A method for optimal control problems

An adaptative antidissipative method for optimal control problems

Olivier Bokanowski*, Nadia Megdich**, Hasnaa Zidani**

* Lab. Jacques-Louis Lions, Université Pierre et Marie Curie, 175 Rue Chevaleret 75013 Paris, France. Email: boka@math.jussieu.fr
** ENSTA, UMA, 32 Boulevard Victor, 75739 Paris Cedex 15, France.
Emails: {Nadia.Megdich, Hasnaa.Zidani}@ensta.fr.

RÉSUMÉ. On étudie une méthode numérique pour les équations HJB provenant des problèmes de contrôle optimal avec contraintes sur l'état. Plus précisément on présente un schéma antidissipatif sur une grille adaptative. La grille adaptative est générée en utilisant la structure des quadtree linéaires. Cette technique facilite le stockage et la maniabilité des mailles.

ABSTRACT. We deal with a numerical method for HJB equations coming from optimal control problems with state constraints.
More precisely, we present here an antidissipative scheme applied on an adaptative grid. The adaptative grid is generated using linear quadtree structure. This technique of adaptation facilitates stocking data and dealing with large numerical systems.

MOTS-CLÉS : problèmes de contrôle optimal, équations HJB, schéma antidissipatif, quadtree linéaire.

KEYWORDS : optimal control problems, HJB equations, antidissipative scheme, linear quadtree.

Volume 5 – 2006, pages 256 à 271 – **ARIMA**

for the approximation of discontinuous value functions. Indeed, the interpolation steps produce more or less numerical diffusion, which causes an increasing loss of precision mainly around the discontinuities. To our knowledge, the only scheme which doesn't use any interpolation technique is the one based on the viability algorithm developed by P. Saint Pierre and his co-authors [SP]. But, as already shown in [BMMZ] this scheme still diffuses.

The approximation method we study here is a mixture of the antidissipative *UltraBee* (UB) scheme [DL, BZ] and of an adaptative gridding technique. The UltraBee scheme has been studied by B. Désprès and F. Lagoutière [DL] for solving the transport equation with positive constant velocity. It has been extended by O. Bokanowski and H. Zidani [BZ] for the transport equation with a changing sign velocity and applied for the resolution of Hamilton Jacobi equations (3) on a regular grid.
In our case, the value function takes only values 0 and 1 (the value 1 coding in fact the $+\infty$ value). In this special situation, the UltraBee scheme has a nice property : it is able to localize accurately the discontinuity of v corresponding to the interface Γ_t separating the region where $v(t,.)$ takes the value 1 from the region where it takes the value 0. This property allows us to design a simple method for adaptative gridding. Moreover, the real calculations at every time $t^n = n\Delta t$ (Δt being the time step) have only to be done on a small neighborhood of the interface Γ_{t^n}. Hence adaptative gridding is particularly interesting in our case. Moreover, we use linear quadtrees which provide a good way to handle easily adaptative grids and to achieve a significant save of memory.

Adaptative gridding for solving HJB equations has already been studied in the case of a continuous value function [CY1, CY2, Gr]. In [Gr] for example, L.Grune has handled the Semi-Lagrangian scheme to solve (3) and explained the criteria he used for the refinement and coarsening steps. These criteria are based on a fixed tolerance for the interpolation error. The presence of discontinuities in our case makes these criteria no more suitable.

The paper is organized as follows. In section 2 we give the formulation of the UltraBee scheme and some of its properties. In section 3 we present the adaptative technique that we use and explain the steps of the proposed method. Finally in section 4, we give several numerical simulations in 2 dimensions coming from control problems and propagating front problems.

2. The UltraBee scheme

Notice that when we deal with only one control, the HJB equation (3) becomes a transport equation. Hence we will first present the UltraBee scheme in this simple case in one space dimension ($n = 1$).

• In the case when f is of changing sign, the *UltraBee generalized scheme* (UB-G) [BZ] is defined as follows

a) if $\nu_j > 0$, $U^{n,L}_{j+\frac{1}{2}} = \min(\max(U^n_{j+1}, b^{n,+}_j), B^{n,+}_j)$ as proposed in (7).

b) if $\nu_j < 0$, we define symetrically $U^{n,R}_{j-\frac{1}{2}} = \min(\max(U^n_{j-1}, b^{n,-}_j), B^{n,-}_j)$ with $b^{n,-}_j = \frac{1}{|\nu_j|}(U^n_j - \max(U^n_j, U^n_{j+1})) + \max(U^n_j, U^n_{j+1})$, and $B^{n,-}_j = \frac{1}{|\nu_j|}(U^n_j - \min(U^n_j, U^n_{j+1})) + \min(U^n_j, U^n_{j+1})$.

c) if $\nu_j \leq 0$ and $\nu_{j+1} \geq 0$, $U^{n,L}_{j+\frac{1}{2}} = U^n_j$, $U^{n,R}_{j+\frac{1}{2}} = U^n_{j+1}$.

d) if $\nu_j\nu_{j+1} > 0$, $U^{n,R}_{j+\frac{1}{2}} = U^{n,L}_{j+\frac{1}{2}}$ (if $\nu_j > 0$) or $U^{n,L}_{j+\frac{1}{2}} = U^{n,R}_{j+\frac{1}{2}}$ (if $\nu_{j+1} < 0$).

When the velocity is constant, and under the CFL condition,

$$|\nu_j| \leq 1 \; \forall j \in \mathbb{Z}, \tag{8}$$

one interesting property of the UltraBee scheme is an exact advection [DL, Theorem 3] for a class of step functions defined by : $\exists k^0 \in [0,1[$ such that $\forall j \in \mathbb{Z}$,

$$U^0_{3j+1} = U^0_{3j}, \quad U^0_{3j+2} = k^0 U^0_{3j+1} + (1-k^0)U^0_{3j+3}. \tag{9}$$

Exact advection means that the computed value U^n_j is the exact mean value,

$$U^n_j = \frac{1}{\Delta x}\int_{M_j} u(t^n, x)dx,$$

where u is the exact solution of the advection problem. For the convergence proofs of the UltraBee scheme, we refer to [DL, BZ].

2.2. HJB equation

Here, we are still in dimension 1. First, we consider the simple change of variable,

$$\hat{v}(t,x) = v(T-t,x), \; \forall t \in [0,T], \; \forall x \in \boldsymbol{R}.$$

Then the function $\hat{v}$ satisfies

$$\begin{cases} \hat{v}_t(t,x) - \min\limits_{a\in\mathcal{A}} f(x,a).\hat{v}_x(t,x) = 0, \; \forall (t,x) \in [0,T]\times\mathcal{K}, \\ \hat{v}(0,x) = \varphi(x), \; \forall x \in \mathcal{K}. \end{cases} \tag{10}$$

The application of UB-G to the HJB equation (10) consists, on a regular grid $\mathcal{G}$ of $\mathcal{K}$, in the following steps (UB-HJB) :

• Step 1 : We compute the discrete initial condition

$$V^0_j = \frac{1}{\Delta x}\int_{M_j} \varphi(x)dx, \; \forall j \in J := \{j \in \mathbb{Z}, \; M_j \in \mathcal{G}\}.$$

3.1. Linear quadtrees

As we deal with adaptative grids, we look for a technique that facilitates stocking and finding data relative to each cell of the grid. This technique is explained by I. Gargantini in [Ga] and uses the notion of *linear quadtree*. If we represent our final adapted grid by a tree, each cell is a leaf (final node of the tree) and the initial quadrant (all the domain $\mathcal{K}$ before adaptation) is the root of the tree. The method for stocking data using quadtrees is based on coding each leaf of the tree with a quaternary function. This code representation is implicitly the path from the root to the concerned leaf.
Every code is composed of 0, 1, 2, 3. When dividing a cell into four subcells, the NW quadrant is indexed by 0, the NE by 1, the SW by 2 and the SE by 3. The code of each subcell is the concatenation of the code of the mother cell with the index of the subcell (as shown in figure 3.1). Here cells 20, 21, 22 and 23 are sisters and 2 is the mother cell.

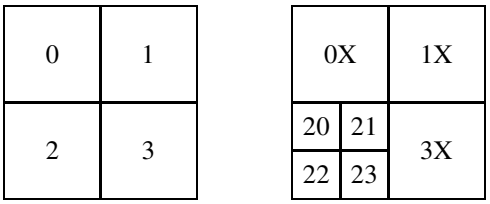


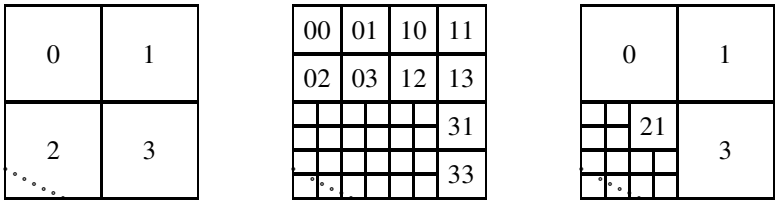
Figure 1. *Refinement of a cell by quadtrees*

Notice that when coding the grid using a tree, all intermediate cells have to be memorized, for example we memorize cells 2, 20, 21, 22, 23. However, in a linear quadtree, we have to stock only final cells of the grid, i.e. cells 20, 21, 22, 23. Then, to find the intermediate cells, we have just to truncate the codes. Furthermore the use of linear quadtrees allows to manage efficiently the adapted grid. In fact, thanks to fast algorithms, operations like encoding a cell into its quaternary code and finding adjacencies of a cell are run in logarithmic time [Ga].

3.2. Algorithm of the method

Our contribution consists in finding a suitable criterion to adapt the computational domain. This criterion must be compatible with the fact that we deal with mean values on each cell and that our value function is discontinuous. Let $L_{\max}$ be a fixed integer that corresponds to the maximal level of refinement. We set $\Delta X^1_{\min} = \frac{|X^1_{\max} - X^1_{\min}|}{2^{L_{\max}}}$ and $\Delta X^2_{\min} = \frac{|X^2_{\max} - X^2_{\min}|}{2^{L_{\max}}}$. Then $(\Delta X^1_{\min}, \Delta X^2_{\min})$ is the minimal cell size. The maximal level $L_{\max}$ is chosen such that the following CFL condition holds :

$$\max(|\frac{f_1(x_j, a_i)\Delta t}{\Delta X^1_{\min}}|, |\frac{f_2(x_j, a_i)\Delta t}{\Delta X^2_{\min}}|) \leq 1, \ \forall j \in J, \ \forall i = 1, .., N_a. \qquad (12)$$



$\mathcal{G}^{0,1}$ $\mathcal{G}^{0,3}$ $\mathcal{G}^{0}$

Figure 2. *Construction of $\mathcal{G}^0$, the discontinuity is plotted with dots.*

Step 2 : UB-HJB computation and construction of $\mathcal{G}^{n+1}$.

– Step 2.0 : Do an iteration of UB-HJB scheme on the cells of minimal size $(\Delta X^1_{\min}, \Delta X^2_{\min})$ of $\mathcal{G}^n$. We obtain V^{n+1} on $\mathcal{G}^n$.

– Step 2.1 : Define $\mathcal{G}^{n+1,0} := \mathcal{G}^n$, and $V^{n+1,0} := V^{n+1}$ on $\mathcal{G}^n$.

– Step 2.2 : For $1 \leq l \leq L_{\max} - 1$, for all cells $M_j \in \mathcal{G}^{n+1,l} \cap \mathcal{G}_l$, compare the value[1] $V_j^{n+1,l}$ on M_j with its neighboring values. If the values are different, then refine cell M_j, attribute to the daugther cells of M_j the same value $V_j^{n+1,l}$. This defines a new grid $\mathcal{G}^{n+1,l+1}$. Set $l = l + 1$, and go to Step 2.2.
Otherwise, set $l = L_{max}$ and go to step 2.3.

– Step 2.3 : Set $\widetilde{\mathcal{G}}^{n+1,L_{\max}} := \mathcal{G}^{n+1,L_{\max}}$ and $\widetilde{V}^{n+1,L_{\max}} := V^{n+1,L_{\max}}$.

– Step 2.4 : For $l = L_{\max}, \cdots, 2$, do a corsening step following the same idea as in Step 1.4. If there is no coarsening to do, then set $l = 1$, and go to Step 2.5.

– Step 2.5 : Set $\mathcal{G}^{n+1} := \widetilde{\mathcal{G}}^{n+1,1}$, and $V^{n+1} := \widetilde{V}^{n+1,1}$. This corresponds to the approximation on $\mathcal{G}^{n+1}$ of the solution $\hat{v}$ of (10) at $t = t^{n+1}$.

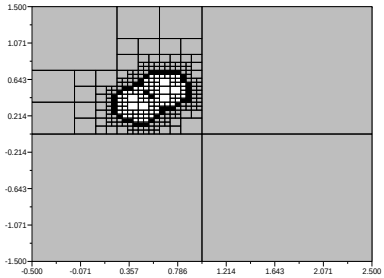
By construction, we have the following equivalence result :
Theorem : Let $L_{\max}$ be a fixed integer. Under the CFL condition (12), the approximation of (10) using the UB-HJB scheme on an adaptative grid gives the same numerical solution as the resolution using the UB-HJB scheme on a regular grid $\mathcal{G}_{L_{\max}}$.

4. Numerical simulations

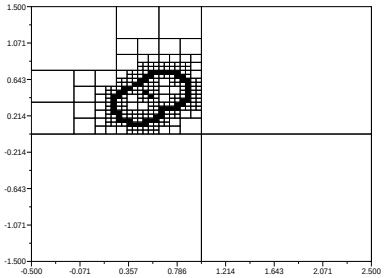
In the graphics through all this section, we use the black color for cells with mean value strictly between 0 and 1, white for cells with value 0 and light gray for cells with value 1. We also use the notation $\mathcal{B}(c_0, r)$ for the ball centered in c_0 and with radius r.

Example 1 : A propagating front problem
We first start with a propagating fronts problem. The initial condition is here two sources

1. Recall that for every $M_j \in \mathcal{G}^{n+1,l} \cap \mathcal{G}_l$, with $l < L_{\max}$, the mean value $V_j^{n+1,l}$ is equal to 0 or 1.

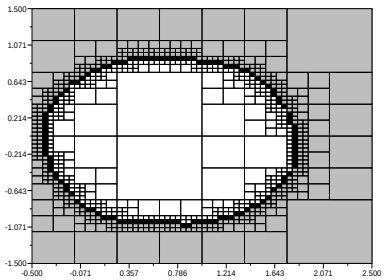


(a) computed solution

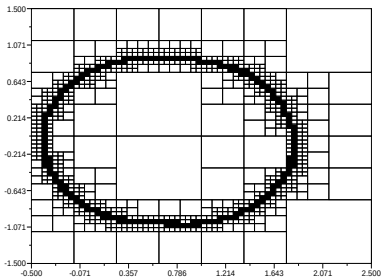


(b) error $(\varepsilon_j^n)_{j\in J}$

Figure 3. *Computed solution and error at T=0.11, # cells=244,* $L_{max} = 6$.



(a) computed solution



(b) error $(\varepsilon_j^n)_{j\in J}$

Figure 4. *Computed solution and error at T=0.87, # cells=817,* $L_{max} = 6$.

Hence when we adapt the grid we improve the precision 3 times without spending any additional memory cost. For refinement levels bigger than 10, it is no more possible to handle calculations on a regular grid. Hence we can not have better precision on a regular grid : this reflects the gain of precision achieved by the use of the adaptative algorithm.

Example 2 : A capture basin problem (Zermelo problem)
Let $\mathcal{K} := [-6, 2] \times [-2, 2]$ and $\mathcal{C} := \mathcal{B}(c_0, r)$ with $c_0 = (0, 0)$ and $r = 0.44$. We define the dynamics $f : \boldsymbol{R}^2 \times \mathcal{A} \to \boldsymbol{R}^2$,

$$f(x, a, \theta) = (1 - \beta x_2^2 + a\cos(\theta), a\sin(\theta)),$$

where the constant $\beta = 0.1$, and $\mathcal{A}$ denotes the set $[0, 0.44] \times [0, 2\pi[$.

Our aim is to approximate the capture basin of $\mathcal{C}$ which is the subset of initial states $x \in \mathcal{K}$ for which exists an admissible control $(a, \theta) \in L^\infty([0, +\infty[; \mathcal{A})$ and a finite time