Computing Equilibria of Dynamic Games

Kenneth Judd and Şevin Yeltekin

July 4, 2011

1 Introduction

In two influential papers, Abreu, Pearce and Stacchetti (APS) (1986, 1990) developed set-valued dynamic programming techniques for solving certain classes of repeated game. They showed that, for these games, the set of sequential equilibrium payoffs can be computed as a fixed point of an operator analogous to the Bellman operator in dynamic programming. These methods can be extended to cover a large class of dynamic games that arise naturally in industrial organization, macroeconomics, and public finance.[1] In the dynamic case, the object of interest is a correspondence that maps a physical state variable to sets of equilibrium payoffs. APS-based methods imply that the equilibrium payoff correspondence is a fixed point of a monotone operator and can be obtained via an iteration of this operator. In principle, this iterative procedure can be implemented numerically to solve for the equilibrium payoff correspondence. However, its practical numerical implementation requires an approximation scheme for correspondences that is efficient and consistent with the underlying structure of the monotone operator. In the first part of this paper, we provide such a

[1] See Atkeson (1991), Phelan and Stacchetti (2001), Sleet (1996) and Fernandez-Villaverde and Tsyvinski (2002) for applications to international lending and macroeconomic policy.

scheme.

As a first step, following Cronshaw and Luenberger's (1994) and Judd, Yeltekin and Conklin's (JYC)(2003) analyses of repeated games, we convexify the dynamic game by introducing a public randomization device. We then show that sets of polygons provide an efficient and consistent means of approximating the equilibrium payoff correspondences from such convexified games. Moreover, they enable us to obtain inner and outer approximations to the equilibrium payoff correspondence. Since the latter lies between these approximations, the difference between them provides us with an error bound for gauging the accuracy of the procedure.[2] This approach improves both on the discretization and linear interpolation methods that have been used elsewhere.[3] The former is non-parsimonious and is vulnerable to the curse of dimensionality, the latter delivers neither an inner nor an outer approximation and lacks error bounds. Additionally, we provide a third algorithm to compute the equilibrium strategies that give rise to the equilibrium values obtained from the inner and outer approximations.

In the second part of the paper, we apply our computational methods to an important dynamic game the industrial organization literature has not been able to analyze successfully: oligopolistic competition with endogenous productive capacity. So far the I.O. literature on dynamic games (See Kreps and Scheinkman (1983), Davidson and Deneckere (1986, 1990) and Benoit and Krishna (1987)) has made crucial simplifying assumptions such as choosing capacity once and for all in period 0 (hence stripping the model of any of its interesting dynamics) and/or restricting attention to symmetric Markov equilibria. We relax both of these assumptions and compute the equilibria of this fully dynamic game using

[2] The usefulness of inner and outer approximations in providing error bounds was first emphasised by JYC (2003) in the context of repeated games.

[3] See Chang (1998) for an application of discrete methods to a repeated policy game. See Phelan and Stacchetti (2001) for an application of linear interpolation methods to a dynamic policy game.

our newly developed methods. Our results indicate that there is rich set of equilibrium outcomes that had not been discovered due to these restrictive assumptions. These include, for example, asymmetric equilibria where market power (determined by the capacity size and share of total output) fluctuates between firms and equilibria in which firms do strive for cooperation after a phase of uncooperative behavior that involves over investment and over production. The results indicate that symmetric Markov games are not the right games to use for studying collusion or cooperation, as the nature of cooperation evolves with the history of firm interaction and investment. These history dependent outcomes can only be captured by considering all equilibria. The techniques developed in the first part of the paper can compute all history-dependent equilibria.

2 Supergames with State Variables

We start our analysis by describing supergames with state variables. N infinitely lived agents play a dynamic game. Let the state of the i-th agent be denoted by the variable $x_i \in X_i$. Let $x \in X = \times_{i=1}^{N} X_i$ denote an 'aggregate state'. The actions of player i in each stage game and in each state come from a finite set A_i. An action profile is any element of $A = \times_{i=1}^{N} A_i$. The aggregate state evolves according to a function $g : A \times X \to X$.

Each player has a per period payoff function $\Pi_i : A \to \Re$. The minimal and maximal payoffs for each player in each state are defined as:

$$\underline{\Pi}_{i,x} \equiv \min_{a \in A} \Pi_i(a, x)$$

$$\overline{\Pi}_{i,x} \equiv \max_{a \in A} \Pi_i(a, x)$$

In each state x, the set of equilibrium payoffs is thus contained within the compact set:

$$W_x = \times_{i=1}^{N} [\underline{\Pi}_{i,x}, \overline{\Pi}_{i,x}].$$

Let $W : X \rightrightarrows \Re^N$ be the correspondence that maps x to W_x. The graph of W is clearly compact.

The action space for the dynamic game is A^∞. A t-period history, h_t, is a sequence $\{\{a_s, x_s\}_{s=0}^{t-1}, x_t\}$ with $x_s = g(x_{s-1}, a_{s-1})$ and $a_s \in A$. The set of t-period histories are denoted H_t. The i-th agent chooses sequences of points in A_i^∞ according to the function:

$$w_i(a^\infty, x^\infty) = \frac{1-\delta}{\delta} E_0 \Sigma_{t=1}^{\infty} \delta^t \Pi_i(a_t, x_t). \tag{1}$$

where β is the common discount factor amongst agents. A strategy for player i is a sequence of functions $\{\sigma_{i,t}\}_{t=0}^{\infty}$ with $\sigma_{i,t} : H_t \to A_i$.

Let $V^* \equiv \{V_x^* | x \in X\}$ denote the subgame perfect equilibrium payoff correspondence and $\mathcal{P}$ the set of all correspondences $\mathcal{W} : X \rightrightarrows \Re^N$ such that i) the graph of $\mathcal{W}$ is compact and ii) the graph of $\mathcal{W}$ is contained within the graph of $\mathcal{P}$. V^* may be shown to be an element of $\mathcal{P}$.

We follow the recursive approach of APS. In the recursive formulation each subgame perfect equilibrium payoff vector is supported by a profile of actions consistent with Nash play in the current period and a vector of continuation payoffs that are themselves payoffs in some subgame perfect equilibrium. The key to finding V^* is the construction of *self-generating* correspondences. The concept may be formalized by introducing the operator $B^* : \mathcal{P} \to \mathcal{P}$.

Let $\mathcal{W} \in \mathcal{P}$. $B^*(\mathcal{W})_x$ is the set of possible payoffs consistent with Nash play in state x today and continuation values drawn from $\mathcal{W} \equiv \{\mathcal{W}_x \mid x \in X\}$:

$$B^*(\mathcal{W})_x = \cup_{(a,w)}\{(1-\delta)\Pi(a,x)+\delta w\}$$

subject to:

$$w \in \mathcal{W}_{g(a,x)}$$

and for each $\forall i \in N$, $\forall \tilde{a} \in A_i$

$$IC\,(i,a,\tilde{a},w) \geq 0 \tag{2}$$

where Equation (2) is the incentive compatibility condition. More precisely,

$$IC\,(i,a,\tilde{a},w) = (1-\delta)\Pi_i(a_i,a_{-i},x)+\delta w_i - \big((1-\delta)\Pi_i(\tilde{a},a_{-i},x)+\delta\mu_{i,g(\tilde{a},a_{-i},x)}\big)$$

is a function that expresses the gain to player i from playing a and receiving a continuation value of w_i instead of playing $\tilde{a}$ and receiving continuation value $\mu_{i,g(\tilde{a},a_{-i},x)}$. For each agent i, and action profile a, the next period state will be $g(a,x)$ and the utility promise w_i has to be taken from the set of possible values in the state $g_{a,x}$. If agent i deviates with action $\tilde{a}$ in state x, then his current payoff is $\Pi_i(\tilde{a},a_{-i},x)$ plus the worst payoff for player i in state $g(\tilde{a},a_{-i},x)$, expressed as $\mu_{i,g(\tilde{a},a_{-i},x)}$. IC is the temptation to deviate by playing $\tilde{a}$ and must be non-negative for all players i and all deviations $\tilde{a}$ in order for a to be incentive compatible.

Therefore b is in $B^*(\mathcal{W})_x$ if there is a continuation payoff $w \in \mathcal{W}_{g(a,x)}$ such that b is the value of playing a today, and, for each i, player i will choose to play a_i because he believes that to do otherwise would yield him the worst continuation payoff in next period's state $g\,(a,x)$.

A correspondence $\mathcal{W}$ is self-generating if $\mathcal{W} \subseteq B^*(\mathcal{W})$.[4] Using an extension of the

[4] Here and for the remainder of the paper, we denote a correspondence $\mathcal{W}$ as a subset of $\tilde{\mathcal{W}}$, if each component of $\mathcal{W}$, i.e., $\mathcal{W}_x$, is a subset of the corresponding component of $\tilde{\mathcal{W}}$.

arguments of Cronshaw and Luenberger (1990,1994), it is straightforward to show that any self-generating correspondence is contained within the equilibrium payoff correspondence, V^* and that the equilibrium payoff correspondence is itself self-generating. One can then establish that a unique maximal fixed point of B^* exists and that the equilibrium payoff correspondence is the maximal fixed point of the operator B^*.

The operator B^* has important properties that can be exploited for computing the equilibrium correspondence. Specifically, B^* is monotone in the set inclusion ordering, so that if $\mathcal{W}_1 \subseteq \mathcal{W}_2$, then $B^*(\mathcal{W}_1) \subseteq B^*(\mathcal{W}_2)$. Additionally, it preserves compactness. It is then straightforward to show that the equilibrium value correspondence V^*, the maximal fixed point of the mapping B^*, may be obtained by repeatedly applying this operator to a correspondence that is known to contain V^*.

3 Approximating Value Correspondences

Approximating the iteration described in APS requires that the candidate value correspondences be efficiently represented on a computer and that the monotonicity property of the B^*-operator is preserved. We proceed in two steps. First, we convexify the underlying game and its equilibrium value correspondence via public randomization. Then we develop methods for approximating convex-valued correspondences.

3.1 Public Randomization

We alter the game described in Section 2 by including public randomization. We assume that in each stage of the game, there is a public lottery that depends on the players' current actions and the support of the lottery is contained in $\mathcal{W}_{g(a,x)}$. The public lottery specifies the continuation values for the next period. If $\mathcal{W}$ is the possible continuation value correspondence at time t of $S^{\infty}(X)$, then $co(\mathcal{W})$ is the value of correspondence of possible time t ex-ante (before the outcome of the lottery) continuation values for $S^{\infty}(X)$ with public randomization. Then $B^{*}(co(\mathcal{W}))$ the equilibrium value correspondence at t, and $co(B^{*}(co(\mathcal{W})))$ is the set of ex-ante continuation values available at $t-1$ for $S^{\infty}(X)$ with public randomization. Let V be the ex-ante continuation value correspondence which can occur in equilibrium for the supergame with public randomization. Therefore, V is convex-valued and bounded. Let B be monotone operator for the game with public randomization. Therefore B takes the form:

$$B(\mathcal{W}) = co(B^{*}(co(\mathcal{W}))), \qquad \mathcal{W} \in \mathcal{P}. \tag{3}$$

It is straightforward to show that B is monotone and V is the largest fixed point of B.

4 Outer and Inner Approximations

We use two types of schemes to approximate convex-valued correspondences: inner and outer approximations. The two approximations are defined precisely below.

Definition 1. *If $Z \subset Y \subset \Re^{N}$ is set of m points, then the* inner approximation *to Y generated by Z is $co(Z)$. Furthermore if the correspondence Z is such that cardinality of Z_x is finite for each x, and $Z \subset \mathcal{W}$, then an* inner approximation *to the correspondence $\mathcal{W}^{I}$ is $co(Z_x)$ for each $x \in X$.*

Definition 2. *If Z is a set of m points on the boundary of a set $Y \subset \Re^{N}$ and $R \subset \Re^{N}$ a set of m corresponding subgradients oriented such that $(\mathbf{z}^{l} - \mathbf{y}) \cdot \mathbf{r}^{l} > 0$ for $\mathbf{y} \in Y$, then the* outer approximation *of Y generated by (Z, R) is*

$$Y^{O} = \bigcap_{l=1}^{m} \{\mathbf{z} \in \Re^{N} | \mathbf{r}^{l} \cdot \mathbf{z} \leq \mathbf{r}^{l} \cdot \mathbf{z}^{l}\}$$

The outer approximation *to correspondence $\mathcal{W}$, $\mathcal{W}^O$, generated by (Z, R) is defined similarly.*

The critical property of B is that it maps correspondences into convex-valued correspondences and that it is monotone. We define inner and outer monotone approximations of the operator B that preserve these critical properties.

Definition 3. *A mapping $B^I : \mathcal{P} \to \mathcal{P}$ ($B^O : \mathcal{P} \to \mathcal{P}$) is an inner (outer) approximation of B if*

1. *$\forall\ \mathcal{W} \in \mathcal{P},\ B^I(\mathcal{W}) \subseteq B(\mathcal{W})$ $(B^O(\mathcal{W}) \supseteq B(\mathcal{W}))$, and*

2. *$\forall\ \mathcal{W},\ \mathcal{W}' \in \mathcal{P},\ if\ \mathcal{W} \subseteq \mathcal{W}'$ then $B^I(\mathcal{W}) \subseteq B^I(\mathcal{W}')$ $(B^O(\mathcal{W}) \subseteq B^O(\mathcal{W}'))$.*

4.1 Computing the Outer Approximation

The definition of an outer monotone approximation directly implies the following proposition which relates the maximal fixed point of B^O to the maximal fixed point of B and also provides a sufficient condition for the outer approximation scheme to converge.

Proposition 1. *Suppose $B^O(\cdot)$ is an outer monotone approximation of $B(\cdot)$. Then the maximal fixed point of B^O contains V. More precisely, if $\mathcal{W} \supseteq B^O(\mathcal{W}) \supseteq V$, then $B^O(\mathcal{W}) \supseteq B^O(B^O(\mathcal{W})) \supseteq \cdots \supseteq V$.*

Proof. $\cdots$ □

The following Lemma asserts that the correspondence W is a good initial guess for the outer approximation to the equilibrium value correspondence in all games.

Lemma 1. *$W \supseteq B^O(W) \supseteq V$.*

Problems without state variables only require the approximation of a single value set at each iteration. In contrast, the state variable problem requires that a collection of approximated value sets, one for each element of the (finite) state space be found at each iteration. Additionally, the latter problem requires that the implications of each action profile (and each deviation from an action profile) for the future state be found, and that future continuation values be obtained appropriately from the relevant continuation value set.

Outer Approximation Algorithm for Supergames with State Variables: Hyperplane Method

1. Inputs:

   (a) Subgradients: Set of normals

   $$R_x = \{\mathbf{r}_x^1, ..., \mathbf{r}_x^{m_x}\}$$

   (b) Set of boundary points

   $$Z_x = \{\mathbf{z}_x^1, ..., \mathbf{z}_x^{m_x}\}$$

   Inputs (a) and (b) define the correspondence $\mathcal{W}$. For each $x \in X$

   $$\mathcal{W}_x = \cap_{j=1}^{m_x}\{\mathbf{b} \in \Re^N |\ \mathbf{r}_x^j \cdot (\mathbf{b} - \mathbf{z}_x^j) \leq 0\}$$

2. Choose search subgradients:

   $$S_x = \{\mathbf{s}_x^1, ..., \mathbf{s}_x^{l_x}\}$$

3. Compute the new subgradient and boundary point sets, R^+ and Z^+, that together represent an outer approximation to $B(\mathcal{W})$. For each $x \in X$ and each $\mathbf{s} \in S_x$ do

(a) for each action profile $a \in A$ do

$$c(a, s, x) = \max_{w} \mathbf{s} \cdot [(1-\delta)\Pi(a, x) + \delta w] \quad s.t.$$

$$(i)\ w \in \mathcal{W}_{g(a,x)}$$

$$(ii) \forall i \in N, \forall \tilde{a} \in A_i, IC\,(i, a, \tilde{a}, w) \geq 0.$$

Let $w^*(a, \mathbf{s}, x)$ be the *argmax* of the previous problem.

Let $v(a, \mathbf{s}, x) = (1-\delta)\Pi(a, x) + \delta w^*(a, \mathbf{s}, x)$ be the corresponding vector of player payoffs.

If the above problem is not feasible, then $c(a, \mathbf{s}, x) = -\infty$, $w^* = \emptyset$ and $v(a, \mathbf{s}, x)$ is a vector of $-\infty$.

(b) Choose action profile that maximizes the weighted value

$$\widetilde{a}(\mathbf{s}, x) \in argmax_a \{c(a, \mathbf{s}, x)\}$$

Let $\widetilde{v}(\mathbf{s}, x) = v(\widetilde{a}(\mathbf{s}, x), \mathbf{s}, x)$ be the corresponding vector of payoffs.

4. Update R and Z:

(a) The new set of normals is

$$R_x^+ = S_x$$

(b) The new set of boundary points is

$$Z_x^+ = \{\widetilde{v}(\mathbf{s}, x) \mid \mathbf{s} \in S_x\}$$

R^+ and S^+ together define the outer approximation to $B(\mathcal{W})$, $\mathcal{W}^+$. For each x

$$\mathcal{W}_x^+ = \cap_{j=1}^{l_x} \{\mathbf{b} \in \Re^N | \ \mathbf{r}_x^j \cdot \mathbf{b} \leq \mathbf{r}_x^j \cdot \mathbf{z}_x^j\}, \quad \mathbf{r}_x \in R_x^+, \ \mathbf{z}_x \in Z_x^+$$

(c) Check for convergence:

Stop if the Hausdorff distance between $\mathcal{W}_x^+$ and $\mathcal{W}_x$ is less than $\epsilon > 0$ for all $x \in X$; else set $\mathcal{W}_x = \mathcal{W}_x^+$, $R_x = R_x^+$, $Z_x = Z_x^+$ and go back to Step 2.

------------------------------------ ---------

Before we proceed with the details of the inner approximation scheme, we provide some definitions. We use the following characterization of a convex hull in our inner and strategy algorithms.

If Z is a finite set of L points in $\Re^N$: $Z = \{z_i | i = 1, ...L\}$, then the *convex hull of* Z is defined as:

$$co(Z) = \left\{ b \in \Re^N | \ \exists \lambda \geq 0, \sum_{i=1}^{L} \lambda_i = 1, b = \sum_{i=1}^{L} \lambda_i z_i \right\}.$$

In the algorithm descriptions below, we will economize on the notation by using this definition of $co(Z)$. For example, a maximization problem of the form

$$\max_{w \in \Re^N} \cdots \tag{4}$$

$$(i) w \in co(Z)$$

$$\cdot \cdot\cdot$$

will mean

$$\begin{aligned} \max_{w \in \Re^N,\ \lambda \in \Re^{m_{g(a,x)}}} \quad & \cdots \\ & (i) \quad \sum_{i=1}^{m_{g(a,x)}} \lambda_i = 1, \lambda_i \geq 0, i = 1, ..., m_{g(a,x)} \\ & (ii) \ w = \sum_{i=1}^{m_{g(a,x)}} \lambda_i z_{i,g(a,x)}, \\ & \cdots \end{aligned} \tag{5}$$

This notational shorthand will make it easier to see the essential details of the algorithms

----

Inner Approximation Algorithm for Supergames with State Variables

1. Input: Sets of points:

$$Z_x = \{\mathbf{z}_x^1, ..., \mathbf{z}_x^{m_x}\}$$

which represents the correspondence $\mathcal{W}$, where for each x, $\mathcal{W}_x = co(Z_x)$.

2. Choose search subgradients

$$S_x = \{\mathbf{s}_x^1, ..., \mathbf{s}_x^{l_x}\}$$

3. Compute new Z^+ that represents an inner approximation of $B(\mathcal{W})$. For each $x \in X, s \in S_x$ do:

   (a) For each action profile $a \in A$ do

$$\begin{aligned} c(a, \mathbf{s}, x) = \max_{w \in \Re^N} \; & \mathbf{s} \cdot [(1-\delta)\Pi(a,x) + \delta w] \;\; s.t. \\ & (i) \; w \in co(Z_{g(a,x)}) \\ & (ii) \forall i \in N, \; \forall \widetilde{a} \in A_i, \;\; IC\,(i, a, \tilde{a}, w) \geq 0 \end{aligned} \tag{6}$$

   Let $w^*(a, \mathbf{s}, x)$ be the *argmax* of (**??**) and define

$$v(a, \mathbf{s}, x) = (1-\delta)\Pi(a,x) + \delta w^*(a, \mathbf{s}, x)$$

   to be the corresponding vector of player payoffs. If (**??**) is not feasible, then $c(a, \mathbf{s}, x) = -\infty$, $w^* = \emptyset$ and $v(a, \mathbf{s}, x)$ is a vector of $-\infty$.

(b) Determine $\{\Omega_x^a\}$:[5]

$$\Omega_x^a = \{v(a, \mathbf{s}, x) \mid v(a, \mathbf{s}, x) > -\infty,\ \mathbf{s} \in S_x\}$$

(c) Choose action profile that maximizes the weighted value:

$$\widetilde{a}(\mathbf{s}, x) \in argmax_a\{c(a, \mathbf{s}, x)\}.$$

Let $\widetilde{v}(\mathbf{s}, x) = v(\widetilde{a}(\mathbf{s}, x), \mathbf{s}, x)$ be the corresponding vector of payoffs.

4. Update correspondence Z: For each $x \in X$

$$Z_x^+ = Z_x \cup \{\tilde{v}(s, x) |\ s \in S\}.$$

The set of Z_x^+ represents an inner approximation of $B(\mathcal{W})$ and for each x, $\mathcal{W}_x^+ = co(Z_x^+)$.

5. Check for convergence:

Stop if the Hausdorff distance between $\mathcal{W}_x^+$ and $\mathcal{W}_x$ is less than $\epsilon > 0$, for all $x \in X$; else set $\mathcal{W}_x = \mathcal{W}_x^+$, $Z_x = Z_x^+$ and go back to Step 2.

----

[5] Until one is close to convergence, one does not need to record these sets. However, these sets are essential for the output from this algorithm which will then be used to compute equilibrium paths.

Finally, we provide a two-step algorithm to find equilibrium strategies. In the inner approximation, we have computed the equilibrium values that can be supported by each specific action profile in each state, denoted by Ω_x^a. An equilibrium value v_x is supported by a convex set $\Lambda_x = \{(\lambda_a, v_x^a)|a \in A\}$ where λ_a is a probability vector over action profiles and v_x^a is a vector of values such that $v_x^a \in co(Z_x^a)$ and $v_x = \sum_a \lambda_a v_x^a$. A pure strategy implementation of the payoff v_x is any such $\{(\lambda_a, v_x^a)|a \in A\}$ where $\lambda_a = 1$ for some a. In that case, $w(a,x) = \delta^{-1}[(v_x - (1-\delta)\pi(a,x)]$ is in $V_{g(a,x)}^I$. However, there may not be an element of Λ_x that represents a pure strategy. In that case, the only way to support a payoff v_x is with some non-degenerate mixture of v_x^a. In our algorithm, we first check to see if there is some pure strategy equilibria continuation, otherwise one has pick some element of Λ_x.

Factorization/ Equilibrium Strategy Algorithm

1. Input: $v_x \in V_x^I = co(Z_x)$

2. Choose equilibrium actions and continuation values that support v_x.

   - Check if v_x is pure-strategy implementable. If $v_x \in co(\Omega_x^a)$ for some a, then $w(a,x) = \delta^{-1}[(v_x - (1-\delta)\pi(a,x)] \in V_{g(a,x)}^I$ and therefore a is pure strategy action profile that supports v_x. If there are multiple such a, choose an a according to some criterion.

   - If there is no a s.t. $v_x \in co(\Omega_x^a)$, then pick some element (according to some criterion) of Λ_x s.t. $v_x = \sum_a \lambda_a v_x^a$.

It is important to note that any v_x can be implemented by more than one pure strategy action profile, in this case, the algorithm chooses an a according to some criterion. Examples of criteria that can be used in the algorithm include maximizing total current profits, or minimizing current profits or symmetry in current payoffs. It is also possible to make a random choice from Λ_x. A sample equilibrium path can be constructed by using the continuation value $w(a, x)$ as the new value input in step 1 and reapplying the algorithm.

------------------------------------ ---------

5 Cournot competition with capacity constraints

This section provides an example of a dynamic game with state variables. Specifically, it extends a standard textbook Cournot oligopoly model to a dynamic setting with capital investment. In this environment, firms participate in a market for a finished good. Firm i has sales of $q_i \in Q_i$, and unit cost c_i. The firm's technology is augmented by the incorporation of a capacity constraint which depends upon the total number of machines that a firm has. If firm i has $k_i \in K_i$ machines then the output it can produce in the current period is bounded above by k_i. The cost of maintaining a machine for a period is given by MC. Each firm can alter the level of its capital stock by purchasing and installing a new machine at a cost of FC. To abstract from entry/exit and financing issues, we further assume that firms have an endowment of one machine, for which they need not pay the maintenance cost. If there exists a market for the resale of capital, the investment that the firms undertake is described as reversible. Otherwise, it is described as irreversible. If investment is reversible, then a firm can also divest, by selling an existing machine at a price of SP.

5.1 Reversible Investment

In this section, we assume that investment in capital is reversible: firms can sell their extra machines at a price of SP. Let $C(k_i, k_i')$ denote the cost of maintaining k_i machines this period and altering the number of machines to k_i' in the next period. Then $C(\cdot)$ takes the following form:

$$C(k_i, k_i') = \begin{cases} MC * (k_i - 1) + FC * (k_i' - k_i) & \text{if } k_i' \geq k_i \\ MC * (k_i - 1) - SP * (k_i - k_i') & \text{if } k_i' \leq k_i \end{cases}$$

Firm i's current profits are given by:

$$\Pi_i(\mathbf{q}, k_i, k_i') = q_i(p(\mathbf{q}) - c) - C(k_i, k_i'),$$

where the market price is given by a linear demand curve:

$$p(\mathbf{q}) \;=\; \max\,\{a - \sum_i^N b_i q_i, 0\}. \tag{7}$$

The stage game action spaces for this problem are sets of outputs and capital stocks (i.e. stocks of machines), for each firm. The state space is the set of feasible capital stocks for both firms. Thus, $A_i = Q_i \times K_i$ and $X = K_1 \times \cdots \times K_N$. Firm strategies are collections of functions that map from histories of outputs and capital stocks into current output and capital choices. As usual, each firm attempts to maximize its average discounted profits.

For the next few numerical examples, we assume there are 2 firms and consider a finite action version of the game. To this end we discretize the action space over q_i and k_i. Firms with full capacity can choose between 16 different quantities from the interval $[0, \bar{Q}]$. If they are in state 1, where they only possess their endowment capacity, they can at most produce $\bar{Q}/2$. If they purchase an additional machine, they can produce up to $\bar{Q}$. Therefore firms can choose to be in two different states, corresponding to their endowment capacity or 1 extra machine respectively. With all possible combinations of both firms' capacity, there are four different states of the capacity game in total. The fixed parameters are : $\{a = 6,\ b = 0.6, \beta = 0.8,\ FC = 2.5,\ MC = 1.5,\ \bar{Q} = 6\}$.

5.1.1 Example 1: Alternating Leadership

In this particular reversible investment example, the scrap value of a machine, SP, is set to 1.5. The per unit cost of production $c = 0.6$. In Tables 1 and 2, we display the solution to the monopolist's problem and the symmetric Nash collusion payoffs, capacity and production. With this set of parameters, a monopolist would prefer to have 3 machines, produce at the maximum capacity and receive an average discounted payoff of 22.80. Since machines are indivisible, the symmetric Nash collusion payoffs are lower than the shared monopoly payoffs. The symmetric Nash collusion choice of capacity is 1 machines per firm, 2 for the industry.

Table 1: Monopoly

k	q	k'	$V(k)$
1	3.0	3	20.24
2	6.0	3	22.06
3	9.0	3	22.80
4	9.0	3	22.80

Table 2: Symmetric Nash Collusion

k	q	k'	$V(k)$
1	3.0	1	10.80
2	4.5	1	11.07

To compute equilibrium correspondence that includes all equilibrium values, we apply our outer and inner approximation schemes using 48 different hyperplanes. Figure 1 displays the monotonic property of the B^I operator. Repeated application of the B^I operator produces a sequence of expanding value sets that contains the value set from the previous iteration (Figure 1 displays the state (1,1) only.). After 50 iterations, the resulting inner approximation is virtually the same as the outer approximation, with a slight difference in the northeast quadrant of the set. Hence the error in our approximation is very small. Although Figure 1 shows the results from the state (1,1), the results are similar for the other three capacity states.

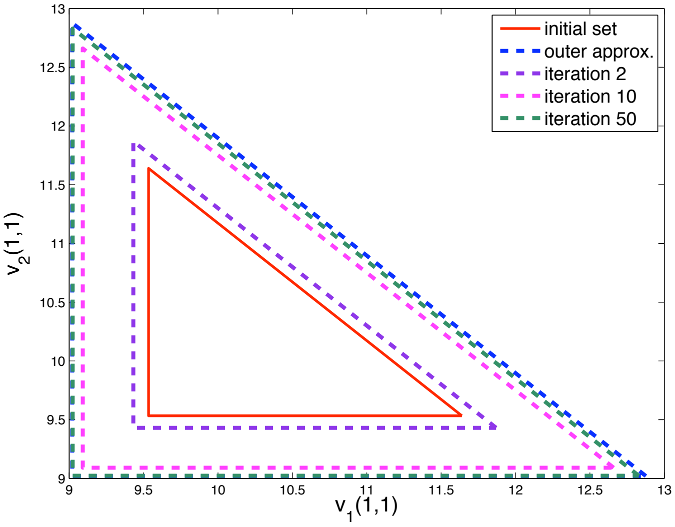


Figure 1: Reversible investment: Monotone operator and convergence

In Figure 2, we show the inner approximation for all states. The lowest symmetric payoffs are obtained in state (1,1) and the highest symmetric payoffs are obtained in state (2,2). The highest payoff for an individual firm is the asymmetric states, when the firm has more capacity then it's competitor. In Figure (3), the same four sets of equilibrium values are plotted, along with their outer approximations, and hence their error bounds. Additionally the values corresponding to the perfect shared collusion steady states are marked by a triangle

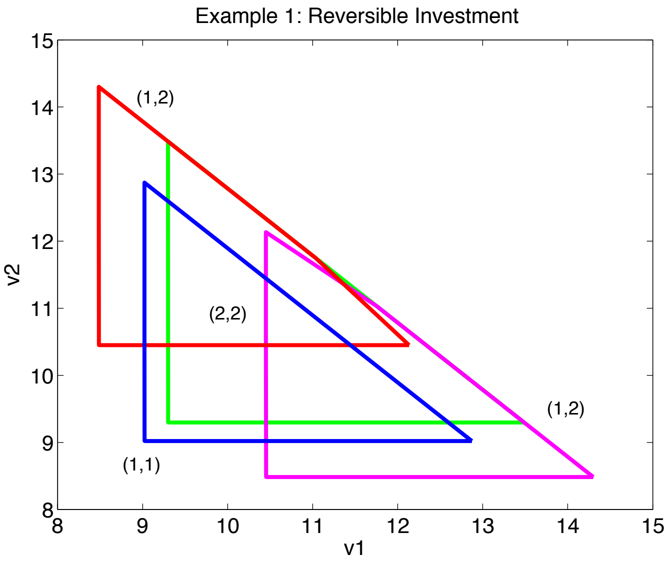


Figure 2: Reversible investment: Inner approximation

for the symmetric states (1,1) and (2,2). Note that in both of these states, the firms can achieve a higher payoff in the dynamic game then they do in the repetition of symmetric collusion. Figure (4) shows a sample equilibrium path that delivers payoff marked 1 on the pareto frontier of the set in state (1,1). At point 1, firm 1 achieves a higher payoff than firm 2. This is supported by continuation values that come from state (2,1), which themselves are supported by continuation equilibrium play that allows firm 1 to lead with a large capacity and higher output, but with continuation values that are decreasing over time after the initial jump to state (2,1). In other words, Firm 1 is able to achieve higher payoff and market leadership early on, but this equilibrium play can only be supported by allowing Firm 2 to eventually earn higher profits in the future even though the firms continue to stay in state (2,1) and hence allowing firm 1 to be the larger firm in terms of capacity. Table 3

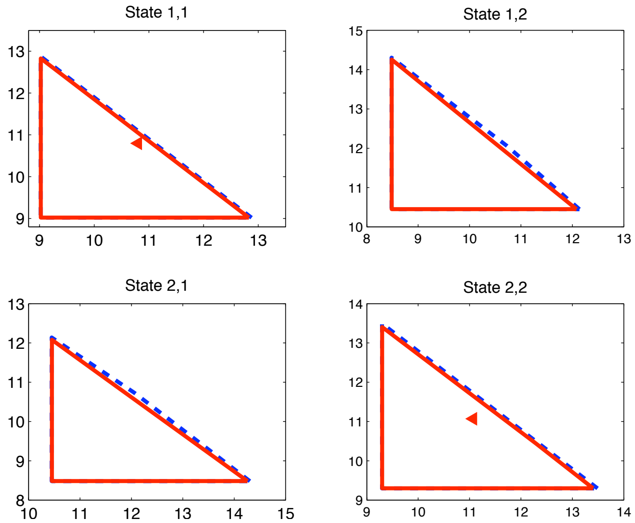


Figure 3: Reversible Investment: Inner and outer approximation

shows the equilibrium values, continuation payoffs and capacity and production choices at each of those eight nodes. After reaching the pair of equilibrium values (11.2342 , 11.3407) in state (2,1), the continuation of this equilibrium allows Firm 1 to become the market leader and Firm 2 to divest by moving to the state (1,2). Figure (5) shows the continuation states associated with the payoffs that lie on the boundary of state (1,1), and it's clear that one can also construct an equilibrium path that allow Firm 2 to be larger firm, with increasing payoffs to Firm 1. The symmetric point on the boundary of (1,1), on the other hand, can only be supported with the help of public randomization, specifically a randomization between continuations that come from states (2,1) and (1,2).

Analysis of such capacity games that restrict capacity choices to the initial period or restrict equilibrium play to be symmetric would simply miss the strategic interaction and higher payoffs in the dynamics game.[insert footnote with references of such analysis] In this

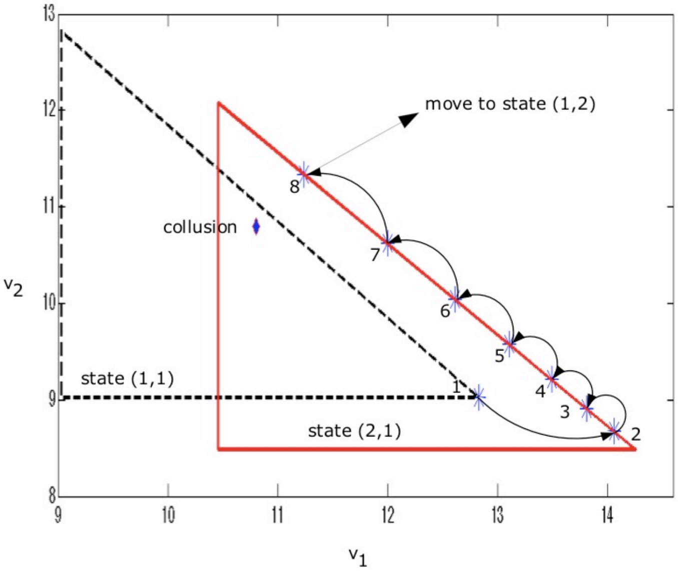


Figure 4: Reversible investment $c = 0.6$: Equilibrium path

Table 3: Equilibrium Path

$Node$	v_1	v_2	k_1	k_2	q_1	q_2
1	12.8289	9.0232	1	1	3.0	3.0
2	14.0571	8.6750	2	1	6.0	3.0
3	13.8064	8.9118	2	1	6.0	3.0
4	13.4930	9.2078	2	1	6.0	3.0
5	13.1012	9.5777	2	1	6.0	3.0
6	12.6115	10.0401	2	1	6.0	3.0
7	11.9994	10.6181	2	1	6.0	3.0
8	11.2342	11.3407	2	1	6.0	3.0

setting, a monopolist would choose to have three machines in steady state, produce 9 units of output with a profit of 22.80. Since machines are indivisible, it's not possible for the two firms to reach shared monopoly payoffs in symmetric states. Firms are able to achieve a higher payoff than symmetric collusion by alternating market power. This can be readily seen in Figure 5, where for select equilibrium values in state (1,1), the state that continuation values come from are displayed. The Pareto frontier in this symmetric state is supported by continuation values from the Pareto frontier of the asymmetric states, which themselves are supported by continuation values from the same or the alternative asymmetric states.

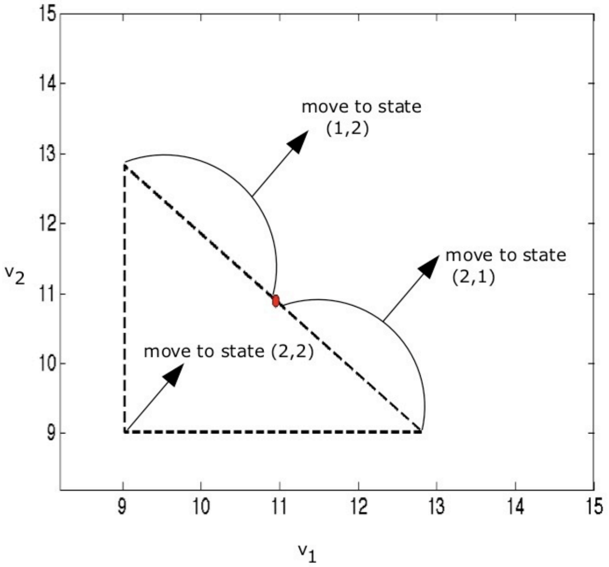


Figure 5: Reversible investment $c = 0.6$: Equilibrium path

Figure (6) displays the continuation values associated with the worst equilibrium payoffs in states $(1, 1)$ and $(2, 2)$. The worst equilibrium outcomes are very severe. In state $(1, 1)$, the worst equilibrium payoff is supported by producing at full capacity in this period, investing in new capacity, and producing at the higher capacity thereafter. In other words, the worst equilibrium payoff involves firms saturating the market, building further capacity and saturating the market even more in each and every period. The worst equilibrium payoff in state (2,2) is an absorbing state and it's associated with a stationary equilibrium.

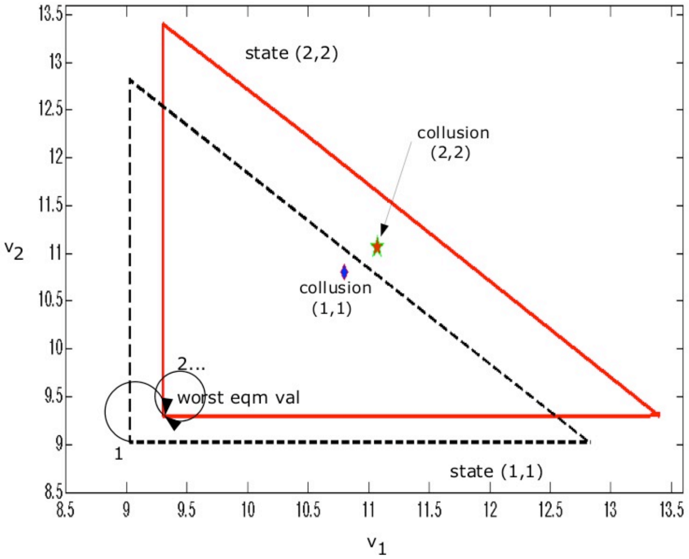


Figure 6: Uncooperative behavior $c = 0.6$: Equilibrium path

When firms have higher per unit cost of production the worst equilibrium associated with $c = 0.6$ becomes to costly to enforce. In the case of $c = 0.9$, after a short punishment phase, firms move to a more cooperative equilibrium.

Continuation in 2,1 or 1,2. point 18 is not exactly on the boundry. A little inside.

Table 4: Equilibrium Path

$Node$	v_1	v_2	k_1	k_2	q_1	q_2
1	7.47280025107915	7.47280025107915	1	1	8	8
2	7.57200031384894	7.57200031384894	2	2	16	16
3	7.59000039231118	7.59000039231118	2	2	16	16
4	7.61250049038897	7.61250049038897	2	2	16	16
5	7.64062561298621	7.64062561298621	2	2	16	16
6	7.67578201623276	7.67578201623276	2	2	16	16
7	7.71972752029095	7.71972752029095	2	2	16	16
8	7.77465940036369	7.77465940036369	2	2	16	16
9	7.84332425045461	7.84332425045461	2	2	16	16
10	7.92915531306827	7.92915531306827	2	2	16	16
11	8.03644414133533	8.03644414133533	2	2	16	16
12	8.17055517666916	8.17055517666916	2	2	16	16
13	8.33819397083645	8.33819397083645	2	2	16	16
14	8.54774246354557	8.54774246354557	2	2	16	16
15	8.80967807943196	8.80967807943196	2	2	16	16
16	9.13709759928994	9.13709759928994	2	2	16	16
17	9.54754361279848	9.54754361279842	2	2	16	16
18	10.0594295159981	10.0594295159980	2	1		
			1	2		

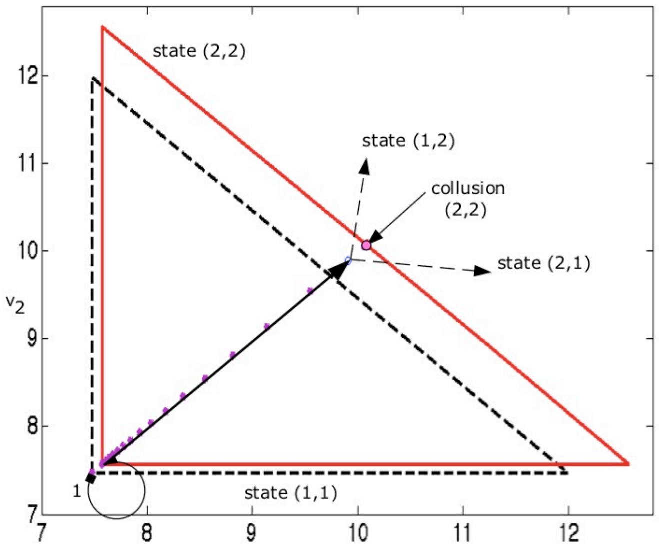
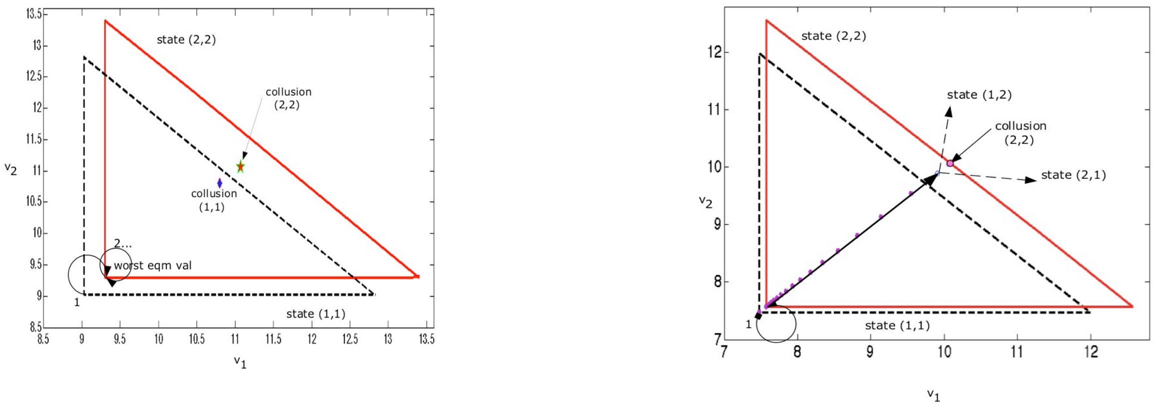


Figure 7: Uncooperative behavior $c = 0.9$: Equilibrium path



(a) Uncooperative behavior: $c = 0.6$

(b) Striving for cooperation $c = 0.9$

Figure 8: Worst equilibrium and its Continuation