Avoiding the Curse of Dimensionality in Dynamic Stochastic Games*

Ulrich Doraszelski† Harvard University

Kenneth L. Judd‡ Hoover Institution and NBER

November 16, 2004

Abstract

Discrete-time stochastic games with a finite number of states have been widely applied to study the strategic interactions among forward-looking players in dynamic environments. However, these games suffer from a "curse of dimensionality" since the cost of computing players' expectations over all possible future states increases exponentially in the number of state variables. We explore the alternative of continuous-time stochastic games with a finite number of states, and show that continuous time has substantial computational and conceptual advantages. Most important, continuous time avoids the curse of dimensionality, thereby speeding up the computations by orders of magnitude in games with more than a few state variables. Overall, the continuous-time approach opens the way to analyze more complex and realistic stochastic games than currently feasible.

*We thank Ken Arrow, Lanier Benkard, Michaela Draganska, Sarit Markovich, Ariel Pakes, Katja Seim, Gabriel Weintraub, and the participants of SITE 2004 for their comments and suggestions.

†Cambridge, MA 02138, U.S.A., doraszelski@harvard.edu.

‡Stanford, CA 94305-6010, U.S.A., judd@hoover.stanford.edu.

1 Introduction

The usefulness of discrete-time stochastic games with a finite number of states is limited by their computational burden; in particular, there is a "curse of dimensionality" since the cost of computing players' expectations over all possible future states increases exponentially in the number of state variables. We examine the alternative of continuous-time games with a finite number of states and show that they avoid the curse of dimensionality. Hence, continuous-time games with more than a few state variables are orders of magnitude faster to solve than their discrete-time counterparts. In addition, we argue that continuous-time formulations of games are as natural, if not more natural, than discrete-time specifications. Overall, continuous time offers a computationally and conceptually promising approach to modeling dynamic strategic interactions.

Discrete-time stochastic games with a finite number of states have a long tradition in economics. Dating back to Shapley (1953), they have become central to the analysis of strategic interactions among forward-looking players in dynamic environments. A well-known example is the Ericson & Pakes (1995) (hereafter, EP) model of dynamic competition in an oligopolistic industry with investment, entry, and exit, which has triggered a large and active literature in industrial organization (see Pakes (2000) for a survey) and, most recently, has been used also in other fields such as international trade (Erdem & Tybout 2003) and finance (Goettler, Parlour & Rajan 2004). Since models like these are generally too complex to be solved analytically, Pakes & McGuire (1994) (hereafter, PM1) present an algorithm to solve numerically for a Markov perfect equilibrium.

Unfortunately, the range of applications of discrete-time, finite-state stochastic games is limited by their high computational cost. As Pakes & McGuire (2001) (hereafter, PM2) point out, computing players' expectations over all possible future states of the game is subject to a curse of dimensionality in that the computational burden is increasing exponentially in the number of state variables, i.e., the dimension of the state vector. Suppose that a player can move to one of K states from one period to the next. Given that there are K possibilities for each of N players, there are K^N possibilities for the future state of the game, and computing the expectation over all these successor states therefore involves summing over K^N terms. Because of this exponential increase of the computational burden, applications of discrete-time games are constrained to a handful of players. The computational burden also restricts heterogeneity among players. For example, a typical application of EP's framework may allow the competing firms to differ from each other in terms of either their production capacity or their product quality, but not both. In short, the computational constraints are often binding in important problems and, as Pakes (2000) contends, this causes modeling choices to "become dominated by their computational (rather than their substantive) implications" (p. 38).

In this paper we develop the alternative of continuous-time stochastic games with a

finite number of states and propose suitable algorithms.[1] To the extent that continuous-time, finite-state Markov processes are less familiar than their discrete-time counterparts, continuous-time games may be slightly more cumbersome to formulate. However, they have substantial advantages. First, continuous time avoids the curse of dimensionality in computing expectations. In contrast to a discrete-time game, the possibility of two or more players' states changing simultaneously disappears in a continuous-time game under standard assumptions on the transition laws. This is not a restriction on the behavior of players; rather it reflects the fact that changes happen one by one as time passes. The absence of simultaneous changes implies that the expectation over successor states in the discrete-time game is replaced by a much smaller sum in the continuous-time game and results in a simpler, and computationally much more tractable, model: while computing the expectation over successor states in the discrete-time game involves summing over K^N terms, it merely requires adding up $(K-1)N$ terms in the continuous-time game. This eliminates the curse of dimensionality and accelerates the computations by orders of magnitude for games with more than a few state variables. For example, the discrete-time algorithm uses over 84 hours per iteration in a model with $N = 14$ state variables and $K = 3$ possible transitions per state variable while our continuous-time algorithm uses 4.27 seconds per iteration, over $70,000$ times faster.

Second, prior to adding them up, both the continuous- and the discrete-time algorithms need to look up in computer memory each of the terms that enter in the expectation over successor states. This requires the algorithms to compute the addresses of the successor states in computer memory and imposes a further cost. One way to speed up the computations is to compute these addresses once and then store them for future reference. Precomputed addresses decrease running times but increase memory requirements. Therefore, this computational strategy is infeasible in all but the smallest discrete-time games since the number of successor states, K^N, is quite large, but it is feasible in continuous-time games since the number of successor states, $(K-1)\,N$, is much smaller. Precomputed addresses give a further advantage to continuous time: with precomputed addresses the continuous-time algorithm uses 2.93 seconds per iteration in the above example with $N = 14$ state variables compared to 4.27 seconds without precomputed addresses. Combining these gains, continuous time is over $100,000$ times faster than discrete time.

In sum, each iteration of the continuous-time algorithm is far faster than its discrete-time equivalent. Partly offsetting this is the fact that for comparable games the continuous-time algorithm needs more iterations to converge to the equilibrium. However, the loss in the number of iterations is small when compared to the gains from avoiding the curse of dimensionality and precomputed addresses. In the above example with $N = 14$ state

[1] Our approach differs from continuous-time games with a continuum of states which date back to Isaacs (1954) (zero-sum games) and Starr & Ho (1969) (nonzero-sum games); see Basar & Olsder (1999) for a standard presentation of differential games and Dockner, Jorgensen, Van Long & Sorger (2000) for a survey of applications.

variables, continuous time beats discrete time by a factor of almost $30,000$. To put this number in perspective, while it takes about 20 minutes to compute the equilibrium of the continuous-time game, it would take over a year to do the same in discrete time!

The curse of dimensionality in integration is recognized as an important problem in numerical analysis in general (see, e.g., Davis & Rabinowitz (1984) on integration). To alleviate its impact on computing equilibria of discrete-time, finite-state stochastic games, PM2 develop a stochastic approximation algorithm. Their idea is to create approximations to players' expectations over all possible future states and update them each time a state is visited by a random draw from the set of successor states. Similar to Monte Carlo integration, many visits to a state are required to reduce the approximation error to an acceptable level and obtain useful estimates of these expectations.

In addition to breaking the curse of dimensionality in computing expectations over successor states, PM2 address another issue in computing equilibria of dynamic stochastic games, namely the large size of the state space. If the states of the game are given by the Cartesian product of the states of the players, then the number of states suffers from yet another curse of dimensionality. However, many games, in particular all applications of EP's framework, make additional assumptions on the model's primitives (i.e., payoff functions and transition laws) and restrict attention to symmetric and anonymous equilibria. The number of states that have to be examined to compute a symmetric and anonymous equilibrium grows polynomially instead of exponentially in the number of state variables (see Section 3.4 for details). Even though there is no curse of dimensionality in the formal sense, the polynomial growth is arguably a challenge. The PM2 algorithm addresses it by tracking the states that appear to be visited frequently in equilibrium, i.e., are in the ergodic set, and ignoring the rest.

In this paper we compute players' values (i.e., payoffs) and policies (i.e., strategies) at all states, making no attempt to address the large size of the state space. We do this for a variety of reasons. First, many applications require knowledge of the equilibrium on states outside the ergodic set. For example, in any model of a young and growing industry, it is unlikely that the initial state and the transition path are in the ergodic set. Similarly, if the goal is to study the effect of a change in antitrust policy, then the initial state generated by the old regime may not be in the ergodic set induced by the equilibrium under the new regime, so that the transition from the old to the new regime cannot be accurately captured unless the equilibrium is computed on the transient states. In practice, this can be done via multiple restarts of the algorithm, but at additional cost. Second, as PM2 acknowledge, their algorithm needs to be significantly altered in order to solve models in which behavior depends on players' values and policies "off the equilibrium path," as is typically the case in models of collusion, since off-path states are by definition never visited in equilibrium (PM2, p. 1278). Third, the ergodic set is large in many dynamic stochastic games, so that there is little gain from focusing on the ergodic set. For example, in Doraszelski & Markovich

(2004) the ergodic set consists of the entire state space. Fourth, the number of states is independent of the concept of time. In order to contrast the discrete- and continuous-time approaches to stochastic games, we attend to issues such as those related to computing the expectation over successor states that are specific to the concept of time. We note, however, that our continuous-time algorithm can be extended to focus on the ergodic set and that this may result in improvements similar to those reported in PM2 in some applications.

Since PM2 exploit other ideas besides stochastic approximation whereas we restrict attention to the problem of computing the expectation over successor states, it is difficult to compare their algorithm with our continuous-time approach. However, to give the reader some basis for comparison, we note that PM2 report that their algorithm cuts running time roughly in half (relative to PM1) in a model with 6 state variables where the ergodic set comprises about 3.3% of all states. They also project that it reduces running time by a factor of 250 in a model with 10 state variables and an ergodic set containing 0.4% of all states. In contrast, our continuous-time approach avoids approximations altogether, computes the equilibrium on the entire state space, and still reduces running time by a factor of 12 and 524, respectively, in similar models with 6 and 10 state variables.

Besides their computational advantages, continuous-time games have a number of features that may be useful in modeling dynamic strategic interactions. First, continuous time gives the researcher more freedom to choose functional forms that are not only tractable but also easy to interpret. For example, one can more easily specify proportional depreciation in continuous-time models. Second, in continuous-time models there is no limit to the number of changes in the state of the game that can occur in any finite interval of time. This makes it easier to interpret data that does not arrive at fixed points in time. In general, the frequency of changes in the state of the game is governed by players' actions in equilibrium and not predetermined by the unit of time as in discrete-time models. Third, in continuous-time models players are able to react swiftly to changes in the strategic situation. To the extent that the state space is fairly coarse in many applications, changes typically have a significant impact on the environment and a swift reaction may thus be deemed more realistic than the delayed response of discrete-time models.

From the standpoint of theory a continuous-time model is similar to a discrete-time model with short periods. Indeed, as the length of a period goes to zero, the differences between continuous- and discrete-time models disappear. Practical considerations, however, prohibit short periods in discrete-time models. In a discrete-time model the period length is implicitly determined by the discount rate, and the lower the discount rate, the slower is the convergence of the discrete-time algorithm (see Section 5.2 for details). In a continuous-time model, on the other hand, the length of a period is essentially zero, but we show that this does not pose a problem for the continuous-time algorithm. Moreover, it is precisely in the limit that the curse of dimensionality disappears and we obtain a dramatic reduction in the computational burden. Thus, from the standpoint of computation, continuous-time

models are often superior to discrete-time models.

Overall, the computational and conceptual advantages of continuous-time games are substantial and open the way to study more complex and realistic stochastic games than currently feasible. In addition, the much smaller computational burden of continuous-time games has at least two other benefits. First, the quite large computational burden of discrete-time games often limits the researcher to computing the equilibrium for just a few sets or, in the extreme, for just one set of parameter values (e.g., Fershtman & Pakes 2000). While one parameterization is sufficient to demonstrate that something can happen in equilibrium, one parameterization is insufficient to delineate the conditions under which it does. Neither does one parameterization suffice to explore the comparative statics/dynamics properties of the equilibrium. Gaining a more thorough understanding of strategic behavior in dynamic settings therefore requires the ability to compute equilibria quickly for many different parameterizations. Second, our continuous-time approach may be useful in empirical work on stochastic games since many standard estimation procedures require computing the equilibrium hundreds or even thousands of times.[2] But even if the goal is simply to conduct policy experiments based on estimated parameters, the ability to compute equilibria quickly is key to establishing the robustness of the conclusions.

The remainder of the paper is organized as follows. Section 2 describes the basic elements of discrete- and continuous-time stochastic games with a finite number of states. Section 3 presents the computational strategies for both models and shows that continuous time avoids the curse of dimensionality inherent in discrete-time models. Section 4 formulates discrete- and continuous-time versions of the quality ladder model used in PM1. Section 5 compares the performance of the discrete- and continuous-time algorithms and Section 6 argues that continuous-time models have a number of conceptual advantages in addition to their computational advantages. Section 7 concludes.

2 Models

In this section we describe the discrete- and continuous-time approaches to finite-state stochastic games.

2.1 Discrete-Time Model

A discrete-time stochastic game with a finite number of states is often just called a "stochastic game" (Filar & Vrieze 1997, Basar & Olsder 1999). The EP model of industry dynamics is an example of this type of game. Time is discrete and the horizon is infinite. We let Ω denote the finite set of possible states; the state of the game in period t is $\omega_t \in \Omega$. We

[2]Recently two-step estimation procedures have been proposed (Aguirregabiria & Mira 2002, Bajari, Benkard & Levin 2004, Pakes, Ostrovsky & Berry 2004, Pesendorfer & Schmidt-Dengler 2003) that avoid computing the equilibrium but entail a loss of efficiency.

assume that there are N players. Player i's action (also called his control or policy) in period t is $x_t^i \in \mathbb{X}^i(\omega_t)$, where $\mathbb{X}^i(\omega_t)$ is the set of feasible actions for player i in state ω_t. We make no specific assumptions about $\mathbb{X}^i(\omega_t)$, which may be one- or multidimensional, discrete or continuous. The collection of players' actions in period t is $x_t = \left(x_t^1, \ldots, x_t^N\right)$. We follow the usual convention of letting x_t^{-i} denote $\left(x_t^1, \ldots, x_t^{i-1}, x_t^{i+1}, \ldots, x_t^N\right)$.

The state follows a controlled discrete-time, finite-state, first-order Markov process. Specifically, if the state in period t is ω_t and the players choose actions x_t, then the probability that the state in period $t+1$ is ω' is $\Pr(\omega'|\omega_t, x_t)$. In applications such as EP, ω_t is a vector partitioned into $(\omega_t^1, \ldots, \omega_t^N)$, where ω_t^i denotes the (one or more) coordinates of the state that describe player i (e.g., the player's production capacity and/or product quality). We refer to ω_t^i as the state of player i and to ω_t as the state of the game. Many applications assume that transitions in player i's state are controlled by player i's actions and are independent of the actions of other players and transitions in their states. In this case the law of motion can be written as

$$\Pr\left(\omega'|\omega_t, x_t\right) = \prod_{i=1}^{N} \Pr^i\left(\left(\omega'\right)^i|\omega_t^i, x_t^i\right),$$

where $\Pr^i\left(\left(\omega'\right)^i|\omega_t^i, x_t^i\right)$ is the transition probability for player i's state. Our example in Section 4 assumes independent transitions since this allows us to cleanly illustrate the computational advantages of continuous time but, as we point out in Section 3.3, our insights are not limited to this special case.

We decompose payoffs into two components. First, in period t player i receives a payoff equal to $\pi^i(x_t, \omega_t)$ when players' actions are x_t and the state is ω_t. For example, if ω_t is a list of firms' capacities and x_t lists their output and investment decisions, then $\pi^i(x_t, \omega_t)$ represents firm i's profit from product market competition net of investment expenses. Second, at the end of period t player i receives a payoff if there is a change in the state. Specifically, $\Phi^i(x_t, \omega_t, \omega_{t+1})$ is the change in the wealth of player i at the end of period t if the state moves from ω_t to $\omega_{t+1} \neq \omega_t$ (think of the transition as occurring at the end of the period) and players' actions were x_t.[3] For example, if a firm searches for a buyer of a piece of equipment it wants to sell and sets a reservation price, both the search effort and the reservation price are coded in x_t^i. If the firm succeeds in finding an acceptable buyer, the state changes and the firm receives a payment equal to $\Phi^i(x_t, \omega_t, \omega_{t+1})$. In general, $\Phi^i(x_t, \omega_t, \omega_{t+1})$ depends on the nature of the transition (e.g., selling some or all equipment) and may be affected by the search effort of the firm prior to the sale as well as its reservation price. While $\pi^i(x_t, \omega_t)$ is paid out at the beginning of the period, we assume that $\Phi^i(x_t, \omega_t, \omega_{t+1})$ accrues at the end. This representation of payoffs allows us to capture many features of models of industry dynamics, including entry and exit.

Players discount future payoffs using a discount factor $\beta \in [0, 1)$. The objective of player

[3] We set $\Phi^i(x_t, \omega_t, \omega_t) = 0$ without loss of generality.

i is to maximize the expected net present value of its future cash flows

$$\mathrm{E}\left\{\sum_{t=0}^{\infty}\beta^t\left(\pi^i\left(x_t,\omega_t\right)+\beta\Phi^i\left(x_t,\omega_t,\omega_{t+1}\right)\right)\right\},$$

where $\Phi^i(x_t,\omega_t,\omega_{t+1})$ is discounted (relative to $\pi^i(x_t,\omega_t)$) due to our assumption that it accrues at the end of the period after a change in the state has occurred.[4]

As is done in many applications of dynamic stochastic games, we focus on Markov perfect (a.k.a., feedback) equilibria. Hence, player i's strategy maps the set of possible states Ω into his set of feasible actions $\mathbb{X}^i(\omega_t)$. Let $V^i(\omega)$ denote the expected net present value of future cash flows to player i if the current state is ω. Suppose that the other players use strategies $X^{-i}(\omega)$. Then the Bellman equation for player i is

$$V^i\left(\omega\right)=\max_{x^i}\pi^i\left(x^i,X^{-i}\left(\omega\right),\omega\right)+\beta\mathrm{E}_{\omega'}\left\{\Phi^i\left(x^i,X^{-i}\left(\omega\right),\omega,\omega'\right)+V^i\left(\omega'\right)|\omega,x^i,X^{-i}\left(\omega\right)\right\}. \tag{1}$$

The Bellman equation adds the current cash flow of player i, $\pi^i(x^i,X^{-i}(\omega),\omega)$, to the appropriately discounted expected future cash flow,

$$\mathrm{E}_{\omega'}\left\{\Phi^i(x^i,X^{-i}\left(\omega\right),\omega,\omega')+V^i\left(\omega'\right)|\omega,x^i,X^{-i}\left(\omega\right)\right\},$$

where the expectation is taken over the successor states ω'. Player i's strategy is given by

$$X^i\left(\omega\right)=\arg\max_{x^i}\pi^i\left(x^i,X^{-i}\left(\omega\right),\omega\right)+\beta\mathrm{E}_{\omega'}\left\{\Phi^i\left(x^i,X^{-i}\left(\omega\right),\omega,\omega'\right)+V^i\left(\omega'\right)|\omega,x^i,X^{-i}\left(\omega\right)\right\}. \tag{2}$$

Each player has his own version of equations (1) and (2). The system of equations defined by the collection of (1) and (2) for each player $i=1,\ldots,N$ and each state $\omega\in\Omega$ defines a Markov perfect equilibrium.[5]

2.2 Continuous-Time Model

We next describe the continuous-time stochastic game with a finite number of states. As with the discrete-time model, the horizon is infinite, the state of the game at time t is $\omega_t\in\Omega$, there are N players, and player i's action at time t is denoted by $x_t^i\in\mathbb{X}^i(\omega_t)$. The key difference is that the state in the continuous-time model follows a controlled continuous-time, finite-state Markov process. In discrete time the time path of the state is a sequence, but in continuous time the path is a piecewise-constant, right-continuous function of time. Jumps occur at random times according to a controlled nonhomogenous Poisson process. At time t the hazard rate of a jump occurring is $\phi(x_t,\omega_t)$. If a jump occurs at time t, then

[4]Discounting $\Phi^i(\cdot)$ is without loss of generality because it can always be replaced by $\tilde{\Phi}^i(\cdot)=\beta\Phi^i(\cdot)$, the net present value of $\Phi^i(\cdot)$ at the beginning of the period.

[5]Similar to PM1 and PM2, we restrict attention to pure-strategy equilibria. Doraszelski & Satterthwaite (2003) establish the existence of such equilibria in EP's framework.

the probability that the state moves to ω' is $f(\omega'|\omega_{t^-}, x_{t^-})$, where $\omega_{t^-} = \lim_{s \to t^-} \omega_s$ is the state just before the jump and $x_{t^-} = \lim_{s \to t^-} x_s$ are players' actions just before the jump. That is, $f(\omega'|\omega_{t^-}, x_{t^-})$ characterizes the transitions of the embedded first-order Markov process. Since a jump from a state to itself does not change the game, we simply ignore it and instead adjust, without loss of generality, the hazard rate of a jump occurring so that $f(\omega_{t^-}|\omega_{t^-}, x_{t^-}) = 0$.

This decomposition of jumps into a hazard rate and a transition probability is a convenient representation of the controlled continuous-time, finite-state Markov process. Over a short interval of time of length $\Delta > 0$ the law of motion is

$$\begin{aligned} \Pr(\omega_{t+\Delta} \neq \omega_t | \omega_t, x_t) &= \phi(x_t, \omega_t)\Delta + O(\Delta^2), \\ \Pr(\omega_{t+\Delta} = \omega' | \omega_t, x_t, \omega_{t+\Delta} \neq \omega_t) &= f(\omega' | \omega_t, x_t) + O(\Delta). \end{aligned}$$

In the special case of independent transitions, player i's state evolves according to

$$\begin{aligned} \Pr^i(\omega^i_{t+\Delta} \neq \omega^i_t | \omega^i_t, x^i_t) &= \phi^i(x^i_t, \omega^i_t)\Delta + O(\Delta^2), \\ \Pr^i\left(\omega^i_{t+\Delta} = (\omega')^i | \omega^i_t, x^i_t, \omega^i_{t+\Delta} \neq \omega^i_t\right) &= f^i\left((\omega')^i | \omega^i_t, x^i_t\right) + O(\Delta), \end{aligned}$$

and $\phi(x_t, \omega_t) = \sum_{i=1}^N \phi^i(x^i_t, \omega^i_t)$ is the hazard rate of a change in the state of the game. This last equality reveals a critical fact about continuous-time Markov processes: during a short interval of time, there will be (with probability infinitesimally close to one) at most one jump. In the discrete-time model we must keep track of all possible combinations of players' transitions between time t and time $t+1$. The possibility of two or more players' states changing simultaneously disappears in the continuous-time model; this results in a simpler, and computationally much more tractable, model.

The remaining aspects of the continuous-time model are essentially the same as in the discrete-time model. The payoff of player i consists of two components. First, player i receives a payoff flow equal to $\pi^i(x_t, \omega_t)$ when players' actions are x_t and the state is ω_t. Second, $\Phi^i(x_{t^-}, \omega_{t^-}, \omega_t)$ is the instantaneous change in the wealth of player i at time t if the state moves from ω_{t^-} to $\omega_t \neq \omega_{t^-}$ and players' actions just before the jump were x_{t^-}. Similar to the discrete-time model, $\pi^i(x_t, \omega_t)$ may capture firm i's profit from product market competition net of investment expenses and $\Phi^i(x_{t^-}, \omega_{t^-}, \omega_t)$ the scrap value that the firm receives upon exiting the industry or the setup cost that it incurs upon entering the industry. Unlike the discrete-time model, there is a clear-cut distinction between $\pi^i(x_t, \omega_t)$ and $\Phi^i(x_{t^-}, \omega_{t^-}, \omega_t)$ in the continuous-time model: $\pi^i(x_t, \omega_t)$ represents a flow of money, expressed in dollars per unit of time, whereas $\Phi^i(x_{t^-}, \omega_{t^-}, \omega_t)$ represents a change in the stock of wealth, expressed in dollars. As in the discrete-time game, this representation of payoffs can represent many dynamic phenomena; for example, the Appendix gives details on modeling entry and exit in our continuous-time game.

Players discount future payoffs using a discount rate $\rho > 0$. The objective of player i is

to maximize the expected net present value of its future cash flows

$$\mathrm{E}\left\{\int_0^\infty e^{-\rho t}\pi^i\left(x_t,\omega_t\right)dt+\sum_{m=1}^{\infty}e^{-\rho T_m}\Phi^i\left(x_{T_m^-},\omega_{T_m^-},\omega_{T_m}\right)\right\},$$

where T_m is the random time of the m'th jump in the state, $x_{T_m^-}$ are players' actions just before the m'th jump, $\omega_{T_m^-}$ is the state just before the m'th jump, and ω_{T_m} is the state just after the m'th jump.

The Bellman equation for player i is similar to the one in discrete time. To see this note that over a short interval of time of length $\Delta>0$ player i solves the dynamic programming problem given by

$$\begin{aligned}V^i\left(\omega\right) &= \max_{x^i}\pi^i\left(x^i,X^{-i}(\omega),\omega\right)\Delta\\ &\quad+\left(1-\rho\Delta\right)\Bigg\{\left(1-\phi\left(x^i,X^{-i}\left(\omega\right),\omega\right)\Delta-O\left(\Delta^2\right)\right)V^i\left(\omega\right)\\ &\quad+\left(\phi\left(x^i,X^{-i}\left(\omega\right),\omega\right)\Delta+O\left(\Delta^2\right)\right)\\ &\qquad\times\Big(\mathrm{E}_{\omega'}\left\{\Phi^i\left(x^i,X^{-i}\left(\omega\right),\omega,\omega'\right)+V^i\left(\omega'\right)|\omega,x^i,X^{-i}\left(\omega\right)\right\}+O\left(\Delta\right)\Big)\Bigg\},\end{aligned}$$

which, as $\Delta\to 0$, simplifies to the Bellman equation

$$\begin{aligned}\rho V^i\left(\omega\right) &= \max_{x^i}\pi^i\left(x^i,X^{-i}\left(\omega\right),\omega\right)-\phi\left(x^i,X^{-i}\left(\omega\right),\omega\right)V^i\left(\omega\right)\\ &\quad+\phi\left(x^i,X^{-i}\left(\omega\right),\omega\right)\mathrm{E}_{\omega'}\left\{\Phi^i(x^i,X^{-i}\left(\omega\right),\omega,\omega')+V^i\left(\omega'\right)|\omega,x^i,X^{-i}\left(\omega\right)\right\}\quad(3)\end{aligned}$$

Hence, $V^i(\omega)$ can be interpreted as the asset value to player i of participating in the game. This asset is priced by requiring that the opportunity cost of holding it, $\rho V^i(\omega)$, equals the current cash flow, $\pi^i(x^i,X^{-i}\left(\omega\right),\omega)$, plus the expected capital gain or loss conditional on a jump occurring,

$$\mathrm{E}_{\omega'}\left\{\Phi^i(x^i,X^{-i}\left(\omega\right),\omega,\omega')+V^i\left(\omega'\right)|\omega,x^i,X^{-i}\left(\omega\right)\right\}-V^i(\omega),$$

times the hazard rate of a jump occurring, $\phi(x^i,X^{-i}\left(\omega\right),\omega)$.

In the special case of independent transitions, player i solves the problem

$$\begin{aligned}
V^i(\omega) &= \max_{x^i} \pi^i\left(x^i, X^{-i}(\omega), \omega\right)\Delta \\
&\quad + (1-\rho\Delta)\Bigg\{\left(1-\phi\left(x^i, X^{-i}(\omega), \omega\right)\Delta - O\left(\Delta^2\right)\right)V^i(\omega) \\
&\quad + \left(\phi^i\left(x^i, \omega^i\right)\Delta + O\left(\Delta^2\right)\right) \\
&\qquad \times\Big(\mathrm{E}_{(\omega')^i}\left\{\Phi^i\left(x^i, X^{-i}(\omega), \omega, \left(\omega'\right)^i, \omega^{-i}\right) + V^i\left(\left(\omega'\right)^i, \omega^{-i}\right)|\omega^i, x^i\right\} + O(\Delta)\Big) \\
&\quad + \sum_{j\neq i}\left(\phi^j\left(X^j(\omega), \omega^j\right)\Delta + O\left(\Delta^2\right)\right) \\
&\qquad \times\Big(\mathrm{E}_{(\omega')^j}\left\{\Phi^i\left(x^i, X^{-i}(\omega), \omega, \left(\omega'\right)^j, \omega^{-j}\right) + V^i\left(\left(\omega'\right)^j, \omega^{-j}\right)|\omega^j, X^j(\omega)\right\} + O(\Delta)\Big)\Bigg\},
\end{aligned}$$

which, as $\Delta \to 0$, simplifies to the Bellman equation

$$\begin{aligned}
\rho V^i(\omega) &= \max_{x^i} \pi^i\left(x^i, X^{-i}(\omega), \omega\right) - \phi\left(x^i, X^{-i}(\omega), \omega\right)V^i(\omega) \\
&\quad + \phi^i\left(x^i, \omega^i\right)\mathrm{E}_{(\omega')^i}\left\{\Phi^i\left(x^i, X^{-i}(\omega), \omega, \left(\omega'\right)^i, \omega^{-i}\right) + V^i\left(\left(\omega'\right)^i, \omega^{-i}\right)|\omega^i, x^i\right\} \\
&\quad + \sum_{j\neq i}\phi^j\left(X^j(\omega), \omega^j\right)\mathrm{E}_{(\omega')^j}\left\{\Phi^i\left(x^i, X^{-i}(\omega), \omega, \left(\omega'\right)^j, \omega^{-j}\right) + V^i\left(\left(\omega'\right)^j, \omega^{-j}\right)|\omega^j, X^j(\omega))\right\}.
\end{aligned} \tag{4}$$

Similar to the discrete-time model, player i's strategy is found by carrying out the maximization on the RHS of equation (3) or (4).

3 Computational Strategies

Next we present the computational strategies for the discrete- and continuous-time models and show that continuous time avoids the curse of dimensionality inherent in discrete-time models.

3.1 Discrete-Time Algorithm

The algorithm is iterative. First we order the states in Ω and make initial guesses for the value $V^i(\omega)$ and the policy $X^i(\omega)$ of each player $i = 1, \ldots, N$ in each state $\omega \in \Omega$. Then we update these guesses as we proceed through the state space in the pre-specified order. Specifically, in state $\omega \in \Omega$, given old guesses $V^i(\omega)$ and $X^i(\omega)$ we compute new guesses

$\hat{V}^i(\omega)$ and $\hat{X}^i(\omega)$ for each player $i = 1, \ldots, N$ as follows:

$$\begin{aligned}
\hat{X}^i(\omega) &\leftarrow \arg\max_{x^i} \pi^i\left(x^i, X^{-i}(\omega), \omega\right) \\
&\quad +\beta \mathrm{E}_{\omega'}\left\{\Phi^i\left(x^i, X^{-i}(\omega), \omega, \omega'\right) + V^i\left(\omega'\right) | \omega, x^i, X^{-i}(\omega)\right\}, \qquad (5) \\
\hat{V}^i(\omega) &\leftarrow \pi^i\left(\hat{X}^i(\omega), X^{-i}(\omega), \omega\right) \\
&\quad +\beta \mathrm{E}_{\omega'}\left\{\Phi^i\left(\hat{X}^i(\omega), X^{-i}(\omega), \omega, \omega'\right) + V^i\left(\omega'\right) | \omega, \hat{X}^i(\omega), X^{-i}(\omega)\right\}. \qquad (6)
\end{aligned}$$

Note that the old guesses for the policies of player i's opponents, $X^{-i}(\omega)$, and the old guess for player i's value, $V^i(\omega)$, are used when computing the new guesses $\hat{V}^i(\omega)$ and $\hat{X}^i(\omega)$. This procedure is, therefore, a Gauss-Jacobi scheme at each state $\omega \in \Omega$.

There are two ways to update $V^i(\omega)$ and $X^i(\omega)$. PM1 suggest a Gauss-Jacobi scheme that computes $\hat{V}^i(\omega)$ and $\hat{X}^i(\omega)$ for all players $i = 1, \ldots, N$ and *all* states $\omega \in \Omega$ before replacing the old guesses with the new guesses, as in

$$\begin{aligned}
X^i(\omega) &\leftarrow \hat{X}^i(\omega), \\
V^i(\omega) &\leftarrow \hat{V}^i(\omega).
\end{aligned}$$

Their value function iteration approach is also called a pre-Gauss-Jacobi method in the literature on nonlinear equations (see Judd (1998) for a more extensive discussion of Gauss-Jacobi and Gauss-Seidel methods). In contrast to PM1, we employ the block Gauss-Seidel scheme that is typically used for discrete-time stochastic games with a finite number of states (e.g., Benkard 2004). In our block Gauss-Seidel scheme, immediately after computing $\hat{V}^i(\omega)$ and $\hat{X}^i(\omega)$ for all players $i = 1, \ldots, N$ and a *given* state $\omega \in \Omega$, we replace the old guesses with the new guesses for that state. This has the advantage that "information" is used as soon as it becomes available. The algorithm cycles through the state space until the changes in the value and policy functions are deemed small.

3.2 Continuous-Time Algorithm

In its basic form our computational strategy adapts the block Gauss-Seidel scheme to the continuous-time model. The sole change is that to update players' values and policies in

state $\omega \in \Omega$, we replace equations (5) and (6) by

$$\begin{aligned}
\hat{X}^i(\omega) &\leftarrow \arg\max_{x^i} \pi^i\left(x^i, X^{-i}(\omega), \omega\right) - \phi\left(x^i, X^{-i}(\omega), \omega\right) V^i\left(x^i, X^{-i}(\omega), \omega\right) \\
&\quad + \phi\left(x^i, X^{-i}(\omega), \omega\right) \mathrm{E}_{\omega'}\left\{\Phi^i\left(x^i, X^{-i}(\omega), \omega, \omega'\right) + V^i\left(\omega'\right) | \omega, x^i, X^{-i}(\omega)\right\}, \quad (7) \\
\hat{V}^i(\omega) &\leftarrow \frac{1}{\rho + \phi\left(\hat{X}^i(\omega), X^{-i}(\omega), \omega\right)} \pi^i\left(\hat{X}^i(\omega), X^{-i}(\omega), \omega\right) \\
&\quad + \frac{\phi\left(\hat{X}^i(\omega), X^{-i}(\omega), \omega\right)}{\rho + \phi\left(\hat{X}^i(\omega), X^{-i}(\omega), \omega\right)} \\
&\quad \times \mathrm{E}_{\omega'}\left\{\Phi^i\left(\hat{X}^i(\omega), X^{-i}(\omega), \omega, \omega'\right) + V^i\left(\omega'\right) | \omega, \hat{X}^i(\omega), X^{-i}(\omega)\right\}. \quad (8)
\end{aligned}$$

The remainder of the algorithm proceeds as before. Note that by dividing through by $\rho + \phi(\hat{X}^i(\omega), X^{-i}(\omega), \omega)$, we ensure that equation (8) is contractive for a given player (holding fixed the policies of all players) since

$$\frac{\phi(\hat{X}^i(\omega), X^{-i}(\omega), \omega)}{\rho + \phi(\hat{X}^i(\omega), X^{-i}(\omega), \omega)} < 1$$

as long as the hazard rate is bounded above. Note that the contraction factor varies with players' policies. In the discrete-time model, by contrast, the contraction factor equals the discount factor β. Unfortunately, the system of equations that defines the equilibrium is not contractive, and hence neither our continuous- nor our discrete-time algorithm is guaranteed to converge.

3.3 Avoiding the Curse of Dimensionality

The key difficulty of the discrete-time model is computing the expectation over successor states in equations (5) and (6). Dropping the distinction between old and new guesses and setting $\Phi^i(X(\omega), \omega, \omega') = 0$ to simplify the notation, this expectation is

$$\mathrm{E}_{\omega'}\left\{V^i\left(\omega'\right) | \omega, X(\omega)\right\} = \sum_{\{\omega' : \Pr(\omega' | \omega, X(\omega)) > 0\}} V^i\left(\omega'\right) \Pr\left(\omega' | \omega, X(\omega)\right), \quad (9)$$

which involves summing over all states ω' such that $\Pr(\omega' | \omega, X(\omega)) > 0$. A clean case arises if transitions are independent across players and each transition is restricted to going one level up, one level down, or staying the same, i.e., $(\omega')^i \in \left\{\omega^i - 1, \omega^i, \omega^i + 1\right\}$. Then the expectation consists of 3^N terms,

$$\mathrm{E}_{\omega'}\left\{V^i\left(\omega'\right) | \omega, X(\omega)\right\} = \sum_{\left\{\omega' : (\omega')^i \in \{\omega^i - 1, \omega^i, \omega^i + 1\}, i = 1, \ldots, N\right\}} V^i\left(\omega'\right) \prod_{i=1}^{N} \Pr^i\left(\left(\omega'\right)^i | \omega^i, X^i(\omega)\right).$$

More generally, if each player can move to one of K states, then the expectation involves summing over K^N terms and grows exponentially in N.

The main advantage of the continuous-time model now becomes clear. If transitions are independent across players and each transition is limited to going one level up or down, i.e., $\omega_{t+1}^i \in \left\{\omega_t^i - 1, \omega_t^i + 1\right\}$, then the N-dimensional expectation over successor states decomposes into N one-dimensional expectations, each of which consists of 2 terms.[6] In fact, we have

$$\mathrm{E}_{(\omega')^j}\left\{V^i\left(\left(\omega'\right)^j, \omega^{-j}\right)|\omega^j, X^j(\omega)\right\} = \sum_{(\omega')^j \in \{\omega^j - 1, \omega^j + 1\}} V^i\left(\left(\omega'\right)^j, \omega^{-j}\right) f^j\left(\left(\omega'\right)^j |\omega^j, X^j(\omega)\right).$$

In the continuous-time model, we need to sum over a total of $2N$ terms compared to 3^N terms in the discrete-time model. More generally, if each player can move to one of K states, then computing the expectation over successor states involves summing over $(K-1)N$ terms in the continuous-time model but K^N terms in the discrete-time model. Since $(K-1)N$ grows linearly rather than exponentially with N, computing the expectation over successor states is no longer subject to the curse of dimensionality.

The curse of dimensionality becomes even more severe in applications where each player is described by $D > 1$ coordinates of the state (e.g., Benkard 2004, Langohr 2003). In this case computing the expectation over successor states in the discrete-time model involves summing over K^{ND} terms compared to $(K-1)ND$ terms in the continuous-time model. What matters is the total number of coordinates of the state vector. The curse of dimensionality is just as severe in a single-agent dynamic programming problem with a ND-dimensional state vector as in a N-player discrete-time stochastic game with a ND-dimensional state vector. Similarly, common states that affect the current payoffs of all players are computationally more burdensome in the discrete- than in the continuous-time model. Suppose, for example, that in addition to players' states that describe firm-specific production capacities there is a common state such as industry demand (e.g., Besanko & Doraszelski 2004). If the common state can move to L possible levels and each player can move to one of K states, then the summation is over LK^N terms in discrete time but $L - 1 + (K-1)N$ terms in continuous time.

In contrast to common states, common shocks that affect the states the players can move to in a uniform fashion contribute equally to the number of summands in both models. EP, for example, assume that firm i's state evolves according to the law of motion $\left(\omega'\right)^i = \omega^i + \tau^i - \eta$, where $\tau^i \in \{0,1\}$ is a binary random variable governed by firm i's investment decision and $\eta \in \{0,1\}$ is an industry-wide depreciation shock. Hence, computing the expectation over successor states in the discrete-time model involves summing over 2×2^N terms compared to $2 \times 2N$ terms in the continuous-time model. Nevertheless, as long as the

[6] Here we exploit the fact that, unlike in the discrete-time model, there is no need to explicitly consider the possibility of remaining in the same state.

transition probabilities for the coordinates of the state exhibit less than perfect correlation the continuous-time model has a significant advantage over the discrete-time model.[7]

3.4 Precomputed Addresses, Symmetry, and Anonymity

The first advantage of continuous time is that it avoids the curse of dimensionality in computing the expectation over successor states. We next describe a way to further speed up this computation. To understand this suggestion we need to briefly discuss the nuts-and-bolts of computer storage. Any algorithm must store the value and policy functions in some table that we denote $\mathbb{M}$. Each row of this table corresponds to a state $\omega \in \Omega$ and contains the vector $\left(V^1(\omega), \ldots, V^N(\omega), X^1(\omega), \ldots, X^N(\omega)\right)$ of values and policies for all players in that state. Consider the expectation over successor states in the discrete-time model as given by equation (9). To compute this sum, the algorithm must find the rows and columns with the relevant information in table $\mathbb{M}$, implying that the sum is really

$$\sum_{\{\omega' : \Pr(\omega'|\omega, \mathbb{M}[R(\omega),(N+1,\ldots,2N)])>0\}} \mathbb{M}\left[R\left(\omega'\right), C\left(\omega', i\right)\right] \Pr\left(\omega'|\omega, \mathbb{M}\left[R\left(\omega\right), (N+1, \ldots, 2N)\right]\right), \tag{10}$$

where $C\left(\omega', i\right)$ is the column in row $R\left(\omega'\right)$ that contains the value for player i in state ω' and $N+1, \ldots, 2N$ are the columns in row $R(\omega)$ that contain the policies for players $j = 1, \ldots, N$ in state ω. In the continuous-time model the expression for the expectation over successor states is analogous except that $\Pr(\cdot)$ is replaced by $f(\cdot)$. Equation (10) displays all the computations that must occur in evaluating $\mathrm{E}_{\omega'}\left\{V^i\left(\omega'\right)|\omega, X(\omega)\right\}$ and emphasizes that there are two kinds of costs involved. The first is the summation over all states ω' such that $\Pr(\omega'|\omega, X(\omega)) > 0$ and the second is the computation of the address, $R(\omega')$ and $C(\omega', i)$, of the value of player i at each of them. One way to reduce running times is to precompute these addresses and store them along with the values and policies for state ω. More precisely, for each successor state ω' of state ω we append a vector $(R(\omega'), C(\omega', 1), \ldots, C(\omega', N))$ of precomputed addresses to the vector $\left(V^1(\omega), \ldots, V^N(\omega), X^1(\omega), \ldots, X^N(\omega)\right)$ of values and policies.

Precomputed addresses decrease running times but increase memory requirements since $N+1$ numbers need to be stored for each successor state. The practicality of this computational strategy hinges on the number of successor states. As we have shown in Section 3.3, this number is much smaller in the continuous- than in the discrete-time model. For example, if transitions are independent across players and each transition is restricted to going one level up, one level down, or staying the same, then there are $2N$ successor states in the continuous-time model but 3^N in the discrete-time model. Hence, this computational

[7] It is possible to specify models that are as demanding in continuous time than they are in discrete time. Consider, for example, the law of motion that assigns equal probability to transitions from any state $\omega \in \Omega$ to any other state $\omega' \in \Omega$, where $\omega' \neq \omega$: $\Pr\left(\omega'|\omega, x\right) = \frac{1}{|\Omega|-1}$ in discrete time translates into $\phi(x, \omega) = 1$ and $f\left(\omega'|\omega, x\right) = \frac{1}{|\Omega|-1}$ in continuous time, and the expectation over successor states involves $|\Omega| - 1$ terms in both cases. We are, however, not aware of an economic problem that leads to such a specification.

strategy is infeasible except in the smallest discrete-time models. Precomputed addresses are therefore another advantage that is essentially only available in continuous time.

The usefulness of precomputed addresses further depends on how hard it is to evaluate $R(\omega)$ and $C(\omega, i)$. In some cases, this is quite easy and there is little to be gained from this computational strategy. For example, suppose that the set of player i's possible states is $\{1, \ldots, M\}$. In the absence of restrictions such as symmetry and anonymity the state space is $\Omega = \{1, \ldots, M\}^N$. Hence, ω is the base M representation of

$$R(\omega) = \omega^1 + \left(\omega^2 - 1\right) M + \left(\omega^3 - 1\right) M^2 + \ldots + \left(\omega^N - 1\right) M^{N-1}$$

and $C(\omega, i) = i$.

Evaluating $R(\omega)$ and $C(\omega, i)$, however, becomes much harder once symmetry and anonymity are invoked, as is always done in applications of EP's framework in order to slow down the growth of the state space in N and M. Under suitable conditions on the model's primitives (i.e., payoff functions and transition laws), it is possible to restrict attention to symmetric and anonymous equilibria. In a symmetric equilibria, if $V^1(\omega)$ denotes player 1's value in state $\omega = \left(\omega^1, \omega^2, \ldots, \omega^N\right)$, then player i's value is given by

$$V^i(\omega) = V^1(\omega^i, \omega^2, \ldots, \omega^{i-1}, \omega^1, \omega^{i+1}, \ldots)$$

and similarly for player i's policy. Therefore, symmetry allows us to focus on the problem of player 1. Furthermore, anonymity (also called exchangeability) says that player 1 does not care about the identity of its competitors, only about the distribution of their states. Hence, for all $2 \leq j < k$,

$$V^1(\omega) = V^1(\omega^1, \ldots, \omega^{j-1}, \omega^k, \omega^{j+1}, \ldots, \omega^{k-1}, \omega^j, \omega^{k+1}, \ldots)$$

and similarly for player i's policy (see Doraszelski & Satterthwaite (2003) for a detailed discussion of symmetry and anonymity).

In practice, symmetry and anonymity are imposed by limiting the computation of players' values and policies to states in the set $\bar{\Omega} = \{(\omega_1, \omega_2, \ldots, \omega_N) \in \Omega : \omega_1 \leq \omega_2 \leq \ldots \leq \omega_N\}$.[8] Whereas Ω grows exponentially in N, $\bar{\Omega}$ grows polynomially. More specifically, imposing symmetry and anonymity reduces the number of states to be examined from $|\Omega| = M^N$ to $|\bar{\Omega}| = \frac{(N+M-1)!}{N!(M-1)!}$, but makes $R(\omega)$ and $C(\omega, i)$ much harder to compute. Pakes, Gowrisankaran & McGuire (1993) and Gowrisankaran (1999) propose slightly different methods for mapping the elements of $\bar{\Omega}$ into consecutive integers. These methods form the basis for computing $R(\omega)$, but require that $\omega \in \bar{\Omega}$. While this is achieved by sorting the coordinates of the vector ω, sorting implies that $C(\omega, i)$ is no longer always equal to i. Suppose that the state of the game is $(1, 1, 3)$ and that firm 1 moves to state 2. Hence, the

[8] Some additional restrictions are needed to obtain a symmetric and anonymous equilibria. If $N = 2$, for example, symmetry requires that $V^1(1, 1) = V^2(1, 1)$.

state becomes $(2,1,3)$ or, after sorting, $(1,2,3)$ so that $C((2,1,3),1)=2$, $C((2,1,3),2)=1$, and $C((2,1,3),3)=3$. Since evaluating $R(\omega)$ and $C(\omega,i)$ is rather involved, there is a lot to be gained from precomputed addresses, see Section 5.

4 Example: The Pakes & McGuire (1994) Quality Ladder Model

We use the quality ladder model developed in PM1 to demonstrate the computational advantages of continuous time in Section 5. Below we first describe their model and then we reformulate it in continuous time. We want to focus on a simple example that highlights the numerical issues. We also want to avoid existence problems that may arise from the discrete nature of firms' entry and exit decision (see Doraszelski & Satterthwaite (2003) for details and a way to resolve these difficulties). Therefore, we abstract from entry and exit, and set $\Phi^i(x,\omega,\omega')=0$. This allows us to make clean performance comparisons between the discrete- and continuous-time algorithms. Since entry and exit are important features of the EP model of industry dynamics we describe in the Appendix how to add them to the continuous-time model.

4.1 Discrete-Time Model

The quality ladder model assumes that there are N firms with vertically differentiated products engaged in price competition. Firm i produces a product of quality ω^i. Quality is assumed to be discrete, i.e., $\omega^i \in \{1,\ldots,M\}$, and evolves over time in response to investment and depreciation. The state space is $\Omega=\{1,\ldots,M\}^N$. We first describe price competition and then turn to quality dynamics.

Demand Each consumer purchases at most one unit of one product. The utility consumer k derives from purchasing product i is $g(\omega^i)-p^i+\epsilon^{ik}$, where

$$g(\omega^i)=\begin{cases} 3\omega^i-4, & \omega^i\le 5,\\ 12+\ln\left(2-\exp\left(16-3\omega^i\right)\right), & \omega^i>5\end{cases}$$

maps the quality of the product into the consumer's valuation for it and ϵ^{ik} represents taste differences among consumers. There is a no-purchase alternative, product 0, which has utility ϵ^{0k}. We assume that the idiosyncratic shocks $\epsilon^{0k},\epsilon^{1k},\ldots,\epsilon^{Nk}$ are independently and identically extreme value distributed across products and consumers; therefore, the demand for firm i's product is

$$q^i(p^1,\ldots,p^N;\omega)=m\frac{\exp\left(g(\omega^i)-p^i\right)}{1+\sum_{j=1}^N\exp\left(g(\omega^j)-p^j\right)},$$

where $m > 0$ is the size of the market (the measure of consumers).

Price competition In each period, firm i observes the quality of its and its rivals' products and chooses the price p^i of product i to maximize profits, thereby solving

$$\max_{p^i \geq 0} q^i(p^1, \ldots, p^N; \omega) \left(p^i - c\right),$$

where $c \geq 0$ is the marginal cost of production. The first-order condition of firm i is

$$0 = \frac{\partial}{\partial p^i} q^i(p^1, \ldots, p^N; \omega) \left(p^i - c\right) + q^i(p^1, \ldots, p^N; \omega).$$

It can be shown that in a given state ω there exists a unique Nash equilibrium $\left(p^1(\omega), \ldots, p^N(\omega)\right)$ of the product market game (see Caplin & Nalebuff 1991). The Nash equilibrium is found easily by numerically solving the system of first-order conditions.

Law of motion Firm i's state ω^i represents the quality of its product in the present period. The quality of firm i's product in the subsequent period is determined by its investment $x^i \geq 0$ in quality improvements and by depreciation. The outcomes of the investment and depreciation processes are assumed to be stochastic. If the investment is successful, then the quality increases by one level. Expenditures in investment enhance the probability of success; more specifically, the probability of success is $\frac{\alpha x^i}{1+\alpha x^i}$, where $\alpha > 0$ is a measure of the effectiveness of investment. If the firm is hit by a depreciation shock, then the quality decreases by one level; this happens with probability $\delta \in [0, 1]$. Note that we differ from the original quality ladder model of PM1 in that our depreciation shocks are independent across firms whereas PM1 assume an industry-wide depreciation shock. We do this to focus on the key issue related to the curse of dimensionality in discrete-time models.

Combining the investment and depreciation processes, if $\omega^i \in \{2, \ldots, M-1\}$, then the quality of firm i's product changes according to the transition probability

$$\Pr^i(\left(\omega'\right)^i | \omega^i, x^i) = \begin{cases} \frac{(1-\delta)\alpha x^i}{1+\alpha x^i}, & \left(\omega'\right)^i = \omega^i + 1, \\ \frac{1-\delta+\delta\alpha x^i}{1+\alpha x^i}, & \left(\omega'\right)^i = \omega^i, \\ \frac{\delta}{1+\alpha x^i}, & \left(\omega'\right)^i = \omega^i - 1. \end{cases}$$

Since firm i cannot move further down (up) from the lowest (highest) product quality, we set

$$\begin{aligned} \Pr^i(\left(\omega'\right)^i | 1, x^i) &= \begin{cases} \frac{(1-\delta)\alpha x^i}{1+\alpha x^i}, & \left(\omega'\right)^i = 2, \\ \frac{1+\delta\alpha x^i}{1+\alpha x^i}, & \left(\omega'\right)^i = 1, \end{cases} \\ \Pr^i(\left(\omega'\right)^i | M, x^i) &= \begin{cases} \frac{1-\delta+\alpha x^i}{1+\alpha x^i}, & \left(\omega'\right)^i = M, \\ \frac{\delta}{1+\alpha x^i}, & \left(\omega'\right)^i = M - 1. \end{cases} \end{aligned}$$

Payoff function The per-period payoff of firm i is derived from the Nash equilibrium of the product market game and given by

$$\pi^i(x,\omega) \equiv q^i(p^1(\omega),\ldots,p^N(\omega);\omega)(p^i(\omega)-c) - x^i,$$

where we have subtracted investment x^i from the profit from price competition.

Parameterization We use the same parameter values as PM1. The size of the market is $m = 5$, the marginal cost of production is $c = 5$, the effectiveness of investment is $\alpha = 3$, and the depreciation probability is $\delta = 0.7$. We follow PM1 in first assuming that the discount factor is $\beta = 0.925$, which corresponds to a yearly interest rate of 8.1%, and that the number of quality levels per firm is $M = 18$, but we also examine other values for β and M in Section 5.

4.2 Continuous-Time Model

In the interest of brevity, we start by noting that the details of price competition remain unchanged. In the continuous-time model we can thus reinterpret $\pi^i(x,\omega)$ as the payoff flow of firm i.

Law of motion To make the continuous- and discrete-time models comparable, we use the same law of motion as described for the discrete-time model. Therefore, the hazard rate for the investment project of firm i being successful is given by $\frac{\alpha x^i}{1+\alpha x^i}$, the same choice as for the success probability in the discrete-time model. This is appropriate since the expected time to the first success is $\frac{1+\alpha x^i}{\alpha x^i}$ in both models. Similarly, the depreciation hazard in the continuous-time model equals the depreciation probability, δ, in the discrete-time model.

Jumps in firm i's state thus occur according to a Poisson process with hazard rate

$$\phi^i(x^i,\omega^i) = \frac{\alpha x^i}{1+\alpha x^i} + \delta,$$

and when a jump occurs, firm i's state changes according to the transition probability

$$f^i(\left(\omega'\right)^i|\omega^i,x^i) = \begin{cases} \frac{\alpha x^i}{(1+\alpha x^i)\phi^i(x^i,\omega^i)}, & \left(\omega'\right)^i = \omega^i + 1, \\ \frac{\delta}{\phi^i(x^i,\omega^i)}, & \left(\omega'\right)^i = \omega^i - 1 \end{cases}$$

if $\omega^i \in \{2,\ldots,M-1\}$. Since firm i cannot move further down (up) from the lowest (highest) product quality, we set

$$\begin{aligned} \phi^i(x^i,1) &= \frac{\alpha x^i}{1+\alpha x^i}, \qquad & f^i(2|1,x^i) &= 1, \\ \phi^i(x^i,M) &= \delta, \qquad & f^i(M-1|M,x^i) &= 1. \end{aligned}$$

Parameterization Whenever possible we use the same parameter values in the continuous- as in the discrete-time model. Moreover, we can easily match the discrete-time discount factor β to the continuous-time discount rate ρ: if Δ is the unit of time in the discrete-time model, then β and ρ are related by $\beta = e^{-\rho\Delta}$ or, equivalently, by $\rho = -\frac{\ln\beta}{\Delta}$. We take $\Delta = 1$ to obtain $\rho = -\ln\beta$.

5 Computational Advantages of Continuous Time

This section illustrates the computational advantages of continuous time using the quality ladder model of Section 4 as an example. Even though this is one specific example, it is useful for many purposes. First, the results related to the curse of dimensionality are clearly robust since they simply involve floating point operations related to computing the expectation over successor states. The burden of such computations depends on neither functional forms nor parameter values. Also, as we have pointed out in Section 3.3, what matters is the total number of coordinates of the state vector. Hence, the N-firm quality ladder model should be viewed as representative of dynamic stochastic games with N-dimensional state vectors. Second, the results related to the rate of convergence may depend on functional forms and parameter values but there is no reason to believe that our example is atypical. Third, we use our example to illustrate a strategy for diagnosing convergence. Our systematic approach to devising stopping rules contrasts with the commonly used *ad hoc* approaches and is thus in itself a contribution to the economics literature on numerically solving dynamic stochastic games.

5.1 Time per Iteration

Continuous time avoids the curse of dimensionality in the expectation over successor states. Since the algorithms for both discrete and continuous time perform this computation once for each state and each firm in each iteration, we divide the time it takes to complete one iteration by the number of states and the number of firms. Tables 1 and 2 summarize the results for the three algorithms presented in Section 3 – the discrete-time algorithm, the continuous-time algorithm without precomputed addresses, and the continuous-time algorithm with precomputed addresses.[9] Table 1 assumes $M = 18$ quality levels per firm and up to $N = 8$ firms just as PM1 do; Table 2 reduces M to 9 in order to accommodate a larger number of firms. Both tables also report the number of states after symmetry and anonymity are invoked, $\frac{(N+M-1)!}{N!(M-1)!}$, and the number of unknowns, which equals one value and one policy per state and firm, along with the ratio of discrete to continuous time without precomputed addresses, the ratio of continuous time without to with precomputed addresses, and the ratio of discrete time to continuous time with precomputed addresses.

[9]The programs are written in ANSI C and compiled with Microsoft Visual C++ .NET 2003 (code optimization enabled). All computations are carried out on an IBM ThinkPad T40 with a 1.6GHz Intel

			discrete time		continuous time without precomputed addresses		continuous time with precomputed addresses		ratio discrete to continuous time without precomputed addresses	ratio continuous time without to with precomputed addresses	ratio discrete to continuous time with precomputed addresses
#firms	#states	#unknowns	secs.	perct.	secs.	perct.	secs.	perct.			
2	171	684	1.07(-6)	55%	7.13(-7)	41%	5.85(-7)	36%	1.50	1.22	1.83
3	1140	6840	1.61(-6)	76%	6.67(-7)	44%	5.26(-7)	38%	2.41	1.27	3.06
4	5985	47880	3.30(-6)	87%	6.68(-7)	49%	5.10(-7)	41%	4.94	1.31	6.48
5	26334	263340	8.05(-6)	98%	7.06(-7)	49%	5.24(-7)	43%	11.40	1.35	15.36
6	100947	1211364	2.15(-5)	97%	7.51(-7)	52%	5.37(-7)	46%	28.57	1.40	40.00
7	346104	4845456	6.19(-5)	100%	7.74(-7)	56%	5.47(-7)	49%	80.00	1.42	113.21
8	1081575	17305200	1.65(-4)	100%	8.23(-7)	58%	5.92(-7)	56%	200.28	1.39	278.44

Table 1: Time per iteration per state per firm and percentage of time spent on computing the expectation. (k) is shorthand for $\times 10^k$. Quality ladder model with $M = 18$ quality levels per firm and a discount factor of 0.925.

			discrete time		continuous time without precomputed addresses		continuous time with precomputed addresses		ratio discrete to continuous time without precomputed addresses	ratio continuous time without to with precomputed addresses	ratio discrete to continuous time with precomputed addresses
#firms	#states	#unknowns	secs.	perct.	secs.	perct.	secs.	perct.			
2	45	180	9.78(-7)	52%	6.89(-7)	42%	5.67(-7)	33%	1.42	1.22	1.73
3	165	990	1.45(-6)	74%	6.36(-7)	44%	5.05(-7)	38%	2.29	1.26	2.88
4	495	3960	2.90(-6)	88%	6.36(-7)	48%	4.75(-7)	43%	4.55	1.34	6.10
5	1287	12870	6.94(-6)	96%	6.42(-7)	53%	4.77(-7)	46%	10.81	1.35	14.57
6	3003	36036	1.81(-5)	98%	6.88(-7)	55%	4.88(-7)	45%	26.34	1.41	37.12
7	6435	90090	5.02(-5)	100%	7.33(-7)	53%	5.11(-7)	48%	68.48	1.43	98.26
8	12870	205920	1.31(-4)	100%	7.77(-7)	55%	5.24(-7)	50%	168.33	1.48	249.38
9	24310	437580	3.82(-4)	100%	7.77(-7)	62%	5.39(-7)	53%	492.16	1.44	709.04
10	43758	875160	1.07(-3)	100%	8.34(-7)	64%	5.94(-7)	44%	1282.19	1.40	1800.00
11	75582	1662804	2.99(-3)	100%	8.42(-7)	67%	5.77(-7)	56%	3557.14	1.46	5187.50
12	125970	3023280	8.20(-3)	100%	8.60(-7)	68%	5.95(-7)	60%	9533.08	1.44	13770.00
13	203490	5290740	2.42(-2)	100%	9.22(-7)	69%	6.20(-7)	61%	26235.65	1.49	39033.56
14	319770	8953560	6.76(-2)	100%	9.53(-7)	72%	6.55(-7)	59%	70946.70	1.45	103195.27

Table 2: Time per iteration per state per firm and percentage of time spent on computing the expectation. (k) is shorthand for $\times 10^k$. Quality ladder model with $M = 9$ quality levels per firm and a discount factor of 0.925.

Avoiding the curse of dimensionality in the expectation over successor states yields a significant advantage only if this particular computation takes up a large fraction of the running time. Tables 1 and 2 show that this is the case: the discrete-time algorithm spends more than 50% of its time on it if $N = 2$, about 90% if $N = 4$, and essentially 100% if $N \geq 6$. Hence, computing the expectation over successor states is indeed the bottleneck of the discrete-time algorithm. The continuous-time algorithms, in contrast, spend between 33% and 72% of their time on it.

Even in its basic form the continuous-time algorithm is far faster than the discrete-time algorithm. The gain from continuous time increases from 50% if $N = 2$ to a factor of 200 if $N = 8$ in case of $M = 18$ (Table 1) and from 42% if $N = 2$ to a factor of 70,947 if $N = 14$ in case of $M = 9$ (Table 2). In line with theory the computational burden grows exponentially in N in discrete time but approximately linearly in continuous time. Consequently, the gain from continuous time explodes in the dimension of the state vector.

Precomputed addresses yield further gains: the continuous-time algorithm without precomputed addresses takes about 20% to 50% more time per iteration than the continuous-time algorithm with precomputed addresses. Compounding the gains from continuous time and precomputed addresses yields a total gain over discrete time that ranges from 83% if $N = 2$ to a factor of 278 if $N = 8$ in case of $M = 18$ (Table 1) and from 73% if $N = 2$ to a factor of 103,195 if $N = 14$ in case of $M = 9$ (Table 2).

In sum, the continuous-time algorithms are orders of magnitude faster than their discrete-time counterpart for games with more than a few state variables. Most of the gain is from avoiding the curse of dimensionality, but the precomputed addresses, a computational strategy that is effectively constrained to continuous time, also make a significant contribution.

5.2 Number of Iterations

While each iteration is far faster in the continuous- than in the discrete-time algorithm, this does not prove that the equilibrium of continuous-time models is faster to compute since the model is not solved until the iterations of the algorithm converge. Indeed, there are good reasons to think that the continuous-time algorithm will need more iterations to converge. Suppose that the strategic elements in the stochastic game were eliminated; in that case, the stochastic game reduces to a disjoint set of single-agent dynamic programming problems. Hence, a value function iteration approach (also called a pre-Gauss-Jacobi method) would converge at rate β in discrete time. As we have pointed out in Section 3.2, the continuous-time contraction factor

$$\eta(X(\omega), \omega) = \frac{\phi\left(X(\omega), \omega\right)}{\rho + \phi\left(X(\omega), \omega\right)},$$

is not constant but varies with players' policies from state to state. It has a simple interpretation: $\eta(X(\omega), \omega)$ is the expected net present value of a dollar delivered at the next time

Pentium M processor and 1.5GB memory running Microsoft Windows XP Professional.

the state changes if the current state is ω and players' policies are $X(\omega)$. This is easily seen in the special case of $\rho \ll \phi(X(\omega), \omega) = 1$ since

$$\eta(X(\omega), \omega) = \frac{1}{\rho + 1} \approx 1 - \rho = 1 + \ln \beta \approx \beta.$$

In general, if the discount rate ρ is large or if the hazard rate $\phi\left(X\left(\omega\right), \omega\right)$ is small, then $\eta(X(\omega), \omega)$ is small and there is a strong contraction aspect to a value function iteration approach. However, $\eta(X(\omega), \omega)$ could be close to one if the discount rate is small or if the hazard rate is large, in which case a value function iteration approach would converge slowly. Since $\phi\left(X\left(\omega\right), \omega\right) = \sum_{i=1}^{N} \phi^i\left(X^i\left(\omega\right), \omega^i\right)$ in the special case of independent transitions, this in particular suggests that convergence could be slow if the number of players N is large.

The above facts lead us to worry about the rate of convergence of the continuous-time algorithm. A fair comparison between the discrete- and continuous-time algorithms requires a careful application of accuracy estimates and stopping rules. Let $V^i(\omega)$ and $X^i(\omega)$ denote the value and policy of player i in state ω at the beginning of an iteration and $\hat{V}^i(\omega)$ and $\hat{X}^i(\omega)$ his value and policy at the end of the iteration. We need a measure of the distance between two sets of value functions. We want this measure to be unit-free and to describe the relative difference. Therefore, we define the L_∞-relative difference between $\hat{V}$ and V to be

$$E\left(\hat{V}, V\right) = \left\| \frac{\hat{V} - V}{1 + |\hat{V}|} \right\| = \max_{i=1,\ldots,N} \max_{\omega \in \Omega} \left| \frac{\hat{V}^i(\omega) - V^i(\omega)}{1 + |\hat{V}^i(\omega)|} \right|.$$

We similarly define $E\left(\hat{X}, X\right)$.

Table 3 compares the discrete- and continuous-time algorithms.[10] It presents the number of iterations until the distance between subsequent iterates as measured by $E\left(\hat{V}, V\right)$ and $E\left(\hat{X}, X\right)$ are below a prespecified tolerance of either 10^{-4} or 10^{-8}.[11] In addition, Table 3 presents the number of iterations until the distance between the current iterate $\hat{V}$ and $\hat{X}$ and the "true" solution V_∞ and X_∞ is below a prespecified tolerance. To obtain V_∞ and X_∞ we ran the algorithm until the distance between subsequent iterates failed to decrease any further. The iterations continued until both $E\left(\hat{V}, V\right)$ and $E\left(\hat{X}, X\right)$ were less than 10^{-13} and, in some cases, less than 10^{-15}. The final iterates were considered the true solution since they satisfied the equilibrium conditions essentially up to machine precision.

In light of our previous discussion we expect the number of iterations to be sensitive to the number of firms and the discount factor. Hence, Table 3 assumes $N \in \{3, 6, 9, 12\}$ and $\beta = e^{-\rho} \in \{0.925, 0.98, 0.99, 0.995\}$. We omit the cases with $N = 12$ in discrete time because one iteration takes more than 3 hours, thus making it impractical to compute the

[10] Whether or not we use precomputed addresses in continuous time is immaterial for the number of iterations to convergence.

[11] The starting values are $V(\omega) = \frac{\pi(\omega)}{1-\beta}$ and $X(\omega) = 0$ in discrete time and $V(\omega) = \frac{\pi(\omega)}{\rho}$ and $X(\omega) = 0$ in continuous time.

#firms	discount factor	discrete time				continuous time				ratio			
		distance betw. iterations $< 10^{-4}$	distance to truth $< 10^{-4}$	distance betw. iterations $< 10^{-8}$	distance to truth $< 10^{-8}$	distance betw. iterations $< 10^{-4}$	distance to truth $< 10^{-4}$	distance betw. iterations $< 10^{-8}$	distance to truth $< 10^{-8}$	distance betw. iterations $< 10^{-4}$	distance to truth $< 10^{-4}$	distance betw. iterations $< 10^{-8}$	distance to truth $< 10^{-8}$
3	0.925	99	118	182	201	131	212	364	446	0.76	0.56	0.50	0.45
3	0.98	304	412	594	702	313	776	1238	1699	0.97	0.53	0.48	0.41
3	0.99	519	782	1104	1367	455	1531	2320	3393	1.14	0.51	0.48	0.40
3	0.995	923	1543	2100	2719	589	3042	4343	6779	1.57	0.51	0.48	0.40
6	0.925	99	118	182	201	220	364	581	725	0.45	0.32	0.31	0.28
6	0.98	387	494	673	780	742	1674	2395	3324	0.52	0.30	0.28	0.23
6	0.99	743	983	1286	1525	1198	3379	4593	6761	0.62	0.29	0.28	0.23
6	0.995	1362	1900	2408	2945	1832	6797	8729	13637	0.74	0.28	0.28	0.22
9	0.925	100	119	182	201	232	404	647	818	0.43	0.29	0.28	0.25
9	0.98	386	492	670	775	1100	2363	3235	4493	0.35	0.21	0.21	0.17
9	0.99	751	988	1289	1526	1927	4973	6447	9469	0.39	0.20	0.20	0.16
9	0.995	1469	2003	2509	3042	3129	10148	12452	19365	0.47	0.20	0.20	0.16
12	0.925					227	412	669	854				
12	0.98					1276	2721	3668	5106				
12	0.99					2447	6023	7637	11181				
12	0.995					4217	12580	15085	23304				

Table 3: Number of iterations to convergence. Quality ladder model with $M = 9$ quality levels per firm.

true solution. We see that the continuous-time algorithm needs more iterations to converge than its discrete-time counterpart, and that this gap widens very slightly as we increase β (decrease ρ). On the other hand, the number of iterations needed by the discrete-time algorithm remains more or less constant as we increase the number of firms whereas the number of iterations needed by the continuous-time algorithm increases rapidly as we go from $N = 3$ to $N = 6$. Fortunately, the number of iterations increases slowly as we go from $N = 6$ to $N = 9$ and remains more or less constant thereafter, so that the gap between the algorithms stabilizes.

5.3 Time to Convergence

The continuous-time algorithm suffers an "iteration penalty" because $\eta(X(\omega), \omega)$ substantially exceeds the discrete-time discount factor β. Even though the continuous-time algorithm needs more iterations, the loss in the number of iterations is small when compared to the gain from avoiding the curse of dimensionality. Table 4 illustrates this comparison and the total gain from continuous time. Continuous time beats discrete time by 60% if $N = 3$, a factor of 12 if $N = 6$, a factor of 209 if $N = 9$, a factor of $3,977$ if $N = 12$, and a factor of $29,734$ if $N = 14$. To put these numbers in perspective, in case of the 14-firm quality ladder model it takes about 20 minutes to compute the equilibrium of the continuous-time game, but it would take over a year to do the same in discrete time!

#firms	discrete time (mins.)	continuous time (mins.)	time per iteration	ratio number of iterations	time to convergence
2	1.80(-4)	1.12(-4)	1.73	0.93	1.61
3	1.42(-3)	8.83(-4)	2.88	0.56	1.60
4	1.13(-2)	4.43(-3)	6.10	0.42	2.54
5	8.78(-2)	1.70(-2)	14.57	0.36	5.18
6	6.42(-1)	5.34(-2)	37.12	0.32	12.03
7	4.44(0)	1.47(-1)	98.26	0.31	30.19
8	2.67(1)	3.56(-1)	249.38	0.30	74.94
9	1.66(2)	7.95(-1)	709.04	0.29	208.85
10	*9.28(2)*	1.77(0)	1800.00	*0.29*	*523.72*
11	*4.94(3)*	3.30(0)	5187.50	*0.29*	*1498.33*
12	*2.46(4)*	6.18(0)	13770.00	*0.29*	*3977.26*
13	*1.27(5)*	1.13(1)	39033.56	*0.29*	*11246.96*
14	*6.00(5)*	2.02(1)	103195.27	*0.29*	*29734.23*

Table 4: Time to convergence. (k) is shorthand for $\times 10^k$. Convergence criterion is "distance to truth$< 10^{-4}$." Entries in italics are based on an estimated 119 iterations to convergence in discrete time. Quality ladder model with $M = 9$ quality levels per firm and a discount factor of 0.925.

5.4 Stopping Rules

In practice it is rarely feasible to compute the true solution V_∞ and X_∞. Rather we compute the distance between subsequent iterates and terminate the algorithm once $E\left(\hat{V}, V\right)$ and $E\left(\hat{X}, X\right)$ is below a prespecified tolerance. Yet we really want to know $E\left(\hat{V}, V_\infty\right)$ and $E\left(\hat{X}, X_\infty\right)$ in order to assess the accuracy of our computations. Table 3 suggests that the distance to the true solution may be far greater than the distance between subsequent iterates. Fortunately, as we show below, the two concepts are closely related, and we exploit this fact in devising stopping rules. Note that the choice of stopping rule is especially important since convergence is linear in the Gauss-Seidel schemes that we use to compute equilibria.

Our approach to devising stopping rules applies some ideas from the theory of sequences. Consider a sequence of points $\{z_l\}_{l=0}^{\infty}$ that satisfies

$$\|z_{l+1} - z_l\| \leq \theta \|z_l - z_{l-1}\|,$$

where $\theta < 1$ is a contraction factor that determines the rate of convergence. Then the distance to the limit z_∞ satisfies

$$\|z_{l+1} - z_\infty\| \leq \frac{\|z_l - z_{l-1}\|}{1-\theta}.$$

First, define $\delta_l = \|z_l - z_{l-1}\|$ and suppose that $\delta_{l+1} = \theta\delta_l$. Then, for all l and all k, $\delta_l = \theta^k \delta_{l-k}$ or

$$\theta = \left(\frac{\delta_l}{\delta_{l-k}}\right)^{\frac{1}{k}}. \tag{11}$$

In our computations we observe δ_l but not θ. Equation (11) gives us a way to estimate θ from δ_l, the distance between iterates l and $l-1$, and δ_{l-k}, the distance between iterates $l-k$ and $l-k-1$.

Next, define $\varepsilon_l = \|z_l - z_\infty\|$. Then, approximately, we have $\varepsilon_l = \delta_l / (1-\theta)$. With δ_l and θ in hand our task is to determine the number of additional iterations k that are required to ensure that the distance between iterate $l+k$ and the limit is below a prespecified tolerance $\bar{\varepsilon}$:

$$\varepsilon_{l+k} = \frac{\delta_{l+k}}{1-\theta} = \frac{\theta^k \delta_l}{1-\theta} = \bar{\varepsilon}.$$

Hence, the number of additional iterations as a function of the rate of convergence is

$$K(\theta) = \frac{\ln(\bar{\varepsilon}/\delta_l) + \ln(1-\theta)}{\ln\theta}. \tag{12}$$

It is common practice to terminate the algorithm once the distance between subsequent iterates is below $\bar{\varepsilon}$. However, the distance to the true solution could be a factor $(1 -$

$\theta)^{-1}$ greater than $\bar{\varepsilon}$. Equation (12) relates $E\left(\hat{V}, V\right)$ and $E\left(\hat{X}, X\right)$ with $E\left(\hat{V}, V_\infty\right)$ and $E\left(\hat{X}, X_\infty\right)$ and, along with equation (11), forms the basis of our strategy for diagnosing convergence.

The first step is to use equation (11) to estimate the rate of convergence θ. Table 5 presents the results for discrete as well as continuous time assuming $N \in \{3, 6, 9, 12\}$ and $\beta = e^{-\rho} \in \{0.925, 0.98, 0.99, 0.995\}$. Several remarks are in order. First, while the estimate in principle could vary from one iteration to the next, it turns out to be nearly constant after the first several iterations. Second, for any given N and β, the continuous-time rate of convergence exceeds its discrete-time counterpart. This is in line with the "iteration penalty" of the continuous-time algorithm. Third, the discrete-time rate of convergence is smaller than the discount factor β. This reflects the fact that we are using Gauss-Seidel schemes instead of Gauss-Jacobi schemes such as value function iteration to compute equilibria.

#firms	discount factor	discrete time	continuous time
3	0.925	0.8962	0.9611
3	0.98	0.9690	0.9901
3	0.99	0.9845	0.9951
3	0.995	0.9922	0.9975
6	0.925	0.8962	0.9747
6	0.98	0.9681	0.9944
6	0.99	0.9832	0.9973
6	0.995	0.9912	0.9987
9	0.925	0.8962	0.9779
9	0.98	0.9681	0.9957
9	0.99	0.9830	0.9980
9	0.995	0.9912	0.9990
12	0.925		0.9793
12	0.98		0.9961
12	0.99		0.9982
12	0.995		0.9991

Table 5: Estimated rate of convergence. Estimated from the distance between iterations at 10^{-8}. Quality ladder model with $M = 9$ quality levels per firm.

The second step is to use equation (12) to predict the number of additional iterations required to reduce the distance to the true solution to $\bar{\varepsilon}$. Equation (12) does an excellent job here. For example, if $N = 6$ and $\beta = 0.925, 0.98, 0.99, 0.995$, then the estimated continuous-time rates in Table 5 imply $K(\theta) = 144, 931, 2168, 4910$. From Table 3 the actual numbers are $144, 929, 2168, 4908$. Overall, the discrepancy between the predicted and the actual number of additional iterations is negligible. Devising stopping rules without knowing the

true solution is feasible; indeed, a careful examination of the iteration history suffices to assess the accuracy of the computations.

6 Conceptual Advantages of Continuous Time

In Section 5 we have emphasized the computational advantages of continuous time. In addition, as we discuss below, continuous time has a number of conceptual advantages.

6.1 Flexibility and Interpretability of Model Specifications

Discrete-time models often have difficulty capturing dynamic phenomena. Consider, for example, depreciation of machinery. Suppose that firm i owns ω^i machines in the present period and that each machine has a probability of 0.2 per period of breaking down independent of other machines. Then $(\omega')^i \in \{0, 1, \ldots, \omega^i\}$ is binomially distributed, and firm i will own anywhere between 0 and ω^i machines in the subsequent period. While this is a natural way to model stochastic depreciation, it aggravates the curse of dimensionality in discrete-time models because in an industry with N firms the expectation over successor states is comprised of $\prod_{i=1}^{N}(1+\omega^i)$ terms. A possible shortcut is to focus on the expected number of machines rather than their entire distribution (e.g., Benkard 2004). If a firm has 5 machines this period, then in expectation it will have 4 next period. The case of, say, 7 machines is not as easy to model since $7(1-0.2) = 5.6$ is not an integer. One could assume that the firm will have either 5 or 6 machines next period and adjust the transition probabilities so that the expectation equals 5.6. In this case, however, the variance of the depreciation shock varies from state to state. In general, discrete time forces one to choose between making a peculiar assumption about the nature of transitions or exacerbating the curse of dimensionality.

In continuous time, by contrast, depreciation is easy to model. We just say that each machine has a hazard rate of 0.2 of breaking down independent of other machines, so that the hazard rate of a jump occurring in firm i's state is $0.2\omega^i$. This exactly models a stochastic exponential depreciation rate, but it does not affect the number of terms that enter the expectation over successor states: since the machines owned by the N firms break down one at a time, computing the expectation over successor states involves summing over N terms.

Besides allowing for more flexible model specifications, continuous time also facilitates their interpretation. In a discrete-time model the transition probabilities cannot exceed one. This forces one to look for tractable functional forms. A popular choice is to assume that the probability of an investment success is $\frac{\alpha x^i}{1+\alpha x^i}$ (e.g., PM1). This form is highly stylized and the parameter α is hard to interpret. In a continuous-time model it suffices to ensure that the hazard rates are nonnegative. For example, $\left(x^i\right)^\gamma$ is a familiar constant elasticity form for the success hazard which can be used in continuous- but not in discrete-

time models. The parameter γ is simply the elasticity of the success hazard with respect to investment expenditures or, equivalently, (the negative of) the elasticity of the expected time to an investment success. Since they are often used in empirical studies, easy-to-interpret functional forms such as constant elasticity may also facilitate parameterizing the model.

6.2 Richness of Stochastic Outcomes

Many dynamic stochastic games such as the quality ladder model of Section 4 restrict a player's transitions to immediately adjacent states. This imposes a sense of continuity – the player cannot go from state 3 to state 5 without passing through state 4 – although the number of states is finite. While often natural, this "continuity" assumption has different consequences for discrete- and continuous-time models.

In discrete-time models it implies that the state changes by at most one unit in any given period. Hence, the minimum amount of time that is required to change the state by n units is n periods. Discrete-time models have limited flexibility in modeling the frequency of changes. In continuous-time models, by contrast, this "continuity" assumption just says that the state changes by one unit at a time, but that does not constrain the number of changes that can occur in any finite interval of time. The frequency of changes is governed by players' actions in equilibrium and not predetermined by the unit of time. Continuous time thus allows for a much richer range of stochastic outcomes over any finite interval of time.

Figure 1 illustrates this point by comparing the equilibrium value function for the discrete- and continuous-time version of the quality ladder model with $N = 1$ firm. The difference is largest in state $\omega = 1$ with $V(1) = 69.59$ in discrete time and $V(1) = 112.92$ in continuous time. The reason is that the monopolist is stuck with low quality and thus low profits for a very long time in the discrete-time model whereas it is able to quickly reach states with high quality in the continuous-time model.

In discrete-time models β determines both the discount rate and the period length. Hence, in order to enrich the range of outcomes over a given interval of time, one could think about shortening the length of a period by taking the discount factor close to one. However, as Table 3 shows, the number of iterations to convergence increases with β. Taking β close to one is thus not a practical way to model short periods. In contrast, in continuous-time models the length of a period is essentially zero and completely independent of the discount rate ρ.

6.3 Realism of Strategic Interactions

Discrete time may result in unrealistic patterns of strategic interactions. For example, consider two firms that are both trying to expand their capacity and assume that each would want to cease its investment once the other succeeds. As long as the success of an

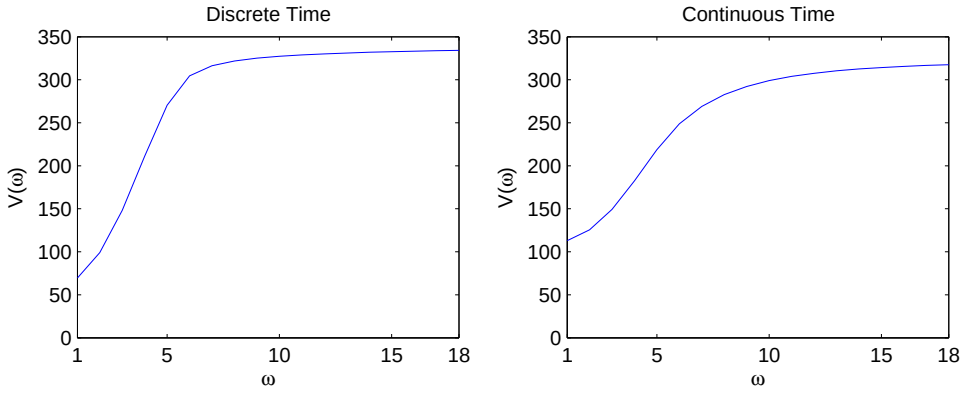

Figure 1: Equilibrium value function. Quality ladder model with $N = 1$ firm, $M = 18$ quality levels, and a discount factor of 0.925.

investment project is uncertain, in a discrete-time model there is some chance that both firms succeed in a given period. This results in excess capacity that makes both firms regret their previous investments and perhaps spurns some efforts to disinvest. In a continuous-time model, by contrast, this cannot happen since at most one firm succeeds at a given point in time and the other promptly adjusts and ceases its investment. In short, there will be no "mistakes" in a continuous-time model.

In a discrete-time model players are also able to respond quickly to changes in the strategic situation provided that the length of a period is sufficiently short. However, as we have pointed out above, practical considerations dictate high discount rates and thus long response times. In many applications of EP's framework such as the quality ladder model of Section 4 the state space is fairly coarse. Thus, changes typically have a large effect on the environment and while a response time of a few days, weeks, or even months may be plausible, a response time of one or more years is not. In this case continuous time yields a more realistic description of players' ability to react swiftly to changes.

7 Concluding Remarks

Discrete-time stochastic games with a finite number of states suffer from a curse of dimensionality in computing players' expectations over all possible future states in that their computational burden increases exponentially in the number of state variables. We develop the alternative of continuous-time stochastic games with a finite number of states and demonstrate that continuous-time games avoid the curse of dimensionality, thereby speeding up the computations by orders of magnitude for games with more than a few state variables. We further speed up the computations with precomputed addresses, a computational strategy that is effectively constrained to continuous time. Besides their computational advantages, continuous-time games have conceptual advantages in terms of the flexibility and

interpretability of the model specifications, the richness of stochastic outcomes over any finite interval of time, and the realism of strategic interactions. Overall, the computational and conceptual advantages of continuous-time games are substantial and open the way to study more complex and realistic stochastic games than currently feasible.

The methods in this paper are just the beginning of what can be done to speed up the computation of equilibria of dynamic stochastic games. In particular, extending our continuous-time algorithms to focus on the ergodic set as suggested by PM2 may lead to further gains in some applications. The more general observation is that computing equilibria is just a problem of solving a large system of nonlinear equations. While the number of unknowns is large, each unknown appears in a rather small subset of equations. This sparse structure is implicitly used in all available methods and can be further exploited. Since the size of the problem is typically very large, a direct application of Newton's method or other solution methods for nonlinear equations is impractical, and some type of Gaussian scheme is necessary. However, there are many variations of the block Gauss-Seidel scheme that have not been explored and it is highly likely that there are some superior approaches available. In future work we plan to examine alternative block structures, methods within blocks, and acceleration methods.

Appendix

Below we show how to add entry and exit to the continuous-time quality ladder model of Section 4. Recall that $\omega^i \in \{1, \ldots, M\}$ describes the quality of firm i's product. To model entry and exit, we add $M+1$ to the set of firm i's feasible states and assume that $\omega^i = M+1$ designates firm i as being inactive in the product market game. The state space thus becomes $\Omega = \{1, \ldots, M, M+1\}^N$. Once an incumbent firm exits the industry, it transits from state $\omega^i \neq M+1$ to state $(\omega')^i = M+1$. It then becomes a potential entrant that, upon entry, transits from state $\omega^i = M+1$ to state $(\omega')^i \neq M+1$. These transitions are under the control of firms. Specifically, firm i's action $x^i = (x^{i,1}, x^{i,2})$ is now a vector instead of a scalar. Let $x^{i,1} \geq 0$ denote firm i's investment in quality improvements and let $h^1(x^{i,1})$ denote the hazard rate of an investment success. In addition, let $x^{i,2} \geq 0$ denote firm i's "exit intensity" if it is an incumbent firm or its "entry intensity" if it is a potential entrant. The exit (entry) intensity $x^{i,2}$ translates into a hazard rate $h^2(x^{i,2})$ of exiting (entering) the industry. If an incumbent firm exits the industry, it receives a scrap value. We make the scrap value a decreasing function of the exit intensity. That is, if a firm is in a hurry to exit, it receives less for its assets. Hence, $x^{i,2}$ can be thought of as reducing the firm's reservation price for selling its assets. Conversely, if a potential entrant enters the industry, it pays a setup cost, which we take to be an increasing function of the entry intensity.

The details of entry and exit are as follows: Suppose first that firm i is an incumbent firm, i.e., $\omega^i \neq M+1$. Jumps in firm i's state occur according to a Poisson process with hazard rate

$$\phi^i(x^i, \omega^i) = h^1(x^{i,1}) + \delta + h^2(x^{i,2}),$$

and when a jump occurs, firm i's state changes according to the transition probability

$$f^i(\left(\omega'\right)^i|\omega^i,x^i)=\begin{cases}\frac{h^1(x^{i,1})}{\phi^i(x^i,\omega^i)}, & \left(\omega'\right)^i=\omega^i+1,\\ \frac{\delta}{\phi^i(x^i,\omega^i)}, & \left(\omega'\right)^i=\omega^i-1,\\ \frac{h^2(x^{i,2})}{\phi^i(x^i,\omega^i)}, & \left(\omega'\right)^i=M+1\end{cases}$$

if $\omega^i\in\{2,\ldots,M-1\}$.[12] Note that the last line captures the possibility of exit. Upon exit the incumbent firm receives a scrap value and the instantaneous change in wealth is

$$\Phi^i(x,\omega^i,\omega^{-i},M+1,\left(\omega'\right)^{-i})=\kappa-x^{i,2}.$$

More elaborate specifications are possible, e.g., the value of a firm's assets may depend on its state as in $\Phi^i(x,\omega^i,\omega^{-i},M+1,(\omega')^{-i})=\kappa(\omega^i)-x^{i,2}$, where $\kappa(\omega^i)$ is a (presumably increasing) function of ω^i.

Suppose next that firm i is a potential entrant, i.e., $\omega^i=M+1$. It is natural to assume that a potential entrant cannot invest in order to improve the quality of its product before it has actually entered the industry. Jumps in firm i's state thus occur according to a Poisson process with hazard rate

$$\phi^i(x^i,M+1)=h^2(x^{i,2}),$$

and when a jump occurs, firm i's state changes according to the transition probability

$$f^i(\omega^e|M+1,x^i)=1,$$

where $\omega^e\in\{1,\ldots,M\}$ is the (exogenously given) initial quality of a firm's product. Upon entry the potential entrant pays a setup cost and the instantaneous change in wealth is

$$\Phi^i(x,M+1,\omega^{-i},\omega^e,\left(\omega'\right)^{-i})=-\left(\kappa^e+x^{i,2}\right).$$

Finally, since a potential entrant is inactive in the product market game, its payoff flow is

$$\pi^i(x,M+1,\omega^{-i})=0.$$

The above formulation of entry and exit differs from the one proposed by PM1. In the background of their model is an infinite pool of potential entrants. Among these potential entrants one is selected at random in each period and given a chance to enter the industry. The potential entrant is therefore short-lived and bases its entry decision solely on the value of immediate entry; it does not take into account the value of deferred entry. In addition, PM1 assume that by exiting the industry an incumbent firm *de facto* exits the game. In contrast, we assume that there is a fixed number of firms and that each firm may be either an incumbent firm or a potential entrant at any given point in time. Moreover, when exiting the firm takes the possibility that it may enter the industry at some later point into account and, conversely, when entering the firm takes the possibility that it may exit the industry at some later point into account. Exiting is thus tantamount to "mothballing" and entering to resuming operations. The advantage of this formulation of entry and exit is that it leads to a game with a finite and constant number of players. Whether one uses our formulation or the one proposed by PM1 is immaterial for the purposes of this paper

[12] As discussed in Section 4, if $\omega^i=1$ or if $\omega^i=M$, then the hazard rate and the transition probability need to be adjusted.

since the computational advantages of continuous time are exactly the same in both.

References

Aguirregabiria, V. & Mira, P. (2002), Sequential simulation-based estimation of dynamic discrete games, Working paper, Boston University, Boston.

Bajari, P., Benkard, L. & Levin, J. (2004), Estimating dynamic models of imperfect competition, Working paper no. 10450, NBER, Cambridge.

Basar, T. & Olsder, J. (1999), *Dynamic noncooperative game theory*, 2nd edn, Society for Industrial and Applied Mathematics, Philadelphia.

Benkard, L. (2004), 'A dynamic analysis of the market for wide-bodied commercial aircraft', *Review of Economic Studies* **71**(3), 581–611.

Besanko, D. & Doraszelski, U. (2004), 'Capacity dynamics and endogenous asymmetries in firm size', *Rand Journal of Economics* **35**(1), 23–49.

Caplin, A. & Nalebuff, B. (1991), 'Aggregation and imperfect competition: On the existence of equilibrium', *Econometrica* **59**(1), 26–59.

Davis, P. & Rabinowitz, P. (1984), *Methods of numerical integration*, 2nd edn, Academic Press, New York.

Dockner, E., Jorgensen, S., Van Long, N. & Sorger, G. (2000), *Differential games in economics and management science*, Cambridge University Press, Cambridge.

Doraszelski, U. & Markovich, S. (2004), Advertising dynamics and competitive advantage, Working paper, Hoover Institution, Stanford.

Doraszelski, U. & Satterthwaite, M. (2003), Foundations of Markov-perfect industry dynamics: Existence, purification, and multiplicity, Working paper, Hoover Institution, Stanford.

Erdem, E. & Tybout, J. (2003), Trade policy and industrial sector reponses: Using evolutionary models to interpret the evidence, Working paper no. 9947, NBER, Cambridge.

Ericson, R. & Pakes, A. (1995), 'Markov-perfect industry dynamics: A framework for empirical work', *Review of Economic Studies* **62**, 53–82.

Fershtman, C. & Pakes, A. (2000), 'A dynamic oligopoly with collusion and price wars', *Rand Journal of Economics* **31**, 294–326.

Filar, J. & Vrieze, K. (1997), *Competitive Markov decision processes*, Springer, New York.

Goettler, R., Parlour, C. & Rajan, U. (2004), 'Equilibrium in a dynamic limit order market', *Journal of Finance* **forthcoming**.

Gowrisankaran, G. (1999), 'Efficient representation of state spaces for some dynamic models', *Journal of Economic Dynamics and Control* **23**, 1077–1098.

Isaacs, R. (1954), *Differential games*, John Wiley & Sons, New York.

Judd, K. (1998), *Numerical methods in economics*, MIT Press, Cambridge.

Langohr, P. (2003), Competitive convergence and divergence: Capability and position dynamics, Working paper, Northwestern University, Evanston.

Pakes, A. (2000), A framework for applied dynamic analysis in I.O., Working paper no. 8024, NBER, Cambridge.

Pakes, A., Gowrisankaran, G. & McGuire, P. (1993), Implementing the Pakes-McGuire algorithm for computing Markov perfect equilibria in Gauss, Working paper, Yale University, New Haven.

Pakes, A. & McGuire, P. (1994), 'Computing Markov-perfect Nash equilibria: Numerical implications of a dynamic differentiated product model', *Rand Journal of Economics* **25**(4), 555–589.

Pakes, A. & McGuire, P. (2001), 'Stochastic algorithms, symmetric Markov perfect equilibrium, and the "curse" of dimensionality', *Econometrica* **59**(5), 1261–1281.

Pakes, A., Ostrovsky, M. & Berry, S. (2004), Simple estimators for the parameters of discrete dynamic games (with entry/exit examples), Working paper no. 10506, NBER, Cambridge.

Pesendorfer, M. & Schmidt-Dengler, P. (2003), Identification and estimation of dynamic games, Working paper no. 9726, NBER, Cambridge.

Shapley, L. (1953), 'Stochastic games', *Proceedings of the National Academy of Sciences* **39**, 1095–1100.

Starr, A. & Ho, Y. (1969), 'Nonzero-sum differential games', *Journal of Optimization Theory and Applications* **3**(3), 184–206.