Multigrid methods for two-player zero-sum stochastic games

Marianne Akian Sylvie Detournay

July 11, 2011

Abstract

We develop a fast numerical algorithm for large scale zero-sum stochastic games with perfect information, which combines policy iteration and algebraic multigrid methods. This algorithm can be applied either to a true finite state space zero-sum two player game or to the discretization of an Isaacs equation. We present numerical tests on discretizations of Isaacs equations or variational inequalities. We also develop a full multi-level policy iteration, similar to FMG, which allows one to improve substantially the computation time for solving some variational inequalities.

1 Introduction

Consider a game on a finite state space $\mathcal{X}$ with discounted infinite horizon payoff. Each pair of strategies of the two players determines a Markov chain on $\mathcal{X}$. The value v of the game satisfies the following dynamic programing equation [42]:

$$v(x) = \max_{a\in\mathcal{A}(x)} \left(\min_{b\in\mathcal{B}(x,a)} \left(\sum_{y\in\mathcal{X}} \mu\, p(y\,|\,x,a,b)\, v(y) + r(x,a,b) \right) \right) \qquad \forall x \in \mathcal{X}. \tag{1}$$

where $v(x)$ is the value of the game starting from the state $x \in \mathcal{X}$, $r(x,a,b)$ is the paiment made by the second player to the first player when the Markov chain is in state x, if the players choose the actions a and b respectively at the current time, $p(y\,|\,x,a,b)$ is the transition probability of the Markov Chain from state x to state y, given the actions a and b at the current time, and $\mu < 1$ is the discount factor.

Equation (1) may also be obtained after a suitable discretization of Hamilton-Jacobi-Bellman or Isaacs partial differential equations :

$$\max_{a\in\mathcal{A}} \left(\min_{b\in\mathcal{B}} \left(\sum_{ij} \mathrm{a}_{ij}(a,b,x) \frac{\partial^2 v(x)}{\partial x_i \partial x_j} + \sum_j \mathrm{b}_j(a,b,x) \frac{\partial v(x)}{\partial x_j} - \lambda v(x) + r(x,a,b) \right) \right) = 0 \text{ for } x \text{ in } \mathcal{X} \tag{2}$$

which are dynamic programing equations of zero-sum stochastic differential games. This is the case for instance of Markov chain discretizations [30, 31], monotone discretizations [7]. Other possible discretizations include full discretizations of semi-Lagrangian type [5], and max-plus finite element method [3] for deterministic games or control roblems.

One can solve classically (1) by applying the fixed point method which is known as the value iteration algorithm [8]. The iterations are cheap but their convergence slows considerably as μ approaches one, which holds when the discretization step h for (2) is small, since then $\mu = 1 - O(\lambda h^2)$. Another approach consists in the so called policy iteration algorithm, initially introduced by Howard [29] for one player games. For two players games, Hoffman and Karp [26] were the first to give the algorithm in the special mean-payoff case when the Markov chains associated to any policies of the two players are all irreductible. In 1967, Dernado, in his study of contracting maps [17], proposed a generalization of Howard's algorithm with an approximate solution and convergence results. He applied his method to Shapley's zero-sum two player stochastic terminating games. In the nighties (95), the policy iteration method was clearly established for two player zero-sum discounted stochastic games in Puri thesis [39]. In 2006, Cochet-Terrasson and Gaubert [14] gave an algorithm for the general ergodic case.

Recall that a (pure feedback) policy $\bar{\alpha}$ for the first player maps any $x \in \mathcal{X}$ to an action $\bar{\alpha}(x) \in \mathcal{A}(x)$. Then, the policy algorithm consists in improving at each step a policy for the first player, by computing the value of the game for this current policy, then finding the optimal policy for the corresponding value function, that is the policy optimizing the expression inside the "max" in (1). Computing the above value functions is performed itself using the policy iteration algorithm for a one-player game. It is well known that, by monotonicity of dynamic programming operators, the sequence of value functions of the (internal) policy iteration for a one-player "min" game is non increasing (see for instance [40, 32, 9]). From the same property, the sequence of value functions of the policy iteration for two-player game (1) is non decreasing (see [39, 14]). In both cases, the iteration stops after a finite time when the sets of actions are finite (see again [9, 32, 39]).

However, basic policy iteration can be exponential in time. Recall that the number of iterations is bounded by the number of possible strategies, which is exponential in $|\mathcal{X}|$. Friedmann has shown [25] that a strategy improvement algorithm requires an exponential number of iterations for some particular parity games, this result can be extended to other types of games, and to undiscounted or ergodic (mean-payoff) stochastic control problems (one-player games) as shown by Fearnley [20, 21]. However, in all cases, policy algorithm converges faster than the value iteration algorithm and in practice it ends in few steps (see for instance large scale random examples for deterministic games shown in [18]). In addition, under regularity assumptions, the policy iteration algorithm for a one player game with infinite action spaces is equivalent to Newton's method, thus can have a super-linear convergence in the neighborhood of the solution, see [40, 2, 4] for order $p > 0$ superlinear convergence under some regularity and

strong convexity assumptions, and [10] for superlinear convergence under more general regularity assumptions.

In each internal iteration of the policy iterations, one needs to solve a linear system of equations, the dimension of which is equal to the cardinality $|\mathcal{X}|$ of the state space $\mathcal{X}$. When (1) is coming from the discretization of the Isaacs partial differential equation (2), these linear systems correspond to discretizations of linear elliptic equations, hence may be solved in the best case in a time in the order of $|\mathcal{X}|$, by using multigrid methods. In general, using the nice monotonicity properties of the underlying linear systems, one may expect the same complexity when solving them by an algebraic multigrid method.

In the present paper, we introduce an algorithm, named AMGπ, which is the combination of policy iterations with the algebraic multigrid method (AMG) introduced by Brandt, McCormick and Ruge [12, 13], see also Ruge and Stüben [41]. This algorithm can be applied either to a true finite state space zero-sum two player game or to the discretization of an Isaacs equation.

Such an association of multigrid methods with policy iteration has already been used and studied in the case of one player games, that is discounted stochastic control problems (see Hoppe [27, 28] and Akian [1, 2] for Hamilton-Jacobi-Bellman equations or variational inequalities, Ziv and Shimkin [38] for AMG with learning methods). However, it is new in the case of two player games.

We have implemented this algorithm (in C) and shall present numerical tests on discretizations of Isaacs or Hamilton-Jacobi-Bellman equations or variational inequalities, while comparing AMGπ with policy iteration with direct solvers.

In some reachability (or pursuit-evasion) games, the number of policy iterations is typically in the order of the diameter of the graph of the controlled Markov chain, which is in the order of $1/h$, where h is the discretization step. However, as for Newton's algorithm, convergence can be improved by starting the policy iteration with a good initial guess, close to the solution. In this way, we developed a full multi-level policy iteration, similar to FMG. It consists in solving the problem at each grid level by performing policy iterations (combined with algebraic multigrid method) until a convergence criterion is verified, then to interpolate the strategies and value to the next level, in order to initialize the policy iterations of the next level, until the finest level is attained. For one-player discounted games with infinite number of actions and under regularity assumptions, one can show [2, 1] that this full multi-level policy iteration has a computing time in the order of $|\mathcal{X}|$. Below, we give numerical examples on variational inequalities for two player games, the computation time of which is improved substantially using the full multi-level policy iteration instead of AMGπ.

The paper is organized as follow. The three following sections are some recalls about basic definitions on the subject. In Section 2, we introduce the definition of a two player zero-sum stochastic game with finite state space and the corresponding dynamic programming equation. Section 3 is about two player zero-sum stochastic differential games, we recall here the definition of the Isaacs equation, the variational inequalities and the discretization scheme that we use. Section 4 is devoted to the numerical background needed to solve the dynamic

programming equation, including the policy iteration algorithm and the algebraic multigrid method. Section 5 describes our algorithms AMGπ and full multi-level AMGπ. We present in Section 6 some numerical tests on Isaacs equations and variational inequalities. Last section gives concluding remarks.

2 Two player zero-sum stochastic games: the discrete case

The class of two player zero-sum stochastic game was first introduced by Shapley in the early fifties [42]. We recall in this section the definition of these games in the case of finite state space and discrete time (for more details see [42, 22, 43]).

We consider a finite state space $\mathcal{X} = \{1, \ldots, n\}$. A stochastic process $(\xi_k)_{k\geq 0}$ on $\mathcal{X}$ gives the state of the game at each point time k, called stage. At each of these stages, both players have the possibility to influence the course of the game.

The stochastic game $\Gamma(x_0)$ starting from $x_0 \in \mathcal{X}$ is played in stages as follows. The initial state x_0 is given and known by the players. The player who plays first, says MAX, chooses an action a_0 in a set of possible actions $\mathcal{A}(x_0)$. Then the second player, called MIN chooses an action b_0 in a set of possible actions $\mathcal{B}(x_0, a_0)$. The actions of both players and the current state determine the payment $r(x_0, a_0, b_0)$ made by MIN to MAX and the probability distribution $p(\cdot|x_0, a_0, b_0)$ of the new state x_1. Then the game continue in the same way with state x_1 and so on.

At a stage k, each player chooses an action knowing the history defined by $h_k = (x_0, a_0, b_0, \cdots, x_{k-1}, a_{k-1}, b_{k-1}, x_k)$ for MAX and (h_k, a_k) for MIN. We call a strategy or policy for a player, a rule which tells him the action to choose in any situation. They are several classes of strategies. A behavior or randomized strategy for MAX (resp. MIN) is represented by a sequence $\bar{\alpha} := (\bar{\alpha}_0, \bar{\alpha}_1, \cdots)$ (resp. $\bar{\beta} := (\bar{\beta}_0, \bar{\beta}_1, \cdots)$) where $\bar{\alpha}_k$ (resp. $\bar{\beta}_k$) is a map which to the history h_k (resp. (h_k, a_k)) at stage k associates a probability distribution on the possible actions space $\mathcal{A}(x_k)$ (resp. $\mathcal{B}(x_k, a_k)$). A Markovian (or feedback) strategy is a strategy which only depends on the information of the current stage k: $\bar{\alpha}_k$ (resp. $\bar{\beta}_k$) depends only on x_k (resp. (x_k, a_k)), then $\bar{\alpha}_k(h_k)$ (resp. $\bar{\alpha}_k(h_k, a_k)$) will be denoted $\bar{\alpha}_k(x_k)$ (resp. $\bar{\beta}_k(x_k, a_k)$). It is said stationary if it is independent of k, then $\bar{\alpha}_k$ is also denoted by $\bar{\alpha}$ and $\bar{\beta}_k$ by $\bar{\beta}$. A strategy of any type is said pure if for any stage k, the values of $\bar{\alpha}_k$ (resp. $\bar{\beta}_k$) are Dirac probability measures at a certain action in $\mathcal{A}(x_k)$ (resp. $\mathcal{B}(x_k, a_k)$) then we shall denote also by $\bar{\alpha}_k$ (resp. $\bar{\beta}_k$) the map which to the history assigns the only possible action in $\mathcal{A}(x_k)$ (resp. $\mathcal{B}(x_k, a_k)$).

A strategy $(\bar{\alpha}_k)_{k\geq 0}$ (resp. $\left(\bar{\beta}_k\right)_{k\geq 0}$) together with an initial state determines stochastic processes, $(\zeta_k)_{k\geq 0}$ for the actions of MAX, $(\eta_k)_{k\geq 0}$ for the actions of

MIN and $(\xi_k)_{k\geq 0}$ for the states with

$$P(\xi_{k+1} = y \,|\, \iota_k = h, \zeta_k = a, \eta_k = b) \;=\; p(y \,|\, x, a, b) \tag{3a}$$

$$P(\zeta_k \in A \,|\, \iota_k = h) \;=\; \bar{\alpha}_k(x)(A) \tag{3b}$$

$$P(\eta_k \in B \,|\, \iota_k = h, \zeta_k = a) \;=\; \bar{\beta}_k(x, a)(B) \tag{3c}$$

where the process ι_k is defined by $(\xi_0, \zeta_0, \eta_0, \ldots, \xi_k)$, h is the history vector at time k: $(x_0, a_0, b_0, \ldots, x_{k-1}, a_{k-1}, b_{k-1}, x)$ and for all measurable sets A (resp. B) in $\mathcal{A}(x)$ ($\mathcal{B}(x, a)$ resp). For instance, for each pair of Markovian stationary strategies $(\bar{\alpha}, \bar{\beta})$ of the two players, the state process $(\xi_k)_{k\geq 0}$ is a Markov chain on $\mathcal{X}$ with transition probability

$$P(\xi_{k+1} = y \,|\, \xi_k = x) \;=\; p^{\bar{\alpha}(x), \bar{\beta}(x,\cdot)}(y \,|\, x)$$

where

$$p^{\alpha,\beta}(y \,|\, x) \;=\; \int_{a\in\mathcal{A}(x)} \int_{b\in\mathcal{B}(x,a)} p(y \,|\, x, a, b)\; d(\beta(a))(b)\, d\alpha(a)$$

with $x, y \in \mathcal{X}$ and $(\xi_k, \zeta_k, \eta_k)_{k\geq 0}$ is a Markov Chain on $\{(x, a, b) \,|\, x \in \mathcal{X}, a \in \mathcal{A}(x), b \in \mathcal{B}(x, a)\}$.

The payoff of the game $\Gamma(x_0)$ starting from $x_0 \in \mathcal{X}$ is the expected sum of all rewards that MAX wants to maximize and MIN to minimize. In this paper we consider discounted games Γ_μ with discount factor $0 < \mu < 1$. When the strategies $\bar{\alpha}$ for MAX and $\bar{\beta}$ for MIN are fixed, the payoff of the game $\Gamma_\mu(x_0, \bar{\alpha}, \bar{\beta})$ starting from x_0 is

$$J(x_0, \bar{\alpha}, \bar{\beta}) \;=\; \mathbb{E}_{x_0}^{\bar{\alpha},\bar{\beta}} \left[\sum_{k=0}^{\infty} \mu^k r(\xi_k, \zeta_k, \eta_k) \right],$$

where $\mathbb{E}_{x_0}^{\bar{\alpha},\bar{\beta}}$ denotes the expectation for the probability law determined by (3). A discounted game can be seen equivalently as a game which has, in each stage, a stopping probability equal to $1 - \mu$, independent of the actions taken by both players. The value of the game starting from $x_0 \in \mathcal{X}$, $\Gamma_\mu(x_0)$, is then given by

$$v(x_0) \;=\; \sup_{\bar{\alpha}} \inf_{\bar{\beta}} J(x_0, \bar{\alpha}, \bar{\beta}), \tag{4}$$

where the supremum is taken among all strategies $\bar{\alpha}$ for MAX and the infimum is taken over all strategies $\bar{\beta}$ for MIN. Note that a non terminating game without any discount factor (or $\mu = 1$) is called ergodic.

We are concerned in finding optimal strategies for both players and the value of the discounted game Γ_μ in each point. These are given by the dynamic programming equation [42] defined below.

Theorem 2.1 (Dynamic programming equations [42]). *Assume $\mathcal{A}(x)$ and $\mathcal{B}(x, a)$ are finite sets for all $x \in \mathcal{X}$, $a \in \mathcal{A}(x)$. Then, the value v of the stochastic game*

Γ_μ, *defined in* (4), *is the unique solution* $v : \mathcal{X} \to \mathbb{R}$ *of the following dynamic programing equation:*

$$v(x) \;=\; \underbrace{\max_{a\in\mathcal{A}(x)} \left(\min_{b\in\mathcal{B}(x,a)} \left(\sum_{y\in\mathcal{X}} \mu\, p(y \,|\, x,a,b)\, v(y) \;+\; r(x,a,b) \right) \right)}_{=F(v;x)} \qquad \forall x \in \mathcal{X}. \tag{5}$$

Moreover, optimal strategies are obtained for both players by taking pure Markovian strategies $\bar{\alpha}$ *for* MAX *and* $\bar{\beta}$ *for* MIN *such that* $\bar{\alpha}(x)$ *attains the maximum in* (5) *for each* x *in* $\mathcal{X}$ *and that* $\bar{\beta}(x,a)$ *attains the minimum for* $F(v;x,a)$ *at all* a *in* $\mathcal{A}(x)$ *and* x *in* $\mathcal{X}$ *where*

$$F(v;x,a) \;=\; \min_{b\in\mathcal{B}(x,a)} \left(\sum_{y\in\mathcal{X}} \mu\; P(y|x,a,b)\; v(y) \;+\; r(x,a,b) \right). \tag{6}$$

We denote by F the dynamic programing operator from $\mathbb{R}^{\mathcal{X}}$ to itself which maps v to the function

$$\begin{array}{rccc} F(v): & \mathcal{X} & \to & \mathbb{R} \\ & x & \to & F(v;x) \end{array} \tag{7}$$

where $F(v;x)$ is defined in (5). This operator is monotone and contracting with constant $1-\mu$ in the sup-norm, i.e. $\|F(v) - F(v')\|_\infty \le (1-\mu)\|v - v'\|_\infty$ for all $v, v' \in \mathbb{R}^{\mathcal{X}}$.

3 Two player zero-sum stochastic differential games: the continuous case

Now we give some examples of partial differential equations associated to stochastic differential games. After applying a suitable discretization, these partial differential equations yield dynamic programming equations of finite state space zero-sum two player games, which were described in the previous section.

3.1 Isaacs equations with Dirichlet boundary conditions.

Assume now that the state space is a regular open subset $\mathcal{X}$ of $\mathbb{R}^d$. Suppose a probability space Ω is given, as also a filtration $(\mathcal{F}_t)_{t\ge 0}$ on it. We consider games where the dynamics is governed by the following stochastic differential equation :

$$d\xi_t \;=\; \mathrm{b}(\xi_t, \zeta_t, \eta_t)\, dt + \sigma(\xi_t, \zeta_t, \eta_t)\, dW_t, \tag{8}$$

with initial state $\xi_0 = x \in \mathcal{X}$ and where W_t is a m-dimensional Wiener process on $(\Omega, (\mathcal{F}_t)_{t\ge 0})$, ζ_t and η_t are adapted stochastic processes taking values in

respectively closed subsets of $\mathbb{R}^p$ and $\mathbb{R}^q$, $\mathcal{A}$ and $\mathcal{B}$. We shall also denote by $\bar{\alpha}$ (resp. $\bar{\beta}$) the strategy of player MAX (resp. MIN) which determines the process ζ_t (resp. η_t).

We denote by τ the first exit time of ξ_t of $\mathcal{X}$. The discounted payoff of the game stopped at the boundary, with discount rate λ, is

$$J(x;\bar{\alpha},\bar{\beta}) \;=\; \mathbb{E}_x^{\bar{\alpha},\bar{\beta}}\left[\int_0^\tau e^{-\lambda t} r(\xi_t,\zeta_t,\eta_t)\,dt + e^{-\lambda\tau}\psi(\xi_\tau)\,|\,\xi_0 = x\right]. \qquad (9)$$

The value function of the differential stochastic game starting from x is defined by

$$v(x) \;=\; \sup_{\bar{\alpha}}\;\inf_{\bar{\beta}}\; J(x;\bar{\alpha},\bar{\beta})$$

where the supremum is taken among all strategies $\bar{\alpha}$ for MAX and the infimum is taken over all strategies $\bar{\beta}$ for MIN.

As previously, we are interested in finding the value function of the game and the corresponding optimal strategies. Denote by $L(v;(x,a,b))$ the second order elliptic partial differential operator :

$$L(v;(x,a,b)) \;:=\; \sum_{ij} \mathrm{a}_{ij}(a,b,x)\frac{\partial^2 v(x)}{\partial x_i \partial x_j} + \sum_j \mathrm{b}_j(a,b,x)\frac{\partial v(x)}{\partial x_j} - \lambda v(x),$$

with $(\mathrm{a}_{ij})_{i,j=1,..,d} = \dfrac{1}{2}\sigma\sigma^t$. The value of the game v is solution (under regularity assumptions on Ω and on the functions b, σ, r and ψ) of the dynamic programming equation, called Isaacs equation:

$$\begin{cases} \max\limits_{a\in\mathcal{A}}\left(\min\limits_{b\in\mathcal{B}}\left(L(v;(x,a,b)) + r(x,a,b)\right)\right) \;=\; 0 \;\text{ for } x \text{ in } \mathcal{X} \\ v \;=\; \psi \;\text{ on } \partial\mathcal{X}. \end{cases} \qquad (10)$$

This has been shown in the viscosity sense in [23]. See also [15] and references therein for uniqueness of the solution of (10). If v is a classical solution of (10) and $\bar{\alpha}$ and $\bar{\beta}$ are such that $\bar{\alpha}(x)$ realizes the maximum for MAX and $\bar{\beta}(x,a)$ the minimum for MIN in (10) for all x in $\mathcal{X}$ and $a \in \mathcal{A}(x)$, and such actions are unique, then the strategy $\zeta_t = \bar{\alpha}(\xi_t)$ (resp. $\eta_t = \bar{\beta}(\xi_t,\zeta_t)$) is a stationary Markovian pure strategy for (8) and (9).

Note that for a game with one player, i.e. for a stochastic control problem, equation (10) is the so-called Hamilton-Jacobi-Bellman equation.

3.2 Variational inequalities

We consider games in which some of the players have the choice of stopping the game at any moment, for detailed description see [24]. These are called optimal stopping time games. We assume here that MAX has this ability and that his possible actions consist only in choosing the stopping time κ of the game ($0 \le \kappa \le \tau$) or equivalently an element of the action space $\{1,2\}$ where

1 means that the game is continuing, 2 that the game stops, with $\zeta_s = 2$ and $\xi_s = \xi_t$ for $s \geq t$ when $\zeta_t = 2$ (i.e. $\mathrm{b}(x, 2, b) = 0$, $\sigma(x, 2, b) = 0\ \forall b, x$ in (8)). The second player MIN plays as previously and we consider the same model.

The discounted payoff is now:

$$J(x; \kappa, \bar{\beta}) \ = \ \mathbb{E}_x^{\kappa, \bar{\beta}} \left[\int_0^{\kappa} e^{-\lambda t} r(\xi_t, \eta_t)\, dt + e^{-\lambda \kappa} \psi(\xi_\kappa) \,|\, \xi_0 = x \right]$$

and the value function of the game starting from x is defined by

$$v(x) \ = \ \sup_{\kappa} \ \inf_{\bar{\beta}} \ J(x; \kappa, \bar{\beta})$$

where the supremum is taken among all stopping times $\kappa \leq \tau$ (or equivalently all strategies $\bar{\alpha}$ determining ζ) and the infimum is taken over all strategies $\bar{\beta}$ for MIN.

Then the value of the game is solution of the following variational inequality:

$$\begin{cases} \max \{ \underbrace{\min_{b \in \mathcal{B}} \left(L(v; (x, b)) + r(x, b) \right)}_{①}, \underbrace{\psi - v}_{②} \} \ = \ 0 \ \text{ for } x \text{ in } \mathcal{X} \\ v \ = \ \psi \ \text{ on } \partial \mathcal{X} \end{cases} \tag{11}$$

As for (10), if v is a classical solution of (11), if for all x in $\mathcal{X}$: $\bar{\alpha}(x)$ is equal to 1 or 2 if resp. ① or ② is maximum in (11) and if for all x in $\mathcal{X}$: $\bar{\beta}(x, 1)$ is the action $b \in \mathcal{B}$ which realize the minimum in ①, then an optimal stationary Markovian pure strategy is obtained by taking $\eta_t = \bar{\beta}(\xi_t, 1)$ and κ is equal to the first time when $\bar{\alpha}(\xi_t) = 2$. So this equation behaves as Equation (10) but where the first player has a discrete action space equal to $\{1, 2\}$, 1 meaning continue to play and 2 meaning stop the game. This variational inequality can be treated with the same methods as for (10).

In another usual way, the variational inequality of (11) can also be written:

$$\begin{cases} \min_{b \in \mathcal{B}} \left(L(v; (x, b)) + r(x, b) \right) \leq 0 \\ \psi - v \leq 0 \\ \text{equality must hold in one of the two inequalities above.} \end{cases} \tag{12}$$

3.3 Discretization

Several discretization methods may transform equations (10) or (11) into a dynamic programing equation of the form (5). This is the case when using Markov discrezation of the diffusions (10) as in [30, 31] and in general when using discrezation schemes that are monotone in the sense of [7]. One can obtain such discretizations by using the simple finite difference scheme below when there are no mixed derivative (that is $\sigma\sigma^T$ is a diagonal matrix). Under less restrictive assumptions on the coefficients, finite difference schemes with larger stencil also lead to monotone schemes [11, 35]. In the deterministic case (when $\sigma \equiv 0$), one can also use semi-Lagrangian scheme [5, 6] or max-plus finite

element method [3], both of them having the property of leading to a discrete equation of the form (5).

We suppose that $\mathcal{X}$ is the d-dimensional open unit cube. Let $h = 1/m$ ($m \in \mathbb{N}^*$) denote the finite difference step in each coordinate direction, e_i the unit vector in the i^{th} coordinate direction, and $x = (x_1, \ldots, x_d)$ a point of the uniform grid $\mathcal{X}_h = \mathcal{X} \cap (h\mathbb{Z})^m$. Equation (10) is discretized by replacing the first and second order derivatives of v by the following approximation, for $i, j = 1, \ldots, d$:

$$\frac{\partial v(x)}{\partial x_i} \sim \frac{v(x + he_i) - v(x - he_i)}{2h} \tag{13}$$

or

$$\frac{\partial v(x)}{\partial x_i} \sim \begin{cases} \dfrac{v(x + he_i) - v(x)}{h} & \text{when} \quad \mathrm{b}_i(x, a, b) \geq 0 \\ \dfrac{v(x) - v(x - he_i)}{h} & \text{when} \quad \mathrm{b}_i(x, a, b) < 0. \end{cases} \tag{14}$$

$$\frac{\partial^2 v}{\partial x_i^2}(x) \quad \sim \quad \frac{v(x + he_i) - 2v(x) + v(x - he_i)}{h^2}, \tag{15}$$

$$\tag{16}$$

Approximation (13) may be used when L is uniformly elliptic and h is small, whereas (14) has to be used when L is degenerate (see [30, 31]). For equations (10) and (11), these differences are computed in the entire grid $\mathcal{X}_h$, by prolonging v on the "boundary", $\partial\mathcal{X}_h := \partial\mathcal{X} \cap (h\mathbb{Z})^m$ using Dirichlet boundary condition:

$$v(x) \; = \; \psi(x) \qquad \forall\, x \in \partial\mathcal{X} \cap (h\mathbb{Z})^m.$$

We obtain a system of N_h non linear equations of N_h unknowns, the values of the function $v_h : x \in \mathcal{X}_h \mapsto v_h(x) \in \mathbb{R}$:

$$\max_{a \in \mathcal{A}} \big(\min_{b \in \mathcal{B}} \; (\, L_h(v_h; (x, a, b)) + r(x, a, b) \,) \big) \; = \; 0 \quad \forall\, x \in \mathcal{X}_h \;\; , \tag{17}$$

where $N_h = \sharp\mathcal{X}_h \sim 1/h^d$ and L_h is a function which to $v \in \mathbb{R}^{\mathcal{X}_h}$, $x \in \mathcal{X}_h$, $a \in \mathcal{A}$, $b \in \mathcal{B}$ associates the approximation of $L(v; (x, a, b))$.

When there are no mixed derivatives ($\mathrm{a}_{i,j}(x, a, b) = 0$ if $i \neq j$), the discretization is monotone in the sense of [7], then if (10) has a unique viscosity solution, the solution v_h of (17) converges uniformly to the solution v of (10) [7]. Moreover, multiplying Equation (17) by ch^2 with c small enough, it can be rewritten in the form (5), with a discount factor $\mu = \lambda ch^2$. A similar result holds for the discretization of (11) (by multiplying only the diffusion part by ch^2).

4 Background for numerical solution of discrete dynamic programming equations

4.1 Policy Iterations

As recalled in the introduction, one can classically solve the dynamic programming equations (5) by applying fixed point iterations to the map F defined at (7). This algorithm is known as the value iteration algorithm [8]. It has a linear convergence with a factor for the sup-norm less than or equal to the contraction factor of F, that is $1 - \mu$ (see Section 2). Since in the case of discretization $\mu = \lambda c h^2$, the convergence rate of the value iteration slows considerably as the discretization step goes to zero.

Here, we consider the *policy iteration* algorithm for two players discounted games as defined in Puri thesis [39]. (Recall that other references for policy iteration for one-player and two-player games are Howard [29], Hoffman and Karp [26], Dernado [17], Cochet-Terrasson and Gaubert [14].) It consists in nested iterations on pure feedback strategies (also called policies) and value functions. Recall that a pure feedback strategy $\bar{\alpha}$ for MAX maps any point x of $\mathcal{X}$ to a possible action $\bar{\alpha}(x)$ in $\mathcal{A}(x)$. If such a strategy $\bar{\alpha}$ is fixed, then the value v of the game is $v(x;\bar{\alpha}) = \inf_{\bar{\beta}} J(x,\bar{\alpha},\bar{\beta})$. The function $v(\cdot;\bar{\alpha}) : x \to v(x;\bar{\alpha})$ is solution of the equation $v(x;\bar{\alpha}) = F(v;x,\bar{\alpha}(x))\ \forall x \in \mathcal{X}$ where $F(v;x,a)$ is defined in (6). Equivalently, v is solution of the equation $v = F^{\bar{\alpha}}(v)$ where $F^{\bar{\alpha}}$ is the operator depending on $\bar{\alpha}$ which maps v to the function

$$\begin{array}{rccl} F^{\bar{\alpha}}(v): & \mathcal{X} & \to & \mathbb{R} \\ & x & \to & F(v;x,\bar{\alpha}(x)) \ . \end{array}$$

The policy iteration algorithm is given in Algorithm 1.

Algorithm 1 Policy Iteration

Given an initial policy $\bar{\alpha}_0$ for MAX, the policy iteration consists in applying successively the two following steps:

1. Compute the value v^{n+1} of the game with fixed feedback strategy $\bar{\alpha}_n$, that is the solution of
$$v^{n+1} \ = \ F^{\bar{\alpha}_n}(v^{n+1})$$

2. Improve the policy: Find the optimal feedback strategy $\bar{\alpha}_{n+1}$ of MAX for the value v^{n+1}:
$$\bar{\alpha}_{n+1}(x) \ \in \ \operatorname*{argmax}_{a\in\mathcal{A}(x)} F(v^{n+1};x,a) \quad \forall x \in \mathcal{X}.$$

The first step is performed itself using the policy iteration algorithm for a one-player game. That is, given an initial feedback strategy for MIN $\bar{\beta}_{n,0}$, which maps each point x of $\mathcal{X}$ to an action of MIN $\bar{\beta}_{n,0}(x) \in \mathcal{B}(x,\bar{\alpha}_n(x))$, we iterate

on MIN policies $\bar{\beta}_{n,k}$ and value functions $v^{n,k}$. Recall that the value of the game when the strategies of both players are fixed to $\bar{\alpha}$ and $\bar{\beta}$ is the payoff : $v(x;\bar{\alpha},\bar{\beta}) = J(x,\bar{\alpha},\bar{\beta})$. The function $v(\cdot;\bar{\alpha},\bar{\beta}) : x \to v(x;\bar{\alpha},\bar{\beta})$ is solution of the linear system $v(x;\bar{\alpha},\bar{\beta}) = F(v;x,\bar{\alpha}(x),\bar{\beta}(x))\ \forall x \in \mathcal{X}$ where

$$F(v;x,a,b) \ = \ \sum_{y\in\mathcal{X}} \mu\ P(y|x,a,b)\ v(y) \ + \ r(x,a,b). \tag{18}$$

Equivalently, v is solution of the linear system $v = F^{\bar{\alpha},\bar{\beta}}(v)$ where $F^{\bar{\alpha},\bar{\beta}}$ denotes the operator, depending on $\bar{\alpha}$ and $\bar{\beta}$, which to v associates the function

$$\begin{array}{rccl} F^{\bar{\alpha},\bar{\beta}}(v): & \mathcal{X} & \to & \mathbb{R} \\ & x & \to & F(v;x,\bar{\alpha}(x),\bar{\beta}(x)) \end{array}$$

where $F(v;x,a,b)$ is defined by (18). Then at each step k of the interior policy iteration, one computes $v^{n,k+1}$, the value of the game with fixed strategies $\bar{\alpha}_n$ for MAX and $\bar{\beta}_{n,k}$ for MIN as the solution of

$$v^{n,k+1} = F^{\bar{\alpha}_n,\bar{\beta}_{n,k}}(v^{n,k+1}) \tag{19}$$

and improve the policy for MIN by :

$$\bar{\beta}_{n,k+1}(x) \ \in \ \underset{b\in\mathcal{B}(x,\bar{\alpha}_n(x))}{\operatorname{argmin}} \ F(v^{n,k+1};x,\bar{\alpha}_n(x),b) \qquad \forall x \in \mathcal{X}.$$

Each policy iteration strictly improve the current strategy, hence it can never visit twice the same policy. Moreover, it produces a non decreasing (resp. non increasing) sequence of values $(v^n)_{n\geq 1}$ (resp. $(v^{n,k})_{k\geq 1}$) of the external loop (resp. internal loop), see for instance [40, 32, 9] for one player games and [39, 14] for two player games. Hence, if the action sets for both players are finite in each point of $\mathcal{X}$, the policy iterations stop after a finite time [39]. In all cases, this method converges faster than the value iteration algorithm, and the sequence of value functions $(v^n)_{n\geq 1}$ has at least a linear convergence to the solution with a factor for the sup-norm less than or equal to $1-\mu$. In practice it ends in few steps (see for instance large scale random examples for deterministic games shown in [18]).

Moreover, under regularity assumptions, the policy iteration algorithm for a one player game with infinite action spaces is equivalent to Newton's method, thus can have a super-linear convergence in the neighborhood of the solution, see [40, 2, 4] for order $p > 0$ superlinear convergence under some regularity and strong convexity assumptions, and [10] for superlinear convergence under more general regularity assumptions.

Moreover, the policy iteration algorithm for a one player game can be considered as a generalization of Newton's method [40, 2, 4, 10]. Indeed, define $G(v) = F(v) - v$ where F is the programing dynamic operator of a one player game, then the problem is to find the solution of $G(v) = 0$ where each entries of G are concave functions. The policy improvement step can be seen as the

computation of an element of the sup-differential of G in the current approximation of v and the value improvement step compute the zero of the previous sup-differential. When G is regular, the sequence of value functions $(v^n)_{n\geq 1}$ is exactly the sequence of the Newton's algorithm.

4.2 AMG

The linear systems defined in (19) are all of the form $v = \mu M v + r$ where M a Markov matrix. We shall solve them using algebraic multigrid methods which we shall recall in this section.

Standard multigrid was originally created in the seventies to solve efficiently linear elliptic partial differential equations (see for instance [33]). It works as follows. Assume the continuous equation is discretized on a sequence of grids. Each of them, starting from a coarse grid, being a refinement of the previous until a given accuracy is attained. The size of the coarsest grid is chosen such that the cost of solving the problem on it is cheap. Assume also that transfer operators between these grids are given: interpolation and restriction. Then, a multigrid cycle on the finest grid consists in : first, the application of a smoother on the finest grid; then a restriction of the residual on the next coarse grid; then solving the residual problem on this coarse grid using the same multigrid scheme; then, interpolate this solution (which is an approximation of the error) and correct the error on the fine grid; finally, the application of a smoother on the finest grid. If the multigrid components are properly chosen, this process is efficient to find the solution on the finest grid. Indeed, in general the relaxation process is smoothing the error which then can be well approximated by elements in the range of the interpolation. It implies, in good cases, that the contraction factor of the multigrid method is independent of the discretization step and also the complexity is in the order of the number of discretization points. We shall refer to this standard method as geometric multigrid.

Algebraic multigrid method, called AMG, has been initially developed in the early eighties (see for example [13, 12, 41]) for solving large sparse linear systems arising from the discretization of partial differential equations with unstructured grids or PDE's not suitable for the application of the geometric multigrid solver or large discrete problems not derived from any continuous problem.

The AMG method consists of two phases, called "setup phase" and "solving phase". In contrast to geometric multigrids, the mode of constructing the coarse levels (coarse "grids") which constitute the setup phase, is based only on the algebraic equations. The points of the fine grids are represented by the variables and coarse grids by subset of these variables. The selection of those coarse variables and the construction of the transfer operators between levels are done in such a way that the range of the interpolation approximates the errors not reduced by a given relaxation scheme. Then the "solving phase" is performed in the same way as a geometric multigrid method and consists of the application of a smoother and a correction of the error by a coarse grid solution. The whole process is briefly recall below.

Consider a system of n linear equations given in the matrix form:

$$Av = f \tag{20}$$

where the matrix $A \in \mathbb{R}^{n\times n}$ and the vector f $\mathbb{R}^n$ are given, and we are looking for the vector $v \in \mathbb{R}^n$. We call fine grid Ω^h the set of all variables of the system, i.e. $\Omega^h = \{1, \ldots, n\}$.

First, recall that a relaxation method consist of the following approximations:

$$u \leftarrow Su + S_o f \quad \text{with} \quad S = I - S_o A$$

where S is called the smoothing operator and I is the identity operator in $\mathbb{R}^{n\times n}$. The error $e = u - v$ propagates as

$$e \leftarrow Se.$$

The method is said to converge if $\rho(S) < 1$ where $\rho(S) = \max_i |\lambda_i|$ is the spectral radius of S with λ_i his eigenvalues. For example, the smoother operator of the weighted Jacobi method is $S = I - wD^{-1}A$ and that of the Gauss-Seidel is $S = I - L^{-1}A$ where D and L are the diagonal and lower triangular part of the matrix A resp.

Assume Ω^l the grid on level l where level 0 correspond to the finest grid Ω^h. The construction of the coarse grid Ω^{l+1} from the fine grid Ω^l, consists in the splitting of the n_l variables from the grid Ω^l into two distinct subsets, namely C which contains the variables belonging to both grids, Ω^l and Ω^{l+1}, and F the variables belonging to the grid Ω^l only. We have then $\Omega^l = C \cup F$. The coarse grid $\Omega^{l+1} = C$ contains $n_{l+1} = n_{l_C}$ variables. This splitting is based on the "connections" between the variables on level l [13, 41] and such as the range of the associate interpolation or prolongation operator $\mathcal{P}_{l+1}^l$ accurately approximates the errors not efficiently reduced by the relaxation phase (these errors are "smooth" in the algebraic multigrid terminology). The restriction operator $\mathcal{R}_l^{l+1}$ maps residuals from grid Ω^l to the grid Ω^{l+1}. In [13, 41], the operator is fixed to be $\mathcal{R}_l^{l+1} = (\mathcal{P}_{l+1}^l)^T$. The coarse grid operator is defined by $A^{l+1} = \mathcal{R}_l^{l+1} A^l \mathcal{P}_{l+1}^l$ where A^{l+1} is the approximation of A^l on Ω^{l+1} and $A^0 = A$. Similarly, for any vector $v^l \in \mathbb{R}^{n_l}$ we denote $v^{l+1} = \mathcal{R}_l^{l+1} v^l$ its restriction on Ω^{l+1}. This construction can be repeated recursively from the finest level $l = 0$ to the coarsest level L.

The solution phase consists in applying the multigrid cycle described in Algorithm 2, it is called V(ν_1,ν_2)-cycle if $\gamma = 1$ and W(ν_1,ν_2)-cycle if $\gamma = 2$.

Convergence results for AMG are described by using operator norms defined by the following inner products

$$\langle u, v\rangle_0 = \langle Du, v\rangle, \quad \langle u, v\rangle_1 = \langle Au, v\rangle, \quad \langle u, v\rangle_2 = \left\langle D^{-1}Au, Av\right\rangle,$$

where $u, v \in \mathbb{R}^n$, $D = \mathrm{diag}(A)$ is the diagonal part of A, $\langle\cdot,\cdot\rangle$ is the euclidean inner product of $\mathbb{R}^n$ and $\|u\|_i = (\langle u, u\rangle_i)^{1/2}$ $(i = 0, 1, 2)$ are the associated norms.

Algorithm 2 Multigrid scheme $u^l \leftarrow MG(u^l, f^l)$

if $l < L$ **then**
  pre relaxation :
    $u^l \leftarrow Su^l + S_o f^l$ (on Ω^l) $\quad \nu_1$ times
  coarse grid correction :
    $f^{l+1} \leftarrow \mathcal{R}_l^{l+1}(f^l - A^l u^l)$
    $u^{l+1} \leftarrow 0$
    $u^{l+1} \leftarrow MG(u^{l+1}, f^{l+1}) \quad \gamma$ times
    $u^l \leftarrow u^l + \mathcal{P}_{l+1}^l u^{l+1}$
  post relaxation :
    $u^l \leftarrow Su^l + S_o f^l$ (on Ω^l) $\quad \nu_2$ times
else
  Solve $A^L u^L = f^L$
end if

Theorem 4.1 ([41]). *Assume A to be symmetric and positive definite and that the interpolation operator $\mathcal{P}_{l+1}^l$ have full rank and that the restriction and coarse grid operators are defined as before. Furthermore, suppose that for all e^l,*

$$\left\|S^l e^l\right\|_1^2 \leq \left\|e^l\right\|_1^2 - \delta \left\|T^l e^l\right\|_1^2 \qquad \text{where } T^l = I^l - \mathcal{P}_{l+1}^l (A^{l+1})^{-1} \mathcal{R}_l^{l+1} A^l$$

holds with some $\delta > 0$ independently of e^l and l. Then $\delta \leq 1$, and provided that the coarsest grid equation is solved and that at least one smoothing step is performed after each coarse grid correction step, the $V(0,1)$-cycle to solve (20) *has a convergence factor (with respect to the energy norm) bounded above by $\sqrt{1-\delta}$.*

However, as mentioned in [12], a realistic coarsing strategy can hardly satisfy the condition of this theorem for the V-cycle convergence except for linear system arising from regular elliptic PDE's. Weaker conditions exist for two-level convergence for linear systems where the matrix of the system is an M-matrix, symmetric and positive definite, see for instance [13, 12, 19]. Also we can find in the literature, two-grid convergence analysis for non-symmetric linear system in [37], also in [34] which is based on the analysis of AMLI, a block incomplete factorization partitioned in hierarchical form.

Another multigrid scheme called full multigrid (FMG) is based on the idea of starting, at each level, the MG cycle with a good initial guess. This is done by using the solution of the coarse problem Ω^{l+1}, which is a good approximation of the solution on the fine grid Ω^l, to start the multigrid cycle on the fine grid Ω^l. Therefore a transfer operators are define and may differ from the ones used in the multigrid cycle. The FMG scheme starts from the coarsest grid and is given in Algorithm 3.

Algorithm 3 Full multigrid scheme $u^l \leftarrow FMG(f^l)$

if $l < L$ **then**
  $f^{l+1} \leftarrow \mathcal{R}_l^{l+1} f^l$
  $u^{l+1} \leftarrow FMG(f^{l+1})$
  $u^l \leftarrow \mathcal{P}_{l+1}^l u^{l+1}$
end if
$u^l \leftarrow MG(u^l, f^l)$ γ times

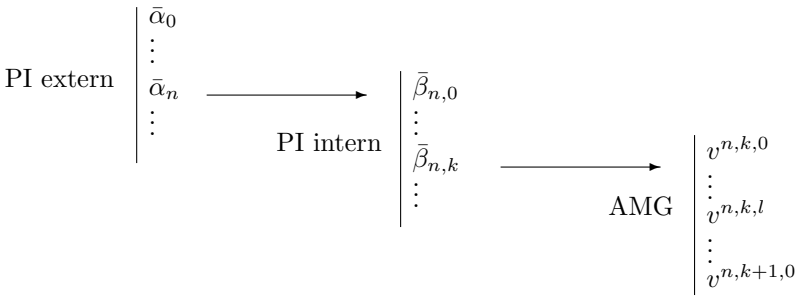

Figure 1: Representation of the nested iterations of AMGπ.

5 A multigrid algorithm for discrete dynamic programming equations

5.1 AMGπ

Recall that in the policy iteration for games at each step k of the interior policy iteration, we have to solve the linear system $v^{n,k+1} = F^{\bar{\alpha}_n, \bar{\beta}_{n,k}}(v^{n,k+1})$ (19). These systems are all of the form $v = \mu M v + r$ with M a Markov matrix and $0 < \mu < 1$ the discounted factor. Since $(I - \mu M)$ are non singular M-matrices, we use AMG for the resolution of those systems. Our algorithm, named AMGπ, is the combination of policy iterations and AMG. The iterations of AMGπ are summarized in the scheme represented in Figure 1 where $(v^{n,k,0}, \cdots, v^{n,k,l}, \cdots, v^{n,k+1,0})$ is a sequence of value functions generated by the multigrid solver. The algebraic multigrid methods allows us to solve linear systems arising from either the discretizations of Isaacs or Hamilton-Jacobi-Bellman equation either a true finite state space zero-sum two player games. For the true finite state space zero-sum two player games, we use the AGMG algorithm of [36] which is more adapted for non symmetric problems.

In the one player game case, convergence results have been established by Hoppe [27, 28] and Akian [1, 2].

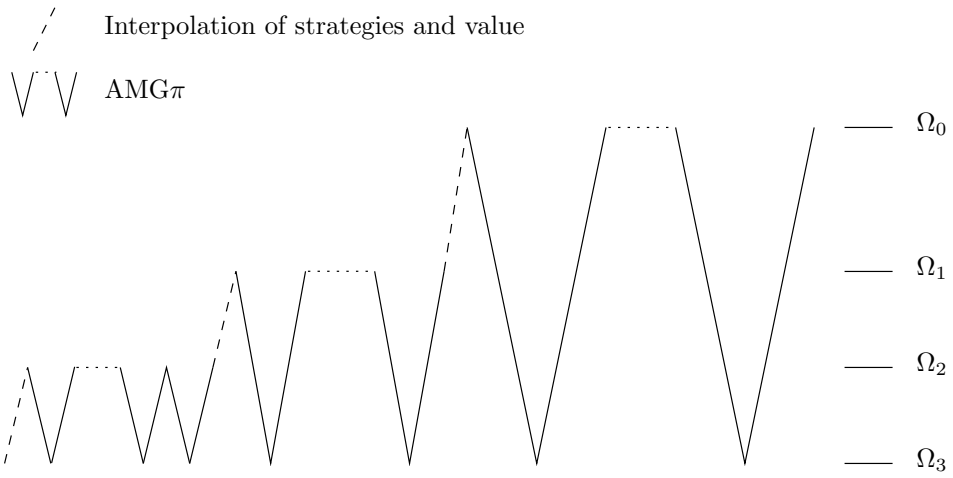

Figure 2: Full multi-level AMGπ with AMGπ V-cycles

5.2 Full multi-level AMGπ

Recall that the number of policy iterations can be exponential. As for Newton's algorithm, convergence can be improved by starting the policy iteration with a good initial guess, close to the solution. In this way, we developed a full multi-level scheme called full multi-level AMGπ. As in standard FMG, it consists in solving the problem at each grid level by performing policy iterations AMGπ until a convergence criterion is verified, then to interpolate the strategies and value to the next level, in order to initialize the policy iterations of the next level. This scheme is repeated until the finest level is attained.

Figure 2 illustrates the full multi-level AMGπ algorithm when V-cycles are use in AMGπ. The dashed lines represent the interpolation of the solution and strategies from a coarse grid Ω^{l+1} to the next fine grid Ω^l. The continuous V-lines are the V-cycles AMGπ which are not fixed in number since at each level AMGπ cycles are performed until a given criterion is attained.

Note that our full multi-level program only apply to stochastic differential games since for them coarse representation, including equations and strategies, can be easily constructed by tacking different size of discretization step. We use linear interpolations between the levels of the full multi-level algorithm to interpolate values and strategies.

For one-player discounted games with infinite number of actions and under regularity and strong convexity assumptions, it is shown in [2, 1] that this full multi-level policy iteration has a computing time in the order of $|\mathcal{X}|$.

6 Numerical results

In this section, we apply our programs AMGπ and Full multilevel AMGπ, which were implemented in C, to examples of Isaac's equations and Variational Inequalities.

6.1 Parameters and notations

We first give some implementation details of our code and the parameters that were use for the tests. Then we give the details of the notations for the results.

Our AMG solver implements the construction phase, including the coarsing scheme and definition of interpolation, described in [41] and the general recursive multigrid cycles for the solution phase. In the tests, W(1,1)-cycles were used. The chosen smoother is a CF relaxation method, a Gauss Seidel relaxation scheme that relax first on C points and than on F points.

We implemented the policy iterations algorithm as explain in section 4.1. The policy iterations stop when the norm of the residual, $\|r\|_{L_2}$, is smaller than a given value denoted by ϵ. For the iterations on MIN policies (i.e. internal loop of the policy iteration algorithm), the residual is $r = F^{\bar{\alpha}_n}(v) - v$ and for the iterations on MAX policies (i.e. external loop of the policy iteration algorithm), $r = F(v) - v$ which is the residual of the game.

Our AMGπ code is the combination of our policy iteration algorithm and the AMG solver.

Our full multi-level AMGπ algorithm is the implementation of the method explain in the previous section 5.2. The stopping criterion at each level of the full multi-level AMGπ is $\|r\|_{L_2} < c\, {h_l}^2$ where h_l is the discretization step size of the current coarse level Ω_l and $c = 0.1$.

The numerical results represented in the tables use the following notations: n denotes the iteration over MAX policies and k is the corresponding number of iterations for MIN policies. The residual error of the game is denoted by r and the exact error by $e = F(v) - u$ where u is the exact solution of the game. Infinity norm and discrete L_2 norm are given for each of them.

6.2 Isaacs equations

The first example concern a diffusion problem where the dynamic programing equation is given by

$$\begin{cases} \Delta v + \|\nabla v\|_2 - 0.5\,\|\nabla v\|_2^2 + f = 0 \text{ in } \mathcal{X} \\ v = g \text{ on } \partial\mathcal{X} \end{cases} \tag{21}$$

where $\mathcal{X} =]0,1[\times]0,1[$ is the unit square. The exact solution is $v(x_1, x_2) = sin(x_1) \times sin(x_2)$ on $\mathcal{X} = [0,1] \times [0,1]$ and is represented in Figure 3.

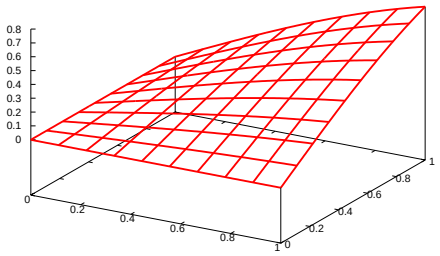

Figure 3: Graph of $sin(x_1) \times sin(x_2)$ on $\mathcal{X} = [0,1] \times [0,1]$.

Table 1: Numerical results for equation (21).

Policy iteration with LU

n	k	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	3	$8.51e-7$	$5.96e-7$	$4.47e-2$	$2.48e-2$	$1.40e+2$
2	2	$2.44e-8$	$6.16e-9$	$1.84e-4$	$1.05e-4$	$2.31e+2$
3	1	$7.38e-13$	$2.03e-13$	$4.13e-6$	$2.16e-6$	$2.77e+2$

AMGπ

n	k	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	3	$8.51e-7$	$5.96e-7$	$4.47e-2$	$2.48e-2$	$2.65e+1$
2	2	$2.44e-8$	$6.16e-9$	$1.84e-4$	$1.05e-4$	$4.59e+1$
3	1	$7.92e-13$	$2.02e-13$	$4.13e-6$	$2.16e-6$	$5.56e+1$

Equation (21) is equivalent to :

$$\begin{cases} \Delta v + \max\limits_{\|\alpha\|_2 \leq 1} (\alpha \cdot \nabla v) + \min\limits_{\beta} \left(\beta \cdot \nabla v + \dfrac{\|\beta\|_2^2}{2} \right) + f = 0 \text{ in } \mathcal{X} \\ v = g \text{ on } \partial\mathcal{X} \end{cases} \tag{22}$$

Numerical results were performed on equations (22) discretized on a grid with 1025 nodes in each directions, i.e. with a discretization step of $h = 1/2^{10}$ in each directions. The stopping criterion for the policy iterations is $\epsilon = 1e-10$. First table in 1 shows the results of the policy iteration algorithm with a direct solver LU (we used the package UMFPACK [16]) and the second table in 1 the results of AMGπ.

We observe in both tables that AMGπ solves the problem faster than the policy iteration with a direct solver. Also remark that only three steps on MAX policies are needed and a total of six internal loops which involves the resolution of six linear systems. This is due to the fact that the solution is regular.

The number of iterations for the linear solver AMG are illustrated in Table

n	k	AMG	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	2	4|3	$2.121e-04$	$1.489e-04$	$4.443e-02$	$2.497e-02$	$6.000e-02$
2	1	3	$3.737e-06$	$1.130e-06$	$1.177e-04$	$5.467e-05$	$8.000e-02$

Table 2: Numerical results with a 65×65 points grid, computed by AMGπ for equation (21).

n	k	AMG	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	2	4|3	$5.363e-05$	$3.764e-05$	$4.462e-02$	$2.491e-02$	$2.200e-01$
2	1	3	$9.628e-07$	$2.805e-07$	$1.388e-04$	$6.935e-05$	$3.300e-01$

Table 3: Numerical results with a 129×129 points grid, computed by AMGπ for equation (21).

n	k	AMG	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	2	4|3	$1.349e-05$	$9.464e-06$	$4.467e-02$	$2.485e-02$	$1.070e+00$
2	1	3	$2.441e-07$	$6.992e-08$	$1.498e-04$	$7.717e-05$	$1.560e+00$

Table 4: Numerical results with a 257×257 points grid, computed by AMGπ for equation (21).

n	k	AMG	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	2	4|3	$3.383e-06$	$2.373e-06$	$4.468e-02$	$2.482e-02$	$4.730e+00$
2	1	3	$6.145e-08$	$1.746e-08$	$1.554e-04$	$8.114e-05$	$6.860e+00$

Table 5: Numerical results with a 513×513 points grid, computed by AMGπ for equation (21).

n	k	AMG	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	2	4|3	$8.470e-07$	$5.940e-07$	$4.468e-02$	$2.479e-02$	$2.023e+01$
2	1	3	$1.541e-08$	$4.362e-09$	$1.582e-04$	$8.314e-05$	$2.902e+01$

Table 6: Numerical results with a 1025×1025 points grid, computed by AMGπ for equation (21).

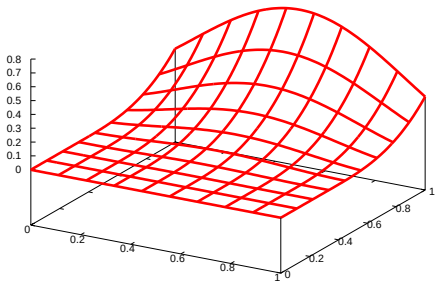

Figure 4: Graph of the solution of equation (23).

2 to 6, for the same example discretized on grids with different mesh sizes. The second column of each tables contained the number of iterations of AMG. The stopping criterion for the policy iterations of AMGπ is $0.1\,h^2$ where h is the discretization step size. We can see that the number of iterations for AMG is independent of the size of the problem.

6.3 Variational inequalities

Next tests concern optimal stopping time games where the variational equality is given by

$$\begin{cases} \max\{\underbrace{\Delta v - 0.5\,\|\nabla v\|_2^2 + f}_{①}\,,\,\underbrace{\phi - v}_{②}\} \;=\; 0 \text{ in } \mathcal{X} \\ v \;=\; \phi \text{ on } \partial\mathcal{X} \end{cases} \tag{23}$$

where $\mathcal{X} =]0,1[\times]0,1[$ is the unit square. As for (11), in each $x \in \mathcal{X}$, action space of player MAX is $\{1,2\}$, 1 means continue to play and 2 means stop the game. When MAX decides to stop the game, he receives $\phi(x)$. Player MIN plays as in (22). The exact solution v is represented in Figure 4.

For numerical purpose, the equations ① and ② are simplified separately by keeping equations (12) true. This example leads to a free boundary problem for the actions of MAX and numerical results with AMGπ are shown geometrically in Figure 5 where only the actions of MAX are represented. Equation (23) is discretized on a grid with 1025 points in each directions, green points represent action 1 of MAX and blue points action 2. Optimal solution is to have only green points above the red line and only blue points under. We start the test with blue points for MAX in the whole domain.

We observe in this Figure 5 and Table 7 that AMGπ find the solution after 702 iterations and in approximately two hours and fifteen minutes. The stopping criterion for policy iterations of AMGπ in this test is $\epsilon = 1e-14$. As mention before the number of policy iterations can be exponential in the cardinality of

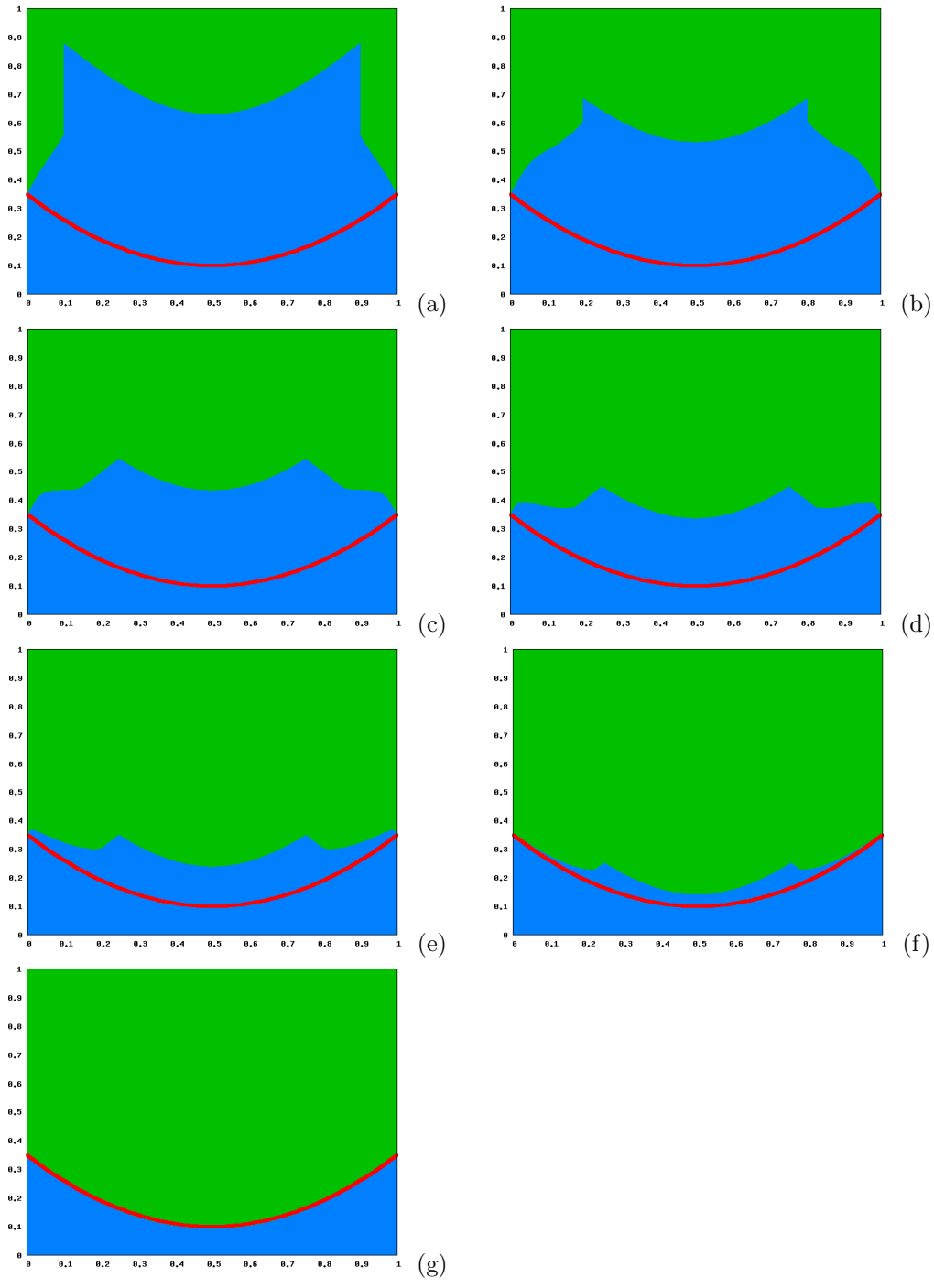

Figure 5: Application of AMGπ to the free boundary problem (23) for a 1025×1025 points grid : (a) after 100 iterations, (b) after 200 iterations, (c) after 300 iterations, (d) after 400 iterations, (e) after 500 iterations, (f) after 600 iterations and (g) after 700 iterations.

Table 7: Numerical results for optimal stopping time game (23) with a 1025×1025 points grid, computed by AMGπ.

n	k	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
1	0	$3.645e-01$	$9.195e-03$	$7.243e-01$	$1.998e-01$	$1.790e+00$
2	4	$1.497e-01$	$1.347e-03$	$3.782e-01$	$1.218e-01$	$1.376e+01$
3	4	$1.094e-01$	$8.839e-04$	$3.767e-01$	$1.213e-01$	$2.492e+01$
...						
100	3	$1.744e-02$	$4.444e-05$	$2.392e-01$	$8.016e-02$	$1.009e+03$
...						
200	3	$7.398e-03$	$1.879e-05$	$1.222e-01$	$3.996e-02$	$2.214e+03$
...						
300	3	$2.510e-03$	$8.779e-06$	$5.614e-02$	$1.728e-02$	$3.619e+03$
...						
400	2	$1.258e-03$	$4.363e-06$	$2.321e-02$	$6.519e-03$	$4.770e+03$
...						
500	2	$4.761e-04$	$1.620e-06$	$6.601e-03$	$1.532e-03$	$5.861e+03$
...						
600	2	$8.857e-05$	$2.781e-07$	$7.274e-04$	$9.598e-05$	$7.045e+03$
...						
650	2	$1.533e-05$	$4.231e-08$	$1.538e-04$	$6.331e-05$	$7.630e+03$
...						
700	1	$5.647e-08$	$8.734e-11$	$1.571e-04$	$6.619e-05$	$8.134e+03$
701	1	$1.207e-08$	$2.267e-11$	$1.571e-04$	$6.619e-05$	$8.141e+03$
702	1	$9.992e-16$	$7.284e-17$	$1.571e-04$	$6.619e-05$	$8.148e+03$

Table 8: Numerical results for optimal stopping time game (23) with a 1025×1025 points grid, computed by full multi-level AMGπ.

n	k	$\|r\|_\infty$	$\|r\|_{L_2}$	$\|e\|_\infty$	$\|e\|_{L_2}$	cpu time s
nodes in each directions : 3, step size : $5.00e-01$						
1	1	$2.17e-01$	$2.17e-01$	$1.53e-01$	$1.53e-01$	$<<1$
2	1	$1.14e-02$	$1.14e-02$	$3.30e-02$	$3.30e-02$	$<<1$
nodes in each directions : 5, step size : $2.50e-01$						
1	2	$2.17e-04$	$8.26e-05$	$3.02e-02$	$1.71e-02$	$<<1$
nodes in each directions : 9, step size : $1.25e-01$						
1	2	$4.99e-03$	$1.06e-03$	$1.65e-02$	$7.99e-03$	$<<1$
2	1	$2.68e-03$	$5.41e-04$	$1.66e-02$	$8.15e-03$	$<<1$
3	1	$2.72e-04$	$5.49e-05$	$1.68e-02$	$8.30e-03$	$<<1$
nodes in each directions : 17, step size : $6.25e-02$						
1	1	$1.73e-03$	$2.80e-04$	$9.68e-03$	$4.37e-03$	$1.00e-02$
2	1	$6.40e-04$	$7.43e-05$	$8.96e-03$	$4.10e-03$	$1.00e-02$
3	1	$1.65e-07$	$2.13e-08$	$9.01e-03$	$4.14e-03$	$1.00e-02$
nodes in each directions : 33, step size : $3.12e-02$						
1	1	$1.36e-04$	$1.03e-05$	$4.94e-03$	$2.16e-03$	$1.00e-02$
2	1	$6.86e-05$	$2.54e-06$	$4.77e-03$	$2.09e-03$	$2.00e-02$
nodes in each directions : 65, step size : $1.56e-02$						
1	1	$3.32e-05$	$8.94e-07$	$2.49e-03$	$1.07e-03$	$4.00e-02$
2	1	$1.30e-05$	$2.28e-07$	$2.45e-03$	$1.05e-03$	$6.00e-02$
nodes in each directions : 129, step size : $7.81e-03$						
1	1	$4.86e-06$	$8.55e-08$	$1.25e-03$	$5.33e-04$	$1.50e-01$
nodes in each directions : 257, step size : $3.91e-03$						
1	1	$1.23e-06$	$2.15e-08$	$6.29e-04$	$2.66e-04$	$6.00e-01$
nodes in each directions : 513, step size : $1.95e-03$						
1	1	$2.57e-07$	$4.04e-09$	$3.15e-04$	$1.33e-04$	$2.62e+00$
nodes in each directions : 1025, step size : $9.77e-04$						
1	1	$1.31e-07$	$1.90e-09$	$1.57e-04$	$6.63e-05$	$1.17e+01$
2	1	$6.77e-08$	$5.83e-10$	$1.57e-04$	$6.62e-05$	$2.11e+01$

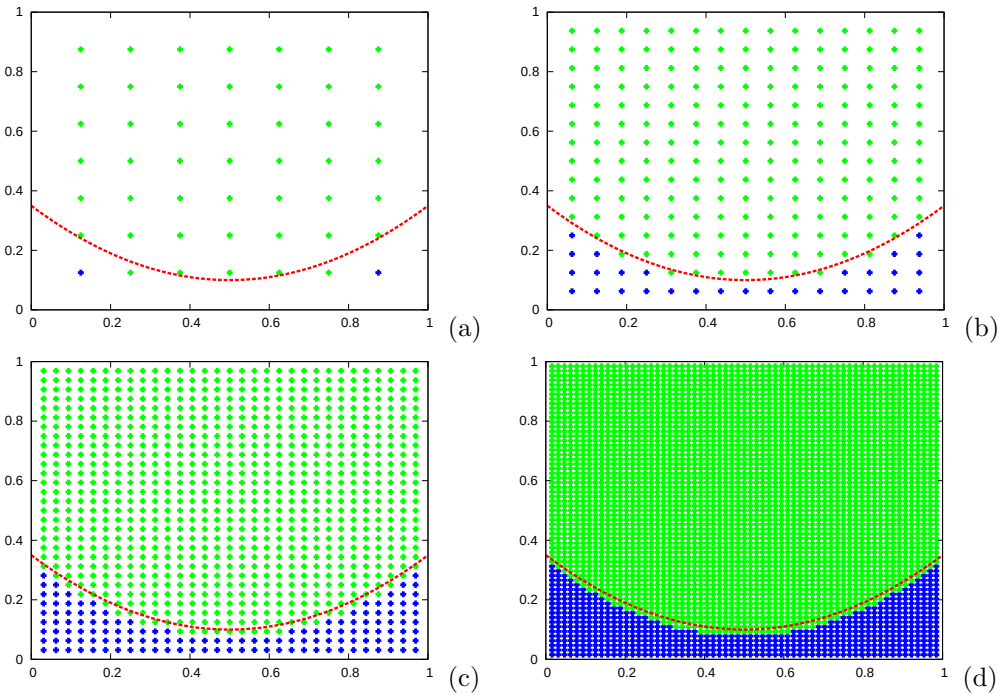

Figure 6: Application of Full multilevel AMGπ to the free boundary problem (23) for: (a) 9×9 points grid, (b) 17×17 points grid, (c) 33×33 points grid, (d) 65×65 points grid.

$\mathcal{X}$. However, by starting AMGπ with a good initial guest, numerical results can be improved. With this example we show the advantage of using the full multi-level AMGπ. Indeed, numerical results of the application of the full multi-level AMGπ to problem (23) with a 1025×1025 points grid, are display in table 8. We observe that our algorithm has solve the problem in 23 seconds. Geometrical representation of MAX actions, obtained at the end of the resolution on four successive levels during the Full multi-level AMGπ resolution, are shown in Figure 6. With less nodes (coarse grids), the algorithm can find a good approximation of the solution in a few iterations. The interpolation of this solutions and the corresponding strategies, are used to start AMGπ on the next fine level and only a few number of policy iterations are needed on each levels.

7 Conclusion and perspective

We have presented our algorithm AMGπ for solving general two player zero-sum stochastic games. This program combines the policy iteration algorithm with algebraic multigrid methods. Our experiences on Isaacs equations show better results for AMGπ in comparison with policy iteration combined to a direct linear solver.

Furthermore, we also introduced a full multi-level AMGπ algorithm for solving two player zero-sum stochastic differential games. The numerical results on variational inequalities presented here show that our full multi-level algorithm improves substantially the computation time of policy iteration algorithm. Indeed the computation time of full multi-level AMGπ algorithm seems to be in the order of the number of discretization point whereas that of a amgπ algorithm is 400 times greater, since the number of necessary policy iteration to solve the equation is in the order of the diameter of the graph of the controlled Markov Chain.

This full multi-level AMGπ uses coarse grids discretizations of the partial differential equation and so cannot be applied directly to the dynamic programming equation of a two player zero-sum stochastic game with finite state space. One may ask if adapting the full multi-level AMGπ algorithm to this kind of games is possible. Indeed, the complexity of two player zero-sum stochastic games is still unsettled, one only knows that it belongs to the complexity class of NP$\cap$coNP [39], and any new approach maybe useful to understand this complexity.

References

[1] Marianne Akian. Analyse de l'algorithme multigrille FMGH de résolution d'équations d'Hamilton-Jacobi-Bellman. In *Analysis and optimization of systems (Antibes, 1990)*, volume 144 of *Lecture Notes in Control and Inform. Sci.*, pages 113–122. Springer, Berlin, 1990.

[2] Marianne Akian. *Méthodes multigrilles en contrôle stochastique*. Institut National de Recherche en Informatique et en Automatique (INRIA), Rocquencourt, 1990. Thèse, Université de Paris IX (Paris-Dauphine), Paris, 1990.

[3] Marianne Akian, Stéphane Gaubert, and Asma Lakhoua. The max-plus finite element method for solving deterministic optimal control problems: basic properties and convergence analysis. *SIAM J. Control Optim.*, 47(2):817–848, 2008.

[4] Randolph E. Bank and Donald J. Rose. Analysis of a multilevel iterative method for nonlinear finite element equations. *Math. Comp.*, 39(160):453–465, 1982.

[5] M. Bardi, M. Falcone, and P. Soravia. Fully discrete schemes for the value function of pursuit-evasion games. In *Advances in dynamic games and applications (Geneva, 1992)*, volume 1 of *Ann. Internat. Soc. Dynam. Games*, pages 89–105. Birkhäuser Boston, Boston, MA, 1994.

[6] Martino Bardi, Maurizio Falcone, and Pierpaolo Soravia. Numerical methods for pursuit-evasion games via viscosity solutions. In *Stochastic and differential games*, volume 4 of *Ann. Internat. Soc. Dynam. Games*, pages 105–175. Birkhäuser Boston, Boston, MA, 1999.

[7] G. Barles and P. E. Souganidis. Convergence of approximation schemes for fully nonlinear second order equations. *Asymptotic Anal.*, 4(3):271–283, 1991.

[8] Richard Bellman. *Dynamic programming*. Princeton University Press, Princeton, N. J., 1957.

[9] D. P. Bertsekas. *Dynamic programming*. Prentice Hall Inc., Englewood Cliffs, NJ, 1987. Deterministic and stochastic models.

[10] Olivier Bokanowski, Stefania Maroso, and Hasnaa Zidani. Some convergence results for Howard's algorithm. *SIAM J. Numer. Anal.*, 47(4):3001–3026, 2009.

[11] J. Frédéric Bonnans and Housnaa Zidani. Consistency of generalized finite difference schemes for the stochastic HJB equation. *SIAM J. Numer. Anal.*, 41(3):1008–1021 (electronic), 2003.

[12] A. Brandt, S. McCormick, and J. Ruge. Algebraic multigrid (AMG) for sparse matrix equations. In *Sparsity and its applications (Loughborough, 1983)*, pages 257–284. Cambridge Univ. Press, Cambridge, 1985.

[13] Achi Brandt. Algebraic multigrid theory: the symmetric case. *Appl. Math. Comput.*, 19(1-4):23–56, 1986. Second Copper Mountain conference on multigrid methods (Copper Mountain, Colo., 1985).

[14] Jean Cochet-Terrasson and Stéphane Gaubert. A policy iteration algorithm for zero-sum stochastic games with mean payoff. *C. R. Math. Acad. Sci. Paris*, 343(5):377–382, 2006.

[15] Michael G. Crandall, Hitoshi Ishii, and Pierre-Louis Lions. User's guide to viscosity solutions of second order partial differential equations. *Bull. Amer. Math. Soc. (N.S.)*, 27(1):1–67, 1992.

[16] Timothy A. Davis. Algorithm 832: Umfpack v4.3—an unsymmetric-pattern multifrontal method. *ACM Trans. Math. Softw.*, 30:196–199, June 2004.

[17] Eric V. Denardo. Contraction mappings in the theory underlying dynamic programming. *SIAM Rev.*, 9:165–177, 1967.

[18] Vishesh Dhingra and Stéphane Gaubert. How to solve large scale deterministic games with mean payoff by policy iteration. In *valuetools '06: Proceedings of the 1st international conference on Performance evaluation methodolgies and tools*, page 12, New York, NY, USA, 2006. ACM.

[19] Robert D. Falgout, Panayot S. Vassilevski, and Ludmil T. Zikatanov. On two-grid convergence estimates. *Numer. Linear Algebra Appl.*, 12(5-6):471–494, 2005.

[20] John Fearnley. Exponential lower bounds for policy iteration. In *Automata, Languages and Programming*, pages 551–562, 2010.

[21] John Fearnley. Exponential lower bounds for policy iteration, 2010. arXiv:1003.3418v1.

[22] Jerzy Filar and Koos Vrieze. *Competitive Markov decision processes*. Springer-Verlag, New York, 1997.

[23] W. H. Fleming and P. E. Souganidis. On the existence of value functions of two-player, zero-sum stochastic differential games. *Indiana Univ. Math. J.*, 38(2):293–314, 1989.

[24] Avner Friedman. Stochastic games and variational inequalities. *Arch. Rational Mech. Anal.*, 51:321–346, 1973.

[25] Oliver Friedmann. An exponential lower bound for the parity game strategy improvement algorithm as we know it. In *LICS*, pages 145–156. IEEE Computer Society, 2009.

[26] A. J. Hoffman and R. M. Karp. On nonterminating stochastic games. *Management Sci.*, 12:359–370, 1966.

[27] Ronald H. W. Hoppe. Multigrid methods for Hamilton-Jacobi-Bellman equations. *Numer. Math.*, 49(2-3):239–254, 1986.

[28] Ronald H. W. Hoppe. Multigrid algorithms for variational inequalities. *SIAM J. Numer. Anal.*, 24(5):1046–1065, 1987.

[29] Ronald A. Howard. *Dynamic programming and Markov processes*. The Technology Press of M.I.T., Cambridge, Mass., 1960.

[30] Harold J. Kushner. *Probability methods for approximations in stochastic control and for elliptic equations*. Academic Press [Harcourt Brace Jovanovich Publishers], New York, 1977. Mathematics in Science and Engineering, Vol. 129.

[31] Harold J. Kushner and Paul G. Dupuis. *Numerical methods for stochastic control problems in continuous time*, volume 24 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, 1992.

[32] P.-L. Lions and B. Mercier. Approximation numérique des équations de Hamilton-Jacobi-Bellman. *RAIRO Anal. Numér.*, 14(4):369–393, 1980.

[33] Stephen F. McCormick, editor. *Multigrid methods*, volume 3 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1987.

[34] C. Mense and R. Nabben. On algebraic multi-level methods for non-symmetric systems—comparison results. *Linear Algebra Appl.*, 429(10):2567–2588, 2008.

[35] Rémi Munos and Hasnaa Zidani. Consistency of a simple multidimensional scheme for Hamilton-Jacobi-Bellman equations. *C. R. Math. Acad. Sci. Paris*, 340(7):499–502, 2005.

[36] Yvan Notay. An aggregation-based algebraic multigrid method. *Electronic Transactions on Numerical Analysis*, 37:123–146, 2010.

[37] Yvan Notay. Algebraic analysis of two-grid methods: The nonsymmetric case. *Numer. Linear Algebra Appl.*, 17(1):73–96, 2010.

[38] N. Shimkin O. Ziv. Multigrid methods for policy evaluation and reinforcement learning. In *Proc. IEEE International Symposium on Intelligent Control (ISIC05)*. IEEE, 2005.

[39] Anuj Puri. *Theory of hybrid systems and discrete event systems*. PhD thesis, Berkeley, CA, USA, 1995.

[40] Martin L. Puterman and Shelby L. Brumelle. On the convergence of policy iteration in stationary dynamic programming. *Math. Oper. Res.*, 4(1):60–69, 1979.

[41] J. W. Ruge and K. Stüben. Algebraic multigrid. In *Multigrid methods*, volume 3 of *Frontiers Appl. Math.*, pages 73–130. SIAM, Philadelphia, PA, 1987.

[42] L. S. Shapley. Stochastic games. In *Stochastic games and applications (Stony Brook, NY, 1999)*, volume 570 of *NATO Sci. Ser. C Math. Phys. Sci.*, pages 1–7. Kluwer Acad. Publ., Dordrecht, 2003. Reprint of Proc. Nat. Acad. Sci. U.S.A. **3**9 (1953), 1095–1100 [0061807].

[43] Sylvain Sorin. Classification and basic tools. In *Stochastic games and applications (Stony Brook, NY, 1999)*, volume 570 of *NATO Sci. Ser. C Math. Phys. Sci.*, pages 27–36. Kluwer Acad. Publ., Dordrecht, 2003.