Approximation Methods for Large Dynamic Stochastic Games

Matteo Courthoud*

November 17, 2020

Abstract

Dynamic stochastic games notoriously suffer from a curse of dimensionality that makes computing the Markov Perfect Equilibrium of large games infeasible. This article compares the existing approximation methods and alternative equilibrium concepts that have been proposed in the literature to overcome this problem. No method clearly dominates the others but some are dominated in all dimensions. In general, alternative equilibrium concepts outperform sampling-based approximation methods. I propose a new game structure, games with random order, in which players move sequentially and the order of play is unknown. The Markov Perfect equilibrium of this game consistently outperforms all existing approximation methods in terms of approximation accuracy while still being extremely efficient in terms of computational time.

Keywords: Industry dynamics, computation

JEL Classification: C63

*University of Zürich.

1 Introduction

Since Ericson & Pakes (1995) introduced their dynamic quality ladder model, dynamic stochastic games have entered the standard toolkit of both theoretical and applied Industrial Organization economists. However, one of the main problems with the computation and estimation of dynamic stochastic games involves their curse of dimensionality. In particular, as the number of players in the model increases, the computational burden increases exponentially, making an exact numerical solution unfeasible but for small state spaces that are generally not rich enough to capture many economic environments. This has motivated many scholars to find methods to alleviate this constraint. All of these methods involve a trade-off between approximation accuracy and computational time. From a practitioner's perspective, however, it is hard to understand from the current literature which method is preferable, in any given setting.

In this article, I review the most relevant approximation methods that have been developed to solve the curse of dimensionality in the equilibrium computation of dynamic stochastic games. Moreover, I propose a new method which I call *games with random order*. My method outperforms all existing methods in terms of approximation accuracy, while being very efficient in terms of computational time. I start by presenting the original framework from Ericson & Pakes (1995) in Section 2. In Section 3, I provide an overview of the existing approximation methods, highlighting their strength and weaknesses. In Section 4, I present simulation results for the different approximation methods, using the framework of Ericson & Pakes (1995) as a benchmark. Section 5 concludes.

2 Quality Ladder Model

In this section, I review the quality ladder model of Ericson & Pakes (1995) and the related computational algorithm of Pakes & McGuire (1994). I decide to use this particular model as baseline for two main reasons.

First of all, Ericson & Pakes (1995) is generally recognized as the model that introduced computational methods to study imperfect competition in market dynamics. It is indeed one of the most influential papers in industrial organization and economics in general. It has provided the baseline for both theoretical and empirical work on industry dynamics. Some of the most influential computational theory papers include Gowrisankaran (1999), Miao (2005), Asplund & Nocke (2006) and Besanko et al. (2010). On the other hand, thanks to identification results from, among many others, Hotz & Miller (1993), Aguirregabiria & Mira (2002) and Bajari et al. (2007), it has spurred a large empirical literature that includes for example Goettler & Gordon (2011), Gowrisankaran & Rysman (2012), Ryan (2012) and Igami (2017).

Second, it is the model used to test most of the different approximation methods described in this paper. However, in none of the papers introducing a new approximation method for large dynamic stochastic games there is a comparison with alternative approximation methods.

2.1 The Game

Time is discrete and the horizon is infinite. At each point in time, up to N firms are active and compete in prices with differentiated products and the state of the game is represented by a quality vector $\boldsymbol{\omega} = \{\omega_1, ..., \omega_N\} \in \Omega = \{0, ..., M\}^N$, where ω_n represents the quality of firm n and M is the maximum achievable quality level. Let $\omega_n = 0$ denote inactive firms. I will refer to $\boldsymbol{\omega}$ as a state and Ω as the state space. Firms maximize their total future discounted profits. The discount factor is $\beta \in [0, 1)$. The solution concept they consider is Markov Perfect Equilibrium.

The first computational bottleneck of the model arises from the dimension of the state space: M^N. This defines the number of points at which the equilibrium has to be computed, is M^N. Since the game satisfies the symmetry and anonymity assumptions of Doraszelski & Pakes (2007), we can reduce the number the number of relevant states from M^N to $M\binom{M+N-1}{N-1}$, so that it's combinatorial in N instead of exponential. Provide

intuition of magnitude.

Static Profits. Firms compete in prices facing multinomial logit demand. There is a continuum of consumers of mass m. Consumer i's utility from buying one unit of product n is given by $u_{in} = g(\omega_n) - p_n + \varepsilon_{in}$ where

$$g(\omega_n) = \begin{cases} 3\omega_n - 4 & \text{if } 1 \leq \omega_n \leq \omega^* \\ 12 + \log(2 - \exp(16 - 3\omega_n)) & \text{if } \omega^* < \omega_n \leq M \end{cases} \tag{1}$$

where p_n is the price of product n and ε_{in} is the idiosyncratic preference of consumer i for product n. There is an outside option which gives utility $u_{i0} = \varepsilon_{i0}$. The random shocks ε_{in} are assumed to be independent and type 1 extreme value distributed so that the resulting demand function has the logit form

$$D_n(\mathbf{p}; \boldsymbol{\omega}) = m \frac{\exp\left(g\left(\omega_n\right) - p_n\right)}{1 + \sum_{j=1}^{N} \exp\left(g\left(\omega_j\right) - p_j\right)} \tag{2}$$

where $\mathbf{p}$ is the vector of prices. Each firm chooses p_n in order to maximize static profits

$$\pi_n(\mathbf{p}; \boldsymbol{\omega}) = \max_{p_n} \; D_n(\mathbf{p}; \boldsymbol{\omega})(p_n - c) \tag{3}$$

where c is the marginal cost of production, assumed identical across firms. Given a state $\boldsymbol{\omega}$, there exists a unique set of Nash equilibrium prices (Caplin & Nalebuff, 1991).

Investment and Depreciation. Firms can invest in order to improve their quality. Conditional on investing an amount x_n, firm n has a probability $\Pr(x_n) = \frac{\alpha x_n}{1+\alpha x_n}$ of increasing its product quality from ω_n to $\omega_n + 1$. Product quality cannot increase in state $\omega_n = M$. After the investment outcome is realized, with probability $\delta \in (0, 1)$, an industry-wide shock hits the industry, decreasing the quality of each active firm by one unit. Inactive firms and firms in state $\omega_n = 1$ are not affected[1].

Entry and Exit. In order to ensure equilibrium existence, we follow Doraszelski & Satterthwaite (2010) and assume stochastic entry costs and exit scrap values. In each period, each inactive firm (entrant) draws an entry cost ϕ_n^e from a distribution $F^e(\phi_n^e)$. After observing its own cost, each entrant firm simultaneously decides whether to enter the market or not. At the same time, in each period, each active firms draws a scrap value ϕ_n^x from a distribution $F^x(\phi_n^x)$ and decides whether to stay active or exit the industry and collect the scrap value.

Timing. The overall timing of the game is the following:

1. Active firms earn static profits. Active firms observe the static scrap values and decide whether to exit the industry or not. Entrants observe entry costs and decide whether to enter the industry or not. Active firms decide their optimal investment level.
2. Entry and exit take place. Investment costs are paid, conditional of the firms being still active, and investment outcomes are realized.
3. The industry shock is realized.

2.2 Value and Policy Function

Let $V_n(\boldsymbol{\omega})$ denote the value function of firm n in state $\boldsymbol{\omega}$. The value function is implicitly defined by the Bellman equation

$$V_n(\boldsymbol{\omega}) = \pi_n^*(\boldsymbol{\omega}) + \beta \mathbb{E}_{\boldsymbol{\omega}'} \left[V_n(\boldsymbol{\omega}') \;\middle|\; \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi} \right] \tag{4}$$

where $\boldsymbol{X} = \{X_1, ..., X_n\}$ is the set of investment strategies of each firm and $\boldsymbol{\Phi} = \{\Phi_1, ..., \Phi_n\}$ is the set of entry and exit strategies of each firm. The expectation $\mathbb{E}_{\boldsymbol{\omega}'}$ is taken over the realizations of scrap values, entry costs, investment outcomes and the industry-wide shock.

[1] State $\omega_n = 0$ indicates an inactive firm, hence unaffected by industry shocks. State $\omega_n = 1$ is the lowest possible quality level for active firms, where quality cannot decrease any further.

The second computational bottleneck of the model arises from the fact that this expectation consists of $4^{(N-1)}$ terms since each competitor firm state can decrease with the industry shock, stay constant, increase with successful innovation or change to $\omega_n = 0$ through exit.

The exit policy function is given by

$$\Phi_n^*(\boldsymbol{\omega}, \phi_n^x | \omega_n \neq 0) = \arg\max \left\{ \phi_n^x \ , \ \max_{x_n} \ -x_n + \frac{1}{1+\alpha x} \beta W_n^0(\boldsymbol{\omega}, \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi}) + \frac{\alpha x}{1+\alpha x} \beta W_n^1(\boldsymbol{\omega}, \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi}) \right\} \quad (5)$$

where $W_n^k = \mathbb{E}_{\boldsymbol{\omega}'} \Big[V_n(\boldsymbol{\omega}') \Big| \boldsymbol{\omega}, \boldsymbol{X}_{-n}, \boldsymbol{\Phi}_{-n}, \Pr(x_n) = k \Big]$ is the conditional value function and $k \in \{0, 1\}$ is the investment outcome (failure and success, respectively). Given the investment technology, optimal investment is given by

$$x_n^*(\omega) = \max \left\{ 0 \ , \ \frac{1 - \sqrt{\alpha\beta(W_n^1(\boldsymbol{\omega}, \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi}) - W_n^0(\boldsymbol{\omega}, \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi}))}}{\alpha} \right\} \quad (6)$$

The entry policy function is given by

$$\Phi_n^*(\boldsymbol{\omega}, \phi_n^e | \omega_n = 0) = \arg\max \left\{ \phi_n^e \ , \ \beta \ \mathbb{E}_{\boldsymbol{\omega}'} \Big[V_n(\boldsymbol{\omega}') \ \Big| \ \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi} \Big] \right\} \quad (7)$$

Equilibrium. The system of equations given by the value function (eq. 4) and policy functions (eqs. 5, 6, 7) defines the Markov Perfect Equilibrium of the game.

2.3 Solution Method

The solution method proposed by Pakes & McGuire (1994) is value function iteration. One starts with an initial guess for the value and policy functions and calculate the best reply function of a firm to the old guess and updates the value function accordingly. The algorithm stops as soon as the distance between two successive iterations of the value function falls below a threshold $\varepsilon > 0$. It is important to remark that the best reply operator is not a contraction and hence, there is no guarantee of convergence. I find, however, that the algorithm always converges for all the parametrizations explored in the simulations.

3 Approximation Methods

In this section, I review different approximation methods to solve the curse of dimensionality in dynamic stochastic games. Note that there is no conceptual problem in solving games with large number of firms or states. The problem is the computational burden which can make dynamic stochastic games too complex to solve in feasible amounts of time. I explore six approximation methods or alternative equilibrium concepts. They are (1) value function approximation (Keane & Wolpin, 1994), (2) experience-based equilibrium (Pakes & McGuire, 2001), (3) oblivious equilibrium (Weintraub et al., 2008), (4) games in continuous time (Doraszelski & Judd, 2012), (5) constrained optimization (Farias et al., 2012) and (6) games with random moves (Doraszelski & Judd, 2019). These methods can generally be grouped into three classes: methods that rely on a different equilibrium concept (3, 4, 6), methods that rely on learning (2), and methods that rely on function approximation (1, 5). Generally, I find that the first and second classes outperform the third, both in terms of speed and accuracy. Lastly, I also propose a new equilibrium concept, games with random order, which outperforms all other methods in terms of approximation accuracy while still being among the fastest algorithms.

In order to provide a baseline for the different approximation methods, I also include the following algorithm, that I call *baseline approximation* or *myopic equilibrium.* It consists in computing the conditional value function of firm n as

$$W_n^k(\boldsymbol{\omega}) = \frac{\pi_n^*(\boldsymbol{\omega'} | \boldsymbol{\omega}, \Pr(x_n) = k)}{1 - \beta} \quad (8)$$

which corresponds with the discounted future profits of firm n in state $\boldsymbol{\omega}$, conditional on investment outcome $k \in \{0, 1\}$ and keeping all the other present and future industry dynamics constant. This could also being interpreted as a myopic conditional value function, where the firm takes only into account its present profits, its present decisions and the infinite horizon, without taking into account its future actions and its competitor's present and future actions. From this conditional value function, I compute the policy function and the firm value function. This algorithm is extremely fast since it corresponds to a single iteration of the Pakes & McGuire (1994) algorithm, but without the expectation operator. It provides a good approximation of the value function but a poor approximation of the policy functions.

3.1 Value Function Approximation

Keane & Wolpin (1994) proposed an approximation method based on a combination of expected value simulation and value function approximation. In this section, I concentrate only on the value function approximation part.

Their idea is to solve the value function only at a limited number of points in the state space and then to interpolate it over all the remaining points. The resulting approximation crucially depends on

1. the number of points of the state space at which the value function is solved exactly,
2. the choice of these points,
3. the accuracy of the interpolation/prediction algorithm.

The choice of the number of points at which the value function is solved exactly involves a trade-off between speed and accuracy. In Section 4, I report results for a sample size equal to $|\Omega|^{2/3}$. Different concave functions of the dimension of the state space lead to similar results. As suggested in the original paper, I pick the states uniformly at random. For what concerns the interpolation algorithm, I use linear regression over a set of dummy variables for own states and a degree 3 polynomial expansion of the states of the other firms. This method allows different approximation algorithms to be applied to the value function. However, I find that linear regression with a sufficiently rich support is the most efficient in terms of computational time, while still providing great prediction accuracy.

The solution method is value function iteration. However, since the value function is approximated in most of the state space, I average updates over iterations and use an less stringent stopping rule: $\varepsilon = 10^{-2}$. A more stringent stopping rule would lead to more precise approximations, at the expense of computational time. However, with the aforementioned stopping rule, the algorithm is among the slowest.

3.2 Experience-Based Equilibrium

Fershtman & Pakes (2012) proposed a learning-based equilibrium concept based on the computational framework of Pakes & McGuire (2001): the Experience-Based equilibrium. This equilibrium concept dictates that, in equilibrium, players expected values of outcomes are consistent with the actual distribution of outcomes at those states. This implies that firms' actions are optimal given their expectations and their expectations are in line with their experience (hence the name). Note that this equilibrium notion only applies to states visited in equilibrium, while is silent on firm behavior in states outside the recurrent class. This is indeed the main computational advantage of the algorithm: since computations have to be performed only for the states in the recurrent class, this can potentially significantly reduce the computational burden if the recurrent class is small.

In practice, the equilibrium computation algorithm is based on reinforcement learning theory from artificial intelligence. Players start with some initial expectations on the value of their actions ($W_n^0(\boldsymbol{\omega}, k)$, the value of action k of firm n in state $\boldsymbol{\omega}$). Given an action and a realization of the random variables, players update their beliefs on the value of that action from realized profits and state transitions. In particular, players average their observed payoffs in order to form their expectations. In practice, if we denote with $W_n^t(\boldsymbol{\omega}, k)$ the $k-$action

specific value function of firm n in state $\boldsymbol{\omega}$ at iteration t, the updating algorithm is

$$W_n^{t+1}(\boldsymbol{\omega}, k) = \frac{1}{I^t(\boldsymbol{\omega})} V_n^t(\boldsymbol{\omega}, k) + \frac{I^t(\boldsymbol{\omega}) - 1}{I^t(\boldsymbol{\omega})} W_n^t(\boldsymbol{\omega}, k) \tag{9}$$

where $I^t(\boldsymbol{\omega})$ indicates the number of times state $\boldsymbol{\omega}$ was previously visited and $V_n^t(\boldsymbol{\omega}, k)$ is the value obtained from performing the optimal action k at iteration t in state $\boldsymbol{\omega}$ for firm n:

$$V_n^t(\boldsymbol{\omega}, k) = \max_k \ \pi(\boldsymbol{\omega}, k) + W_n^t(\boldsymbol{\omega}, k) \tag{10}$$

Repeating this process starting from the new state generates an iterative sequence of play that ultimately leads to equilibrium beliefs and optimal policies. The authors provide a method to obtain a distance metric to use as a stopping procedure together with an approximation threshold. A more strict approximation threshold leads to a better approximation at the expense of computational time. In my replications I use $\varepsilon = 10^{-2}$ as a distance threshold since it provides a good trade-off between speed and approximation accuracy.

3.3 Oblivious Equilibrium

Weintraub et al. (2008) propose an alternative solution concept for games with a large number of firms: oblivious equilibrium. The name reflects the fact that firms take decisions with a limited perception of the current state. In particular, at each point in time firms observe only their own state and not the state of their competitors. Therefore, firm oblivious strategies depend only on their own state and not on the state of the industry. This greatly reduces the computational burden since the state space is reduced from M^N points to M points and expectations are taken over 4 instead of $4^{(N-1)}$ possible future states.

Since all firms use a common strategy, the state transitions are independent across firms. This implies that we can compute the expected number of firms at quality level ω at time t, $\mathbb{E}[s_t(\omega)]$, as the product of the asymptotic individual probability distributions. The oblivious value function is

$$V_n(\omega_n) = \mathbb{E}_{\boldsymbol{\omega}} \left[\pi_n^*(\boldsymbol{\omega}) \ \middle| \ \omega_n, X_n, \Phi_n \right] + \beta \mathbb{E}_{\omega_n'} \left[V_n(\omega_n') \ \middle| \ \omega_n, X_n, \Phi_n \right] \tag{11}$$

From this expression we observe that only firm n's own strategy determines its state trajectory and ultimately its value. Competitors' strategies enter the value function indirectly in the profit function since other firms' states are not observed and the expectation is taken over their (symmetric) equilibrium strategies.

3.4 Constrained Optimization

Farias et al. (2012) combine a constrained optimization approach with an approximation method using basis functions. Their method relies on the observation that, given the Bellman Operator T, defined as

$$TV_n(\boldsymbol{\omega}) = \pi_n^*(\boldsymbol{\omega}) + \beta \mathbb{E}_{\boldsymbol{\omega}'} \left[V_n(\boldsymbol{\omega}') \ \middle| \ \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi} \right] \tag{12}$$

the value function can be expressed as the solution of the following minimization problem (Bertsekas et al., 1995):

$$\min_{V_n} \ c' V_n \quad s.t. \quad TV_n(\boldsymbol{\omega}) \leq V_n(\boldsymbol{\omega}) \ \forall \boldsymbol{\omega} \in \Omega \tag{13}$$

where c is a component-wise positive vector with the same size of V. As the authors suggest, if one is interested in the long-run behavior of the industry, the weight vector c should be the asymptotic distribution of states. The way to obtain it is to perform a Montecarlo simulation of the industry evolution at each iteration and derive a proxy of the asymptotic distribution from it.

The minimization problem above has a number of variables and constraints equal to the dimension of the state space. In order to decrease the computational burden, the authors suggest a number of steps. First, they

suggest to replace the high-dimensional value function with an approximation based on a lower dimensional combination of basis functions. The larger the number of basis functions, the better the approximation but the slower the algorithm. Second, they suggest to reduce the number of constraints by sampling. As for the weight vector c, an optimal sampling procedure should be proportional to the asymptotic distribution of the states. For what concerns the sample size, the same trade-off as for value function approximation applies: the larger the number of sampled states, the better the approximation but the slower the algorithm. These first two suggestions solve the curse of dimensionality in the number of variables and in the number of constraints. There is however one last bottleneck in the minimization problem: the Bellman Operator has to perform an expected value over a potentially large number of points (exponential in N). The authors suggest to overcome this problem by discretizing the support of the policy functions (e.g. the possible investment levels).

It is important to note that the optimization problem above outputs a value function approximation that is likely to be downwards-biased because of the inequality constraints. The authors suggest inserting a parameter $\theta > 0$ in the constraint in order to relax them: $TV_n(\boldsymbol{\omega}) - \theta \leq V_n(\boldsymbol{\omega})\ ,\ \forall \boldsymbol{\omega} \in \Omega$. However, they do not provide any guideline about how to set such a parameter. I use the distance between sequential function updates as relaxation parameter.

In the simulations, I use the same basis function used for the linear regression algorithm in Section 3.1. I also use the same logic to pick the of constraints: their number is $|\Omega|^{2/3}$ and they are drawn uniformly at random. These two approximations together greatly reduce the computational time, but unfortunately imply a significant approximation error. Therefore, I do not further discretize the action space as it's suggested in the paper. Moreover, I find that iterating the algorithm more than once does not bring significant improvements in accuracy, so I limit the algorithm to two iterations: (1) solve the constrained minimization problem starting with a uniform weight vector c and use the resulting policy function to simulate the industry dynamics, (2) use the estimated asymptotic state distribution c to repeat the first step.

3.5 Games in Continuous Time

Doraszelski & Judd (2012) analyse dynamic stochastic games in continuous time. The main difference with respect to the standard discrete time formulation is that the time path of the state is not a discrete sequence but a right-continuous function of time. Jumps occur at random times according to Poisson processes with hazard rate $\lambda(X_t, \Phi_t, \boldsymbol{\omega}_t)$. The probability that the state changes at time t is given by a function of the state and actions of the players just before time t. Since a change from a state to itself is equivalent to no change of state, one can concentrate only on jumps from a state $\boldsymbol{\omega}$ to a different state $\boldsymbol{\omega'} \neq \boldsymbol{\omega}$. In our specific case, the hazard rate for firm n is $\lambda(X_t, \Phi_t, \boldsymbol{\omega}_t) = \delta + \frac{\alpha x_n}{1+\alpha x_n}$, where the two terms correspond to a jump downwards and upwards, respectively.

Since time is continuous, it's important to distinguish between flow payoffs and changes is wealth that occur when the state jumps. Firm n profits are given by $\pi_n^*(\boldsymbol{\omega}_t)$ and are expressed in dollars per unit of time. Scrap values $\phi_n(\boldsymbol{\omega}_t)$ are instead wealth shocks and are expressed in dollars. The objective function of the firm is the expected discounted total flow of profits plus discrete changes in wealth:

$$V_n(\boldsymbol{\omega}) = \int_0^\infty e^{-\rho t} \pi_n^*(\boldsymbol{\omega}_t) \mathrm{d}t + \sum_{s=0}^{\infty} e^{-\rho T_s} \phi_n(\boldsymbol{\omega}_{T_s}) \tag{14}$$

where ρ is the discount rate. In order to make the models comparable, I set $\rho = -\log(\beta)$.

The equilibrium concept is Markov Perfect Equilibrium. Therefore, even though players' strategies could be time-dependent, the optional strategies are not, which implies that players actions change only when the state changes. The Bellman equation is similar to the one in discrete time

$$V_n(\boldsymbol{\omega}) = \pi_n^*(\boldsymbol{\omega}) + (1 - \rho - \lambda(X, \Phi, \boldsymbol{\omega}))V_n(\boldsymbol{\omega}) + \lambda(X, \Phi, \boldsymbol{\omega}) \mathbb{E}_{\boldsymbol{\omega'}} \Big[\phi_n(\boldsymbol{\omega}) + V_n(\boldsymbol{\omega'}) \ \Big|\ \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi} \Big] \tag{15}$$

which can be rewritten as

$$V_n(\boldsymbol{\omega}) = \frac{1}{\rho + \lambda(X, \Phi, \boldsymbol{\omega}_t)} \left(\pi_n^*(\boldsymbol{\omega}) + \lambda(X, \Phi, \boldsymbol{\omega}) \mathbb{E}_{\boldsymbol{\omega}'} \left[\phi_n(\boldsymbol{\omega}) + V_n(\boldsymbol{\omega}') \;\middle|\; \boldsymbol{\omega}, \boldsymbol{X}, \boldsymbol{\Phi} \right] \right) \tag{16}$$

The solution algorithm is value function iteration, performed in parallel at all states. The computational advantage of expressing the game in continuous time comes from the fact that the probability that more than one coordinate of the state changes simultaneously is zero. Moreover, events which do not change the state can be ignored. Therefore, the number of elements over which the expected value is computed is $3(N-1)$ instead of $4^(N-1)$.

It is important to underline that the equilibrium of the game in continuous time does not have to be interpreted as an approximation of the equilibrium of the games in discrete time. However, modeling the game in continuous time offers considerable computational advantages while still generating similar equilibrium behavior. One important difference concerns the fact that, in the game in continuous time, firms have additional incentives to invest since their investment not only increases the probability of success but also the hazard rate and hence speed of success.

3.6 Games with Random Moves

Doraszelski & Judd (2019) proposed an alternative game structure in which players move in a random sequence. In each period, a firm is taken uniformly at random and gets the opportunity to enter/exit and invest. Static competition takes place in every period. In order to make the model comparable with Ericson & Pakes (1995), the authors divide per period profits by N replacing $\pi_n(\boldsymbol{\omega})$ with $\frac{\pi_n(\boldsymbol{\omega})}{N}$ and take the N^{th} square root of the discount factor replacing β with $\sqrt[N]{\beta}$

I denote with $V_{n|m}(\boldsymbol{\omega})$ the value function of firm n conditional on firm m moving. The value of firm n when it is its turn to move is then

$$V_{n|n}(\boldsymbol{\omega}, \phi_n) = \frac{\pi_n^*(\boldsymbol{\omega})}{N} + \max \left\{ \phi_n \;;\; \max_{x_n} \; -x_n + \sqrt[N]{\beta}\, \mathbb{E}_{k,\omega_n'} \left[V_{n,k}(\boldsymbol{\omega}') \middle| \boldsymbol{\omega}, x_n \right] \right\} \tag{17}$$

The main differences with respect to equation 4 is that the expectation is taken over the future state of firm n alone (ω_n') and with respect to the identity of the next firm to move (k). Hence, the number of terms is $4(N-1)$ instead of $4^{(N-1)}$.

When it's not firm n turn to move, the value function is

$$V_{n|m}(\boldsymbol{\omega}) = \frac{\pi_n^*(\boldsymbol{\omega})}{N} + \sqrt[N]{\beta}\, \mathbb{E}_{k,\omega_m'} \left[V_{n,k}(\boldsymbol{\omega}') \middle| \boldsymbol{\omega}, X_m, \Phi_m \right] \tag{18}$$

Since ex-ante firms do not know who will move next and the mover is selected uniformly at random, the ex-ante value function is given by the average conditional-on-moving value function

$$\bar{V}_n(\boldsymbol{\omega}) \frac{1}{N} \sum_{m=1}^{N} V_{n,m} s(\boldsymbol{\omega}) \tag{19}$$

The equilibrium concept is Markov Perfect Equilibrium where the policy functions are implicitly defined in equation 20. The solution concept is value function iteration. As for the original model, there is no guarantee of convergence since the Bellman Operator does not have to be a contraction. However, in all simulations, the mapping converges to a fixed point given an arbitrarily small convergence threshold.

3.7 Games with Random Order

In this section, I examine an alternative equilibrium concept in which players move sequentially but the order is unknown to the firms and hence random from their point of view. Firms do not observe each other's actions

Parameter	Symbol	Value
Investment efficacy	α	5
Discount factor	β	0.925
Marginal cost	c	5
Industry shock probability	δ	0.7
Convergence accuracy	ε	10^{-10}
Market size	m	5
Entry cost distribution	$F(\phi_{in})$	$[0, 0.4]$
Scrap value distribution	$F(\phi_{out})$	$[0, 0.2]$
Demand parameter	ω^*	5

Table 1: Model Parametrization

nor the evolution of the states. They only observe the state when it's their turn to move and they know they are moving sequentially. As in Doraszelski & Judd (2019), in order to make the results comparable with the original model, I divide per period profits by N replacing $\pi_n(\boldsymbol{\omega})$ with $\frac{\pi_n(\boldsymbol{\omega})}{N}$ and take the N^{th} square root of the discount factor replacing β with $\sqrt[N]{\beta}$.

I denote with $V_{n|m}(\boldsymbol{\omega})$ the value function of firm n conditional of firm m moving. The value of firm n when it is its turn to move is then

$$V_{n|n}(\boldsymbol{\omega}, \phi_n) = \frac{\pi_n^*(\boldsymbol{\omega})}{N} + \max\left\{\phi_n \; ; \; \max_{x_n} -x_n + \sqrt[N]{\beta}\, \mathbb{E}_{K,\omega_n'}\left[V_{n,K}(\boldsymbol{\omega}')\middle|\boldsymbol{\omega}, x_n\right]\right\} \tag{20}$$

where K is one possible order of moves. The main difference with respect to the value function of Doraszelski & Judd (2019) in equation 20 is that the expectation is taken with respect to K instead of k. Therefore, the number of terms over which the expectation is taken is $4(N-1)!$ instead of $4(N-1)$. However, this increase in computational burden could be compensated by a better approximation performance and lower number of iterations (as turns out to be the case).

4 Computational Results

I compare the approximation methods listed in section 3 for different number of firms (N). In Appendix A.5, I compare the approximation methods across number of ladder steps (M). I use the parametrization reported in table 1, which is the original parametrization of Pakes & McGuire (1994) with the only difference concerning scrap values and entry costs. In the original paper, the scrap values and entry costs are deterministic. In order to ensure equilibrium existence, I follow Doraszelski & Satterthwaite (2010) and use stochastic scrap values and entry costs. I use uniform distributions for both random variables with the same expected value and the original deterministic parameters. I report summary statistics of the different equilibria in Appendix A.1.

All computations are coded in MATLAB and performed on a standard machine. I keep the code as plain as possible in order to ensure that the results are comparable across methods. Whenever two algorithms overlap, I use the same code to ensure maximal comparability. I present results relative to the full solution method. In particular, to compare the different value function approximation, my preferred metric is the Weighted Absolute Relative Deviation (WARD) of the value function:

$$WARD(V^{approx}) = \sum_{\boldsymbol{\omega}\in\Omega} \Pr^{pm94}(\boldsymbol{\omega}) \left|\frac{V^{approx}(\boldsymbol{\omega}) - V^{pm94}(\boldsymbol{\omega})}{V^{pm94}(\boldsymbol{\omega})}\right| \tag{21}$$

where $\Pr^{pm94}(\boldsymbol{\omega})$ is the asymptotic probability of state $\boldsymbol{\omega}$ computed using the full solution value function V^{pm94}. In Appendix A.4, I present the same results using the Mean Absolute Relative Distance and the Weighted Squared Relative Distance. The results are very similar.

In Figure 1, I plot the average weighted absolute relative deviation of the value function over the number of firms N, for the different algorithms, over 30 iterations. In Appendix A.3 I report all the main figures with the standard deviation for each algorithm.

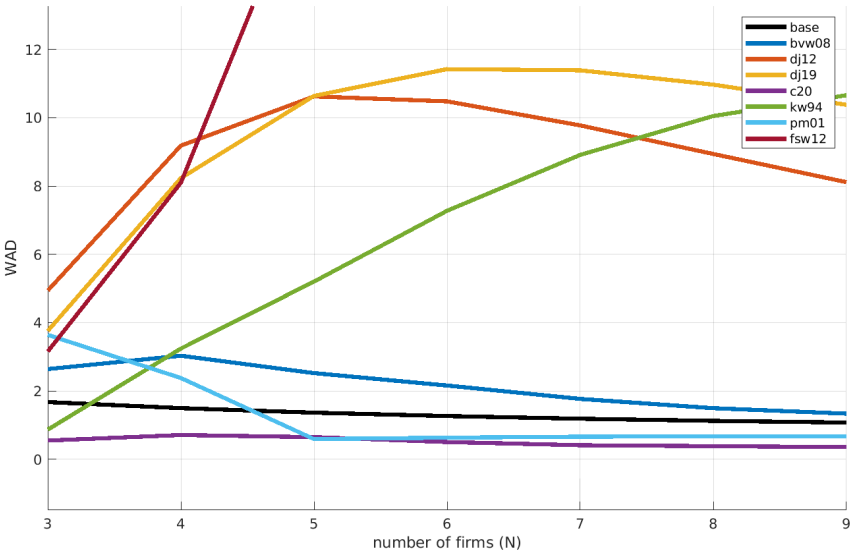
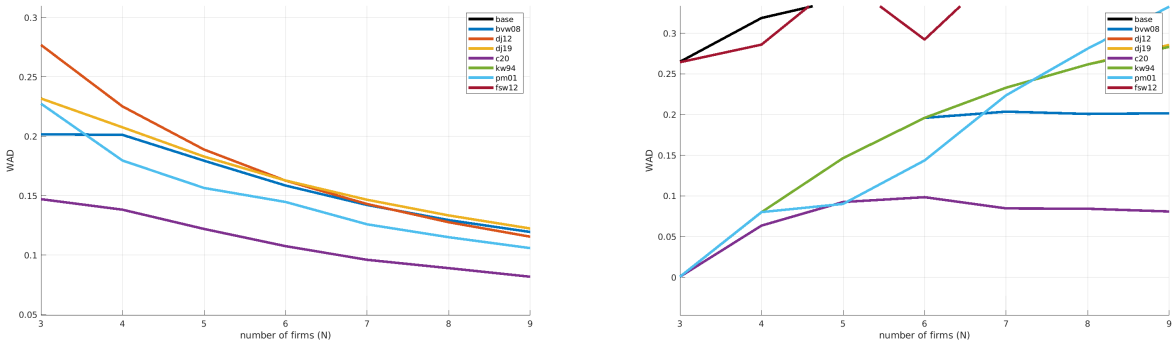


Figure 1: Value function approximation accuracy

For what concerns approximation performance, the simulations results show two clear winners: games with random order (*c20*) and experience-based equilibrium (*pm01*). They are the only two methods that beat the baseline method (*base*) in terms of value function approximation accuracy. The closest runner up the oblivious equilibrium (*bvw08*).

However, a more relevant metric is how well an algorithm approximates the policy functions. In fact, whether we are interested in the computational model per-se or whether we use it as an input into structural estimation, the ultimate object of interest is firm equilibrium behavior. In Figure 2 I plot the relative approximation performance (measured using the Weighted Absolute Relative Deviation) of the investment and exit policy functions for the algorithms considered.



(1) Investment Policy

(2) Exit Policy

Figure 2: Policy functions approximation accuracy

From both figures we observe that games with random order (*c20*) outperforms competing algorithms with experience-base equilibrium (*pm01*) and oblivious equilibrium (*bvw08*) coming close. Moreover, the gap between games with random order (*c20*) and all other methods seems to be widening with the dimension of the state space.

It is important to note that all the other methods except for games in continuous time (*dj12*) and games with random moves (*dj19*) can be tweaked in order to obtain a better performance at the cost of a slower algorithm. However, as we can see from the next graph, given the current parametrization, none of them is sensibly faster than *c20*, *base* or *bvw08*.

In Figure 3, I plot the logarithm of the relative computational time performance with respect to the full solution method over the number of firms N for the different algorithms.

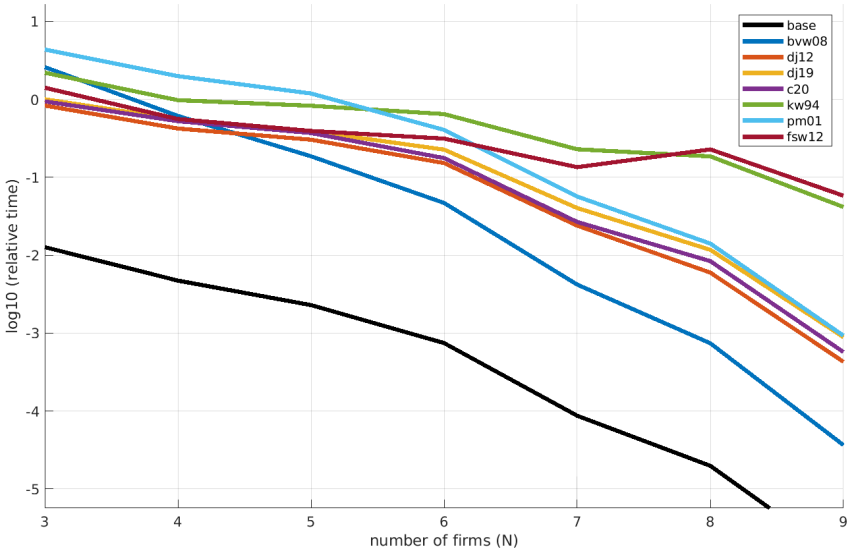


Figure 3: Relative time performance.

For what concerns computational time, the winner in this domain is the baseline approximation (*base*) followed by the oblivious equilibrium (*bvw08*). These two methods outperform all the others by orders of magnitude. Among the rest, experience-based equilibrium (*pm01*), games in continuous time (*dj12*), games with random moves (*dj19*) and games with random order (*c20*) all have a very similar performance in terms of computational time. In Appendix A.3, I report a decomposition of the computational time into time per iteration and number of iterations for all algorithms.

Overall, the simulation results highlight the fact that the choice of the approximation algorithm involves an speed-accuracy trade-off. However, it also shows that some methods dominate others across all dimensions. More generally, alternative equilibrium formulations outperform sampling-based methods.

5 Conclusion

Simulation results show that the new approximation method proposed in this paper, games with random order, outperforms other algorithms in terms of approximation accuracy, while still being very efficient in terms of computational time. For what concerns computational time, the faster algorithm is the oblivious equilibrium of Weintraub et al. (2008). Methods that rely on sampling are generally outperformed and dominated by alternative equilibrium concepts. However, since sampling methods are sensible to the algorithm parametrization, it is hard to draw more general conclusions.

References

Aguirregabiria, V., & Mira, P. (2002). Swapping the nested fixed point algorithm: A class of estimators for discrete markov decision models. *Econometrica*, *70*(4), 1519–1543.

Asplund, M., & Nocke, V. (2006). Firm turnover in imperfectly competitive markets. *The Review of Economic Studies*, *73*(2), 295–327.

Bajari, P., Benkard, C. L., & Levin, J. (2007). Estimating dynamic models of imperfect competition. *Econometrica*, *75*(5), 1331–1370.

Bertsekas, D. P., Bertsekas, D. P., Bertsekas, D. P., & Bertsekas, D. P. (1995). *Dynamic programming and optimal control* (Vol. 1) (No. 2). Athena scientific Belmont, MA.

Besanko, D., Doraszelski, U., Kryukov, Y., & Satterthwaite, M. (2010). Learning-by-doing, organizational forgetting, and industry dynamics. *Econometrica*, *78*(2), 453–508.

Caplin, A., & Nalebuff, B. (1991). Aggregation and imperfect competition: On the existence of equilibrium. *Econometrica: Journal of the Econometric Society*, 25–59.

Doraszelski, U., & Judd, K. L. (2012). Avoiding the curse of dimensionality in dynamic stochastic games. *Quantitative Economics*, *3*(1), 53–93.

Doraszelski, U., & Judd, K. L. (2019). Dynamic stochastic games with random moves. *Quantitative Marketing and Economics*, *17*(1), 59–79.

Doraszelski, U., & Pakes, A. (2007). A framework for applied dynamic analysis in io. *Handbook of industrial organization*, *3*, 1887–1966.

Doraszelski, U., & Satterthwaite, M. (2010). Computable markov-perfect industry dynamics. *The RAND Journal of Economics*, *41*(2), 215–243.

Ericson, R., & Pakes, A. (1995). Markov-perfect industry dynamics: A framework for empirical work. *The Review of Economic Studies*, *62*(1), 53–82.

Farias, V., Saure, D., & Weintraub, G. Y. (2012). An approximate dynamic programming approach to solving dynamic oligopoly models. *The RAND Journal of Economics*, *43*(2), 253–282.

Fershtman, C., & Pakes, A. (2012). Dynamic games with asymmetric information: A framework for empirical work. *The Quarterly Journal of Economics*, *127*(4), 1611–1661.

Goettler, R. L., & Gordon, B. R. (2011). Does amd spur intel to innovate more? *Journal of Political Economy*, *119*(6), 1141–1200.

Gowrisankaran, G. (1999). A dynamic model of endogenous horizontal mergers. *The RAND Journal of Economics*, 56–83.

Gowrisankaran, G., & Rysman, M. (2012). Dynamics of consumer demand for new durable goods. *Journal of political Economy*, *120*(6), 1173–1219.

Hotz, V. J., & Miller, R. A. (1993). Conditional choice probabilities and the estimation of dynamic models. *The Review of Economic Studies*, *60*(3), 497–529.

Igami, M. (2017). Estimating the innovator's dilemma: Structural analysis of creative destruction in the hard disk drive industry, 1981–1998. *Journal of Political Economy*, *125*(3), 798–847.

Keane, M. P., & Wolpin, K. I. (1994). The solution and estimation of discrete choice dynamic programming models by simulation and interpolation: Monte carlo evidence. *the Review of economics and statistics*, 648–672.

Miao, J. (2005). Optimal capital structure and industry dynamics. *The Journal of finance*, *60*(6), 2621–2659.

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*, *69*(5), 1261–1281.

Ryan, S. P. (2012). The costs of environmental regulation in a concentrated industry. *Econometrica*, *80*(3), 1019–1061.

Weintraub, G. Y., Benkard, C. L., & Van Roy, B. (2008). Markov perfect industry dynamics with many firms. *Econometrica*, *76*(6), 1375–1411.

A Appendix

A.1 Summary Statistics

In this section I report summary statistics of the different equilibria analyzed in Section 4. I report the following metrics: average number of firms, average firm state in the quality ladder (conditional on being active), average profits, average investment and average exit probability.

N	Active Firms	State	Profits	Investment	Exit Pr.
3	2.9983	2.8549	2.6267	1.0117	0.9997
4	3.6803	2.6476	2.2088	0.8641	0.9635
5	4.2681	2.4906	1.9469	0.7678	0.9323
6	4.8247	2.3551	1.7432	0.6921	0.9065
7	5.3689	2.2373	1.5745	0.6292	0.8855
8	5.9073	2.1349	1.4334	0.5757	0.8682
9	6.4297	2.0463	1.3162	0.5303	0.8526

Table 2: Summary Statistics

From Table 2 we observe that as the number of firms increases, there are more firms active but each of them makes less profits, invests less and exits less frequently

A.2 Variation

In this section I repeat the main exercise but reporting standard deviations. The area around each line indicates one standard deviation.

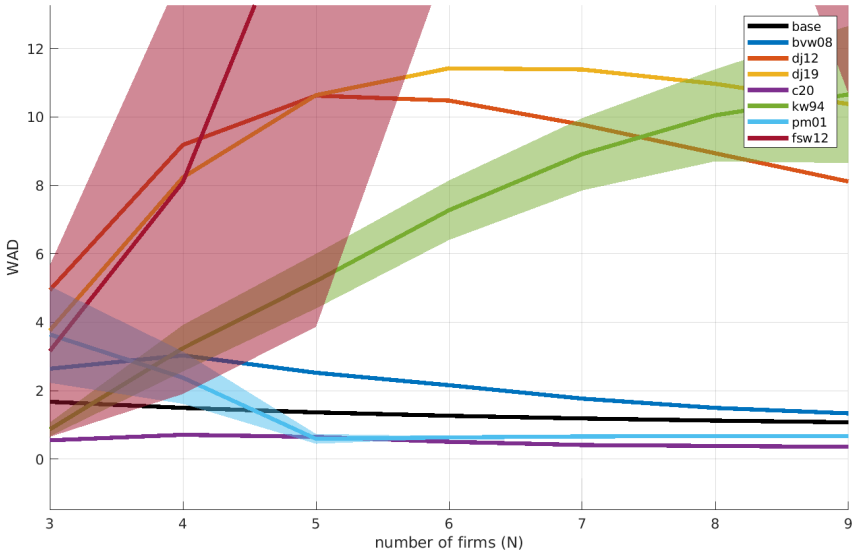
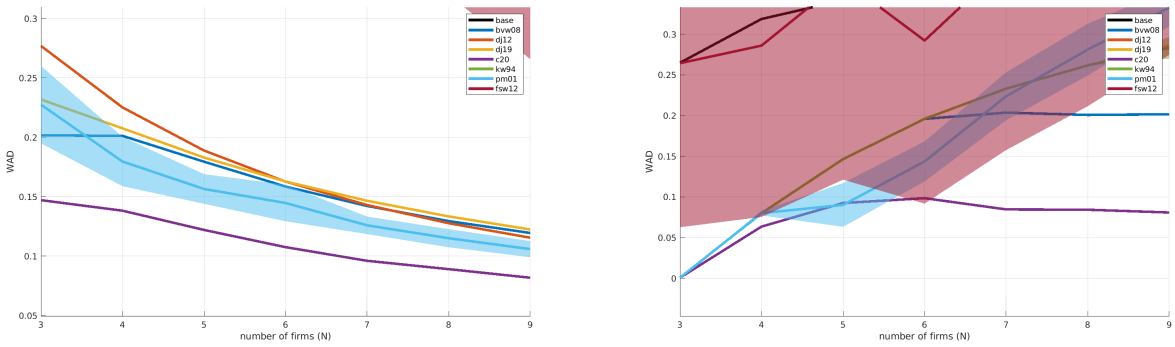


Figure 4: Value function approximation accuracy



(1) Investment Policy

(2) Exit Policy

Figure 5: Policy functions approximation accuracy

From figure 4 and 5 we see that the approximation accuracy of sampling methods, *kw94* and *fsw12*, is extremely volatile. The average values mask a high sensibility to the sampling variation.

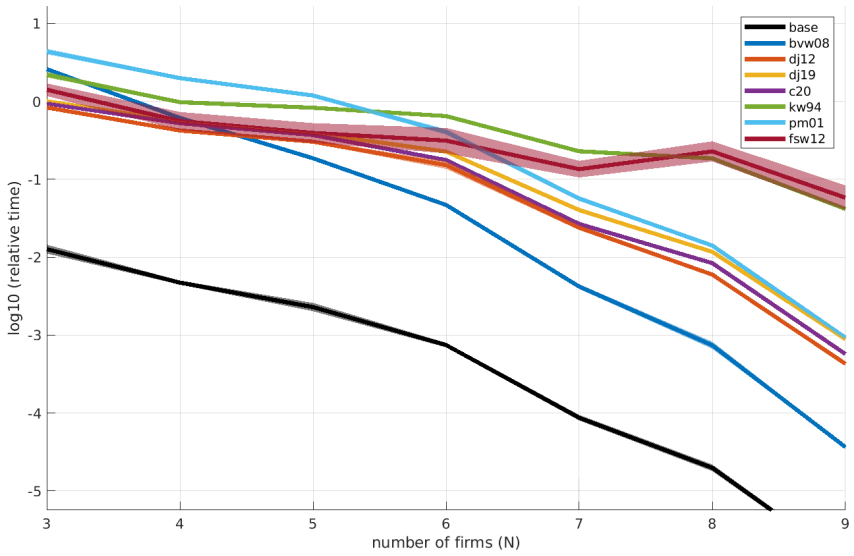
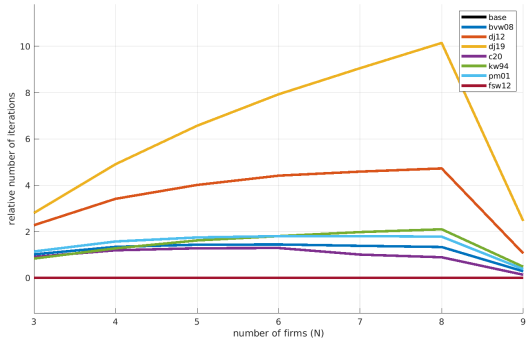


Figure 6: Relative time performance.

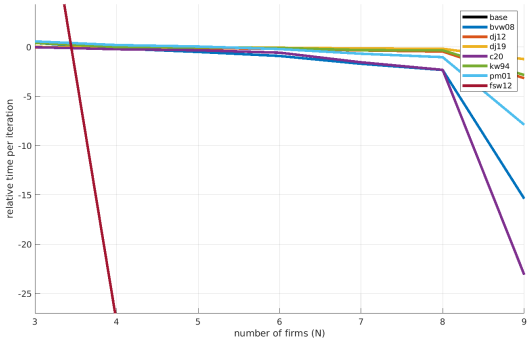
From Figure 6 we observe that the variation in computational time for most algorithms is imperceptible, confirming the robustness of the results. Only *fsw12* shows considerable variation in computational time.

A.3 Computational Time Decomposition

In this section I decompose the computational time into number of iterations and time per iteration. The results are computed over 30 iterations.



(1) Number of Iterations



(2) Time per Iteration

A.4 Performance Metrics

In this section, I consider other performance metrics in order to assess the accuracy of the algorithms. In Section 4 I used the Weighted Absolute Relative Deviation (WARD). Here I report results for two other metrics:

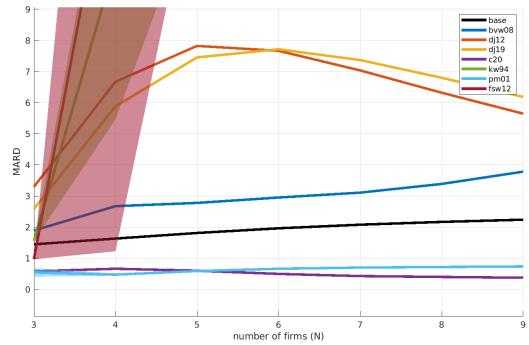
1. Mean Absolute Relative Deviation

$$MARD(V^{approx}) = \frac{1}{|\Omega|} \sum_{\boldsymbol{\omega} \in \Omega} (\boldsymbol{\omega}) \left| \frac{V^{approx}(\boldsymbol{\omega}) - V^{pm94}(\boldsymbol{\omega})}{V^{pm94}(\boldsymbol{\omega})} \right|$$

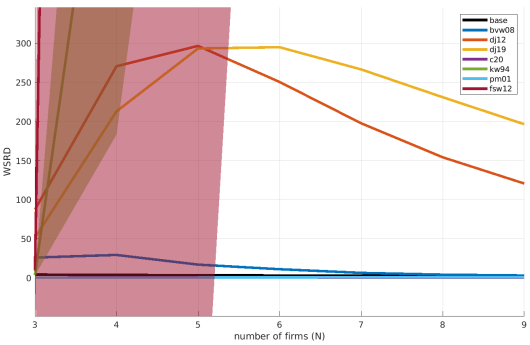
2. Weighted Squared Relative Deviation

$$WSRD(V^{approx}) = \sum_{\boldsymbol{\omega} \in \Omega} \Pr(\boldsymbol{\omega}) \left(\frac{V^{approx}(\boldsymbol{\omega}) - V^{pm94}(\boldsymbol{\omega})}{V^{pm94}(\boldsymbol{\omega})} \right)^2$$

In the next figure, I repeat the same exercise of Figure 1 but for the MARD and WSRD metrics instead of WARD.



(1) Mean Absolute Relative Deviation



(2) Weighted Squared Relative Deviation

From the two figures we observe that the results of Figure 1 are robust to the choice of the distance metric.

A.5 Results for M

In this section I repeat the analysis of Section 4 but for M, the number of states in the quality ladder, instead of N, the number of firms. First, I plot the value function in Figure **??**

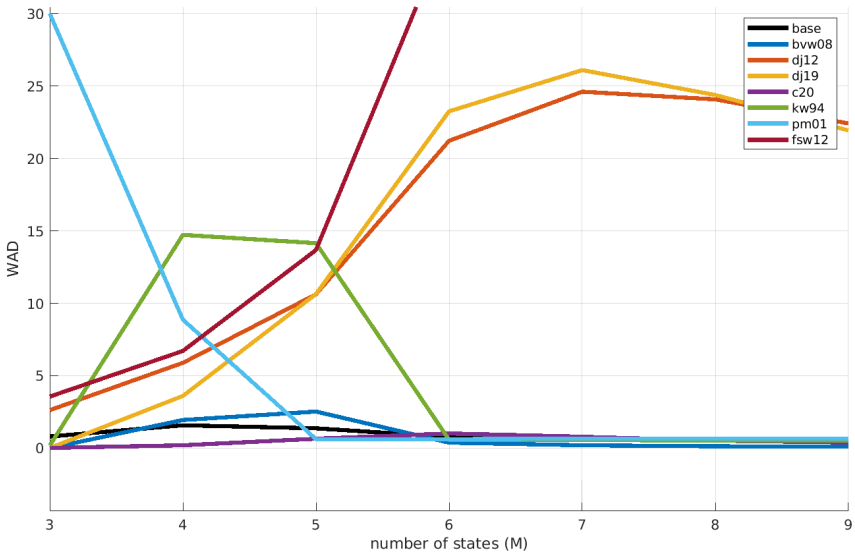
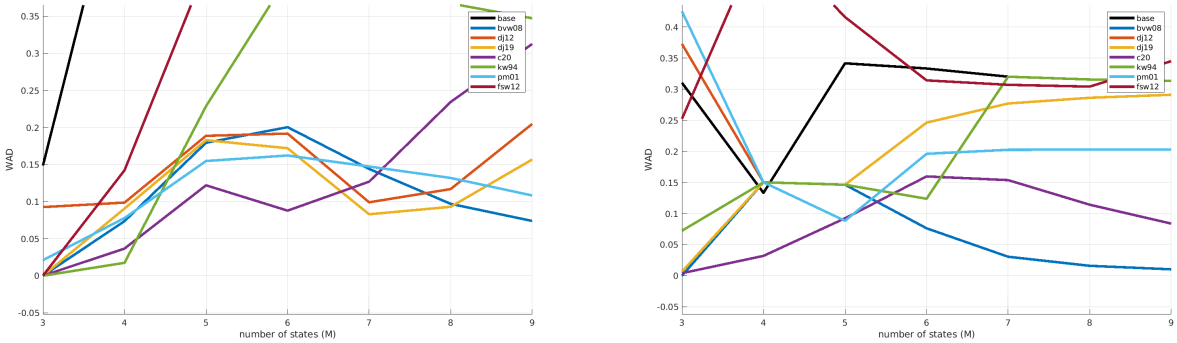


Figure 9: Value Function distance



(1) Investment Policy

(2) Exit Policy

Figure 10: Policy functions approximation accuracy

From Figures 9 and 10, we see a clear distinction between two sets of approximation methods. *dj12*, *dj19* and *fsw12* perform significantly worse than the rest. On the other hand, *c20* and *bvw08* perform best for different values of M.

In Figure 11 I report the relative computational time.

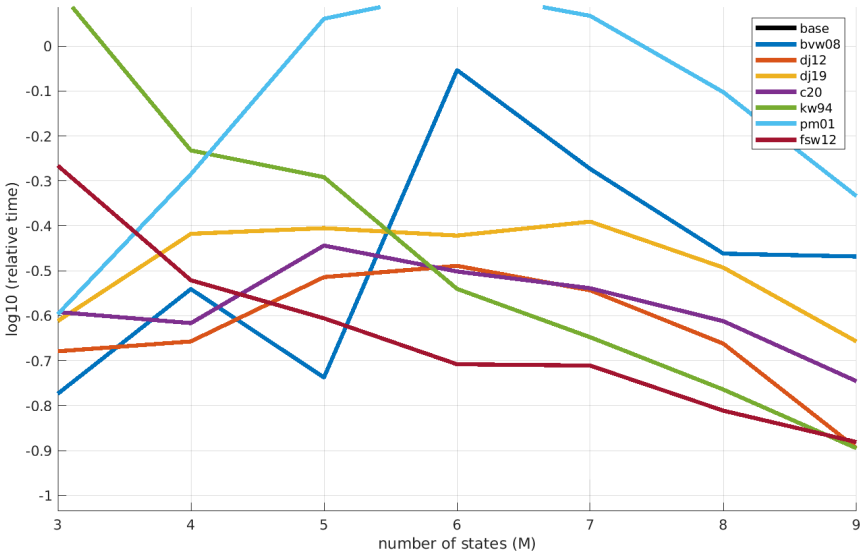


Figure 11: Relative Computational Time

We observe wide heterogeneity in algorithm performance.