11-1-1988

On the Numerical Approximations of an Optimal Correction Problem

M. C. Bancora-Imbert
Universidad Nacional de Rosario

P. L. Chow
Wayne State University, plchow@math.wayne.edu

J. L. Menaldi
Wayne State University, menaldi@wayne.edu

SIAM J. SCI. STAT. COMPUT.
Vol. 9, No. 6, November 1988

ON THE NUMERICAL APPROXIMATION OF AN OPTIMAL CORRECTION PROBLEM*

M. C. BANCORA-IMBERT†, P. L. CHOW‡, AND J. L. MENALDI‡

Abstract. The numerical solution of an optimal correction problem for a damped random linear oscillator is studied. A numerical algorithm for the discretized system of the associated dynamic programming equation is given. To initiate the computation, we adopt a numerical scheme derived from the deterministic version of the problem. Next, a correction-type algorithm based on a discrete maximum principle is introduced to ensure the convergence of the iteration procedure.

Key words. optimal corrections, variational inequalities, free boundaries, numerical solutions

AMS(MOS) subject classifications. 65N10, 93E20, 65K10, 93E25

1. Introduction. We consider the control of a damped linear oscillator excited by a random noise

$$\begin{aligned} &x''(t)+px'(t)+q^2x(t)=rw'(t)+v'(t), \qquad 0<t\leqq T,\\ &x(0)=x, \qquad x'(0)=y, \end{aligned} \tag{1}$$

where p and q are the damping and spring constants; x and y denote the initial position and velocity; r is the intensity of the white-noise $w'(t)$; $v(t)$ is the control momentum at the time t; and T is the horizon. The prime denotes the time derivative. When we set $y=x'$, (1) may be interpreted as a stochastic differential equation in the Ito sense:

$$\begin{aligned} &dx(s)=y(s)\,ds,\\ &dy(s)=-[py(s)+q^2x(s)]\,ds+r\,dw(s)+dv(s),\\ &dt(s)=ds,\\ &x(0)=x, \quad y(0)=y, \quad t(0)=t, \end{aligned} \tag{2}$$

which is defined on a standard Wiener space $(\Omega, \mathcal{F}, P, w(t), \mathcal{F}(t), t\geqq 0)$ satisfying the usual conditions [7]. An admissible control $(v(t), t\geqq 0)$ is a stochastic right-continuous process adapted to $(\mathcal{F}(t), t\geqq 0)$, having locally bounded variation. The fact that the control momentum is required to have bounded variation means that the control is derived from a finite-resource or finite-fuel constraint instead of from the classical finite-energy constraint.

Also, for simplicity, we assume that the expected cost function is of the special form

$$J(x, y, t, v(\cdot))=E\{f(x(T-t), y(T-t))+c|v(T-t)|\}, \tag{3}$$

where f is a smooth function with growth bounded by a polynomial, c is a positive constant, and $|v(T)|$ denotes the total variation on $[0, T]$ of the process $(v(t), t\geqq 0)$. Clearly, f measures the deviations from the rest position and c represents the unit cost of the resource. If we denote by u the minimum cost function, i.e.,

$$u(x, y, t)=\inf\{J(x, y, t, v(\cdot)): v(\cdot)\}, \tag{4}$$

* Received by the editors May 21, 1986; accepted for publication (in revised form) March 3, 1988.

† Departamento de Matemática, Universidad Nacional de Rosario, Avenida Pellegrini 250, 2000 Rosario, Argentina. The work of this author was supported in part by CONICET.

‡ Department of Mathematics, Wayne State University, Detroit, Michigan 48202. The work of these authors has been supported by the National Science Foundation under grants DMS 8702236 and INT 8706083.

then, by properly using the dynamic programming argument, we can show that u satisfies the following differential inequalities:

$$\text{(5)}\quad \begin{aligned} &\text{(i)} \quad \frac{\partial u}{\partial t} + Lu \geqq 0, \\ &\text{(ii)} \quad -c \leqq \frac{\partial u}{\partial y} \leqq c, \end{aligned}$$

with the complementary ψondition

$$\text{(6)}\qquad \left(\frac{\partial u}{\partial t} + Lu\right)\left(\frac{\partial u}{\partial y} + c\right)\left(\frac{\partial u}{\partial y} - c\right) = 0$$

and the terminal condition

$$\text{(7)}\qquad u(\cdot, \cdot, T) = f,$$

where $0 \leqq t \leqq T$, x, y in $\mathscr{R}$ and

$$\text{(8)}\qquad Lu(x, y, t) = \frac{1}{2} r^2 \frac{\partial^2 u(x, y, t)}{\partial y^2} - (py + q^2 x) \frac{\partial u(x, y, t)}{\partial y} + y \frac{\partial u(x, y, t)}{\partial x}.$$

A theoretical study of this problem can be found in [3], Sun and Menaldi [14]; more general similar problems are considered in Menaldi and Robin [11], [12] and Chow et al. [5]. We refer also to Gorbunov [9] for a similar setting of this problem. The preliminary results of this work were summarized in [4].

To solve the problem numerically, we approximate the unknown value function $u(x, y, t)$ as well as an optimal feedback law. Then we replace the unbounded domain in the (x, y, t) space by a rectangular box

$$\text{(9)}\qquad B = \{(x, y, t) \in \mathscr{R}^3 : |x| \leqq a, |y| \leqq b, 0 \leqq t \leqq T\},$$

and we introduce a finite-difference scheme to approximate the variational inequality (5)-(7) and some appropriate condition on the boundary surface of B. To this end, we let

$$\text{(10)}\qquad \Delta x = \frac{a}{M}, \quad \Delta y = \frac{b}{M}, \quad \Delta t = \frac{T}{N}$$

for some positive integers M, N. Let $Q(M, N)$ denote the set of meshpoints in B, i.e.,

$$\text{(11)}\qquad \begin{aligned} Q(M, N) = \{(x_i, y_j, t_n) : x_i = i\Delta x, y_j = j\Delta y, t_n = n\Delta t, i, j = 0, \\ \pm 1, \cdots, \pm M, n = 0, 1, \cdots, N\}. \end{aligned}$$

The nodal value of the approximate solution $\bar{u}$ of $u(x, y, t)$ at a meshpoint (x_i, y_j, t_n) is denoted by

$$\text{(12)}\qquad u_{i,j}^n = \bar{u}(x_i, y_j, t_n).$$

In what follows, we shall present two numerical procedures for solving the problem (5)-(7) corresponding to two different finite-difference schemes for variational inequalities. The first procedure is adapted from a deterministic version of the problem, whereas the second procedure deals with the full problem directly. For convenience, they will be called the first-order and second-order methods, respectively. The former method is simple and explicit, but less accurate, and some condition is needed to ensure stability and convergence. The latter is an implicit scheme, which is always

stable and convergent, as a direct result of the maximum principle. By using the first-order procedure as an initial approximation, the speed of convergence for the second-order procedure is increased.

Let us point out that, roughly speaking, the present problem corresponds to a time-dependent two-dimensional variational inequality, with constraints on the first derivative and with degeneracy in one space-variable. Due mainly to these characteristics, standard algorithms do not produce satisfactory results.

We remark that the same algorithm can be applied to other problems where, for instance, a more general differential operator L can be considered.

2. A first-order numerical method. We discretize the differential operator (8) as follows:

$$\frac{\partial}{\partial t} \approx (u_{i,j}^{n+1} - u_{i,j}^{n})(\Delta t)^{-1}, \tag{13}$$

$$\frac{1}{2} r^2 \frac{\partial^2}{\partial^2 y} \approx \frac{1}{2} r^2 (u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1})(\Delta y)^{-2}, \tag{14}$$

$$\begin{aligned} -(py + q^2 x)\frac{\partial}{\partial y} \approx & \left| pj + q^2 i \Delta x (\Delta y)^{-1} \right| \\ & \cdot \{(u_{i,j-1}^{n} - u_{i,j}^{n})\Pi(pj + q^2 i \Delta x (\Delta y)^{-1} > 0) + (u_{i,j+1}^{n} - u_{i,j}^{n}) \\ & \cdot \Pi(pj + q^2 i \Delta x (\Delta y)^{-1} < 0)\}, \end{aligned} \tag{15}$$

$$y\frac{\partial}{\partial x} \approx |j| \Delta y (\Delta x)^{-1} [(u_{i+1,j}^{n} - u_{i,j}^{n})\Pi(j > 0) + (u_{i-1,j}^{n} - u_{i,j}^{n})\Pi(j < 0)], \tag{16}$$

where $\Pi(\cdot)$ denotes the characteristic function, i.e.,

$$\Pi(j > 0) = 1 \quad \text{if } j > 0 \text{ and } = 0 \text{ otherwise.} \tag{17}$$

Then, the inequalities (5) become

$$\begin{aligned} u_{i,j}^{n} \leqq & c_1(i,j) u_{i,j}^{n+1} + c_2(i,j) \tfrac{1}{2} (u_{i,j+1}^{n+1} + u_{i,j-1}^{n+1}) \\ & + c_3(i,j) u_{i,j+1}^{n} + c_4(i,j) u_{i,j-1}^{n} + c_5(i,j) u_{i+1,j}^{n} + c_6(i,j) u_{i-1,j}^{n}, \end{aligned} \tag{18}$$

$$u_{i,j}^{n} \leqq u_{i,j+1}^{n} + c\Delta y, \tag{19}$$

$$u_{i,j}^{n} \leqq u_{i,j-1}^{n} + c\Delta y, \tag{20}$$

$$c_0(i,j) = (\Delta t)^{-1} + \left| pj + q^2 i \Delta x (\Delta y)^{-1} \right| + |j| \Delta y (\Delta x)^{-1},$$

$$c_1(i,j) = [(\Delta t)^{-1} - r^2 (\Delta y)^{-2}][c_0(i,j)]^{-1},$$

$$c_2(i,j) = [r^2 (\Delta y)^{-2}][c_0(i,j)]^{-1},$$

$$c_3(i,j) = [pj + q^2 i \Delta x (\Delta y)^{-1}]^{-}[c_0(i,j)]^{-1},$$

$$c_4(i,j) = [pj + q^2 i \Delta x (\Delta y)^{-1}]^{+}[c_0(i,j)]^{-1},$$

$$c_5(i,j) = (j)^{+} \Delta y (\Delta x)^{-1} [c_0(i,j)]^{-1},$$

$$c_6(i,j) = (j)^{-} \Delta y (\Delta x)^{-1} [c_0(i,j)]^{-1},$$

and p, q, r, c, are the constants in (1), (2), (3), and $[\cdot]^{+}$ or $[\cdot]^{-}$ denotes the positive or negative part of a real number.

If we impose the stability condition

$$r^2 \Delta t < (\Delta y)^2, \tag{21}$$

then

$$\sum_{1=1}^{6} c_k = 1 \quad \forall i, j,$$

$$c_k(i, j) \geqq 0 \quad \forall i, j, k. \tag{22}$$

To the inequalities (18), (19), and (20) we should add the terminal condition

$$u_{i,j}^{N} = f_{i,j}, \tag{23}$$

where $f_{i,j}$ is the value of the data $f(x, y)$ for $x = i\Delta x$, $y = j\Delta y$.

2.1. Numerical method. Before describing the method, let us introduce some short-hand notation. We shall denote a double array or a matrix, say, $(f_{i,j}, i, j = 0, \pm 1, \pm 2, \cdots, \pm M)$ by $\hat{f}$ and a triple array, $\{u_{i,j}^{n}, i, j = 0, \pm 1, \cdots, \pm M, n = 0, 1, \cdots, N)$ by $\bar{u}$. Similarly we shall write $\hat{u}_n = (u_{i,j}^{n}, i, j = 0, \pm 1, \cdots, \pm M)$, $\hat{u}_{n,k} = (u_{i,j}^{n,k}, i, j = 0, \pm 1, \cdots, \pm M)$ and $\hat{u}_k = (\hat{u}_{n,k}, n = 0, 1, \cdots, N) = (u_{i,j}^{n,k}, i, j = 0, \pm 1, \cdots, \pm M, n = 0, 1, \cdots, N)$. For the double array $\hat{f}, \hat{g}, \cdots$, define

$$\max \text{ (or min)} \hat{f} = \max \text{ (or min)} \{f_{i,j}, i, j = 0, \pm 1, \cdots, \pm M\},$$

and set

$$\min \{\hat{f}, \hat{g}, \cdots\} = \hat{h}$$

where

$$h_{i,j} = \min \{f_{i,j}, g_{i,j}, \cdots\} \quad \forall i, j.$$

Similar conventions are in effect for triple arrays.

For brevity, the meshpoint (x_i, y_i, t_n) will be denoted simply by (i, j, n). Let D be the mesh

$$D = \{(i, j): i, j = 0, \pm 1, \cdots, \pm M\}, \tag{24}$$

which consists of all the spatial meshpoints. Define

$$k_0 = pq^{-2}\Delta x(\Delta y)^{-1} \quad \text{and} \quad i_0 = k_0 M \tag{25}$$

such that $0 < k_0 < 1$ and i_0 is an integer. Subdivide the set D into four parts, D_1, D_2, D_3, D_4, by the lines $i + k_0 j = 0$ and $j = 0$, as shown in Fig. 1. Let Γ_1 denote all points (i, j) on the boundary Γ of D such that one of the following conditions holds:

(a) $i = M$ and $j = 1, 2, \cdots, M$.

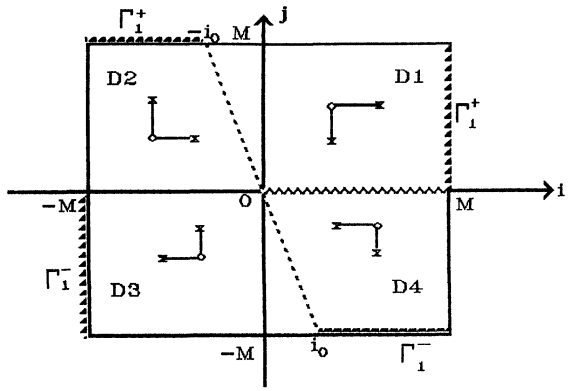

FIG. 1. *First-order method. The values at points marked "x" are given and those marked "o" are to be computed. Initial values are prescribed at points along* OM.

(b) $i=-M$ and $j=-1,-2,\cdots,-M$,
(c) $i=-M,-M+1,\cdots,-i_0$ and $j=M$,
(d) $i=i_0, i_0+1,\cdots,M$ and $j=-M$.

The set Γ_1 will be called the prescribed boundary of D. The numerical method is an iterative scheme in the form $\bar{u}_{k+1}=T\bar{u}_k$, $k=0,1,2,\cdots$. The general procedure will be outlined in what follows.

Initialization. To initiate the iteration, choose the initial iterate $\bar{u}_0$ with constant entries

$$u_{i,j}^{n,0}=\min \hat{f} \quad \forall i,j,n, \tag{26}$$

which satisfy the inequalities (18)-(20).

Iteration. Given the kth iterate $\bar{u}_k$, the next iterate $\bar{u}_{k+1}$ is computed, for $k=1,2,\cdots$, according to the following steps:

Step 1. The terminal condition for $\bar{u}_{k+1}$ is given by

$$\hat{u}_{N,k+1}=\hat{f}, \qquad k=0,1,\cdots.$$

Step 2. For $n=N-1, N-2,\cdots,1,0$, determine the triple array $\bar{\alpha}_{k+1}$ by setting

$$\bar{\alpha}_{k+1}=\min\{f_{i,j}, f_{i,j+1}+c\Delta y, f_{i,j-1}+c\Delta y\}$$

on the prescribed boundary Γ_1, and, in components,

$$\alpha_{i,j}^{n,k+1}=c_1(i,j)u_{i,j}^{n+1,k}+c_2(i,j)2^{-1}(u_{i,j+1}^{n+1,k}+u_{i,j-1}^{n+1,k})$$
$$+c_3(i,j)u_{i,j+1}^{n,k}+c_4(i,j)u_{i,j-1}^{n,k}+c_5(i,j)u_{i+1,j}^{n,k}+c_6(i,j)u_{i-1,j}^{n,k},$$

on the complement $\Gamma_1'=(D-\Gamma_1)$ of Γ_1.

Step 3. For $n=N-1, N-2,\cdots,1,0$, compute the triple array $\bar{\beta}_{k+1}$ by setting

$$\bar{\beta}_{k+1}=\min\{f_{i,j}, f_{i,j+1}+c\Delta y\}$$

on the upper boundary Γ_1^+ with $j=M$, and

$$\beta_{i,j}^{n,k+1}=u_{i,j+1}^{n,k}+c\Delta y \quad \text{on } (D-\Gamma_1^+).$$

Step 4. For $n=N-1, N-2,\cdots,1,0$, find the triple array $\bar{\gamma}_{k+1}$ by letting

$$\bar{\gamma}_{k+1}=\min\{f_{i,j}, f_{i,j-1}+c\Delta y\}$$

on the lower boundary Γ_1^- and

$$\gamma_{i,j}^{n,k+1}=u_{j,j-1}^{n,k}+c\Delta y \quad \text{on } (D-\Gamma_1^-).$$

Step 5. For $n=N-1, N-2,\cdots,1,0$, set

$$\bar{u}_{k+1}=\min\{\bar{\alpha}_{k+1}, \bar{\beta}_{k+1}, \bar{\gamma}_{k+1}\},$$

where by convention the minimum is taken pointwise in (i,j,n).

Similarly, we construct a sequence $\bar{v}_k$, $k=0,1,2,\cdots$, of triple arrays by means of the iterative procedure $\bar{v}_{k+1}=T\bar{v}_k$ as given in Steps 1-5, but, instead of (26), with the initial iterate $\bar{v}_0$ given by

$$v_{i,j}^{n,0}=\max \hat{f} \quad \forall i,j,n. \tag{27}$$

As to be shown later, we claim that the sequences $\{\bar{u}_k\}$ and $\{\bar{v}_k\}$ converge to the same limit $\bar{u}$, which is the unique solution of the problem.

2.2. Algorithm. To implement the above numerical method, we propose a regressive iteration scheme, by which we mean that the iteration process is carried out at each timestep n while the time goes backward step by step. In contrast to a straightforward iteration on the triple array $\bar{u}_k$ or $\bar{v}_k$, the proposed scheme yields a more efficient algorithm. Again referring to Fig. 1 and the notation used in § 2.1, given the terminal condition, choose a suitable initial iterate and assign appropriate data on Γ_1, as well as on the line segment $\overline{OM} = \{0 \leqq i \leqq M, j = 0\}$. This additional set of data is necessary to initiate the iteration process. For each timestep n, the computation will be carried out by parts, from D_1 to D_4 in the counterclockwise direction, comparing the calculated values on $\overline{OM}$ with their previous values. The counterclockwise iteration process will be continued if the absolute value of the difference is greater than a preset precision constant ε, using the new values as the data on $\overline{OM}$. Otherwise, change n to $n-1$ and repeat the sweeping process as before. For convenience, we say that "the iteration converges with ε-precision" if, for some integer $K > 0$, the following condition holds:

$$|\max(\bar{u}_K - \bar{u}_{K-1})| < \varepsilon. \tag{28}$$

Set $\bar{u}_\varepsilon = \bar{u}_K$ and call this a relaxed solution. The proposed numerical algorithm will be given in detail as follows:

(1) First set the terminal condition

$$\bar{u}_N = \hat{f}. \tag{29}$$

(2) For the timesteps $n = N-1, N-2, \cdots, 1, 0$ (in decreasing n), execute Steps 3-7 successively.

(3) Introduce the initial-boundary conditions:

(3.1) On the slit $\overline{OM}$, set

$$u^n_{i,j} = \min \hat{f} \quad \forall n, (i,j) \in \overline{OM}.$$

(3.2) On the prescribed boundary, set

$$u^n_{i,j} = \min\{f_{i,j}, f_{i,j+1} + c\Delta y, f_{i,j-1} + c\Delta y\} \quad \forall n, (i,j) \in \Gamma_1. \tag{30}$$

(4) To calculate the missing values of $\bar{u}_n$ in D_1, for $i = M-1, M-2, \cdots, 1, 0$ (in decreasing i), proceed as follows:

(4.1) For $j = 1, 2, \cdots, M$ (in increasing j) with $i + k_0 j \geqq 0$, compute $\hat{\alpha}_n$ by components,

$$\alpha^n_{i,j} = \min\{c_1(i,j)u^{n+1}_{i,j} + c_2(i,j)\tfrac{1}{2}(u^{n+1}_{i,j+1} + u^{n+1}_{i,j-1}) + c_4(i,j)u^n_{i,j-1} + c_5(i,j)u^n_{i+1,j}, u^n_{i,j-1} + c\Delta y\}.$$

(4.2) For $j = M-1, \cdots, 1, 0$ (in decreasing j), with $i + k_0 j \geqq 0$, compute $\hat{u}_n$ by

$$u^n_{i,j} = \min\{\alpha^n_{i,j}, \alpha^n_{i,j+1} + c\Delta y\}.$$

(5) To find $\hat{u}_n$ in D_2, for $i = -k_0 M, -k_0(M-1), \cdots, -M$ (in decreasing i), proceed as follows:

(5.1) For $j = M-1, \cdots, 1, 0$ (in decreasing j) with $i + k_0 j < 0$, compute $\hat{\alpha}_n$ according to

$$\alpha^n_{i,j} = \min\{c_1(i,j)u^{n+1}_{i,j} + c_2(i,j)\tfrac{1}{2}(u^{n+1}_{i,j+1} + u^{n+1}_{i,j-1}) + c_3(i,j)u^n_{i,j+1} + c_5(i,j)u^n_{i+1,j}, u^n_{i,j+1} + c\Delta y\}.$$

(5.2) For $j = 1, 2, \cdots, M$ (in increasing j) with $i + k_0 j < 0$, compute $\hat{u}_n$ by

$$u^n_{i,j} = \min\{\alpha^n_{i,j}, \alpha^n_{i,j-1} + c\Delta y\}.$$

(6) To determine $\hat{u}_n$ in D_3, for $i=-M,-M+1,\cdots,-1,0$ (in increasing i), proceed as follows:

(6.1) For $j=-1,-2,\cdots,-M$ (in decreasing j) with $i+k_0 j\leqq 0$, compute $\hat{\alpha}_n$ by setting

$$\alpha^n_{i,j}=\min\{c_1(i,j)u^{n+1}_{i,j}+c_2(i,j)\tfrac{1}{2}(u^{n+1}_{i,j+1}+u^{n+1}_{i,j-1})+c_3(i,j)u^n_{i,j+1} + c_6(i,j)u^n_{i-1,j},\, u^n_{i,j+1}+c\Delta y\}.$$

(6.2) For $j=-M+1,\cdots,-1,0$ (in increasing j) with $i+k_0 j\leqq 0$, find $\hat{u}_n$ in components,

$$u^n_{i,j}=\min\{\alpha^n_{i,j},\,\alpha^n_{i,j-1}+c\Delta y\}.$$

(7) Finally to compute $\hat{u}_n$ in D_4, for $i=k_0M,k_0M+1,\cdots,M$, proceed as follows:

(7.1) For $j=-M+1,\cdots,-1,0$ (in increasing j) with $i+k_0 j>0$, find the components of $\hat{\alpha}_n$ by

$$\alpha^n_{i,j}=\min\{c_1(i,j)u^{n+1}_{i,j}+c_2(i,j)\tfrac{1}{2}(u^{n+1}_{i,j+1}+u^{n+1}_{i,j-1})+c_4(i,j)u^n_{i,j-1} + c_5(i,j)u^n_{i+1,j}+c_6(i,j)u^n_{i-1,j},\, u^n_{i,j-1}+c\Delta y\}.$$

(7.2) For $j=-1,-2,\cdots,-M$ (in decreasing j) with $i+k_0 j>0$, compute $\hat{u}$ by

$$u^n_{i,j}=\min\{\alpha^n_{i,j},\,\alpha^n_{i,j+1}+c\Delta y\}.$$

(8) At the end of step (7), compare the values of $\hat{u}$ with those obtained previously to start step (4). If the precision condition (28) is not satisfied, repeat steps (4)-(7). If satisfied, go backward to the $(n-1)$th timestep and restart the above iteration process.

This algorithm yields a sequence $\{\bar{u}_k\}$ of arrays that converges with ε-precision to a relaxed solution. Similarly, by changing the initial data in (3) to

$$v^n_{i,j}=\max\hat{f}\quad\forall n,\,(i,j)\in\overline{OM},$$

we construct a sequence $\{\bar{v}_k\}$, which also approaches a relaxed solution.

2.3. Convergence. Under the stability condition (21), it will be shown in § 4 that two sequences $\{\bar{u}_k\}$ and $\{\bar{v}_k\}$, as defined above, actually converge to the same limit array $\bar{u}$ and satisfy the following properties:

(1) For $k=0,1,2,\cdots$, we have

$$\bar{u}_k\leqq\bar{u}_{k+1}\leqq\bar{u}\leqq\bar{v}_{k+1}\leqq\bar{v}_k, \tag{31}$$

which is understood to hold componentwise.

(2) For every point not on the prescribed boundary Γ_1, at least one of the inequalities (18), (19), and (20) is verified as an equality.

(3) The array $\bar{u}$ is the maximal subsolution of problem (18)-(20) and (22), i.e., if $\bar{w}$ is any other solution to the problem satisfying the terminal condition (29) and the boundary condition (30) (where all equalities are replaced by inequalities), we have $\bar{w}\leqq\bar{u}$ pointwise.

(4) The array $\bar{u}$ is also the minimal supersolution of problem (18)-(20) and (22), i.e., if $\bar{z}$ is any array satisfying at least one of the inequalities in a reversed sense, which also satisfies terminal condition (29) and boundary condition (30), then $\bar{u}\leqq\bar{z}$.

Note that, in view of the monotonicity property (31), the sequences $\{\bar{u}_k\}$ and $\{\bar{v}_k\}$ converge monotonically to the same limit $\bar{u}$, which is the unique solution to the problem.

3. A second-order numerical method. Here, instead of using (14), we discretize the second-order differential operator as follows:

$$\frac{1}{2}r^2\frac{\partial^2}{\partial y^2}\approx\frac{1}{2}r^2(u^n_{i,j+1}-2u^n_{i,j}+u^n_{i,j+1})(\Delta y)^{-2}. \tag{32}$$

Then the inequalities (5) become

$$u^n_{i,j} \leqq d_1(i,j)u^{n+1}_{i,j} + d_2(i,j)u^n_{i,j+1} + d_3(i,j)u^n_{i,j-1} + d_4(i,j)u^n_{i+1,j} + d_5(i,j)u^n_{i-1,j}, \tag{33}$$

$$u^n_{i,j} \leqq u^n_{i,j+1} + c\Delta y, \tag{34}$$

$$u^n_{i,j} \leqq u^n_{i,j-1} + c\Delta y, \tag{35}$$

where

$$\begin{aligned}
d_0(i,j) &= (\Delta t)^{-1} + r^2(\Delta y)^{-2} + |pj + q^2 i\Delta x(\Delta y)^{-1}| + |j|\Delta y(\Delta x)^{-1},\\
d_1(i,j) &= (\Delta t)^{-1}[d_0(i,j)]^{-1},\\
d_2(i,j) &= [\tfrac{1}{2}r^2(\Delta y)^{-2} + (pj + q^2 i\Delta x(\Delta y)^{-1})^-][d_0(i,j)]^{-1},\\
d_3(i,j) &= \tfrac{1}{2}r^2(\Delta y)^{-2} + (pj + q^2 i\Delta x(\Delta y)^{-1})^+][d_0(i,j)]^{-1},\\
d_4(i,j) &= (j)^+\Delta y(\Delta x)^{-1}[d_0(i,j)]^{-1},\\
d_5(i,j) &= (j)^-\Delta y(\Delta x)^{-1}[d_0(i,j)]^{-1},
\end{aligned}$$

and p, q, r, c, are the constants in (1), (2), (3).

Notice that without any stability condition we have

$$\begin{aligned}
&\sum_{k=1}^{5} d_k(i,j) = 1 \quad \forall i,j,\\
&d_k(i,j) \geqq 0 \quad \forall i,j,k.
\end{aligned} \tag{36}$$

As previously, the terminal condition

$$u^N_{i,j} = f_{i,j} \tag{37}$$

is added to the inequalities (33), (34), and (35).

3.1. Numerical method. As with the first-order scheme, the proposed method is an iterative procedure of the form $\bar{u}_{k+1} = S\bar{u}_k$, $k = 0, 1, 2, \cdots$. In this case, the spatial mesh D is divided into two parts, the upper half D^+ and the lower half D^-, as shown in Fig. 2. The prescribed boundary Γ_2 here is the set of points (i,j) satisfying one of the following conditions:

(a) $i = M$ and $j = 1, 2, \cdots, M$,
(b) $i = -M$ and $j = -1, -2, \cdots, -M$,
(c) $i = 0, \pm 1, \cdots, \pm M$ and $j = M$,
(d) $i = 0, \pm 1, \cdots, \pm M$ and $j = -M$.

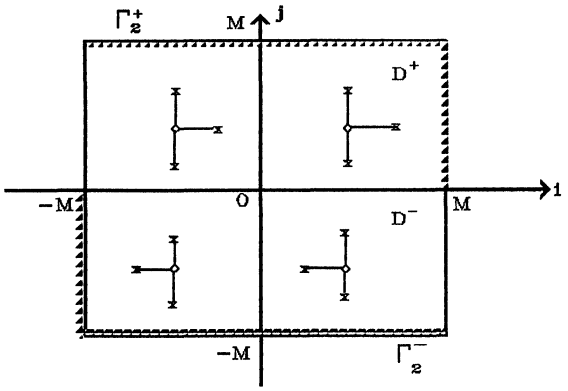

FIG. 2. *Second-order method. The values at points marked "x" are given and those marked "o" are to be computed.*

In what follows, the general numerical procedure will be presented.

Initialization. Take the initial iterate $\bar{u}_0$ to be of constant components

$$u_{i,j}^{n,0} = \min \hat{f} \quad \forall i, j, n.$$

Iteration. Given the kth iterate $\bar{u}_k$, compute $\bar{u}_{k+1}$ for $k = 1, 2, \cdots$, by the following steps:

Step 1. For the terminal condition, set

$$\hat{u}_{N,k+1} = \hat{f}, \quad k = 0, 1, 2, \cdots.$$

Step 2. For $n = 0, 1, 2, \cdots$, set

$$\bar{\alpha}_{k+1} = \bar{u}_k \quad \text{on } \Gamma_2$$

and, in components,

$$\alpha_{i,j}^{n,k+1} = d_1(i,j)u_{i,j}^{n+1} + d_2(i,j)u_{i,j+1}^{n,k} + d_3(i,j)u_{i,j-1}^{n,k} + d_4(i,j)u_{i+1,j}^{n,k} + d_5(i,j)u_{i-1,j}^{n,k}$$

on the complement $\Gamma_2' = (D - \Gamma_2)$.

Step 3. For $n = N-1, N-2, \cdots, 1, 0$, set

$$\bar{\beta}_{k+1} = \bar{u}_k \text{ on the upper boundary } \Gamma^+ \text{ with } j = M,$$

and, by components,

$$\beta_{i,j}^{n,k+1} = u_{i,j+1}^{n,k} + c\Delta y \quad \text{on } (D - \Gamma^+).$$

Step 4. For $n = N-1, N-2, \cdots, 1, 0$, set

$$\bar{\gamma}_{k+1} = \bar{u}_k \text{ on the lower boundary } \Gamma^- \text{ with } j = -M,$$

and, in components,

$$\bar{u}_{k+1} = \min \{\bar{\alpha}_{k+1}, \bar{\beta}_{k+1}, \bar{\gamma}_{k+1}\}.$$

Step 5. For $n = N-1, N-2, \cdots, 1, 0$, set

$$\bar{u}_{k+1} = \min \{\bar{\alpha}_{k+1}, \bar{\beta}_{k+1}, \bar{\gamma}_{k+1}\}.$$

Similarly we may construct a sequence $\{\bar{v}_k\}$ of arrays by the same iterative procedure $\bar{v}_{k+1} = S\bar{v}_k$, $k = 0, 1, 2, \cdots$, by Steps 1-5, but with the initial iterate

$$v_{i,j}^{n,0} = \max \hat{f} \quad \forall i, j, n.$$

3.2. Algorithm. As before, we adopt a regressive iteration process. Referring to Fig. 2, the mesh D is divided into upper and lower parts D^+ and D^-. Given the terminal condition, assign appropriate boundary data on the prescribed boundary Γ_2. For each n, the iterations will be executed alternately between D^+ and D^- until the sequence $\{\bar{u}_k\}$ or $\{\bar{v}_k\}$ converges with ε-precision. Then change n to $n-1$ and repeat the iteration process. To be specific, we propose the following algorithm:

(1) Fix the terminal condition

$$\hat{u}_N = \hat{f}.$$

(2) For $n = N-1, N-2, \cdots, 1, 0$ (in decreasing n), follow steps (3)-(6) below.

(3) Assign the boundary data for every (i,j) on Γ_2,

$$u_{i,j}^n = \min \{f_{i,j}, f_{i,j+1} + c\Delta y, f_{i,j-1} + c\Delta y\} \quad \forall n.$$

(4) Choose the values of the initial iterate in $(D-\Gamma_2)$ to be

$$u^n_{i,j} = \min \hat{f} \quad \forall (i,j) \notin \Gamma_2,$$

or other suitable data, if available.

(5) Compute the values of $\hat{u}_n$ in D^+: For $i = M, \cdots, 0, -1, \cdots, -M+1, -M$ (in decreasing i) and $j = 1, 2, \cdots, M$ (in increasing j), set

$$\begin{aligned} u^n_{i,j} = \min \{ & d_1(i,j) u^{n+1}_{i,j} + d_2(i,j) u^n_{i,j+1} + d_3(i,j) u^n_{i,j-1} \\ & + d_4(i,j) u^n_{i+1,j},\ u^n_{i,j+1} + c\Delta y,\ u^n_{i,j-1} + c\Delta y \}. \end{aligned} \tag{38}$$

(6) Compute $\hat{u}_n$ in D^-: For $i = -M, \cdots, -1, 0, 1, \cdots, M$ (in increasing i) and $j = -M, \cdots, -1, 0$ (in increasing j), set

$$\begin{aligned} u^n_{i,j} = \min \{ & d_1(i,j) u^{n+1}_{i,j} + d_2(i,j) u^n_{i,j+1} + d_3(i,j) u^n_{i,j-1} \\ & + d_5(i,j) u^n_{i-1,j+1},\ u^n_{i,j+1} + c\Delta y,\ u^n_{i,j-1} + c\Delta y \}. \end{aligned} \tag{39}$$

(7) Repeat steps (5) and (6) with the updated values until the iteration converges with ε-precision. Then reset the timestep to $n-1$.

Similarly we may proceed to calculate $\bar{v}_n$ by the same steps except in step (4), replacing $\min \hat{f}$ by $\max \hat{f}$.

3.3. Convergence. The above algorithm defines two sequences $\{\bar{u}_k\}$ and $\{\bar{v}_k\}$. In § 4 we show that they converge to the same limit $\bar{u}$, which satisfies the system of inequalities (33)-(35) subject to terminal condition (37). Furthermore the following properties hold:

(1) For $k = 0, 1, 2, \cdots$, we have

$$\bar{u}_k \leqq \bar{u}_{k+1} \leqq \bar{u} \leqq \bar{v}_{k+1} \leqq \bar{v}_k. \tag{40}$$

(2) For every point in $(D-\Gamma_2)$, at least one of the inequalities (33)-(35) yields an equality for $\bar{u}$.

(3) The array $\bar{u}$ is the maximal subsolution of problem (33)-(34) and (35), along with (36), (37) and the boundary condition in the sense explained in § 2.3.

(4) The array $\bar{u}$ is also the minimal supersolution of the system in the sense explained in § 2.3.

In view of property (40), again we have a monotone convergence for both sequences as in the first-order case.

4. Proof of convergence. The key properties that allow us to show convergence are (22) and (36). Consider the vector space V of all two-dimensional arrays of the form

$$v = (v_{i,j};\ i,j = 0, \pm 1, \cdots, \pm M). \tag{41}$$

On the space V, we define the operators

$$\begin{aligned} (T_1 v)_{i,j} &= c_1(i,j) v_{i,j} + c_2(i,j) \tfrac{1}{2} (v_{i,j+1} + v_{i,j-1}), \\ (T_2 v)_{i,j} &= c_3(i,j) v_{i,j+1} + c_4(i,j) v_{i,j-1} + c_5(i,j) v_{i+1,j} + c_6(i,j) v_{i-1,j}, \end{aligned} \tag{42}$$

$$(M_1 v)_{i,j} = v_{i,j+1} + c\Delta y, \qquad (M_2 v)_{i,j} = v_{i,j-1} + c\Delta y, \tag{43}$$

$$\begin{aligned} (S_1 v)_{i,j} &= d_1(i,j) v_{i,j}, \\ (S_2 v)_{i,j} &= d_2(i,j) v_{i,j+1} + d_3(i,j) v_{i,j-1} + d_4(i,j) v_{i+1,j} + d_5(i,j) v_{i-1,j}, \end{aligned} \tag{44}$$

for every i, j whenever the above expressions are meaningful, and

$$Tv = T_1 v + T_2 v, \qquad Sv = S_1 v + S_2 v, \tag{45}$$

$$(Tv)_{i,j} = v_{i,j} \quad \text{if } (i,j) \in \Gamma_1, \tag{46}$$

$$(Sv)_{i,j} = v_{i,j} \quad \text{if } (i,j) \in \Gamma_2, \tag{47}$$

$$\begin{aligned} (M_1 v)_{i,j} &= v_{i,j} \quad \text{if } j = M, \\ (M_2 v)_{i,j} &= v_{i,j} \quad \text{if } j = -M. \end{aligned} \tag{48}$$

Discrete Maximum Principle. Let u, v be two arrays of the form (41), i.e., u, v in V. Then the condition

$$u_{i,j} \leqq v_{i,j} \quad \forall i, j \tag{49}$$

implies that

$$\begin{aligned} &(T_k u)_{i,j} \leqq (T_k v)_{i,j}, \qquad (M_k u)_{i,j} \leqq (M_k v)_{i,j}, \\ &(S_k u)_{i,j} \leqq (S_k v)_{i,j} \quad \forall i, j \text{ and } k = 1, 2. \end{aligned} \tag{50}$$

□

This property is a direct consequence of the fact that the coefficients $c_k(i,j)$ and $d_k(i,j)$ are nonnegative.

Now the convergence of both algorithms is deduced from this discrete maximum principle. The sequences $\bar{u}_k$ and $\bar{v}_k$, $k = 0, 1, \cdots$ generated by the first-order algorithm satisfy

$$\begin{aligned} \bar{u}^n_{k+1} &= \min \{T_1 \bar{u}^{n+1}_k + T_2 \bar{u}^n_k, M_1 \bar{u}^n_k, M_2 \bar{u}^n_k\}, \\ \bar{v}^n_{k+1} &= \min \{T_1 \bar{v}^{n+1}_k + T_2 \bar{v}^n_k, M_1 \bar{v}^n_k, M_2 \bar{v}^n_k\}, \end{aligned} \tag{51}$$

for any $k = 0, 1, 2, \cdots$ and $n = 0, 1, \cdots, N-1$. We say that $\bar{u}$ is a subsolution if

$$\bar{u} \leqq \min \{T_1 \bar{u} + T_2 \bar{u}, M_1 \bar{u}, M_2 \bar{u}\} \qquad \text{(pointwise sense)}.$$

We say that $\bar{v}$ is a supersolution if

$$\min \{T_1 \bar{v} + T_2 \bar{v}, M_1 \bar{v}, M_2 \bar{v}\} \leqq \bar{v} \qquad \text{(pointwise sense)}.$$

Since $\bar{u}_0$ is constant in i, j, n, we start from

$$\bar{u}^n_0 = T_1 \bar{u}^{n+1}_0 + T_2 \bar{u}^n_0 \quad \text{so that}$$

$$\bar{u}^n_0 \leqq M_k \bar{u}^n_0, \; k = 1, 2 \qquad \text{(pointwise sense)}$$

and also

$$\bar{u}^n_0 \leqq T_1 f + T_2 \bar{u}^n_0,$$

where

$$\bar{u}^n_k = (u^{n,k}_{i,j};\, i, j = 0, \pm 1, \cdots, \pm M), \qquad f = (f_{i,j}; i, j = 0, \pm 1, \cdots, \pm M). \tag{52}$$

Then (51) implies

$$\bar{u}^n_0 \leqq \bar{u}^n_1$$

and by means of the discrete maximum principle (49) and (50), we get

$$T_k \bar{u}^n_0 \leqq T_k \bar{u}^n_1, \quad M_k \bar{u}^n_0 \leqq M_k \bar{u}^n_1, \quad k = 1, 2,$$

from which

$$\bar{u}^n_1 \leqq \bar{u}^n_2$$

follows. On the other hand, $\bar{v}_0$ satisfies

$$\bar{v}^n_0 = T_1 \bar{v}^{n+1}_0 + T_2 \bar{v}^n_0, \quad \bar{v}^n_0 \leqq M_k \bar{v}^n_0, \quad k = 1, 2$$

and also

$$\bar{v}_0^n \geqq T_1 f + T_2 \bar{v}_0^n.$$

Then (51) gives

$$\bar{v}_1^n \leqq \bar{v}_0^n$$

and again, by means of the discrete maximum principle (49) and (50), we obtain

$$T_k \bar{v}_1^n \leqq T_k \bar{v}_0^n, \quad M_k \bar{v}_1^n \leqq M_k \bar{v}_0^n, \quad k = 1, 2$$

and therefore

$$\bar{v}_2^n \leqq \bar{v}_1^n.$$

We also have

$$\bar{u}_1^n \leqq \bar{v}_1^n$$

and, by induction,

$$\bar{u}_0^n \leqq \bar{u}_k^n \leqq \bar{u}_{k+1}^n \leqq \bar{v}_{k+1}^n \leqq \bar{v}_k^n \leqq \bar{v}_0^n \quad \forall n, k. \tag{53}$$

Hence, the sequences generated by the first-order algorithm converge to a unique limit array $\bar{u} = (u_{i,j}^n;\ i, j = 0, \pm 1, \cdots, \pm M;\ n = 0, 1, \cdots, N)$ satisfying

$$\bar{u}^n = \min\{T_1 \bar{u}^{n+1} + T_2 \bar{u}^n,\ M_1 \bar{u}^n,\ M_2 \bar{u}^n\}, \tag{54}$$

$$\bar{u}_k^n \leqq \bar{u}^n \leqq \bar{v}_k^n \quad \forall n, k. \tag{55}$$

The fact that $\bar{u}$ is the maximum subsolution follows in a similar way from the discrete maximum principle.

Analogously, the convergence of the second-order algorithm can be proved. The two sequences $\bar{u}_k$ and $\bar{v}_k$, $k = 0, 1, \cdots$ generated by the second-order algorithm satisfy

$$\begin{aligned} \bar{u}_{k+1}^n &= \min\{S_1 \bar{u}_k^{n+1} + S_2 \bar{u}_k^n,\ M_1 \bar{u}_k^n,\ M_2 \bar{u}_k^n\}, \\ \bar{v}_{k+1}^n &= \min\{S_1 \bar{v}_k^{n+1} + S_2 \bar{v}_k^n,\ M_1 \bar{v}_k^n,\ M_2 \bar{v}_k^n\} \end{aligned} \tag{56}$$

for any $k = 0, 1, \cdots$ and $n = 0, 1, \cdots, N-1$. Then by induction we can show that (53) and (55) hold for these new sequences and also the limit array satisfies

$$\bar{u}^n = \min\{S_1 \bar{u}^{n+1} + S_2 \bar{u}^n,\ M_1 \bar{u}^n,\ M_2 \bar{u}^n\}. \tag{57}$$

Therefore, we have proved the convergence of both (first-order and second-order) algorithms with the error estimate (55).

We remark that several variants of these algorithms can be considered, for instance, the use of other additional boundary conditions together with changes in the definitions (40), (47). Similar algorithms for first-order equations and other kinds of Hamilton-Jacobi-Bellman equations have been used by Falcone [6], Gonzalez and Rofman [8], Bancora-Imbert, Gonzalez, Miellou, and Rofman [1], and others.

5. Numerical results. The second-order algorithm was applied to solve the problem (5)-(7) numerically. As a special case, we took the physical constants in this problem as follows: $q = 0.1$, $r = 0.2$, and $c = 0.1$.

The program was executed with positive $p = 0.01$, $p = 0$, and negative $p = -0.01$. The corresponding numerical results remain unchanged up to three digits after the decimal point. This is due to the fact that the algorithm is unconditionally stable. The negative constant p was chosen in order to test the efficiency of the algorithm in the case where the physical system would be destabilized by negative damping.

The cost function f was assumed to be of the form

$$f(x, y) = x^2 + y^2.$$

A mesh was defined on the box $B = \{(x, y, t) \in \mathcal{R}^3,\ |x| \leqq 1,\ |y| \leqq 1,\ 0 \leqq t \leqq 3\}$. The precision constant was chosen to be $\varepsilon = 10^{-15}$.

Solutions were computed backward in time, with the condition (28) satisfied for each timestep.

We have considered two cases of initializing the iterates.

Case 1. $u_{i,j}^{n,0} = \min f_{i,j}$, for all n ($v_{i,j}^{n,0} = \max f_{i,j}$).

Case 2. $u_{i,j}^{N-1,0} = \min f_{i,j}$ (or the min of a solution coming from a coarser mesh); $u_{i,j}^{n,0} = \min \bar{u}_{i,j}^{n+1}$, for all $n \neq N-1$, where $\bar{u}_{i,j}^{n+1}$ is the relaxed solution obtained in the previous timestep.

The second initialization procedure accelerates the convergence, because we start with a better subsolution.

In the initial phase of computation, we can apply the first-order numerical method with the finest possible mesh to ensure the satisfaction of the stability condition (21), which in our case yields mesh sizes $\Delta x = \Delta y = 0.25$ and $\Delta t = 1$.

This scheme has proved to be faster in obtaining a first approximation to the solution than the second-order one with the same mesh.

As explained in § 2.2, we took the terminal values $u_{i,j}^N = f_{i,j} = (i\Delta x)^2 + (j\Delta y)^2$ in D, the boundary values $u_{i,j}^n = \min\{f_{i,j}, f_{i,j+1} + 0.025, f_{i,j-1} + 0.025\}$ on the prescribed boundary Γ_1 and the initial value $u_{i,j}^{N-1} = 0$ on the slit $\overline{OM}$ (see Fig. 1).

Then, following the iterative algorithm given in § 2.3, we obtained a relaxed solution $\bar{u}_1^\varepsilon$ with precision $\varepsilon = 10^{-15}$.

In Case 2, the relaxed solution thus obtained was used as the initialization data for the second-order numerical scheme with a refined mesh, in which the missing initial values were obtained by interpolation.

In computations we took $M = 8$, $N = 3$ so that $\Delta x = \Delta y = 0.125$, $\Delta t = 1$, and $c\Delta y = 0.0125$.

The terminal and boundary values are the same as before, but with the prescribed boundary Γ_1 replaced by Γ_2 (see Fig. 2). However, the initial data (iterate) is changed to $\bar{u} = \min \bar{u}_1^\varepsilon$ at every point in D. Here the second-order algorithm in § 3.2, can be put in the following form.

(1) Initialization as described above.

(2) Time loop $n = N-1, N-2, \cdots, 1, 0$

(*)
$$\left[\begin{array}{l}
\text{Loop } i = M, M-1, \cdots, 1, 0, -1, \cdots, -M+1, -M \\
\qquad \text{Loop } j = 1, 2, \cdots, M \\
\qquad u_{i,j}^n = \min\{d_1 u_{i,j}^{n+1} + d_2 u_{i,j+1}^n + d_3 u_{i,j-1}^n + d_4 u_{i+1,j}^n, \\
\qquad\qquad u_{i,j+1}^n + c\Delta y, u_{i,j-1}^n + c\Delta y\} \\
\text{Loop } i = -M, -M+1, \cdots, -1, 0, 1, \cdots, M-1, M \\
\qquad \text{Loop } j = -M, -M+1, \cdots, -1, 0 \\
\qquad u_{i,j}^n = \min\{d_1 u_{i,j}^{n+1} + d_2 u_{i,j+1}^n + d_3 u_{i,j-1}^n + d_5 u_{i-1,j}^n, \\
\qquad\qquad u_{i,j+1}^n + c\Delta y,\ u_{i,j-1}^n + c\Delta y\},
\end{array}\right.$$

where (*) stands for "Repeat until the iteration converges with ε-precision."

Later on we made a mesh refinement with $M = 16$ so that $\Delta x = \Delta y = 0.0625$. Some of the results will be shown in the saturation graphs (Figs. 11-13). Here the initialization

data came from the second-order results with a coarser mesh ($M = 8$) since the first-order method is unstable at $M = 8$ for our example.

We have also carried out the calculations on the different meshes with initialization values equal to $v^{n,0} = \max f$ for all n. This yields a decreasing sequence of supersolutions converging to the same solution as obtained by the convergent subsolutions. This was predicted by the theoretical results.

Some numerical results are displayed in the subsequent Figs. 3-17. They were obtained on a computer running with the INTEL 80286 processor (8 MHz 0-wait state) and a 80287 math-coprocessor. Our computer programs were written in (Turbo) Pascal language.

To get some idea about the change in computing time-scale involved, we experimented with mesh refinements. As examples, some execution times versus mesh sizes are given as follows:

Mesh size ($\Delta x = \Delta y$)	0.250	0.125	0.062
Execution time	15 sec.	1 min. 40 sec.	16 min. (per timestep)

We see that the execution time increases rapidly, like a negative cubic power in Δx, as the mesh size decreases. If K is the number of nodes, then the execution time is proportional to $K \ln K$.

Also we wish to point out that the advantage in the use of the mathematical coprocessor is a 40 percent reduction in execution times.

In Figs. 3-6, the graphs show various y-sectional curves for the minimum cost function $u(x, y, t)$ as a function of x, at $t = 2$, where y varies from -1 to 1 with the increment $\Delta y = 0.125$. All curves are convex as expected from the analytical result [3]. We also have symmetry on both variables x, y with respect to the origin: $u^n(x, y) = u^n(-x, -y)$. The y-sections are asymmetric in x. The asymmetry is most pronounced at high speeds ($|y|$ near 1), as shown in Figs. 4 and 6.

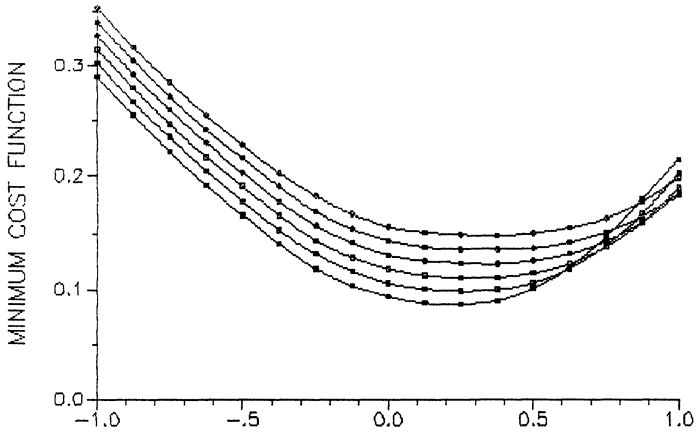

FIG. 3. *y-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, (*the last computed timestep*) *as y varies from* -1 *to* -0.375 *with the increment* 0.125.

These curves show that, for example, given positive velocities, positive displacements produce a higher cost than negative ones. This is because a control will be necessary in the first case to bring the oscillator back to its rest position while it will otherwise come back by itself.

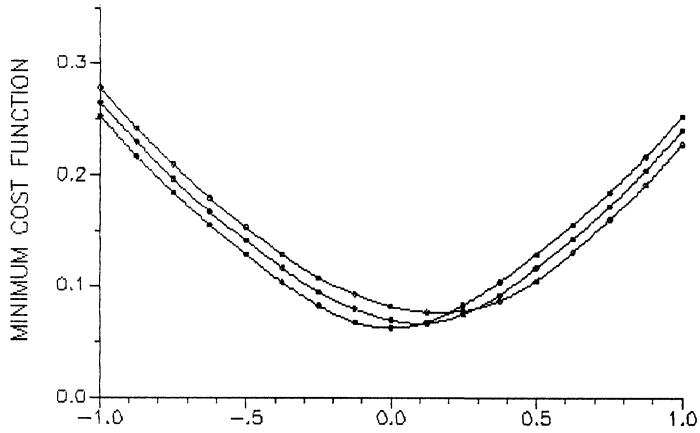

FIG. 4. *y-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, *as y varies from* -0.25 *to* 0 *with the increment* 0.125.

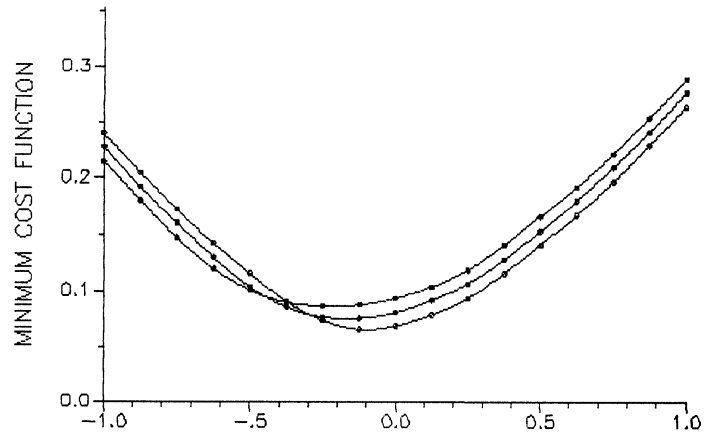

FIG. 5. *y-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, *as y varies from* 0 *to* 0.25 *with the increment* 0.125.

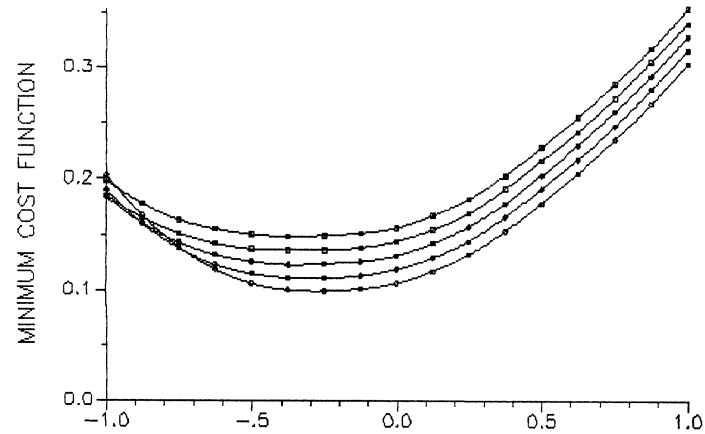

FIG. 6. *y-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, *as y varies from* 0.375 *to* 1 *with the increment* 0.125.

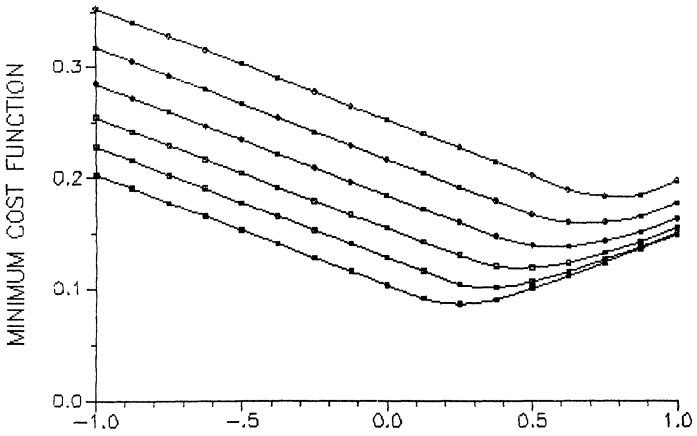

FIG. 7. *x-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, *as* x *varies from* -1 *to* -0.375 *with the increment* 0.125.

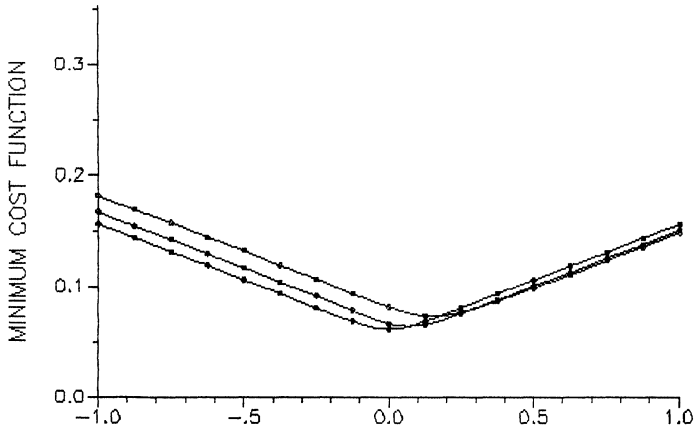

FIG. 8. *x-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, *as* x *varies from* -0.25 *to* 0 *with the increment* 0.125.

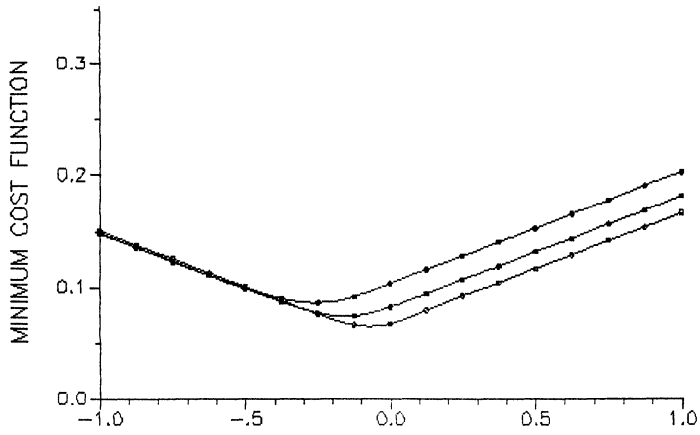

FIG. 9. *x-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, *as* x *varies from* 0 *to* 0.25 *with the increment* 0.125.

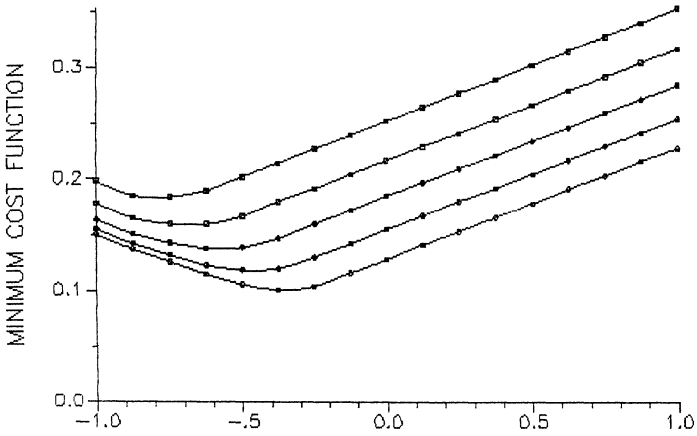

FIG. 10. *x-sectional curves for the minimum cost function* $u(x, y, t)$ *at* $t = 0$, *as* x *varies from* 0.375 *to* 1 *with the increment* 0.125.

The next set of figures, Figs. 7–10, show the x-sectional curves for the minimum cost function u as functions of y, at $t = 2$, where x varies from -1 to 1 with the increment $\Delta x = 0.125$. Note that, at each velocity y, the cost increases with the displacement $|x|$, as expected. Also it is interesting to note the remarkable shift of the low cost region, in Figs. 7 and 10, away from the center, $y = 0$. This means that the velocity $\bar{y}$, at which the cost attains its minimum, decreases for increasing positive displacements and increases otherwise.

In Figs. 11–13 the so-called "saturation graphs" at $t = 2, 1, 0$, are exhibited. In each figure, the computational domain, the unit square $\{-1 \leqq x \leqq 1, -1 \leqq y \leqq 1\}$, is subdivided into three distinct regions, marked by "c," "↑," and "↓." The region "c" is known as the "continuation" set, in which no control action is taken. As soon as the state (x, y) of the system reaches a control region marked by arrows, a control will be applied in the direction of the arrows to bring the system back to the boundary of the region "c." Therefore they may be viewed as control charts at different instants of time. These regions are determined from the numerical solution to the problem by checking which inequality becomes an equality. In "c," the equality holds for (32). Similarly, "↓" and "↑" designate the regions in which the equality holds for (31) and (32), respectively.

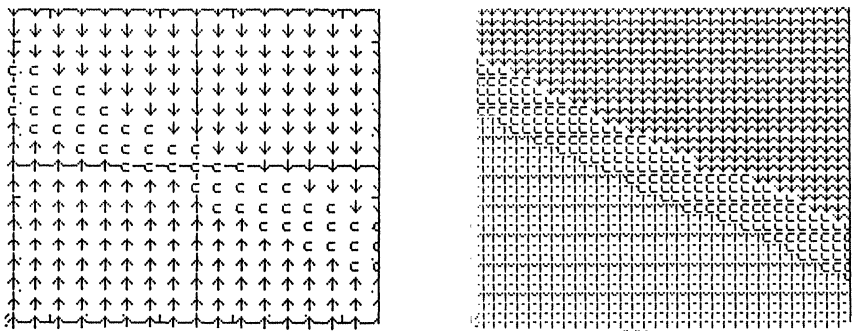

FIG. 11. *Saturation graph for two different meshes* ($M = 8$ *and* $M = 16$) *at* $t = 2$, (*the first computed time step*). *The* $[-1, +1] \times [-1, +1]$ *domain is represented, with* $(0, 0)$ *lying in the middle. In the region marked "*c," *no control is to be applied. In the arrowed regions, a control is to be applied in the direction of an arrow.*

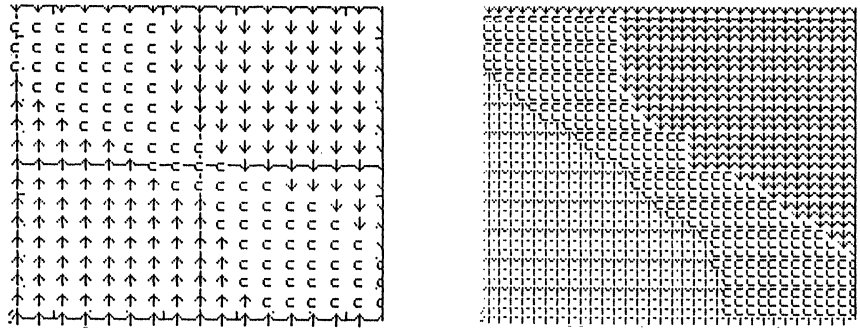

FIG. 12. *Saturation graph for two different meshes* ($M = 8$ *and* $M = 16$) *at* $t = 1$.

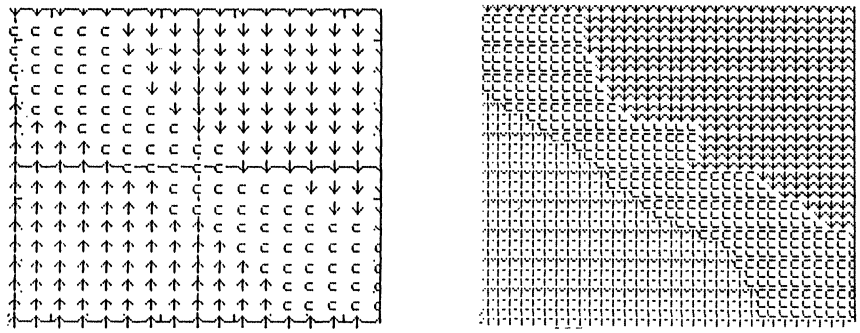

FIG. 13. *Saturation graph for two different meshes* ($M = 8$ *and* $M = 16$) *at* $t = 0$.

In Figs. 14–17 we show the three-dimensional representation of the minimum cost function viewed from different rotation (Twist) and elevation (Tilt) angles. Twist angle 0 corresponds to a South-to-North view. Tilt angle 0 corresponds to no elevation, i.e., a view from the ground. The symmetry with respect to the origin can easily be seen.

We would like to emphasize that the proposed second-order numerical scheme for degenerate problems under consideration is always stable. If regularization procedures, such as the penalty method, were used, we would have obtained a highly ill-conditioned computational problem.

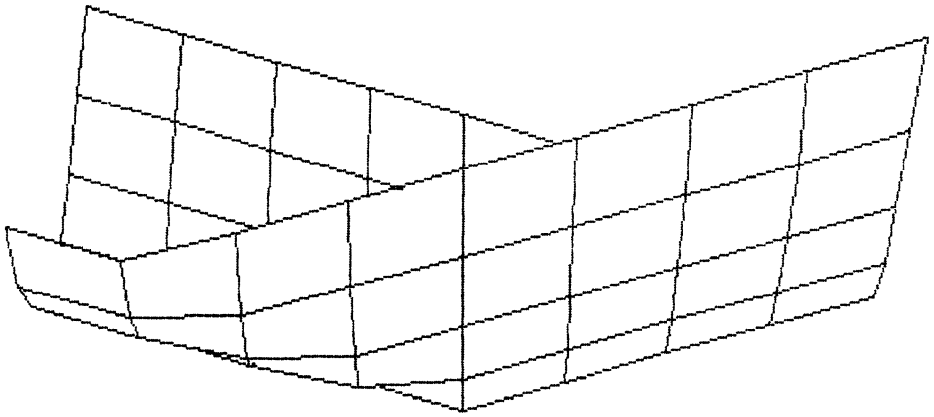

FIG. 14. *Minimum cost function at* $t = 0$. *Twist angle* $= 0$. *Tilt angle* $= 0$.

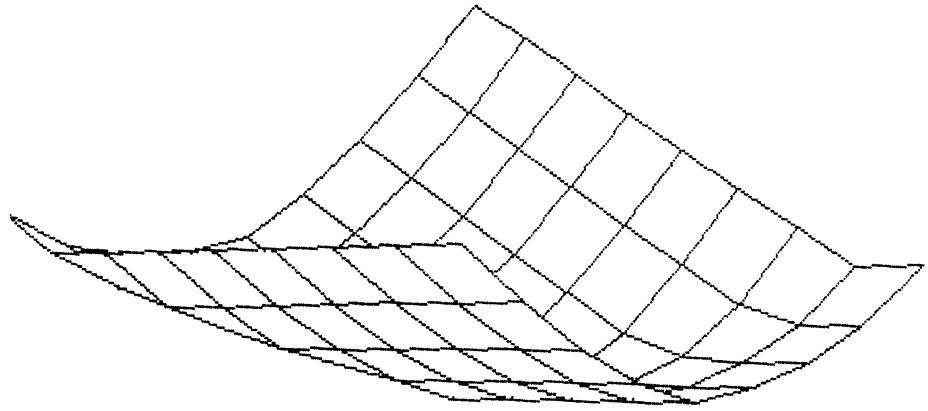

FIG. 15. *Minimum cost function at* $t = 0$. *Twist angle* $= 45$. *Tilt angle* $= 5$.

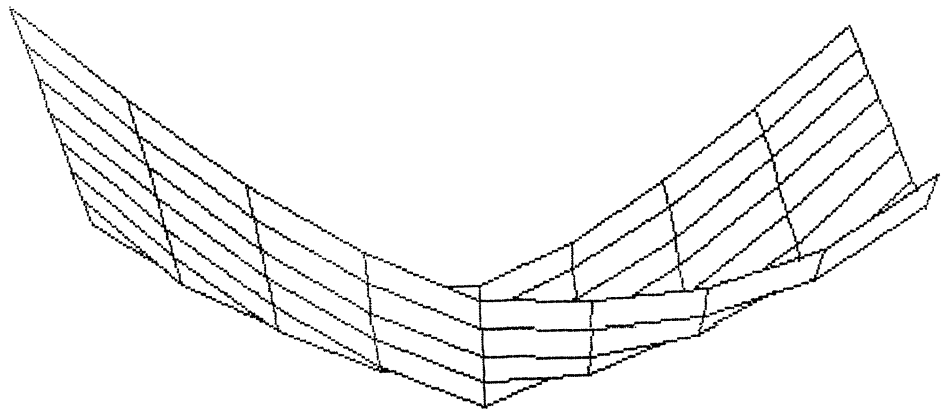

FIG. 16. *Minimum cost function at* $t = 0$. *Twist angle* $= 90$. *Tilt angle* $= 0$.

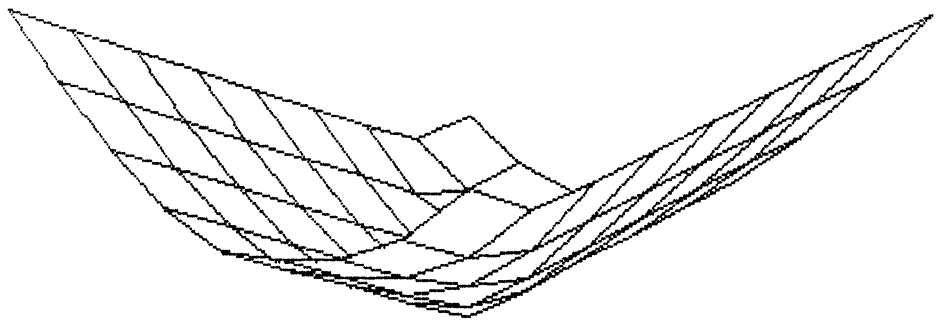

FIG. 17. *Minimum cost function at* $t = 0$. *Twist angle* $= -45$. *Tilt angle* $= 5$.

Also we wish to point out another advantage of the present scheme. Namely, our scheme is essentially "local" in the sense that the new value at each meshpoint is determined by the value of its three immediate, right and left, neighbors (see Fig. 2). Therefore the problem can be solved on computers of relatively small central memory. This "local" characteristic is, of course, important when we embark on a large-scale computation.

6. Final comments. Regarding the relation between the discrete problem (54) or (57) and the continuous problem (5), (6), (7), we would like to mention an error estimate of order $(\Delta t)^{1/2}$ with Δx, Δy dominated by Δt. That estimate (see Chow and Menaldi [4]) is based on the approximation of the diffusion process (2) by a Markov chain. From the probabilistic point of view, the approximations based on a weak convergence in measure were considered by Kushner [10] and Quadrat [13]. For a deterministic optimal control problem, an estimate of order Δt and other similar ones can be found in Capuzzo-Dolcetta [2], Falcone [6], Gonzalez and Rofman [8].

Apparently, problem (57) seems to approximate the continuous problem (5), (6), (7) better than problem (55). The reason is that problem (57) has an interpretation as an optimal control problem similar to the continuous problem (2), (3), (4), in which the diffusion process (2) is replaced by a Markov chain. This last argument also applies to the problem (55), but the Markov chain thus obtained involves the variables x, y, t.

In order to find the optimal feedback law, we need to know at each point (i, j, n) which of the three inequalities, in system (18), (19), (20) (respectively, (33), (34), (35)), is indeed an equality. When (18) (respectively, (33)) is an equality, no control is used. Otherwise an action, according to whether (19) or (20) becomes an equality, is taken in the appropriate direction.

Let us make some comments regarding the boundary conditions due to the use of a bounded mesh. In our example, we would get a better result by using some kind of asymptotic behavior of u at the boundary $y = \pm b$. Formally we argue as follows. For any x in $[-a, a]$ and $y = \pm b$ with b large enough, we have

$$\frac{\partial u}{\partial y} = c \quad \text{for } y = \pm b, \tag{58}$$

since the terminal cost $f(x, y)$ has a superlinear growth in y. On the other hand, for $x = a$, y in $[-b, b]$, we suppose

$$u(x, y, t) = u_0(y, t), \tag{59}$$

from which the inequalities (5) become

$$\begin{aligned} &\text{(i)} \quad \frac{\partial u_0}{\partial t} + L_y u_0 \geqq 0, \\ &\text{(ii)} \quad -c \leqq \frac{\partial u_0}{\partial y} \leqq c, \end{aligned} \tag{60}$$

where

$$L_y u_0(y, t) = \frac{1}{2} r^2 \frac{\partial^2 u_0}{\partial y^2}(y, t) - (py + q^2 a) \frac{\partial u_0}{\partial y}(y, t). \tag{61}$$

Similarly, for $x = -a$, y in $[-b, b]$, we replace a by $-a$ in the above expressions (60), (61). In other words, we impose the boundary condition

$$\frac{\partial u}{\partial x} = 0 \quad \text{for } x = \pm a, \tag{62}$$

which gives a first-order approximation. Some more complicated conditions can also be introduced. For instance, we may assume

$$u(x, y, t) = x|x|^{m-2} u_0(y, t) + |x|^m u_1(t), \tag{63}$$

where $f(x, y)$ grows like $|x|^m$ for large x. Then we obtain

$$Lu = x|x|^{m-2}\left\{\frac{1}{2}r^2\frac{\partial^2 u_0}{\partial y^2} - py\frac{\partial u_0}{\partial y} + myu_1\right\} - q^2|x|^m\frac{\partial u_0}{\partial y} + (m-1)y|x|^{m-2}u_1.$$

By keeping only the terms with mth power in x, we deduce the inequalities

$$\text{(i)} \quad L_0u_0 + \frac{\partial u_0}{\partial t} + myu_1 \geqq 0 \quad (\leqq 0 \text{ if } x < 0),$$

$$\text{(ii)} \quad -q^2\frac{\partial u_0}{\partial y} + \frac{\partial u_1}{\partial t} \geqq 0, \tag{64}$$

$$\text{(iii)} \quad -c \leqq \frac{\partial u_0}{\partial y} \leqq c$$

with the complementary condition: If one of the inequalities of (iii) is strict, then (i) as well as (ii) are equalities. So, for $x = \mp a$, y in $[-b, b]$ with a sufficiently large x, we suppose that (64) holds, subject to the above complementary conditions and an initial condition at $t = 0$.

Finally, we wish to point out some possible extension of the proposed algorithms. Suppose the differential operator L is of the form

$$L = \frac{1}{2}\sigma^2(x, y)\frac{\partial^2}{\partial y^2} + p(x, y)\frac{\partial}{\partial y} + q(x, y)\frac{\partial}{\partial x}. \tag{65}$$

Then, we can discretize it similarly to (13), (14), where r is replaced by $\sigma(i, \Delta x, j\Delta y)$ and

$$p(x, y)\frac{\partial}{\partial y} \approx |p(i\Delta x, j\Delta y)|[(u^n_{i,j-1} - u^n_{i,j})\Pi(p(i\Delta x, j\Delta y) < 0) + (u^n_{i,j+1} - u^n_{i,j})\Pi(p(i\Delta x, j\Delta y) > 0)],$$

$$q(x, y)\frac{\partial}{\partial x} \approx |q(i\Delta x, j\Delta y)|[(u^n_{i-1,j} - u^n_{i,j})\Pi(q(i\Delta x, j\Delta y) < 0) + (u^n_{i+1,j} - u^n_{i,j})\Pi(q(i\Delta x, j\Delta y) > 0)].$$

Analogous extensions can be worked out for higher-dimensional problems.

REFERENCES

[1] M. C. BANCORA-IMBERT, R. GONZALEZ, C. MIELLOU, AND E. ROFMAN, *Numerical optimization of energy-production systems*, Rapport de Recherche No. 306, INRIA, Le Chesnay, France, May 1984.

[2] I. CAPUZZO-DOLCETTA, *On a discrete approximation of Hamilton-Jacobi equation of dynamic programming*, Appl. Math. Optim., 10 (1983), pp. 367-377.

[3] P. L. CHOW AND J. L. MENALDI, *Optimal corrections of a damped linear oscillator under random perturbations*, Transaction of the Second Army Conference on Applied Mathematics and Computing, May 1984, Troy, New York, ARO Report 85-1, pp. 149-158.

[4] ———, *On the numerical solution of a stochastic optimal correction problem*, Transaction of the Third Army Conference on Applied Mathematics and Computing, May 1985, Atlanta, Georgia, ARO Report 86-1, pp. 531-558.

[5] P. L. CHOW, J. L. MENALDI, AND M. ROBIN, *Additive control of stochastic linear systems with finite horizon*, SIAM J. Control Optim., 23 (1985), pp. 858-899.

[6] M. FALCONE, *A numerical approach to the infinite horizon problem of deterministic control theory*, Appl. Math. Optim., 15 (1987), pp. 1-14.

[7] I. I. GIHMAN AND A. V. SKOROHOD, *Stochastic Differential Equations*, Springer-Verlag, New York, Berlin, 1971.

[8] R. GONZALEZ AND E. ROFMAN, *On deterministic control problems: an approximation procedure for optimal cost* I. *The stationary problem*, II. *The nonstationary case*, SIAM J. Control Optim., 23 (1985), pp. 242–285.

[9] V. K. GORBUNOV, *Minimax impulsive correction of perturbations of a linear damped oscillator*, Appl. Math. Mech. (PPM), 40 (1976), pp. 252–259.

[10] H. KUSHNER, *Probability Methods for Approximations in Stochastic Control and for Elliptic Equations*, Academic Press, New York, 1977.

[11] J. L. MENALDI AND M. ROBIN, *On some cheap control problems for diffusion process*, Trans. Amer. Math. Soc., 278 (1983), pp. 771–802.

[12] ———, *On singular stochastic control problems for diffusions with jumps*, IEEE Trans. Automat. Control, 29 (1984), pp. 991–1004.

[13] J. P. QUADRAT, *Existence de solution et algorithme de résolution numérique de problème de contrôle optimal de diffusion stochastique dégénérée ou non*, SIAM J. Control Optim., 18 (1980), pp. 119–226.

[14] M. SUN AND J. L. MENALDI, *Monotone control of a damped oscillator under random perturbations*, J. Math. Control and Inform., to appear.