Journal of Global Optimization **27:** 177–192, 2003.
© 2003 *Kluwer Academic Publishers. Printed in the Netherlands.*

Numerical Solution of Hamilton-Jacobi-Bellman Equations by an Upwind Finite Volume Method

S. WANG[1], L.S. JENNINGS[1] and K.L. TEO[2]
[1]*Centre for Applied Dynamics and Optimization, Department of Mathematics and Statistics, The University of Western Australia, 35 Stirling Highway, Crawley, WA 6009, Australia.*
(E-mail: swang@maths.uwa.edu.au)
[2]*Department of Applied Mathematics, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*

Abstract. In this paper we present a finite volume method for solving Hamilton-Jacobi-Bellman(HJB) equations governing a class of optimal feedback control problems. This method is based on a finite volume discretization in state space coupled with an upwind finite difference technique, and on an implicit backward Euler finite differencing in time, which is absolutely stable. It is shown that the system matrix of the resulting discrete equation is an M-matrix. To show the effectiveness of this approach, numerical experiments on test problems with up to three states and two control variables were performed. The numerical results show that the method yields accurate approximate solutions to both the control and the state variables.

Key words: Optimal feedback control; Hamilton-Jacobi-Bellman equation; finite volume method; Viscosity solution; upwind finite difference.

1. Introduction

Consider the optimal control problem of the form

$$\min_{u\in\mathcal{U}} \quad J(y,s,u) = \int_s^T L(t,x(t),u(t))dt + h(x(T)), \tag{1.1a}$$

$$\text{subjectto} \begin{cases} \dot{x}(t) = f(t,x(t),u(t)), & \text{a.e.} t \in (s,T] \\ x(s) = y, \end{cases} \tag{1.1b}$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ are, respectively, the state and the control, $f : (0,T) \times \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}^n$ is a vector valued function, $(s,y) \in [0,T) \times \mathbb{R}^n$, and $\mathcal{U} \subset \mathbb{R}^m$ is the set of admissible controls. When the initial time $s = 0$ and the initial state y is fixed, the above problem can be solved easily as an open-loop optimal control problem. However, this solution is neither robust nor stable with respect to perturbations in the state x. This is because the solution to the open-loop problem provides an optimal control only along the optimal trajectory x starting from the initial state y. Therefore, a feedback optimal control solution which is defined in a region containing the optimal trajectory is much preferred in practice. This feedback solution gives a 'global' optimal control defined over a time-space region so that if the state is away from the optimal trajectory due to disturbance, a corresponding optimal control can be found for the system. To achieve this, we assume that s and y in (1.1) are not fixed and introduce the value function v defined by

$$v(y, s) = \inf_{u \in \mathcal{U}} J(y, s, u). \tag{1.2}$$

Using the dynamic programming approach, the problem (1.1) can be transformed into the following Hamilton-Jacobi-Bellman equation

$$-\frac{\partial v}{\partial t} + \sup_{u \in \mathcal{U}} \left[-\nabla v \cdot f(t, x, u) - L(t, x, u) \right] = 0, \quad (t, x) \in [0, T) \times \mathbb{R}^n \tag{1.3}$$

with the initial condition

$$v(T, x) = h(x) \tag{1.4}$$

where ∇ denotes the gradient operator with respect to x.

Equation (1.3), together with (1.4) is called an initial value problem. There are two unknown functions in this equation – the value function v and the optimal control u. Thus, finding a feedback control law for (1.1) is equivalent to solving the above initial value problem. Although this initial value problem is defined on the unbounded region $\mathbb{R}^n$, we often restrict our consideration to a bounded region

$$\Omega = (a_1, b_1) \times (a_2, b_2) \times \cdots \times (a_n, b_n) \tag{1.5}$$

where a_i and b_i are constants for $i = 1, 2, ..., n$. This initial value problem with $\mathbb{R}^n$ in (1.3) being replaced by Ω in (1.5) has been discussed in the literature for many years. However, many of these discussions were on its theoretical side such as solvability and smoothness of solutions to (1.3)–(1.4) (cf. [3, 11, 4, 6]). On the other hand, in most practical situations, (1.3)–(1.4) is not analytically solvable, and thus efficient numerical techniques are needed for solving this initial value problem. There are two types of numerical algorithms in the open literature for solving this equation. The first type is based on the discretization of (1.1b) by a finite difference scheme and the optimization of the discrete version of (1.2) (cf.[7, 8, 2]), and the second type to solve the HJB equation (1.3) and (1.4) by a discretization scheme (cf., for example, [5, 10, 13, 9]).

In the present paper we shall present a method for the solution of the HJB equation. Since it is a pure initial value problem, explicit time stepping schemes coupled with stable spatial discretization methods are often used. A typical example is the approach based the forward Euler with an upwind finite difference discussed in [13]. Although the numerical results in [13] are promising, the explicit nature of the method requires that the stability condition on the step lengths in x and t are satisfied, and that the solution region is extended to a trapezoidal region due to the propagation of the scheme. In the extended region, the number of mesh nodes along each component of x is equal to twice the number of time steps needed. Therefore, the computational cost for the scheme will be high for high dimensional problems, since the number of partitions in the time direction is normally large due to the stability condition. An alternative approach is proposed in [9]. In this approach, (1.3) is first perturbed as a second order parabolic equation by adding the diffusion term with a small diffusion coefficient ε. This is known as the viscosity approach in the literature. Then, some artificial Dirichlet/Neumann boundary conditions are defined for the resulting second order system on the boundary of an extended solution region. Finally, a modified characteristic method is applied to the resulting equation.

Following the viscosity and the artificial boundary condition approach, we propose, in this paper, a method based on a finite volume method with an upwind technique and an implicit time-stepping method for the solution of the singularly perturbed equation corresponding to (1.3)–(1.4). This method has the merit that it is absolutely stable because of the implicit nature of the time discretization. At each time step, the discretization method yields a linear system with a positive-definite M-matrix. Thus, it is also stable in space in the case that the singular perturbation parameter $\varepsilon << 1$. Though this approach still needs an extension of the solution region, the number of mesh nodes does not depend on the number of time steps. The rest of the paper is organized as follows. In the next section we will first discuss the viscosity approach to (1.3) to transform it into a convection-diffusion equation. We shall then define appropriate artificial boundary conditions for the resulting equation. In Section 3, we will discuss the space and time discretization of the resulting convection-diffusion equations. Numerical results will be presented in Section 4 to demonstrate the usefulness of this approach.

2. The singularly perturbed problem

In this section, we transform the initial value problem (1.3) and (1.4) into a singularly perturbed convection-diffusion equation. To achieve this we consider the following initial value problem:

$$-\frac{\partial v}{\partial t} - \varepsilon \nabla^2 v + \sup_{u \in \mathcal{U}} \left[-\nabla v \cdot f(t, x, u) - L(t, x, u) \right] = 0, \quad (t, x) \in [0, T) \times \Omega, \tag{2.1a}$$

$$v(T, x) = h(x), \quad x \in \Omega, \tag{2.1b}$$

where Ω is defined in (1.5). This problem differs from (1.3) by the diffusion term $-\varepsilon \nabla^2 v$, where $\varepsilon > 0$ represents a singular perturbation parameter, or physically the diffusion coefficient or viscosity. When $\varepsilon << 1$, (2.1) is called a viscosity approximation to (1.3), which is often used in stochastic control (cf., for example, [1]). The convergence of solutions of (2.1) to solutions of (1.3) as $\varepsilon \to 0^+$ has been proved. For details see [1] and the references therein.

The perturbed system (2.1) is still a pure initial value (Cauchy) problem and can be solved only by explicit time stepping schemes which are only conditionally stable. In order to solve it by unconditionally stable implicit methods, we impose suitable boundary conditions for (2.1). Theoretically, the boundary conditions of the value function $v(x,t)$ defined in (1.2) can be obtained by the interpolant of the numerical solutions of a sequence of open-loop optimal control problems at some chosen points on $\partial\Omega$. However, this is computationally expensive in high dimensions. Alternatively, we introduce artificial boundary conditions. To achieve this, we use the heuristic approach used in [9], i.e, we extend Ω defined in (1.5) to a larger region $\tilde{\Omega} = (\tilde{a}_1, \tilde{b}_1) \times (\tilde{a}_2, \tilde{b}_2) \times \cdots \times (\tilde{a}_n, \tilde{b}_n)$ with constants $\tilde{a}_i$ and $\tilde{b}_i$ satisfying $\tilde{a}_i < a_i$ and $\tilde{b}_i < b_i$ for $i = 1, 2, ..., n$. On this new solution region, we reformulate (2.1) at time equal to T for all t, and the problem can be reformulated as

$$-\frac{\partial v}{\partial t} - \varepsilon\nabla^2 v + \sup_{u\in\mathcal{U}}\big[-\nabla v \cdot f(t,x,u) - L(t,x,u)\big] = 0, \qquad (2.2a)$$

$$(t,x) \in [0,T) \times \tilde{\Omega},$$

$$v(T,x) = h(x), \quad x \in \tilde{\Omega}, \qquad (2.2b)$$

$$v(t,x) = g(t,x), \quad (t,x) \in [0,T) \times \partial\tilde{\Omega}, \qquad (2.2c)$$

where $g(x)$ is an artificial Dirichlet boundary condition. A natural choice is that $g(t,x) = h(x)$. This corresponds to the case that we set $L(t,x,u) = 0$ in (1.1) when $x \in \tilde{\Omega}$. We may also use other boundary conditions. For example, we may set $L(t,x,u)$ to a large value on the boundary to penalize the move of x to leave $\tilde{\Omega}$. Note that the extension of the original region Ω is necessary. This is because the boundary conditions defined above are not exact, and thus the extended part of $\tilde{\Omega}$ is used as a transition region from the solution satisfying the artificial boundary conditions to the solution of (2.1) for small ε. That is, we expect that the solution to (2.2) converges to that of (2.1) in the region Ω as $\varepsilon \to 0+$. The region $\tilde{\Omega} \setminus \Omega$ contains the transition layers. We comment that though this observation is heuristic, it works well in practice, as demonstrated by the numerical examples in Section 4.

Before discussing the discretization of the problem, we first split (2.2) into two equations,

$$-\frac{\partial v}{\partial t} - \varepsilon\nabla^2 v - \nabla v \cdot f(t,x,u^*) - L(t,x,u^*) = 0, \qquad (2.3a)$$

$$u^* = \arg\sup_{u\in\mathcal{U}}\big[-\nabla v \cdot f(t,x,u) - L(t,x,u)\big]. \qquad (2.3b)$$

The first equation is a convection-diffusion equation. When $\varepsilon << 1$, the equation is called convection-dominated, and solutions to this equation normally contain sharp

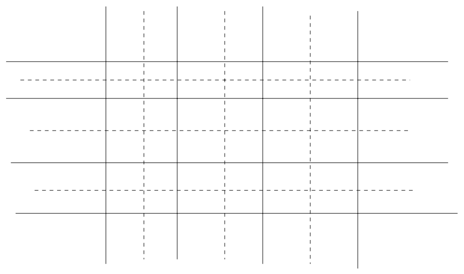

Figure 3.1. 2D rectangular mesh and its Dirichlet tessellation

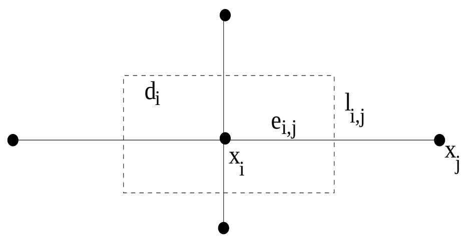

Figure 3.2. A local structure of a 2D mesh.

layers so that conventional numerical methods yield solutions with spurious oscillations. In the next section we propose a first order discretization method which is stable in both time and space.

3. The discretization method

Let us now consider the discretization of (2.3). We start with the discretization of (2.3a)

SPATIAL DISCRETIZATION

Let $\tilde{\Omega}$ be partitioned into a hyper-brick mesh with N (non-Dirichlet) mesh nodes and M mesh edges. Dual to this mesh, we construct a second mesh, called Dirichlet Tessellation (or Voronoi hyper-polyhedra), by connecting all the circumcentres of the neighbouring hyper-bricks. A typical case in two dimensions is shown in Figure 3.1.

Consider a mesh node x_i, its neighouring nodes and the Dirichlet tile associated with x_i. These form a local structure and a two-dimensional example for this is depicted in Figure 3.2. Integrating (2.3) over d_i and using integration by parts,

$$-\int_{d_i} \frac{\partial v}{\partial t} - \varepsilon \int_{\partial d_i} \nabla v \cdot \nu ds + \int_{d_i} \nabla v \cdot f = \int_{d_i} L dx$$

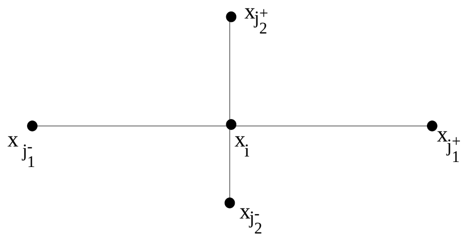

Figure 3.3. Alternative notation for the neighbouring nodes of x_i

where ν denotes the normal direction. Applying the one-point quadrature rule to the first and last term of the above yields

$$-\frac{\partial v_i}{\partial t}|d_i| - \varepsilon \int_{\partial d_i} \nabla v \cdot \nu ds + \int_{d_i} \nabla v \cdot f = L_i |d_i|, \tag{3.1}$$

where v_i and L_i denote respectively the nodal approximations of v and L at x_i and $u^*(t, x^*)$, and $|\cdot|$ denotes the measure (i.e. length, area etc depending on the number of dimensions). Let I_i be the set of indices of neighouring mesh nodes of x_i. From the construction of the meshes we see that ∂b_i consists of a finite number of facets in $(n-1)$-dimensions and each of these facets is a bi-sector of and perpendicular to one of the edges connecting x_i and the nodes with indices in I_i. Therefore, replacing the normal derivatives $\nabla v \cdot \nu$ by the corresponding finite differences we have

$$-\varepsilon \int_{\partial d_i} \nabla v \cdot \nu ds \approx -\varepsilon \sum_{j \in I_i} \frac{v_j - v_i}{|e_{i,j}|} |l_{i,j}| \tag{3.2}$$

where $e_{i,j}$ denotes the edge connecting x_i and x_j and $l_{i,j}$ the facet of b_i bisecting $e_{i,j}$. The 2D case is demonstrated in Figure 3.3.

For x_i, we let j_m^+ and j_m^- denote the indices in I_i such that the vectors $x_i - x_{j_m^-}$ and $x_{j_m^+} - x_i$ are in the same direction as that of the i-th coordinate axis. A 2D example is depicted in Figure 3.3. We now use the following upwind technique to discretize the third term in (3.1): The m-th term is approximated by

$$\begin{aligned}\int_{d_i} \frac{\partial v}{\partial x_m} f_m dx &\approx \begin{cases} \dfrac{v_{j_m^+} - v_i}{|e_{i,j_m^+}|}\sigma_{im}, & \sigma_{im} > 0 \\ \dfrac{v_i - v_{j_m^-}}{|e_{j_m^-,i}|}\sigma_{im}, & \sigma_{im} < 0 \end{cases} \\ &= \left[\frac{1 - \operatorname{sgn}(\sigma_{im})}{2} \frac{v_i - v_{j_m^-}}{|e_{j_m^-,i}|} + \frac{1 + \operatorname{sgn}(\sigma_{im})}{2} \frac{v_{j_m^+} - v_i}{|e_{j_1^+,i}|}\right] \sigma_{im}\end{aligned}$$

where sgn denotes the sign function and $\sigma_{im} = \int_{d_i} f_m(t, x, u^*)dx$ for $m = 1,2,...,n$. Substituting (3.2) and the above into (3.1) we obtain

$$\begin{aligned} & -\frac{\partial v_i}{\partial t}|d_i| + \varepsilon \sum_{j\in I_i} \frac{v_i - v_j}{|e_{i,j}|}|l_{i,j}| \\ & + \sum_{m=1}^{n} \left[\frac{1-\operatorname{sgn}(\sigma_{im})}{2} \frac{v_i - v_{j_m^-}}{|e_{j_m^-,i}|} + \frac{1+\operatorname{sgn}(\sigma_{im})}{2} \frac{v_{j_m^+} - v_i}{|e_{j_m^+,i}|} \right] \sigma_{im} \\ & = L_i |d_i| \end{aligned} \tag{3.3}$$

for $i = 1, 2, ..., N$, or in matrix form

$$-\frac{\partial v_i(t)}{\partial t}|d_i| + A_i(t)\boldsymbol{v}(t) = b_i(t) \tag{3.4}$$

for $i = 1, 2, ..., N$, where $A_i = (a_{i1}, a_{i2}, ..., a_{iN})$ and $\boldsymbol{v} = (v_1, v_2, ..., v_N)$. From the construction of the scheme we see that the number of nonzero elements in A_i is equal to one plus the number of elements in I_i. Also, it is easily seen from (3.3) that

$$\begin{aligned} a_{ii} &= \sum_{j\in I_i} \frac{|l_{i,j}|}{|e_{i,j}|} + \sum_{m=1}^{n} |\sigma_{im}| \left[\frac{1-\operatorname{sgn}(\sigma_{im})}{2|e_{j_m^-,i}|} + \frac{1+\operatorname{sgn}(\sigma_{im})}{2|e_{j_m^+,i}|} \right] \\ a_{ij} &= \begin{cases} -\dfrac{|l_{i,j}|}{|e_{i,j}|}, & \sigma_{ij}\left(\sum_{m=1}^{n}(x_j - x_i)\cdot p_m\right) \geqslant 0 \\ -\dfrac{|l_{i,j}|}{|e_{i,j}|} - \dfrac{|\sigma_{ij}|}{|e_{i,j}|}, & \sigma_{ij}\left(\sum_{m=1}^{n}(x_j - x_i)\cdot p_m\right) < 0 \end{cases}, \qquad j \in I_i \\ a_{ij} &= 0 \qquad j \neq i,\ j \notin I_i, \end{aligned}$$

where p_m denotes the unit vector along the coordinate axis x_m. Obviously, from the above we see that A_i satisfies

$$a_{ii} > 0, \qquad a_{ij} \leqslant 0, \quad \text{and} \quad a_{ii} \geqslant \sum_{\substack{j = 1, \dots, N, \\ j \neq i,}} |a_{ij}|. \tag{3.5}$$

The last inequality becomes strict for some i when one of the neighbouring nodes of i is a boundary node. These ensure that the matrix $A := (a_{ij})$ is an M-matrix, i.e., it is irreducibly diagonally dominant with $a_{ii} > 0$ and $a_{ij} \leqslant 0$ for all $j \neq i$ (cf.[16]). Therefore, the maximum principle is preserved by this semi-discrete system, and the spatial discretization is said to be monotone.

TIME DISCRETIZATION

We now consider the time discretization of (3.3). Let $T = t_0 > t_1 > \cdots > t_K = 0$, and $\Delta t_k = t_k - t_{k-1} < 0$, where $K > 1$ is a positive integer. There are several implicit schemes we can use. For example, the first-order Backward Euler method and the second-order Crank-Nicholson method. Both of these are unconditionally stable. For discussion simplicity, we apply the Backward Euler's scheme to (3.3), yielding

$$-\frac{v_i^{k-1} - v_i^k}{-\Delta t_k} + A_i^k v_i^k = b_i^k,$$

or

$$\frac{v_i^k}{|\Delta t_k|} + A_i^k v_i^k = b_i^k + \frac{v_i^{k-1}}{|\Delta t_k|}$$

for $i = 1, 2, ..., N$ and $k = 1, 2, ..., K$, where A_i^k is the matrix $A_i(t)$ in (3.4) evaluated at $t = t_k$. In matrix form this is

$$(A^k + D^k)\boldsymbol{v}^k = \boldsymbol{b}^k + D^k \boldsymbol{v}^{k-1}. \tag{3.6}$$

for $k = 1, 2, ..., K$, where $\boldsymbol{v}^k = (v_1^k, ..., v_N^k)$, $A^k = (A_i^k)$, and D^k is a diagonal matrix with positive diagonal entries. The system matrix of (3.6) is not symmetric, but its non-zero elements have a symmetric pattern (i.e., if $a_{ij} \neq 0$ then $a_{ji} \neq 0$). From (3.5) we see that $A^k + D^k$ is an M-matrix of order N, and so the full discretized equation (3.6) satisfies the (discrete) maximum principle, or the discretization scheme is monotone. Moreover, the facts that $A^k + D^k$ is an M-matrix and that it is sparse allow us to solve (3.6) by some efficient preconditioned conjugate gradient type algorithms such as the CGS ([12]) and BiGCSTAB ([15]).

DECOUPLING OF THE SYSTEM

In the above discretization, we have assume that the control u^* is known, and thus the system becomes a convection-diffusion equation. However, u^* is coupled with v through the equations (2.3). Let ∇_h denote a difference operator approximating the gradient operator ∇. This can be the forward, backward or central difference operator. Then, the coupled system can be decoupled by the following algorithm in which we drop the superscript *.

Algorithm A:

1. Evaluate $v_i^0 (i = 1, 2, ..., N)$ using the given terminal condition and

$$u_i^0 = \arg\sup_{u \in \mathcal{U}} [-\nabla h(x_i) \cdot f(T, x_i, u) - L(T, x_i, u)].$$

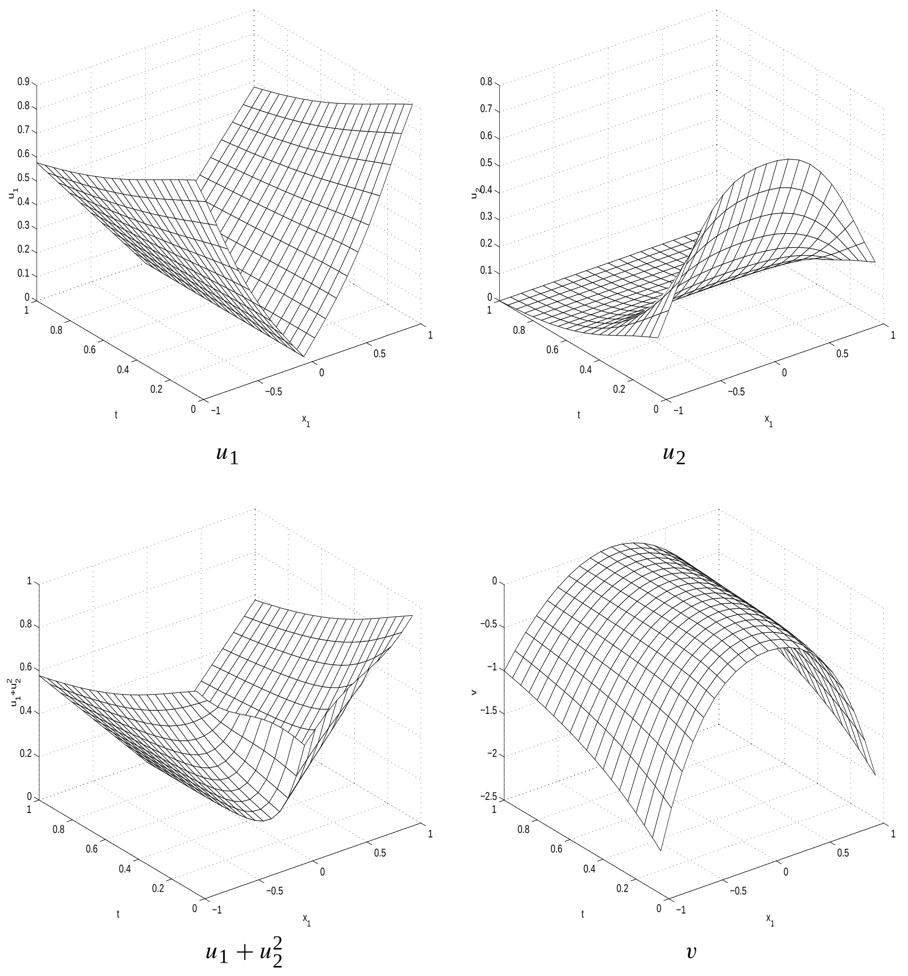

Figure 4.1. Computed $u_1, u_2, u_1 + u_2^2$ and v at $x_2 = 0$ for Test 2.

2. For $k = 1, 2, ..., K$,
   (a) Find u_i^k such that

$$u_i^k = \arg\sup_{u \in \mathcal{U}} [-\nabla_h v_i^k \cdot f(t_k, x_i, u) - L(t_k, x_i, u)]$$

   for $i = 1, 2, ..., N$.
   (b) Solve $(A^k + D^k)\boldsymbol{v}^k = \boldsymbol{b}^k + D^k \boldsymbol{v}^{k-1}$ for $\boldsymbol{v}^k$.

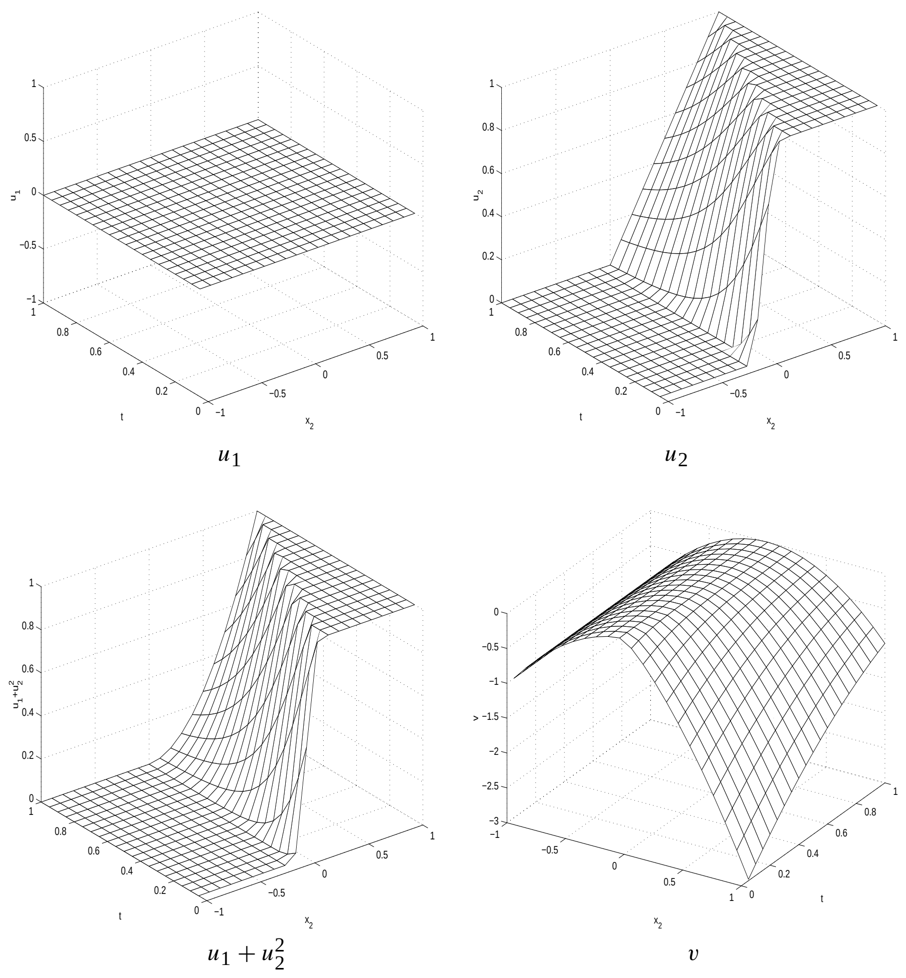

Figure 4.2. Computed $u_1, u_2, u_1 + u_2^2$ and v at $x_1 = 0$ for Test 2.

4. Numerical experiments

To verify the efficiency of the method discussed in the previous sections, some numerical experiments are carried out. For all the test problems below, the viscosity/diffusion coefficient is chosen to be $\varepsilon = 10^{-10}$, $T = 1$ and the region of interest is assumed to be $\Omega = (-1, 1)^n$. In each of the test problems, the extended solution region is chosen to be $\tilde{\Omega} = [-3, 3]^n$ and the homogeneous artificial Dirichlet boundary condition is defined on $\partial\tilde{\Omega}$. In the test problems 2 and 3, the constrained nonlinear optimization problems corresponding to those in Algorithm A are nontrivial, and thus we use the public domain package FFSQP ([14]) to solve these problems. All the computations were performed in Fortran 77 double precision on a Pentium PC under the Linux environment.

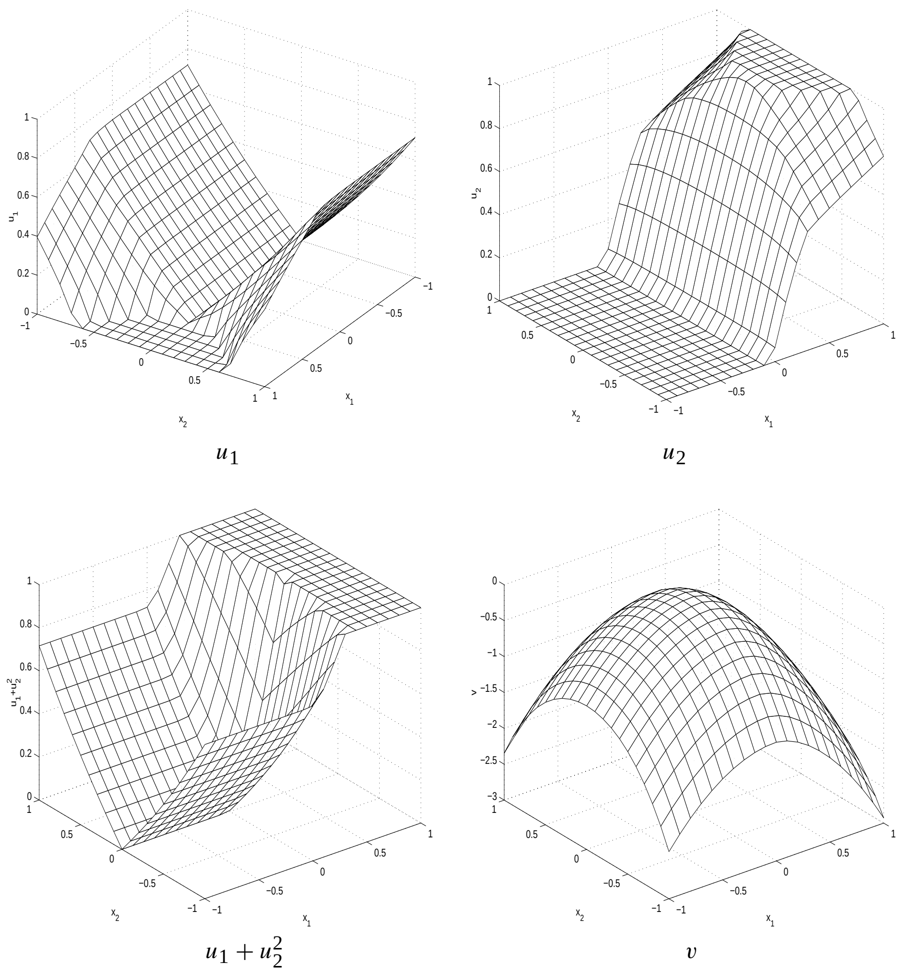

Figure 4.3. Computed $u_1, u_2, u_1 + u_2^2$ and v at $t = 0.5$ for Test 2.

Test 1: Consider the following optimal control problem

$$\begin{aligned} &\text{minimise} && -x^2(1), \\ &\text{subjectto} && \begin{cases} \dot{x}(t) = u(t), & \text{a.e.} t \in [s, 1] \\ x(s) = y, \end{cases} \\ &\text{control} && u(\cdot) : [0, 1] \mapsto \{r \in \mathbb{R} : -1 \leqslant r \leqslant 1\}. \end{aligned}$$

Table 4.I. Convergence history for Test 1.

Mesh	$20 \times 20 \times 10$	$40 \times 40 \times 20$	$80 \times 80 \times 40$	$160 \times 160 \times 80$
Max Error	1.1873	0.3580	0.0875	0.0437

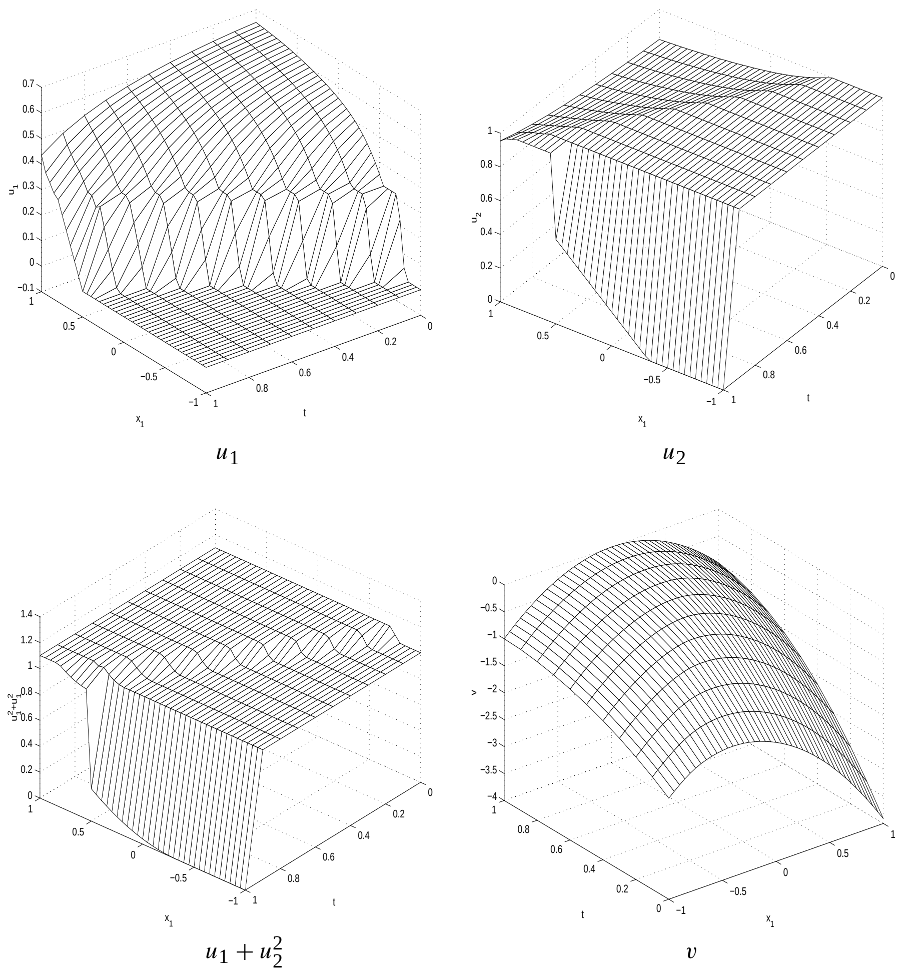

Figure 4.4. Computed u_1, u_2, $u_1^2 + u_2^2$ and v at $x_2 = x_3 = 0$ for Test 3.

The corresponding HJB initial value problem is

$$-v_t + \sup_{-1\leqslant u\leqslant 1} [-uv_x] = 0,$$

$$v(1, x) = -x^2$$

with the exact value function

$$v(t, x) = \begin{cases} -[x + (1 - t)]^2, & \text{if } x \geqslant 0, \\ -[x - (1 - t)]^2, & \text{if } x < 0 \end{cases}$$

$$= -\big[|x| + (1 - t)\big]^2.$$

This is a 1D problem, but we solve it in two dimensions to check our code. The convergence history corresponding to various uniform meshes is shown in the

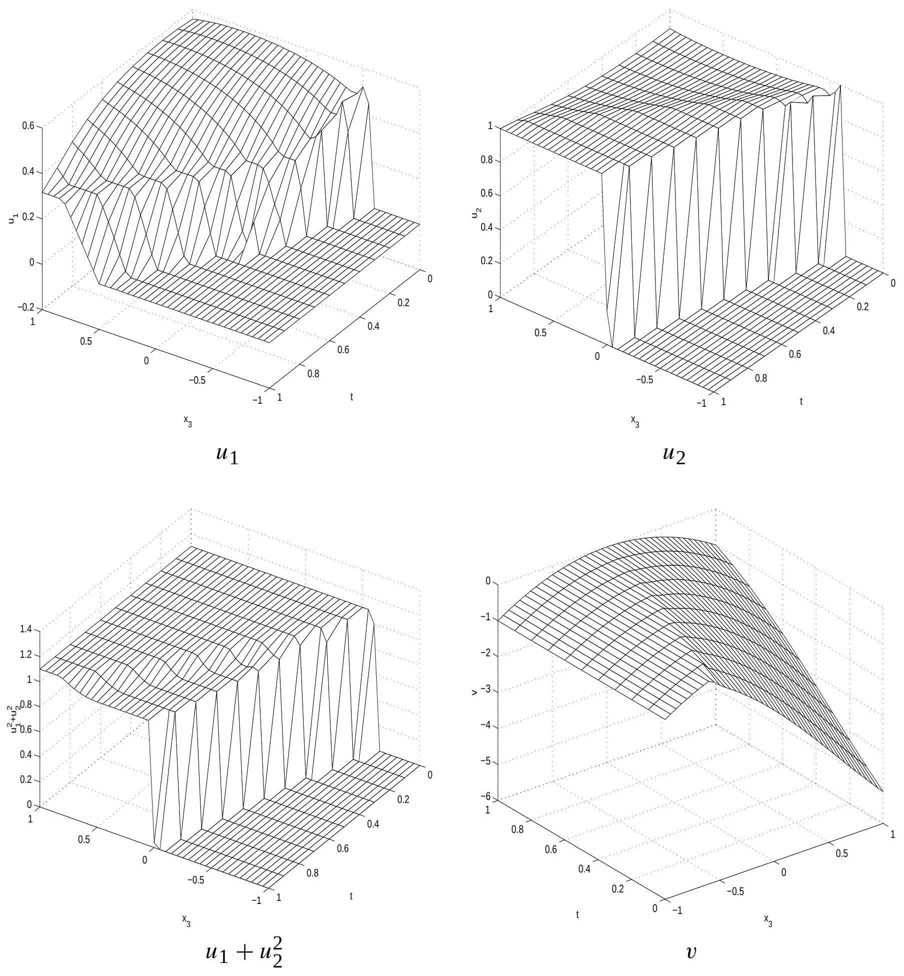

Figure 4.5. Computed $u_1, u_2, u_1^2 + u_2^2$ and v at $x_1 = x_2 = 0$ for Test 3.

Table 4.I. From the table it is seen that the method has a convergence rate of at least first order.

Test 2: Consider the following two-dimensional problem with two control variables.

$$-v_t + \sup_{\substack{0 \leqslant u_1 \leqslant 1, \\ 0 \leqslant u_2 \leqslant 1, \\ u_1 + u_2^2 \leqslant 1.}} \Big[-\frac{1}{2} v_{x_1} u_1 - v_{x_2} u_2 - u_1^3 - u_2^2 \Big] = 0,$$

where $(t, x_1, x_2) \in [0, 1) \times [-1, 1]^2$ and $v(1, x_1, x_2) = -x_1^2 - x_2^2$. To solve this problem we choose the 61×61 uniform mesh in space and 20 time steps. To

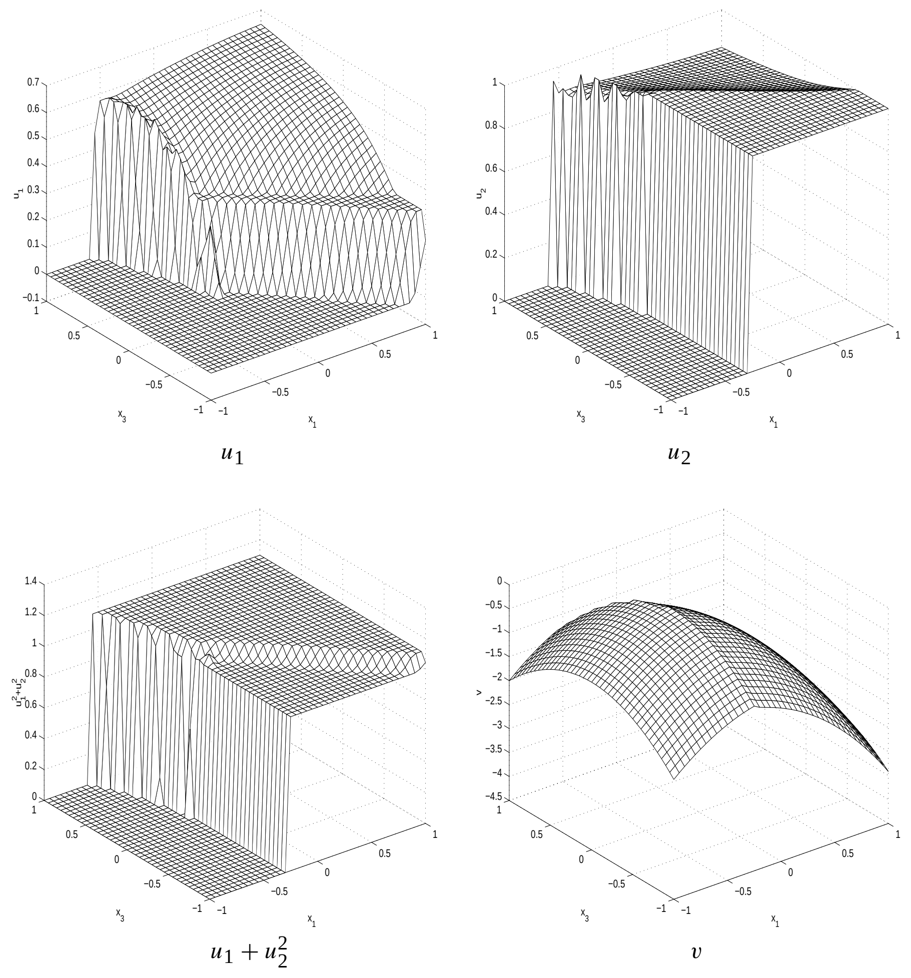

Figure 4.6. Computed $u_1, u_2, u_1^2 + u_2^2$ and v at $x_2 = 0, t = 0.5$ for Test 3.

visualize the solution, we plot in Figures 4.1–4.3 the quantities u_1, u_2, v and $u_1+u_2^2$ at three different cross-sections. From these plots we see that the value function is not smooth and the nonlinear constraint is always satisfied.

Test 3: Consider the following HJB equation

$$-v_t + \sup_{\substack{0 \leqslant u_1 \leqslant 1,\\ 0 \leqslant u_2 \leqslant 1,\\ u_1^2 + u_2^2 \leqslant 1.1}} \left[-u_1 u_2 v_{x_1} - (u_2 - u_1) v_{x_2} - (u_1 + u_2) v_{x_3} - u_1(u_1 + u_2)\right] = 0,$$

where $(t, x_1, x_2, x_3) \in [0, 1) \times [-1, 1]^3$ and $v(1, x_1, x_2, x_3) = -x_1^2 - x_3^2$. This is a problem with three state and two control variables. To our best knowledge, no

3D HJB equations have been solved previously by other numerical methods. To solve this problem we choose 20 time steps and the non-uniform mesh with $61 \times 61 \times 61$ mesh nodes in space obtained as follows: in each axis, the interval $[-1, 1]$ is divided uniformly into 30 uniform subintervals and subregion $[-3, -1] \cup [1, 3]$ is divided uniformed into 20 subintervals. To visualize the numerical result, we plot u_1, u_2, $u_1^2 + u_2^3$ and v at different across-sections in Figures 4.4–4.6. The exact solution to this problem is not known. But from these figure we see, again, that the value function is not smooth and that the numerical controls satisfy all the constraints.

5. Conclusions

In this paper we discussed the upwind finite volume method coupled with the backward Euler scheme for the numerical solution of singularly perturbed HJB equations. It was shown that the system matrix of the resulting linear system is an M-matrix so that the maximum principle is preserved by the discrete system. Non-trivial examples with up to 3 state and 2 control variables and with non-linear constraints on the control were solved to demonstrate the effectiveness of the method. The numerical results showed that the method is efficient for solving practical HJB equations.

Acknowledgements

The authors gratefully acknowledge the support of the Australian Research Council.

References

1. M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman equations*, Birlhäuser, Boston-Basel-Berlin (1997).
2. M. Bardi, M. Falcone, P. Soravia. Numerical methods for pursuit-evasion games via viscosity solutions, in *Stochastic and differential games: theory and numerical methods* eds. M. Bardi, T. Parthasarathy, T.E.S. Raghavan, Birkhäuser, Boston (1999) 105–175.
3. M.G. Crandall and P.L. Lions. Viscosity solutions of Hamilton-Jacobi equations. *Trans. AMS*, **277**, No.1 (1983) 1–42.
4. M.G. Crandall, L.C. Evans and P.L. Lions. Some properties of viscosity solutions of Hamilton-Jacobi equations, *Trans. AMS*, **282**, No.2 (1984) 487–502.
5. M.G. Crandall and P.L. Lions, Two approximations of solutions of Hamilton-Jacobi equations. *Math. Comp.*, **43** (1984) 1–19.
6. M.G. Crandall, H. Ishii and P.L. Lions. User’s guide to viscosity solutions of second order partial differential equations. *Bull. Amer. Math. Soc.*, **27** (1992) 1–67.
7. M. Falcone. Numerical solution of dynamic programming equations. Appendix in [1].

8. M. Falcone and T. Giorgo. An approximation scheme for evolutive Hamilton-Jacobi equations. *Stochastic analysis, control, optimization and applications* eds W.M. McEneaney, G. George Y. and Q. Zhang, Birkhüser Boston, Boston, MA (1999) 289–303.
9. C.-S. Huang, S. Wang and K.L. Teo. Solving Hamilton-Jacobi-Bellman equations by a modified method of characteristic, *Nonlinear Analysis, TMA*, **40** (2000) 279–293.
10. H.J. Kushner and P.G. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*. Springer-Verlag, New York (1992).
11. P.L. Lions. *Generalised solutions of Hamilton-Jacobi equations*. Pitman Research Notes in Mathematics, **69** (1982).
12. P. Sonneveld. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, **10** (1989) 36–52.
13. S. Wang, F. Gao and K.L. Teo. An upwind finite difference method for the approximation of viscosity solutions to Hamilton-Jacobi-Bellman equations. *IMA J. Math. Control & Its Appl.*, **17** (2000) 167–178.
14. J.L. Zhou, A.L. Tits and C.T. Lawrence. User's Guide for FFSQP Version 3.7: A Fortran Code for solving Constrained Optimization Problems, Generating Iterates Satisfying All Inequality and Linear Constraints. Technical Report TR92-107-r2, System Research Center, Univ. of Maryland (1997).
15. H.A. van der vorst. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, **13** (1992) 631–644
16. R.S. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Englewood Cliffs, NJ (1962).