirregabiria V, Mira P. 2010. Dynamic discrete choice structural models: a survey. *J. Econom.* 156:38–67

Andersen L, Broadie M. 2004. Primal-dual simulation algorithm for pricing multidimensional American options. *Manag. Sci.* 50(9):1222–34

Arrow K, Harris T, Marshak J. 1951. Optimal inventory policy. *Econometrica* 19:250–72

Barron A. 1994. Approximation and estimation bounds for neural networks. *Mach. Learn.* 14:115–33

Barto A, Bradtke S, Singh S. 1995. Learning to act using real-time dynamic programming. *Artif. Intell.* 72:81–138

Barto A, Dietterich T. 2004. Reinforcement learning and its relation to supervised learning. In *Learning and Approximate Dynamic Programming: Scaling Up to the Real World*, ed. J Si, A Barto, W Powell, D Wunsch, pp. 46–63. New York: Wiley Intersci.

Bellman R. 1984. *Eye of the Hurricane*. Singapore: World Sci.

Bertsekas D. 1982. Distributed dynamic programming. *IEEE Trans. Autom. Control* 27:610–16

Bertsekas D. 2017. *Dynamic Programming and Optimal Control*, Vol. 1. Belmont, MA: Athena Sci.

Bertsekas D, Tsitsiklis J. 1989. *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NJ: Prentice Hall

Bertsekas D, Tsitsiklis J. 1996. *Neuro-Dynamic Programming*. Belmont, MA: Athena Sci.

Bilonis I, Scheidegger S. 2017. *Machine learning for high-dimensional dynamic stochastic economies*. Work. Pap., Sch. Econ. Bus. Admin., Univ. Lausanne, Switz.

Blackwell D. 1965. Discounted dynamic programming. *Ann. Math. Stat.* 36:226–35

Bloise G, Vailakis Y. 2018. Convex dynamic programming with (bounded) recursive utility. *J. Econ. Theory* 173:118–41

Broadie M, Glasserman P. 2004. A stochastic mesh method for pricing high dimensional American options. *J. Comput. Finance* 7(4):35–72

Brown G. 1951. Iterative solutions of games by fictitious play. In *Activity Analysis of Production and Allocation*, ed. TC Koopmans, pp. 374–76. New York: Wiley

Brumm J, Scheidegger S. 2017. Using adaptive sparse grids to solve high-dimensional dynamic models. *Econometrica* 85:1575–612

Campbell J, Shiller R. 1988. Stock prices, earnings and expected dividends. *J. Finance* 43(3):661–76

Cellan-Jones R. 2014. Stephen Hawking warns artificial intelligence could end mankind. *BBC News*, Dec. 2. **https://www.bbc.com/news/technology-30290540**

Cho S, Lee G, Rust J, Yu M. 2018. *Optimal dynamic hotel pricing*. Work. Pap., Dep. Econ., Georgetown Univ.

Cho S, Lee G, Rust J, Yu M. 2019. *Semi-parametric instrument-free demand estimation: relaxing optimality and equilibrium assumptions*. Work. Pap., Dep. Econ., Georgetown Univ., Washington, DC

Cho S, Rust J. 2010. The flat rental puzzle. *Rev. Econ. Stud.* 77:560–94

Chow CS, Tsitsiklis JN. 1989. The complexity of dynamic programming. *J. Complex.* 5:466–88

Chow GC. 1976. *Analysis and Control of Dynamic Economic Systems*. New York: Wiley

DellaVigna S. 2018. Structural behavioral economics. *In Handbook of Behavioral Economics*, Vol. 1, ed. D Bernheim, S DellaVigna, D Laibson, pp. 621–723. Amsterdam: Elsevier

Dreyfus S. 2002. Richard Bellman on the birth of dynamic programming. *Oper. Res.* 50:48–51

Eckstein Z, Wolpin K. 1989. The specification and estimation of dynamic discrete choice models. *J. Hum. Resour.* 24:562–98

Forbes Insights. 2018. What not to wear: how algorithms are taking uncertainty out of fashion. *Forbes*, July 17. **https://www.forbes.com/sites/insights-intelai/2018/07/17/what-not-to-wear-how-algorithms-are-taking-uncertainty-out-of-fashion/#3e21594186ab**

Frederick S, Loewenstein G, O'Donoghue T. 2002. Time discounting and time preference: a critical review. *J. Econ. Lit.* 40:351–401

Gallego G, van Ryzin G. 1994. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Manag. Sci.* 40:999–1020

Gittins J. 1989. *Multi-Armed Bandit Allocation Indices*. New York: Wiley

Griffiths T, Tenenbaum J. 2009. Theory-based causal induction. *Psychol. Rev.* 116:661–716

Gupta S, Rust J. 2018. *A simple theory of when and when firms go public*. Work. Pap., Dep. Econ., Georgetown Univ., Washington, DC

Hall G, Rust J. 2019. *Econometric methods for endogenously sampled time series: the case of commodity price speculation in the steel market*. Tech. Rep., Georgetown Univ., Washington, DC

Heckman J, Singer B. 2017. Abducting economics. *Am. Econ. Rev.* 107:298–302

Hutchinson J, Meyer RJ. 1994. Dynamic decision making: optimal policies and actual behavior in sequential choice problems. *Mark. Lett.* 5:369–82

Iskhakov F, Rust J, Schjerning B. 2015. Recursive lexicographical search: finding all Markov perfect equilibria of finite state directional dynamic games. *Rev. Econ. Stud.* 83:658–703

Iskhakov F, Rust J, Schjerning B. 2018. The dynamics of Bertrand price competition with cost-reducing investments. *Int. Econ. Rev.* 59(4):1681–731

Judd K. 1998. *Numerical Methods in Economics*. Cambridge, MA: MIT Press

Kahneman D. 2011. *Thinking, Fast and Slow*. New York: Farrar, Straus and Giroux

Leary M, Michaely R. 2011. Determinants of dividend smoothing: empirical evidence. *Rev. Financ. Stud.* 24:3197–249

Magnac T, Thesmar D. 2002. Identifying discrete decision processes. *Econometrica* 70:810–16

Mak T, Cheung P, Lam K, Luk W. 2011. Adaptive routing in network-on-chips using a dynamic-programming network. *IEEE Trans. Ind. Electron.* 58:3701–16

MarketsandMarkets. 2016. *Revenue management market by solutions (risk management, pricing and revenue forecast management, revenue analytics, revenue leakage detection, channel revenue management) by services (professional, managed) by deployment mode—global forecast to 2020*. Tech. Rep., MarketsandMarkets. **https://www.marketsandmarkets.com/Market-Reports/revenue-management-market-264806846.html**

Maskin E, Tirole J. 2001. Markov perfect equilibrium I: observable actions. *J. Econ. Theory* 100:191–219

Massé P. 1944. Application des probabilités en chaîne à l'hydrologie statistique et au jeu des réservoirs. *Soc. Stat. Paris* 86:204–19

McAfee P, te Velde V. 2008. Dynamic pricing with constant demand elasticity. *Prod. Oper. Manag.* 17:432–38

Miller J, Palmer R, Rust J. 1993. Behavior of trading automata in a computerized double auction market. In *The Double Auction Market: Theory, Institutions, and Laboratory Evidence*, ed. D Friedman, J Rust, pp. 155–98. Redwood City, CA: Addison Wesley

Misra S, Nair H. 2011. A structural model of sales-force compensation dynamics: estimation and field implementation. *Quant. Mark. Econ.* 9:211–57

Mnih V, Kavukcuoglu K, Silver D, Rusu A, Veness J, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518:529–33

Myerson R. 1981. Optimal auction design. *Math. Oper. Res.* 6:58–73

Nutt PC. 2002. *Why Decisions Fail*. San Francisco: Berret-Koehler Publ.

Phillips RL. 2005. *Pricing and Revenue Optimization*. Palo Alto, CA: Stanford Univ. Press

Powell W. 2010. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. New York: Wiley

Powell W, Bouzaiene-Ayari B, Lawrence C, Cheng C, Das S, Fiorillo R. 2014. Locomotive planning at Norfolk Southern: an optimizing simulator using approximate dynamic programming. *Interfaces* 44:567–78

Puterman M. 2005. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: Wiley

Renner P, Scheidegger S. 2018. *Machine learning for dynamic incentive problems*. Work. Pap., Dep. Econ., Univ. Lancaster, UK

Robbins H, Munro S. 1951. A stochastic approximation method. *Ann. Math. Stat.* 22:400–25

Rust J. 1994. Structural estimation of Markov decision processes. In *Handbook of Econometrics*, Vol. 4, ed. R Engel, D McFadden, pp. 3081–143. Amsterdam: Elsevier

Rust J. 1996. Numerical dynamic programming in economics. In *Handbook of Computational Economics*, ed. H Amman, D Kendrick, J Rust, pp. 619–730. Amsterdam: Elsevier

Rust J. 1997. Using randomization to break the curse of dimensionality. *Econometrica* 65:487–516

Rust J. 2008. Dynamic programming. In *The New Palgrave Dictionary of Economics*, Vol. 1, ed. SN Durlauf, LE Blume. New York: Palgrave Macmillan. **https://doi.org/10.1057/978-1-349-95121-5_1932-1**

Rust J. 2014. The limits of inference *with* theory: a review of Wolpin 2013. *J. Econ. Lit.* 52:820–50

Rust J. 2017. Dynamic programming, numerical. *Wiley StatsRef*, Feb. 15. **https://doi.org/10.1002/9781118445112.stat07921**

Silver D. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv:1712.01815[cs.AI]

Silver D, Schrittewieser J, Simonyan K, Antonoglu I, Huang A, et al. 2017. Mastering the game of Go without human knowledge. *Nature* 550:354–58

Simon H. 1978. *Rational decision-making in business organizations*. Nobel Memorial Lecture, Dec. 8. **https://www.nobelprize.org/uploads/2018/06/simon-lecture.pdf**

Stokey N, Lucas R. 1989. *Recursive Methods in Economic Dynamics*. Cambridge, MA: Harvard Univ. Press

Sutton R. 1988. Learning to predict by the methods of temporal differences. *Mach. Learn.* 3:9–44

Thaler R, Sunstein C. 2008. *Nudge*. New Haven, CT: Yale Univ. Press

Traub J, Werschulz AG. 1998. *Complexity and Information*. Pisa, Italy: Accad. Naz. Lincei

Tsitsiklis J. 1995. Asynchronous stochastic approximation and Q-learning. *Mach. Learn.* 16:185–202

Wald A. 1947. Foundations of a general theory of statistical decision functions. *Econometrica* 15:279–313

Watkins C. 1989. *Learning from delayed rewards*. PhD Thesis, Cambridge Univ., Cambridge, UK

Wiener N. 1948. *Cybernetics: Or Control and Communication in the Animal and the Machine*. Paris: Hermann & Cie

Wikipedia. 2018. Machine learning. *Wikipedia*. **https://en.wikipedia.org/wiki/Machine_learning**

Zhang K, Wu T, Chen S, Cai L, Peng C. 2017. A new energy efficient VM scheduling algorithm for cloud computing based on dynamic programming. In *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*, pp. 249–54. New York: IEEE