PERGAMON

Nonlinear Analysis 48 (2002) 637–650

Nonlinear Analysis

www.elsevier.com/locate/na

Fully explicit finite-difference methods for two-dimensional diffusion with an integral condition

Mehdi Dehghan

Department of Applied Mathematics, Faculty of Mathematics and Computer Science, Amirkabir University of Technology, No. 424 Hafez Avenue, Tehran, Iran

Received 7 October 1999; accepted 29 February 2000

Dedicated to Professor John R. Cannon on the occasion of his 62nd birthday

Keywords: Fully explicit schemes; Heat equation; Finite-difference techniques; Time dependent; Numerical integration procedures; Two dimensional; Central processor time; Partial differential equations

1. Introduction

Heat conduction and other diffusion processes with an integral condition replacing one boundary condition arises in many important applications in heat transfer [1–3,9,11,15,17], control theory [16], thermoelasticity [5,6] and medical science [4].

The purpose of this article is to present very efficient explicit second- and fourth-order-accurate finite-difference methods for solving the following two-dimensional time-dependent diffusion equation:

$$\frac{\partial u}{\partial t} = \alpha\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{1}$$

with initial condition

$$u(x, y, 0) = f(x, y), \quad 0 \le x, y \le 1, \tag{2}$$

and boundary conditions

$$u(0,y,t)=g_0(y,t), \quad 0\le t\le T, \quad 0\le y\le 1, \tag{3}$$

$$u(1,y,t)=g_1(y,t), \quad 0\le t\le T, \quad 0\le y\le 1, \tag{4}$$

$$u(x,1,t)=h_1(x,t), \quad 0\le t\le T, \quad 0\le x\le 1, \tag{5}$$

$$u(x,0,t)=h_0(x)\mu(t), \quad 0\le t\le T, \quad 0\le x\le 1, \tag{6}$$

and the integral condition

$$\int_0^1\int_0^{s(x)} u(x,y,t)\,\mathrm{d}x\,\mathrm{d}y = m(t), \quad 0\le x,y\le 1, \tag{7}$$

where f, g_0, g_1, h_0, h_1, s and m are known functions, while the functions u and μ are unknown.

The boundary condition (6) is variable separable, with spatial dependence given by $h_0(x)$ and time dependence given by $\mu(t)$.

The existence and uniqueness of the solution of this non-classic problem has been studied in [3]. The organization of this paper is as follows:

Numerical schemes for the solution of (1)–(7) are described in Section 2. The method of incorporating (7) with μ unknown is described in Section 3, and numerical results for various test cases produced by the methods developed, are given in Section 4.

In each case error estimates are given in the maximum norm. Section 5 concludes this paper with a brief summary.

2. The numerical solution with Dirichlet boundary conditions

We divide the domain $[0,1]^2\times[0,T]$ into $M^2\times N$ mesh with spatial step size $h=1/M$ in both x and y directions and the time step size $k=T/N$, respectively. Grid points (x_i,y_j,t_n) are given by

$$x_i=ih, \quad i=0,1,2,\ldots,M, \tag{8}$$

$$y_j=jh, \quad j=0,1,2,\ldots,M, \tag{9}$$

$$t_n=nk, \quad n=0,1,2,\ldots,N, \tag{10}$$

where M is an even integer. We use $u^n_{i,j}$ and μ^n to denote the finite-difference approximations of $u(ih,jh,nk)$ and $\mu(nk)$, respectively. The numerical methods suggested here are based on two ideas: Firstly, the standard second-order FTCS method, or the 9-point finite-difference scheme or the $(1,13)$ fully explicit technique is used to approximate the solution of the two-dimensional diffusion equation, at interior grid points. Secondly, the Simpson's numerical integration scheme [10] is used to evaluate the integral in Eq. (7) approximately.

Using the initial condition

$$u(x,y,0)=f(x,y), \quad 0\le x,y\le 1, \tag{11}$$

Eq. (1) is solved numerically at the spatial points (x_i,y_j), commencing with initial values $u^0_{i,j}=f(x_i,y_j), i,j=0,1,2,\dots,M$, and boundary values

$$u^{n+1}_{0,j}=g_0(y_j,t_{n+1}), \quad j=0,1,2,\dots,M, \tag{12}$$

$$u^{n+1}_{M,j}=g_1(y_j,t_{n+1}), \quad j=0,1,2,\dots,M, \tag{13}$$

$$u^{n+1}_{i,M}=h_1(x_i,t_{n+1}), \quad i=0,1,2,\dots,M, \tag{14}$$

$$u^{n+1}_{i,0}=h_0(x_i)\mu(t_{n+1}), \quad i=0,1,2,\dots,M, \tag{15}$$

for $n=0,1,2,\dots,N-1$, where $h_0(x)$ is given and $\mu(t)$ will be found approximately by using the technique which is described in Section 3.

2.1. The standard FTCS method

This scheme uses the forward-difference approximation for the time derivative and the centred-difference approximation for space derivatives applied at the point (ih,jh,nk), in the following form:

$$\left.\frac{\partial u}{\partial t}\right|^n_{i,j}=\alpha\left.\frac{\partial^2 u}{\partial x^2}\right|^n_{i,j}+\alpha\left.\frac{\partial^2 u}{\partial y^2}\right|^n_{i,j}, \tag{16}$$

$$\frac{u^{n+1}_{i,j}-u^n_{i,j}}{k}=\alpha\frac{(u^n_{i+1,j}-2u^n_{i,j}+u^n_{i-1,j})}{(\Delta x)^2}+\alpha\frac{(u^n_{i,j+1}-2u^n_{i,j}+u^n_{i,j+1})}{(\Delta y)^2}. \tag{17}$$

It follows that

$$u^{n+1}_{i,j}=s_x(u^n_{i-1,j}+u^n_{i+1,j})+s_y(u^n_{i,j-1}+u^n_{i,j+1})+(1-2s_x-2s_y)u^n_{i,j}, \tag{18}$$

for $i,j=1,2,\dots,M-1$, where

$$s_x=\alpha k/(\Delta x)^2, \tag{19}$$

$$s_y=\alpha k/(\Delta y)^2. \tag{20}$$

In the case

$$s_x=s_y=s, \tag{21}$$

(18) becomes

$$u^{n+1}_{i,j}=s(u^n_{i-1,j}+u^n_{i+1,j}+u^n_{i,j-1}+u^n_{i,j+1})+(1-4s)u^n_{i,j}. \tag{22}$$

Values of $u^{n+1}_{i,j}$ on the boundaries $x=0,1$ and $y=0,1$ are provided by the boundary conditions (3)–(6), at the appropriate grid points.

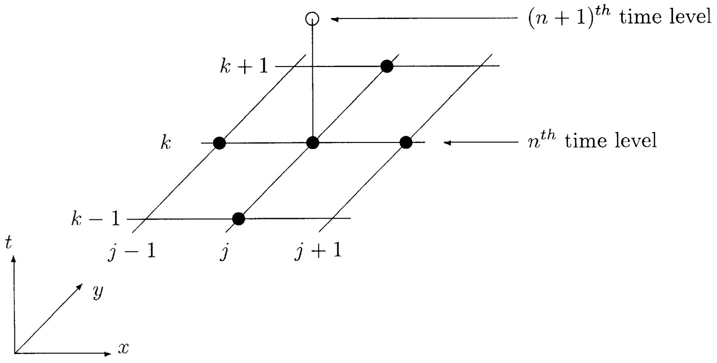

Fig. 1. The standard FTCS computational molecule.

The computational molecule of this method is given in Fig. 1. In the following this will be referred to as the (1,5) method, because the computational molecule involves one gridpoint at the new time level and five at the old level.

The range of stability for this procedure is [12]

$$0 < s \le 1/4. \tag{23}$$

It can be seen that the modified equivalent partial differential equation (MEPDE) of the (1,5) FTCS formula (22) is [18]

$$\frac{\partial u}{\partial t} - \alpha\frac{\partial^2 u}{\partial x^2} - \alpha\frac{\partial^2 u}{\partial y^2} + \frac{\alpha(\Delta x)^2}{12}(6s-1)\frac{\partial^4 u}{\partial x^4}$$
$$+\frac{\alpha\Delta x\Delta y}{2}s^2\frac{\partial^4 u}{\partial^2 x\partial y^2} + \frac{\alpha(\Delta y)^2}{12}(6s-1)\frac{\partial^4 u}{\partial y^4} + O(4) = 0. \tag{24}$$

Formula (22) is second-order-accurate for all $s > 0$ as it can be seen by the modified equivalent partial differential equation analysis.

2.2. The 9-point finite-difference method

The 9-point fully explicit finite-difference method uses a forward-difference approximation for the time derivative and the following weighted approximations for the spatial derivatives:

$$\left.\frac{\partial u}{\partial t}\right|_{i,j}^{n} = \frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{k}, \tag{25}$$

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{i,j}^{n} = \frac{s_y}{2}\times\left(\frac{(u_{i+1,j-1}^{n} - 2u_{i,j-1}^{n} + u_{i-1,j-1}^{n})}{(\Delta x)^2}\right.$$
$$\left. + \frac{(u_{i+1,j+1}^{n} - 2u_{i,j+1}^{n} + u_{i-1,j+1}^{n})}{(\Delta x)^2}\right)$$

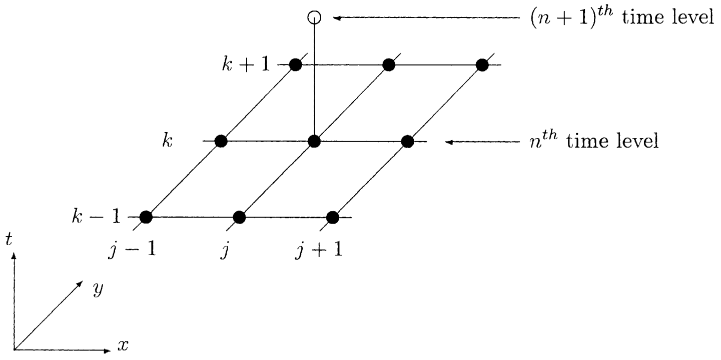

Fig. 2. The 9-point finite-difference computational molecule.

$$+(1-s_y)\times\frac{(u^n_{i+1,j}-2u^n_{i,j}+u^n_{i-1,j})}{(\Delta x)^2}, \tag{26}$$

$$\left.\frac{\partial^2 u}{\partial y^2}\right|^n_{i,j}=\frac{s_x}{2}\times\left(\frac{(u^n_{i-1,j+1}-2u^n_{i-1,j}+u^n_{i-1,j-1})}{(\Delta y)^2}\right.$$
$$\left.+\frac{(u^n_{i+1,j+1}-2u^n_{i+1,j}+u^n_{i+1,j-1})}{(\Delta y)^2}\right)$$
$$+(1-s_x)\times\frac{(u^n_{i,j+1}-2u^n_{i,j}+u^n_{i,j-1})}{(\Delta y)^2}. \tag{27}$$

These approximations produce the following finite-difference equation:

$$u^{n+1}_{i,j}=s_xs_y(u^n_{i-1,j-1}+u^n_{i-1,j+1}+u^n_{i+1,j-1}+u^n_{i+1,j+1})$$
$$+s_y(1-2s_x)(u^n_{i,j-1}+u^n_{i,j+1})+s_x(1-2s_y)(u^n_{i-1,j}+u^n_{i+1,j})$$
$$+(1-2s_x)(1-2s_y)u^n_{i,j}. \tag{28}$$

In the case $s_x=s_y=s$, (28) becomes

$$u^{n+1}_{i,j}=s^2(u^n_{i-1,j-1}+u^n_{i-1,j+1}+u^n_{i+1,j-1}+u^n_{i+1,j+1})$$
$$+s(1-2s)(u^n_{i,j-1}+u^n_{i,j+1}+u^n_{i-1,j}+u^n_{i+1,j})+(1-2s)^2u^n_{i,j}. \tag{29}$$

The computational molecule of this method is given in Fig. 2. In the following this is referred to as the (1,9) method because the computational molecule involves one gridpoint at the new time level and nine at the old level.

The MEPDE for this procedure is given by

$$\frac{\partial u}{\partial t}-\alpha\frac{\partial^2 u}{\partial x^2}-\alpha\frac{\partial^2 u}{\partial y^2}+\frac{\alpha(\Delta x)^2}{12}(6s-1)\frac{\partial^4 u}{\partial x^4}+\frac{\alpha(\Delta y)^2}{12}(6s-1)\frac{\partial^4 u}{\partial y^4}$$

$$-\frac{\alpha(\Delta x)^4}{360}(1-30s+120s^2)\frac{\partial^6 u}{\partial x^6}$$

$$-\frac{\alpha(\Delta y)^4}{360}(1-30s+120s^2)\frac{\partial^6 u}{\partial y^6}+O(6)=0, \tag{30}$$

which has the same second-order error terms as the (1,5) FTCS, without the term involving the cross-derivatives $\partial^4 u/\partial x^2\partial y^2$.

The modified equivalent partial differential equation analysis shows that this is generally second-order accurate, but is fourth-order accurate in the special case $s = 1/6$.

The range of stability corresponding to (29) is [13].

$$0 < s \leq 1/2, \tag{31}$$

which is less restrictive than (23).

The 9-point fully explicit finite-difference formula takes more central processor time than the standard second-order FTCS scheme, because of the inclusion of four additional grid points in the computational molecule, but it has two advantages: firstly, there is the optimal case $s_x = s_y = 1/6$ when it is fourth-order accurate, and secondly, it has a much greater stability range in the (s_x, s_y) plane.

2.3. The (1,13) fully explicit method

This scheme uses the forward-difference approximation for the time derivative and the weighted differencing approximation for the spatial derivatives at the point (ih, jh, nk), in the following form:

$$\left.\frac{\partial u}{\partial t}\right|_{i,j}^{n} = \alpha\left.\frac{\partial^2 u}{\partial x^2}\right|_{i,j}^{n} + \alpha\left.\frac{\partial^2 u}{\partial y^2}\right|_{i,j}^{n}, \tag{32}$$

$$\left.\frac{\partial u}{\partial t}\right|_{i,j}^{n} = \frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{k}, \tag{33}$$

$$\begin{aligned}\left.\frac{\partial^2 u}{\partial x^2}\right|_{i,j}^{n} &= \frac{s_y}{2}\times\left(\frac{(u_{i+1,j-1}^{n}-2u_{i,j-1}^{n}+u_{i-1,j-1}^{n})}{(\Delta x)^2}\right.\\ &\quad\left.+\frac{(u_{i+1,j+1}^{n}-2u_{i,j+1}^{n}+u_{i-1,j+1}^{n})}{(\Delta x)^2}\right)\\ &\quad+(1-6s_x)\times\frac{(u_{i+1,j}^{n}-2u_{i,j}^{n}+u_{i-1,j}^{n})}{(\Delta x)^2}\\ &\quad+(6s_x-s_y)\times\frac{(-u_{i+2,j}^{n}+16u_{i+1,j}^{n}-30u_{i,j}^{n}+16u_{i-1,j}^{n}-u_{i-2,j}^{n})}{12(\Delta x)^2},\end{aligned} \tag{34}$$

$$\left.\frac{\partial^2 u}{\partial y^2}\right|_{i,j}^{n} = \frac{s_x}{2} \times \left(\frac{(u_{i-1,j+1}^n - 2u_{i-1,j}^n + u_{i-1,j-1}^n)}{(\Delta y)^2} + \frac{(u_{i+1,j+1}^n - 2u_{i+1,j}^n + u_{i+1,j-1}^n)}{(\Delta y)^2} \right) + (1-6s_y) \times \frac{(u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n)}{(\Delta y)^2} + (6s_y - s_x) \times \frac{(-u_{i,j-2}^n + 16u_{i,j+1}^n - 30u_{i,j}^n + 16u_{i,j-1}^n - u_{i,j-2}^n)}{12(\Delta y)^2}. \quad (35)$$

In the case $s_x = s_y = s$, this yields the following finite-difference equation:

$$u_{i,j}^{n+1} = \frac{s(1-6s)}{12}(u_{i-2,j}^n + u_{i+2,j}^n + u_{i,j-2}^n + u_{i,j+2}^n) - s^2(u_{i-1,j-1}^n + u_{i-1,j+1}^n + u_{i+1,j-1}^n + u_{i+1,j+1}^n) + \frac{4s(3s-1)}{3}(u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n) + (5s - 10s^2 - 1)u_{i,j}^n. \quad (36)$$

Before this method can be used in practice, however the problem of computing values of $u_{i,j}^{n+1}$, at interior grid points adjacent to the boundaries of the solution domain must be overcome. The difficulty arises when Eq. (36) is applied at a grid point next to a boundary, such as when $i=1$, with $j=1,2,\ldots,M-1$, as the equation then references a value at the exterior grid point $(-\Delta x, M\Delta y)$, in this case the value of $u_{-1,j}^n$.

Several techniques may be used in order to overcome this problem. One procedure is to use extrapolation to approximate the exterior value with the necessary accuracy. Thus we extrapolate a value for $u_{-1,j}^n$, by means of the formula [14]

$$u_{-1,j}^n = 6u_{0,j}^n - 15u_{1,j}^n + 20u_{2,j}^n - 15u_{3,j}^n + 6u_{4,j}^n - u_{5,j}^n. \quad (37)$$

Given this value of $u_{-1,j}^n$, the above equation can be used to find the required values of $u_{1,j}^{n+1}$ at the new time levels. This approach is easy to implement and use little CPU times.

The computational molecule of this method is given in Fig. 3. In the following this is referred to as the (1,13) method because the computational molecule involves one gridpoint at the new time level and 13 at the old level.

The modified equivalent partial differential equation for the (1,13) fully explicit formula which is consistent with the two-dimensional diffusion equation (1), is

$$\frac{\partial u}{\partial t} - \alpha\frac{\partial^2 u}{\partial x^2} - \alpha\frac{\partial^2 u}{\partial y^2} - \frac{\alpha(\Delta x)^4}{12}(1+24s+120s^2)\frac{\partial^6 u}{\partial x^6} - \frac{\alpha(\Delta y)^4}{12}(1+24s+120s^2)\frac{\partial^6 u}{\partial y^6} + O(6) = 0. \quad (38)$$

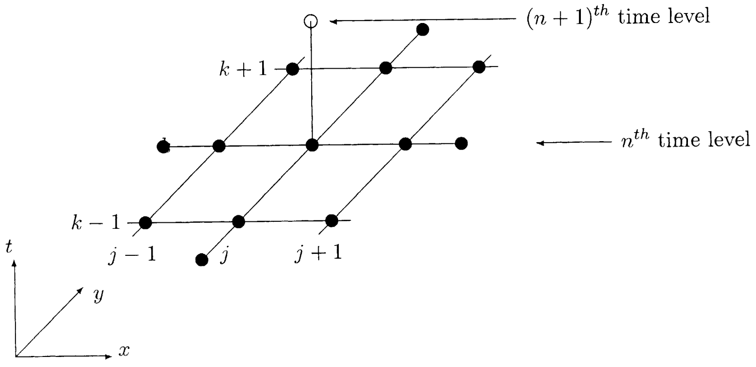

Fig. 3. The (1,13) fully explicit computational molecule.

The modified equivalent partial differential equation analysis shows that this scheme, is fourth-order accurate.

The range of stability for this procedure is

$$0 < s \leq 1/3, \tag{39}$$

which is larger than that for the second-order (1,5) FTCS formula, and smaller than that for the 9-point finite-difference scheme.

The (1,13) fully explicit formula takes more central processor time than the 9-point finite-difference scheme because of the inclusion of some additional grid points in the computational molecule, but its fourth-order accuracy is significant.

3. The numerical integration procedure

In this section we describe the numerical integration procedure, used for (7) employing Simpson's rule which has a truncation error of order four.

The presence of an integral term in a boundary condition can greatly complicate the application of standard numerical techniques [8]. The accuracy of the quadrature must be compatible with that of the discretization of the differential equation [7]. In the following the fourth-order Simpson's composite "one-third" rule is used.

Consider the integral

$$H(x,t^{n+1}) = \int_0^{s(x)} u(x,y,t^{n+1})\,\mathrm{d}y. \tag{40}$$

Application of Simpson's composite "one-third" rule gives

$$\int_0^1 H(x,t^{n+1})\,\mathrm{d}x \simeq \frac{h}{3}\left(H_0^{n+1} + 4\sum_{i=1}^{M/2} H_{2i-1}^{n+1} + 2\sum_{i=1}^{(M/2)-1} H_{2i}^{n+1} + H_M^{n+1}\right), \tag{41}$$

$$H_i^{n+1} = \int_0^{s(ih)} u(x_i,y,t^{n+1})\,\mathrm{d}y, \tag{42}$$

in which

$$H_i^{n+1} = \int_0^{2l_ih} u(x_i, y, t^{n+1})\,\mathrm{d}y + \int_{2l_ih}^{s(ih)} u(x_i, y, t^{n+1})\,\mathrm{d}y, \tag{43}$$

where

$$l_i = [s(ih)/2h], \tag{44}$$

and $[\cdot]$ represents the integer part of the argument. Substituting in the second integral of (43)

$$z_i = y/h - 2l_i, \tag{45}$$

yields

$$\int_{2l_ih}^{s(ih)} u(x_i, y, t^{n+1})\,\mathrm{d}y = h\int_0^{\delta_i} u(x_i, z_i, t^{n+1})\,\mathrm{d}z_i, \tag{46}$$

where

$$\delta_i = (s(ih)/h) - 2l_i. \tag{47}$$

Replacement of u in the integral with a quadratic interpolating polynomial (the Newton's forward-difference formula) [15] through the grid values concerned, gives

$$\begin{aligned}\int_0^{\delta_i} u(x_i, z_i, t^{n+1})\,\mathrm{d}z_i = \int_0^{\delta_i} &[u(x_i, y_{2l_i}, t^{n+1}) + z_i\Delta u(x_i, y_{2l_i}, t^{n+1}) \\ &+ \frac{1}{2}z_i(z_i-1)\Delta^2 u(x_i, y_{2l_i}, t^{n+1})]\,\mathrm{d}z_i + O(h^4),\end{aligned} \tag{48}$$

where

$$\Delta u(x_i, y_{2l_i}, t^{n+1}) = u(x_i, y_{2l_{i+1}}, t^{n+1}) - u_(x_i, y_{2l_i}, t^{n+1}) \tag{49}$$

and

$$\Delta^2 u(x_i, y_{2l_i}, t^{n+1}) = u(x_i, y_{2l_{i+2}}, t^{n+1}) - 2u(x_i, y_{2l_{i+1}}, t^{n+1}) + u(x_i, y_{2l_i}, t^{n+1}). \tag{50}$$

Integrating (49) and collecting like terms then substituting in (43) means that

$$\begin{aligned}H_i^{n+1} = \frac{h}{3}[&u(x_i, 0, t^{n+1}) + 4\sum_{j=1}^{l_i} u(x_i, y_{2j-1}, t^{n+1}) + 2\sum_{j=1}^{l_i-1} u(x_i, y_{2j}, t^{n+1}) \\ &+ u(x_i, y_{2l_i}, t^{n+1}) + 3\delta_i(1 - 3\delta_i/4 + \delta_i^2/6)u(x_i, y_{2l_i}, t^{n+1}) \\ &+ 3\delta_i^2(1 - \delta_i/3)u(x_i, y_{2l_i+1}, t^{n+1}) \\ &+ (\delta_i^2/4)(2\delta_i - 3)u(x_i, y_{2l_i+2}, t^{n+1})] + O(h^4).\end{aligned} \tag{51}$$

However, at all interior points we have computed $u(x_i, y_j, t^{n+1})$ to say, rth order, where

$$u(x_i, y_{2l_i}, t^{n+1}) = u_{i,j}^{n+1} + O(h^r). \tag{52}$$

Substituting these approximations in (51) gives

$$H_i^{n+1}=\frac{h}{3}[u_{i,0}^{n+1}+4\sum_{j=1}^{l_i}u_{i,2j-1}^{n+1}+2\sum_{j=1}^{l_i-1}u_{i,2j}^{n+1}+u_{i,2l_i}^{n+1}$$

$$+3\delta_i(1-3\delta_i/4+\delta_i^2/6)u_{i,2l_i}^{n+1}+3\delta_i^2(1-\delta_i/3)u_{i,2l_i+1}^{n+1}$$

$$+(\delta_i^2/4)(2\delta_i-3)u_{i,2l_i+2}^{n+1}]+O(h^q), \tag{53}$$

where

$$q=\min\{r,4\}. \tag{54}$$

Putting

$$V(t^{n+1})=\int_0^1 H(x,t^{n+1})\,\mathrm{d}x, \tag{55}$$

and using the approximation

$$v^{n+1}=\frac{h}{3}\left(H_0^{n+1}+4\sum_{i=1}^{M/2}H_{2i-1}^{n+1}+2\sum_{i=1}^{(M/2)-1}H_{2i}^{n+1}+H_M^{n+1}\right)+O(h^q), \tag{56}$$

then gives

$$v^{n+1}=\frac{h^2}{9}\left(u_{0,0}^{n+1}+4\sum_{i=1}^{M}u_{2i-1,0}^{n+1}+2\sum_{i=1}^{(M/2)-1}u_{2i,0}^{n+1}+u_{M,0}^{n+1}\right)+R^{n+1}+O(h^q). \tag{57}$$

Note that

$$v^{n+1}=\frac{h}{3}\mu^{n+1}\int_0^1 h_0(x)\,\mathrm{d}x+R^{n+1}+O(h^q), \tag{58}$$

where R^{n+1} is the summation in v^{n+1} excluding the values at the boundary $y=0$.

Since v^{n+1} is an approximation to the left-hand side of (7) it follows that

$$\mu^{n+1}=\frac{m^{n+1}-R^{n+1}}{\frac{h}{3}\int_0^1 h_0(x)\,\mathrm{d}x}+O(h^{q-1}),\qquad \int_0^1 h_0(x)\,\mathrm{d}x\neq 0. \tag{59}$$

As seen above, the order of convergence of μ depends on two things: firstly, the order of the finite-difference formula used at interior gridpoints and, secondly, the order of the numerical quadrature used to approximately evaluate (7). For example, if u^{n+1} is evaluated using a fourth-order formula, when $q=4$, μ^{n+1} is only third-order convergent. If u^{n+1} is found at interior points by a second formula when $q=2$ then μ^{n+1} is only first-order convergent.

4. Numerical tests

Two problems for which exact solutions are known are now used to test the methods described. Firstly, these are applied to solve (1)–(6) with $\mu(t)$ given, in order to test

Table 1
Results for u from Test 1 with $T = 1.0$, $h = 0.05$, $s = 1/4$

			Standard FTCS	9-point	(1,13)	Standard BTCS
x	y	Exact u	Error	Error	Error	Error
0.1	0.1	2.703749	2.2×10^{-4}	2.1×10^{-4}	1.9×10^{-8}	-1.2×10^{-2}
0.2	0.2	2.656094	2.3×10^{-4}	2.3×10^{-4}	2.7×10^{-8}	-1.3×10^{-2}
0.3	0.3	2.568508	2.3×10^{-4}	2.2×10^{-4}	2.8×10^{-8}	-1.3×10^{-2}
0.4	0.4	2.433129	2.0×10^{-4}	2.0×10^{-4}	2.8×10^{-8}	-1.1×10^{-2}
0.5	0.5	2.240845	1.6×10^{-4}	1.5×10^{-4}	2.9×10^{-8}	-8.5×10^{-3}
0.6	0.6	1.981213	1.1×10^{-4}	1.0×10^{-4}	2.5×10^{-8}	-5.8×10^{-3}
0.7	0.7	1.642184	6.8×10^{-5}	6.6×10^{-5}	2.3×10^{-8}	-2.1×10^{-3}
0.8	0.8	1.209929	3.2×10^{-5}	3.0×10^{-5}	1.3×10^{-8}	-8.0×10^{-4}
0.9	0.9	0.668589	8.2×10^{-6}	8.1×10^{-6}	1.0×10^{-8}	-3.4×10^{-4}

the two methods used to compute values of $u_{i,j}^{n+1}$ from $u_{i,j}^{n}$, in the interior of the solution domain. Secondly, these are applied to solve (1)–(7) with $\mu(t)$ not given a priori, thereby testing the algorithm used for the two-dimensional time-dependent problem with an integral condition replacing one boundary condition.

Test 1: Consider (1)–(7) with $\alpha = 1$ and

$$f(x,y) = (1-y)\exp(x), \tag{60}$$

$$g_0(y,t) = (1-y)\exp(t), \tag{61}$$

$$g_1(y,t) = (1-y)\exp(1+t), \tag{62}$$

$$h_0(x) = \exp(x), \tag{63}$$

$$h_1(x) = 0, \tag{64}$$

$$\mu(t) = \exp(t), \tag{65}$$

$$m(t) = 2(11 - 4\exp(1))\exp(t), \tag{66}$$

$$s(x) = x(1-x), \tag{67}$$

for which the exact solution is

$$u(x,y,t) = (1-y)\exp(x+t). \tag{68}$$

The results for $u_{i,j}^{N}$ with $h = 0.05$, $s = 1/4$ at $T = 1.0$, using the three fully explicit finite-difference schemes discussed in Section 2 and defining $\mu(t)$ as in (65) and excluding (66), are shown in Table 1.

Note that the errors obtained when using the (1,13) fully explicit method are generally more than a thousand times smaller than those obtained using the standard FTCS scheme.

Table 2
Results for μ from Test 1 with $h = 0.05$, $s = 1/4$

		Standard FTCS	9-point	(1,13)	Standard BTCS
t	Exact μ	Error	Error	Error	Error
0.1	1.105171	1.5×10^{-4}	1.4×10^{-4}	1.1×10^{-7}	-1.5×10^{-2}
0.2	1.221403	1.6×10^{-4}	1.6×10^{-4}	1.0×10^{-7}	-1.7×10^{-2}
0.3	1.349859	1.8×10^{-4}	1.8×10^{-4}	1.3×10^{-7}	-1.9×10^{-2}
0.4	1.491825	2.0×10^{-4}	2.1×10^{-4}	1.9×10^{-7}	-2.0×10^{-2}
0.5	1.648721	2.2×10^{-4}	2.2×10^{-4}	2.1×10^{-7}	-2.3×10^{-2}
0.6	1.822119	2.5×10^{-4}	2.5×10^{-4}	2.4×10^{-7}	-2.6×10^{-2}
0.7	2.013753	2.7×10^{-4}	2.7×10^{-4}	2.9×10^{-7}	-2.9×10^{-2}
0.8	2.225541	3.0×10^{-4}	3.1×10^{-4}	2.5×10^{-7}	-3.1×10^{-2}
0.9	2.459603	3.3×10^{-4}	3.4×10^{-4}	1.8×10^{-7}	-3.5×10^{-2}
1.0	2.718282	3.7×10^{-4}	3.5×10^{-4}	1.0×10^{-7}	-3.8×10^{-2}

The central processor time (CPU) for the standard FTCS technique was 8.9 s and for the (1,9) FTCS method was 12.3 s and for the (1,13) fully explicit formula was 16.4 s, while the time for the standard implicit method (BTCS) of [3] was 1632 s.

The results obtained for μ with $h = 0.05$, $s = 1/4$, using the three schemes developed with $m(t)$ defined as in (66), and $\mu(t)$ considered to be unknown and found by (59), are shown in Table 2.

Note that the errors with the (1,13) fully explicit finite-difference technique are less than one-thousandth of the errors obtained using the standard FTCS method.

Test 2: Consider (1)–(7) with $\alpha = 1$ and

$$f(x, y) = \exp(x + y), \tag{69}$$

$$g_0(y, t) = \exp(y + 2t), \tag{70}$$

$$g_1(y, t) = \exp(1 + y + 2t), \tag{71}$$

$$h_0(x) = \exp(x), \tag{72}$$

$$h_1(x) = \exp(1 + x + 2t), \tag{73}$$

$$\mu(t) = \exp(2t), \tag{74}$$

$$m(t) = (4\exp(\exp(1)/4) - 4\exp(1/4) - \exp(1) + 1)\exp(2t), \tag{75}$$

$$s(x) = \exp(x)/4, \tag{76}$$

for which the exact solution is

$$u(x, y, t) = \exp(x + y + 2t). \tag{77}$$

The results for $u_{i,j}^N$ with $h = 0.02$, $s = 1/4$ at $T = 1.0$, using the standard FTCS technique and the 9-point finite-difference scheme and the (1,13) fully explicit formula

Table 3
Results for u from Test 2 with $T = 1.0$, $h = 0.05$, $s = 1/4$

			Standard FTCS	9-point	(1,13)	Standard BTCS
x	y	Exact u	Error	Error	Error	Error
0.1	0.1	9.025013	-2.2×10^{-4}	-5.4×10^{-5}	1.6×10^{-8}	1.9×10^{-2}
0.2	0.2	11.023176	$+3.1 \times 10^{-5}$	1.0×10^{-5}	1.9×10^{-8}	1.9×10^{-2}
0.3	0.3	13.463738	$+3.5 \times 10^{-4}$	9.2×10^{-5}	2.1×10^{-8}	1.8×10^{-2}
0.4	0.4	16.444647	$+6.7 \times 10^{-4}$	1.7×10^{-4}	2.3×10^{-8}	1.4×10^{-2}
0.5	0.5	20.905243	$+9.2 \times 10^{-4}$	2.2×10^{-4}	2.5×10^{-8}	1.0×10^{-2}
0.6	0.6	24.532530	$+1.0 \times 10^{-3}$	2.6×10^{-4}	1.8×10^{-8}	6.1×10^{-3}
0.7	0.7	29.964100	$+9.9 \times 10^{-4}$	2.4×10^{-4}	1.6×10^{-8}	2.7×10^{-3}
0.8	0.8	36.598234	$+7.5 \times 10^{-4}$	1.9×10^{-4}	1.4×10^{-8}	6.0×10^{-3}
0.9	0.9	40.447304	$+3.4 \times 10^{-4}$	8.6×10^{-5}	1.1×10^{-8}	-1.6×10^{-3}

Table 4
Results for μ from Test 2 with $h = 0.05$, $s = 1/4$

		Standard FTCS	9-point	(1,13)	Standard BTCS
t	Exact μ	Error	Error	Error	Error
0.1	1.221403	-9.9×10^{-5}	-2.3×10^{-5}	-8.9×10^{-8}	4.2×10^{-3}
0.2	1.491825	-1.3×10^{-4}	-3.0×10^{-5}	-1.0×10^{-7}	5.1×10^{-3}
0.3	1.822119	-1.5×10^{-4}	-3.8×10^{-5}	-1.1×10^{-7}	6.3×10^{-3}
0.4	2.225541	-1.9×10^{-4}	-4.4×10^{-5}	-0.8×10^{-7}	7.5×10^{-3}
0.5	2.718282	-2.3×10^{-4}	-5.5×10^{-5}	-1.2×10^{-7}	9.2×10^{-3}
0.6	3.320117	-2.8×10^{-4}	-6.7×10^{-5}	-1.7×10^{-7}	1.1×10^{-2}
0.7	4.055200	-3.4×10^{-4}	-8.3×10^{-5}	-2.5×10^{-7}	1.4×10^{-2}
0.8	4.953032	-4.2×10^{-4}	-1.0×10^{-4}	-3.6×10^{-7}	1.7×10^{-2}
0.9	6.049647	-5.1×10^{-4}	-1.3×10^{-4}	-3.1×10^{-7}	2.0×10^{-2}
1.0	7.389056	-6.2×10^{-4}	-1.5×10^{-4}	-2.6×10^{-7}	2.5×10^{-2}

solved with given boundary values everywhere, are shown in Table 3. Again, the (1,13) results are much more accurate than those obtained with the standard FTCS method.

The CPU for the standard FTCS method was 13.1 s and for the 9-point finite-difference formula was 14.5 s and for the (1,13) fully explicit scheme was 18.3 s, while the CPU time for the standard implicit method was 1758 s.

The results obtained for μ with $h = 0.02$, $s = 1/4$, using the standard FTCS method and the 9-point finite-difference scheme and the (1,13) fully explicit formula to solve the example non-classic boundary value problem are shown in Table 4. Again errors obtained using the standard FTCS method are generally more than 1000 times larger than for the (1,13) method.

5. Conclusion

In this paper three fully explicit finite-difference schemes, the standard FTCS method and the 9-point finite-difference scheme and the (1,13) fully explicit technique were applied to the two-dimensional diffusion equation with an integral condition replacing one-boundary condition. The latter worked very well for two-dimensional diffusion with an integral condition, because of its fourth-order accuracy. These methods seems particularly suited for parabolic partial differential equations with continuous boundary conditions. A comparison with the standard implicit scheme of [3] for the model problem clearly demonstrates that the new techniques are computationally superior. The numerical tests obtained by using these methods give acceptable results and suggests convergence to the exact solution when h goes to zero.

References

[1] J.R. Cannon, J. van der Hoek, Diffusion subject to specification of mass, J. Math. Anal. Appl. 115 (1986) 517–529.
[2] J.R. Cannon, Y. Lin, An inverse problem of finding a parameter in a semi-linear heat equation, J. Math. Anal. Appl. 145 (2) (1990) 470–484.
[3] J.R. Cannon, Y. Lin, A.L. Matheson, The solution of the diffusion equation in two-space variables subject to the specification of mass, Appl. Anal. J. 50 (1993) 1–19.
[4] V. Capsso, K. Kunisch, A reaction-diffusion system arising in modeling man-environment diseases, Quart. Appl. Math. 46 (1988) 431–449.
[5] W.A. Day, Existence of a property of solutions of the heat equation subject to linear thermoelasticity and other theories, Quart. Appl. Math. 40 (1982) 319–330.
[6] W.A. Day, A decreasing property of solutions of a parabolic equation with applications to thermoelasticity and other theories, Quart. Appl. Math. XLIV (1983) 468–475.
[7] M. Dehghan, Fully implicit finite difference methods for two-dimensional diffusion with a non-local boundary condition, J. Comput. Appl. Math. 106 (1999) 255–269.
[8] G. Fairweather, R.D. Saylor, The reformulation and numerical solution of certain nonclassical initial-boundary value problems, SIAM J. Sci. Stat. Comput. 12 (1) (1991) 127–144.
[9] A. Friedman, Monotonic decay of solutions of parabolic equation with nonlocal boundary conditions, Quart. Appl. Math. XLIX (1986) 468–475.
[10] C.F. Gerald, P.O. Wheatley, Applied Numerical Analysis, Fifth Edition, Addison-Wesley, California, 1994.
[11] B. Kawohl, Remark on a paper by D.A. Day on a maximum principle under nonlocal boundary conditions, Quart. Appl. Math. XLIV (1987) 751–752.
[12] L. Lapidus, G.F. Pinder, Numerical Solution of Partial Differential Equations in Science and Engineering, Wiley, New York, 1982.
[13] A.R. Mitchell, D.F. Griffiths, The Finite Difference Methods in Partial Differential Equations, Wiley, New York, 1980.
[14] B.J. Noye, Numerical Solution of Partial Differential Equations, Lecture Notes, University of Adelaide, South Australia, 1993.
[15] S. Wang, The numerical method for the conduction subject to moving boundary energy specification, Numer. Heat Transfer 130 (1990) 35–38.
[16] S. Wang, Y. Lin, A finite difference solution to an inverse problem determining a control function in a parabolic partial differential equations, Inverse Problems 5 (1989) 631–640.
[17] S. Wang, Y. Lin, A numerical method for the diffusion equation with nonlocal boundary specifications, Int. J. Engng. Sci. 28 (1990) 543–546.
[18] R.F. Warming, B.J. Hyett, The modified equation approach to the stability and accuracy analysis of finite-difference methods, J. Comput. Phys. 14 (2) (1974) 159–179.