JOURNAL OF COMPUTATIONAL PHYSICS **139,** 92–126 (1998)
ARTICLE NO. CP975865

An Adaptive Spline Wavelet ADI (SW-ADI) Method for Two-Dimensional Reaction–Diffusion Equations

Wei Cai and Wu Zhang

Department of Mathematics, University of North Carolina at Charlotte, Charlotte, North Carolina 28223
E-mail: wcai@uncc.edu, wzhang@uncc.edu

Received April 25, 1997; revised September 18, 1997

We study a spline wavelet alternative direction implicit (SW-ADI) algorithm for solving two-dimensional reaction diffusion equations. This algorithm is based on a collocation method for PDEs with a specially designed spline wavelet for the Sobolev space $H^2(I)$ on a closed interval $\boldsymbol{I}$. By using the tensor product nature of adaptive wavelet meshes, we propose a SW-ADI method for two-dimensional problems. The proposed SW-ADI method is an efficient time-dependent adaptive method with second-order accuracy for solutions with localized phenomena, such as in flame propagations. Issues like new boundary wavelets for more accurate boundary conditions, adaptive strategy for two-dimensional meshes, data structure and storage and implementation details, and numerical results will be discussed. © 1998 Academic Press

1. INTRODUCTION

Wavelets have been applied recently to obtain representations of integral and differential operators in many physical problems. To list a few, the pioneering work by Beylkin *et al.* [1] has started several works in deriving sparse matrix representations of integral operators with Daubechies' wavelet basis [2] or wavelet-like basis [3]. This is made possible because of the vanishing moment properties of the basis functions used in the representations and the low rank property of the integral kernels. In the area of differential operators, most of the attempts of using wavelets is to generate an adaptive meshing structure (select a partial set of full wavelet basis functions) upon which differential operations can be carried out, thus reducing the amount of mesh points (number of wavelet basis functions) needed to resolve detailed structures in the solutions of PDEs [4–8]. This is made possible, of course, by the localization properties of the wavelet basis both in space (like traditional finite element basis functions) and in frequency (like the Fourier basis functions). Meanwhile, Harten has introduced a general multiresolution algorithm as a data compression method for the solutions of conservation laws [15].

In this paper, we will further our studies of using a specially designed spline wavelet basis [4] in $H^2(I)$ on a closed interval $\boldsymbol{I}$ for the resolution of the initial boundary value problems, in this case, two-dimensional reaction diffusion equations. There are two equally important issues in the design of adaptive methods for the efficient solution of time-dependent PDEs. First, the ease of generating locally refined adaptive meshes to resolve the gradients of solutions; second, an efficient solver for the resulting algebraic systems (linear or nonlinear). The second issue demands more attention as it is the bottleneck to achieve efficiency of a time-dependent solver of many CFD problems. In most cases, the diffusive terms (the second derivative terms) in the PDEs are treated implicitly to avoid the prohibitive time step restriction, thus producing an algebraic system to be inverted each time step. Preconditioned iterative methods or multigrid methods are the most used ways for the solution of such a system. The wavelet methods turn out to be a good candidate to address both issues. We are proposing to combine the adaptive spline wavelet methods [4] and the traditional alternative direction implicit method to form an efficient adaptive SW-ADI method for reaction diffusion problems where solutions often demonstrate large local variations, such as the dramatic change of temperature profiles across reaction zones. An important feature of the two-dimensional SW-ADI method is its ability to produce localized adaptive mesh in a tensor product fashion. Because of the compactness of the wavelet basis functions, we will be able to produce local meshes of finite element type, but on a tensor product mesh structure. As a result of this unique property, we will be able to combine the ADI approach with the wavelet methods to obtain an efficient second-order time-marching solver for time-dependent problems as long as a factorization of the differential operator is available.

The paper is organized into the following sections. In Section 2, we introduce the reaction–diffusion equations to be studied in this paper. In Section 3, we will reintroduce the wavelet methods proposed in [4], but with significant modifications on the definitions of boundary wavelets. Additional boundary wavelets are introduced to handle more accurately the boundary conditions of the PDEs without using derivative information of the solutions. Derivative matrices for the differential operators based on the new wavelet functions will be given. Also, a fast *discrete wavelet transform*, originally proposed in [4], between function values and wavelet interpolant expansion coefficients will be modified accordingly. In Section 4, we discuss the SW-ADI method which combines spline wavelet methods and ADI schemes. In Section 5, we will provide the general algorithm of the SW-ADI methods and the data structure used in their implementation. In Section 6, we will test the second-order accuracy of the SW-ADI methods for two types of SW-ADI methods, depending on the order of the spline spaces. Also, we will provide numerical results of flame propagations in a channel. In Section 7, we draw some conclusion of this work.

2. REACTION–DIFFUSION EQUATIONS

To investigate the behavior of the numerical algorithm presented in this paper, we will study two-dimensional reaction–diffusion problems. And for the sake of simplicity, we restrict ourselves to the case of a flame propagation in a gaseous medium in which a single one-step chemical reaction takes place; more complex chemistry and three-dimensional case will be considered in our future studies.

Using Lagrangian space coordinates and normalized variables T and Y for the temperature and the mass fraction of the reaction [9], we have the governing equations of the flame as

$$\begin{aligned}\frac{\partial T}{\partial t} &= \Delta T + S(T, Y)\\ \frac{\partial Y}{\partial t} &= \frac{\Delta Y}{Le} - S(T, Y),\end{aligned} \tag{2.1}$$

where the normalized reaction rate S in given by

$$S = \frac{\beta^2}{2Le} Y \exp\left(-\frac{\beta(1-T)}{1-\alpha(1-T)}\right). \tag{2.2}$$

The positive parameters Le, β, and α denote the Lewis number of the reaction, the reduced activation energy of the reaction, and a nondimensional heat-release parameter, respectively. The solution $T(x, y, t)$ and $Y(x, y, t)$ of (2.1) is sought in a region $\Omega \times [0, T]$ with suitable boundary conditions on $\partial\Omega \times [0, T]$, where Ω is a closed region in R^2, $\partial\Omega$ is the boundary of Ω, and $[0, T]$ is the time interval with $0 \le t \le T$.

Equations (2.1) are associated to initial data

$$T(x, y, 0) = T_0(x, y), \quad Y(x, 0) = Y_0(x, y), \tag{2.3}$$

and boundary conditions along x-direction

$$T(x, y, t) = \left.\begin{cases} 0 & (\textit{fresh mixture})\\ 1 & (\textit{burnt}) \end{cases}\right\} x \in \partial\Omega; \tag{2.4a}$$

$$Y(x, y, t) = \left.\begin{cases} 1 & (\textit{fresh mixture})\\ 0 & (\textit{burnt}) \end{cases}\right\} x \in \partial\Omega. \tag{2.4b}$$

and periodic boundary conditions will be assumed along y-direction.

3. WAVELET APPROXIMATIONS

In this section, we discuss wavelet approximation to one-dimensional problems, which can be extended to two-dimensional or three-dimensional problems in a tensor product fashion. We first mention the main results of wavelet methods for initial-boundary value problems of the PDEs in [4]. Furthermore, a set of new "boundary wavelets" are constructed here in order to approximate general boundary conditions and improve the accuracy of derivative approximation with the "*not-a-knot*" end conditions [10].

3.1. *Wavelet Spaces and Function Approximation*

Let I denote a close interval $[0, L]$ and let $H^2(I)$ and $H_0^2(I)$ denote the following two Sobolev spaces with finite L^2-norm up to the second derivatives, i.e.,

$$H^2(I) = \left\{ f(x), x \in I \middle\| f^{(i)} \right\|_2 < +\infty, i = 0, 1, 2 \right\}, \tag{3.1}$$

$$H_0^2(I) = \{ f(x) \in H^2(I) \mid f(0) = f(L) = f'(0) = f'(L) = 0 \}. \tag{3.2}$$

$H_0^2(I)$ is a Hilbert space [11] equipped with the following inner product

$$\langle f, g\rangle = \int_I f''(x)g''(x)\,dx; \tag{3.3}$$

thus,

$$|||f||| = \sqrt{\langle f, f\rangle} \tag{3.4}$$

provides a norm for $H_0^2(I)$.

In [4] we have introduced a basis function set of subspace V_J for a given integer $J \geq 0$ and a fixed interval $I = [0, L]$ with, for example, integer $L > 4$ as

$$V_J = V_0 \oplus W_0 \oplus W_1 \oplus W_2 \oplus \cdots \oplus W_J, \tag{3.5}$$

where

$$V_0 = \operatorname{span}\{\phi_{0,k}(x), 0 \leq k \leq L-3\}, \tag{3.6}$$

$$W_j = \operatorname{span}\{\psi_{j,k}(x), -1 \leq k \leq n_j - 2,\ n_j = 2^j L\}, \quad 0 \leq j \leq J, \tag{3.7}$$

where all the basis functions are defined in [4].

It is shown in [4] that any function $f(x) \in H_0^2(I)$ can be approximated as closely as possible by a function $f_J(x) \in V_J$ for a sufficiently large J, and $f_J(x)$ has a unique orthogonal decomposition under inner product (3.3)

$$f_J(x) = f_0 + g_0 + g_1 + \cdots + g_{J-2} + g_J, \tag{3.8}$$

where $f_0 \in V_0, g_j \in W_j, 0 \leq j \leq J$.

In this paper, we will present two new boundary wavelet functions in order to implement the *not-a-knot* boundary condition. They are defined as

$$\psi_{b0}(x) = -\frac{56}{99}(\psi_{0,-1}(x) + 14\psi_{0,-2}(x)), \tag{3.9}$$

$$\psi_{b1}(x) = \frac{182}{181}\left(\psi(x) + \frac{1}{13}(\psi_{0,-1}(x) + \psi_{0,-2}(x))\right), \tag{3.10}$$

where

$$\psi(x) = -\frac{3}{7}\varphi(2x) + \frac{12}{7}\varphi(2x-1) - \frac{3}{7}\varphi(2x-2), \tag{3.11}$$

and $\psi_{0,-1}(x) = \psi(x+1)$, $\psi_{0,-2}(x) = \psi(x+2)$ and we also consider different collocation points from those proposed in [4] in order to reflect the extra boundary wavelet functions.

First, we redefine the scaling space V_0 and the wavelet spaces W_j. For any $j, k \in Z$, we have

$$\varphi_{0,k}(x) = \varphi(x-k), \qquad 0 \leq k \leq L-4, \tag{3.12}$$

$$\varphi_{0,-3}(x) = \eta_1(x), \qquad \varphi_{0,-2}(x) = \eta_2(x), \tag{3.13}$$

$$\varphi_{0,-1}(x) = \varphi_b(x), \qquad \varphi_{0,L-3}(x) = \varphi_b(L-x), \tag{3.14}$$

$$\varphi_{0,L-2}(x) = \eta_2(L-x), \quad \varphi_{0,L-1}(x) = \eta_1(L-x), \tag{3.15}$$

where

$$\begin{aligned} \eta_1(x) &= (1-x)_+^3, \\ \eta_2(x) &= 2x_+ - 3x_+^2 + \frac{7}{6}x_+^3 - \frac{4}{3}(x-1)_+^3 + \frac{1}{6}(x-2)_+^3, \end{aligned} \tag{3.16}$$

$$\begin{aligned} \varphi(x) &= \frac{1}{6}\sum_{j=0}^{4}\binom{4}{j}(-1)^j(x-j)_+^3, \\ \varphi_b(x) &= \frac{3}{2}x_+^2 - \frac{11}{12}x_+^3 + \frac{3}{2}(x-1)_+^3 - \frac{3}{4}(x-2)_+^3; \end{aligned} \tag{3.17}$$

and

$$\psi_{j,k}(x) = \psi(2^j x - k), \qquad j \ge 0, k = 1, 2, \ldots, n_j - 4, \tag{3.18}$$

$$\psi_{j,-1}(x) = \psi_{b0}(2^j x), \qquad \psi_{j,0}(x) = \psi_{b1}(2^j x), \tag{3.19}$$

$$\psi_{j,n_j-3}(x) = \psi_{b1}(2^j(L-x)), \quad \psi_{j,n_j-2}(x) = \psi_{b0}(2^j(L-x)), \tag{3.20}$$

where $n_j = 2^j L$. The new spaces V_0 and W_j are

$$V_0 = \text{span}\{\phi_{0,k}, -3 \le k \le L-1\}, \tag{3.21}$$

$$W_j = \text{span}\{\psi_{j,k}(x), -1 \le k \le n_j - 2, n_j = 2^j L\}, \quad 0 \le j \le J. \tag{3.22}$$

It can be checked that $\dim V_0 = L+3$, $\dim W_j = n_j$. The collocation points

$$X^{(-1)} = \left\{0, \frac{1}{2}, 1, 2, \ldots, L-1, L-\frac{1}{2}, L\right\} = \left\{x_k^{(-1)}\right\}_{k=1}^{L+3} \quad \text{for } V_0, \tag{3.23}$$

are chosen and

$$\begin{aligned} X^{(j)} &= \left\{\frac{1}{2^{j+2}}, \frac{3}{2^{j+1}}, \frac{5}{2^{j+1}}, \ldots, \frac{2k+3}{2^{j+1}}, \ldots, L - \frac{5}{2^{j+1}}, L - \frac{3}{2^{j+1}}, L - \frac{1}{2^{j+2}}\right\} \\ &= \left\{x_k^{(j)}\right\}_{k=-1}^{n_j-2} \end{aligned} \tag{3.24}$$

for W_j, $j \ge 0$, and the number of collocation points in V_0 is $L+3$ and that in W_j is n_j, which both match the dimensions of spaces V_0 and W_j, $j \ge 0$. The collocation points for each level, $X^{(-1)}$, $X^{(0)}$, $X^{(1)}$, $X^{(2)}$, $X^{(3)}$, are shown in Fig. 1.

The modification on the boundary scaling function and wavelet functions will partially destroy the orthogonality of boundary scaling functions (3.13) to (3.15) with respect to wavelet spaces. This is because we have included the nonhomogeneity in those boundary functions. This approach is equivalent to applying an orthogonal wavelet decomposition to a function minus the boundary nonhomogeneity by subtracting a local boundary term. Such an approach has been discussed in [4]. Moreover, it can still be proven that the *point vanishing property* in [4] holds, i.e., for $j > i$

$$\psi_{j,k}\left(x_l^{(i)}\right) = \delta_{kl}, \tag{3.25}$$

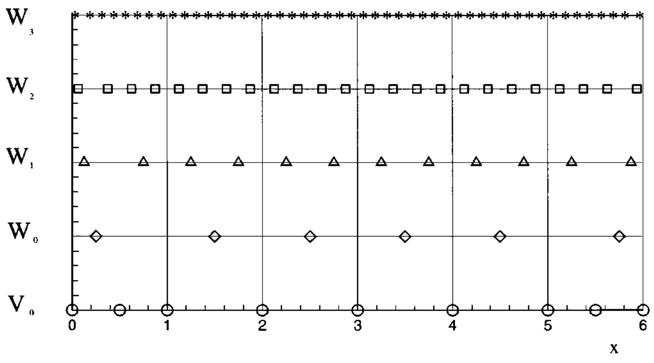

FIG. 1. Collocation point distributions for different space levels.

where $j \geq 0, -1 \leq k \leq n_j - 2$, and $1 \leq l \leq L-1$ if $i = -1; -1 \leq l \leq n_i - 2$ if $i \geq 0$. The *point vanishing property* of (3.25) assures that $\{\psi_{j,k}(x)\}$ forms a hierarchical basis function over the collocation points in (3.23)–(3.24).

Finally, for any function $f(x) \in H^2(I)$, we have an approximate function $f_J(x)$ in the form of

$$f_J(x) = f_0 + g_0 + g_1 + \cdots + g_{J-1} + g_J, \tag{3.26}$$

where $f_0 \in V_0$, $g_j \in W_j, 0 \leq j \leq J$, V_0 and W_j are the redefined scaling and wavelet spaces in (3.21) and (3.22).

3.2. *Function Expansions and Approximation of Derivatives*

Assuming that a function $f(x) \in H^2(I)$, we write its wavelet interpolant $f_J(x) = P_J f(x)$ and approximate derivatives $f_J^{(k)}(x), k = 1, 2$ as

$$f_J(x) =: P_J f(x) = f_{-1}(x) + \sum_{j=0}^{J} f_j(x) \tag{3.27}$$

and

$$f_J^{(k)}(x) = f_{-1}^{(k)}(x) + \sum_{j=0}^{J} f_j^{(k)}(x), \tag{3.28}$$

where

$$f_{-1}(x) = \sum_{k=-3}^{L-1} \hat{f}_{-1,k}\varphi_{0,k}(x) \in V_0 \tag{3.29}$$

and

$$f_j(x) = \sum_{k=-1}^{n_j-2} \hat{f}_{j,k}\psi_{j,k}(x) \in W_j, \quad 0 \leq j \leq J. \tag{3.30}$$

Let us denote the expansion coefficients of (3.27) or (3.29) and (3.30) in a vector form

$$\hat{f}_J = \left(\hat{f}^{(-1)}, \hat{f}^{(0)}, \hat{f}^{(1)}, \ldots, \hat{f}^{(J)}\right)^T \tag{3.31}$$

where

$$\hat{f}^{(-1)} = (\hat{f}_{-1,-3}, \hat{f}_{-1,-2}, \hat{f}_{-1,-1}, \hat{f}_{-1,0}, \ldots, \hat{f}_{-1,L-3}, \hat{f}_{-1,L-2}, \hat{f}_{-1,L-1},)^T, \tag{3.32}$$

$$\hat{f}^{(j)} = (\hat{f}_{j,-1}, \hat{f}_{j,0}, \ldots, \hat{f}_{j,n_j-2})^T, \quad 0 \le j \le J. \tag{3.33}$$

The coefficients $\hat{f}_J$ will be determined by satisfying interpolation conditions at the collocation points defined in (3.23)–(3.24), i.e.,

$$P_J f\left(x_k^{(-1)}\right) = f\left(x_k^{(-1)}\right) =: f_k^{(-1)}, \quad -3 \le k \le L-1, \tag{3.34}$$

$$P_J f\left(x_k^{(j)}\right) = f\left(x_k^{(j)}\right) =: f_k^{(j)}, \quad -1 \le k \le n_j - 2. \tag{3.35}$$

Denoting the values of $f(x)$ at collocation points by

$$f_J = \left(f^{(-1)}, f^{(0)}, f^{(1)}, \ldots, f^{(J)}\right)^T, \tag{3.36}$$

where

$$f^{(-1)} = \left(f_{-3}^{(-1)}, f_{-2}^{(-1)}, f_{-1}^{(-1)}, \ldots, f_{L-3}^{(-1)}, f_{L-2}^{(-1)}, f_{L-1}^{(-1)}\right)^T, \tag{3.37}$$

$$f^{(j)} = \left(f_{-1}^{(j)}, f_0^{(j)}, \ldots, f_{n_j-2}^{(j)}\right)^T, \quad 0 \le j \le J, \tag{3.38}$$

and using the point-vanishing property of (3.25), we obtain

$$\hat{f}^{(-1)} = B^{-1} f^{(-1)}, \tag{3.39}$$

$$\hat{f}^{(j)} = M_j^{-1}\left[f^{(j)} - (P_{j-1}f)^{(j)}\right], \tag{3.40}$$

where $(P_{j-1}f)^{(j)} = \{P_{j-1}f(x_k^{(j)})\}_{k=-1}^{n_j-2}$, B and M_j are the $(L+3)\times(L+3)$, and $n_j \times n_j$ transform matrices, respectively:

$$B = \begin{bmatrix}
1 & 0 & 0 & & & & & & & & & \\
\frac{1}{8} & \frac{19}{48} & \frac{25}{96} & \frac{1}{48} & & & & & & 0 & & \\
 & \frac{1}{6} & \frac{7}{12} & \frac{1}{6} & & & & & & & & \\
 & & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & & & & & & & \\
 & & & \bullet & \bullet & \bullet & & & & & & \\
 & & & & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & & & & & \\
 & & & & & \bullet & \bullet & \bullet & & & & \\
 & & & & & & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & & & \\
 & & & & & & & \frac{1}{6} & \frac{7}{12} & \frac{1}{6} & & \\
 & 0 & & & & & & \frac{1}{48} & \frac{25}{96} & \frac{19}{48} & \frac{1}{8} & \\
 & & & & & & & & 0 & 0 & 1 &
\end{bmatrix} \tag{3.41}$$

$$M_j = \begin{bmatrix} 1 & \frac{9}{362} & & & & & & & & \\ \frac{4}{99} & 1 & -\frac{1}{14} & & & & & 0 & & \\ & -\frac{13}{181} & 1 & -\frac{1}{14} & & & & & & \\ & & -\frac{1}{14} & 1 & -\frac{1}{14} & & & & & \\ & & & \bullet & \bullet & \bullet & & & & \\ & & & & -\frac{1}{14} & 1 & -\frac{1}{14} & & & \\ & & & & & -\frac{1}{14} & 1 & -\frac{13}{181} & & \\ & 0 & & & & & -\frac{1}{14} & 1 & \frac{4}{99} \\ & & & & & & & \frac{9}{362} & 1 \end{bmatrix}. \tag{3.42}$$

Then, the *discrete wavelet transform* (DWT) which maps from f_J to $\hat{f}_J$ will have operation count $O(N \log N)$ as in [4], where $N = 2^{J+1}L + 3$ is the total number of collocation points.

The projection matrix into W_j is written, assuming that the whole collocation point set $x^{(j)}$ for W_j is being used. A different M_j will be defined in Subsection 3.4 when only part of $x^{(j)}$ is used in the approximation as a result of adaptive meshing.

3.3. *Derivative Matrices*

The differential operation of wavelet approximation to a function, given by its wavelet expansion of (3.28), can also be represented by a derivative matrix D. Because of the multiresolution structure of spaces V_j, namely $V_j \subset V_{j+1}$, and $V_J = V_0 \oplus W_0 \oplus \cdots \oplus W_J$, we can assume that $J = 0$ and that the wavelet interpolation $f_0(x) = P_0 f(x)$ for the function $f(x) \in H^2(I)$ can be written as the linear combination of only scaling function in V_0:

$$f_0(x) = \sum_{k=-3}^{L-3} \hat{f}_k \varphi_k(x). \tag{3.43}$$

Let $\{x_0, x_1, \ldots, x_N\}$ be the total collocation point which is the union of partial sets of collocation points $x^{(-1)}$ and $x^{(j)}$, $0 \le j \le J$, in an adaptive situation, and $h_i = x_i - x_{i-1}$, $1 \le i \le N$; then the derivative matrices $D_3^{(2)}$ and $D_3^{(1)}$ for second and first derivatives, respectively, are given as

$$\begin{bmatrix} u''(1) \\ \cdot \\ \cdot \\ u''(N-1) \end{bmatrix} = T_1^{-1} \left\{ T_2 \begin{bmatrix} u(1) \\ \cdot \\ \cdot \\ u(N-1) \end{bmatrix} + u(0) \begin{bmatrix} \frac{2h_2}{h_1+h_2} \\ 0 \\ \cdot \\ 0 \end{bmatrix} + u(N) \begin{bmatrix} 0 \\ \cdot \\ 0 \\ \frac{2h_{N-1}}{h_{N-1}+h_N} \end{bmatrix} \right\}, \tag{3.44a}$$

where

$$T_1 = \begin{bmatrix} \frac{h_1(h_1+2h_2)}{3} & \frac{h_1(h_2-h_1)}{3} & & & & & & \\ \frac{h_2}{6} & \frac{h_2+h_3}{3} & \frac{h_3}{6} & & & 0 & & \\ & & \bullet & \bullet & \bullet & & & \\ & & & \frac{h_i}{6} & \frac{h_i+h_{i+1}}{3} & \frac{h_{i+1}}{6} & & \\ & & & & \bullet & \bullet & \bullet & \\ & 0 & & & & \frac{h_{N-2}}{6} & \frac{h_{N-2}+h_{N-1}}{3} & \frac{h_{N-1}}{6} \\ & & & & & & \frac{h_N(h_{N-1}-h_N)}{3} & \frac{h_N(h_N+2h_{N-1})}{3} \end{bmatrix}_{(N-1)\times(N-1)} \tag{3.44b}$$

$$T_2 = \begin{bmatrix} -2 & \frac{2h_1}{h_1+h_2} & & & \\ \frac{1}{h_2} & -\left(\frac{1}{h_2}+\frac{1}{h_3}\right) & \frac{1}{h_3} & & \\ \bullet & \bullet & \bullet & & \\ & \frac{1}{h_{N-2}} & -\left(\frac{1}{h_{N-2}}+\frac{1}{h_{N-1}}\right) & \frac{1}{h_{N-1}} \\ & & \frac{2h_N}{h_{N-1}+h_N} & -2 \end{bmatrix}_{(N-1)\times(N-1)} \tag{3.44c}$$

$$\begin{aligned} u''(0) &= \frac{h_1+h_2}{h_2}\left[u''(1) - \frac{h_1}{h_1+h_2}u''(2)\right], \\ u''(N) &= \frac{h_{N-1}+h_N}{h_{N-1}}\left[u''(N-1) - \frac{h_N}{h_{N-1}+h_N}u''(N-2)\right]. \end{aligned} \tag{3.44d}$$

$$D_3^{(2)} = T_1^{-1}T_2, \tag{3.44e}$$

if homogeneous boundary conditions, i.e., $u(0) = u(N) = 0$ are used, and

$$\begin{bmatrix} u'(0) \\ \cdot \\ \cdot \\ \cdot \\ u'(N) \end{bmatrix} = H_1^{-1}H_2 \begin{bmatrix} u(0) \\ \cdot \\ \cdot \\ \cdot \\ u(N) \end{bmatrix} = D_3^{(1)} \begin{bmatrix} u(0) \\ \cdot \\ \cdot \\ \cdot \\ u(N) \end{bmatrix}, \tag{3.45a}$$

where

$$H_1 = \begin{bmatrix} \lambda_1 & 1 & & & & & \\ \lambda_1 & 2 & \mu_1 & & & 0 & \\ & \lambda_2 & 2 & \mu_2 & & & \\ & & \bullet & \bullet & \bullet & & \\ & & & \lambda_i & 2 & \mu_i & \\ & & & & \bullet & \bullet & \bullet \\ & 0 & & & \lambda_{N-1} & 2 & \mu_{N-1} \\ & & & & & 1 & \mu_{N-1} \end{bmatrix}_{(N+1)\times(N+1)} \tag{3.45b}$$

$$H_2 = \begin{bmatrix} a_1 & a_2 & a_3 & & & & \\ & c_1 & d_1 & e_1 & & 0 & \\ & & \bullet & \bullet & \bullet & & \\ & & & c_i & d_i & e_i & \\ & & & & \bullet & \bullet & \bullet \\ & 0 & & & c_{N-1} & d_{N-1} & e_{N-1} \\ & & & & b_3 & b_2 & b_1 \end{bmatrix}_{(N+1)\times(N+1)}, \tag{3.45c}$$

where

$$\lambda_i = \frac{h_{i+1}}{h_i + h_{i+1}}, \quad \mu_i = 1 - \lambda_i, \tag{3.46a}$$

$$a_1 = -\lambda_1 \frac{2+\mu_1}{h_1}, \quad a_2 = \lambda_1 \frac{2+\mu_1}{h_1} - \frac{(\mu_1)^2}{h_2}, \quad a_3 = \frac{(\mu_1)^2}{h_2}, \tag{3.46b}$$

$$b_1 = \mu_{N-1} \frac{2+\lambda_{N-1}}{h_N}, \quad b_2 = \frac{(\lambda_{N-1})^2}{h_{N-1}} - \mu_{N-1} \frac{2+\lambda_{N-1}}{h_N}, \quad b_3 = -\frac{(\lambda_{N-1})^2}{h_{N-1}}, \tag{3.46c}$$

$$c_i = -\frac{3\lambda_i}{h_i}, \quad e_i = \frac{3\mu_i}{h_{i+1}}, \quad d_i = -c_i - e_i, \quad 1 \le i \le N-1. \tag{3.46d}$$

As pointed out above, the *not-a-knot* conditions are used in order to obtain the best interpolating results near the boundaries. Similar derivative matrices for spline-based wavelets have been considered in [6], however, with different treatments for the boundaries. To obtain higher accuracy for derivatives, we can also replace $D_3^{(1)}$ and $D_3^{(2)}$ by the derivative matrices $D_5^{(1)}$ and $D_5^{(2)}$, produced by a quintic spline interpolation with the not-a-knot boundary treatment. We leave the derivation of the derivative matrices $D_5^{(1)}$ and $D_5^{(2)}$ to the Appendix.

3.4. *One-Dimensional Data Structure and Adaptive Procedure*

In order to implement an adaptive procedure, it is very important to design an efficient data structure. We consider the data structure for the function values f_J in (3.36) at collocation points. The same data structure will be used for the wavelet coefficients $\hat{f}_J$ of (3.31). Denote

$f_J = \{f^{(-1)}, f^{(0)}, \ldots, f^{(j)}, \ldots, f^{(J)}\}$,

$f^{(j)} = \{f_k^{(j)}\}, \quad 1 \le k \le n_j$, the values of the unknown function on level j,

f_J will be stored as a 1D array, together with the following auxiliary vectors,

pointer (j)—pointer of first element of $f^{(j)}$,

npts(j)—number of collocation points used in W_j, *npts*$(j) = n_j$,

index$(1 : n_j)$—collocation point location indices.

In expression (3.27), $f_J(x)$ is computed using the whole set of collocation points $x^{(j)} = \{x_k^{(j)}\}, 0 \le j \le J$. As we know, most of wavelet expansion coefficients $\hat{f}_{j,k}$ for large j can be ignored within a given tolerance ε, so we can dynamically adjust the number and location of the collocation points used in the wavelet expansion and reduce the cost of the scheme while providing enough resolution in the region where the solution varies significantly.

We now describe the procedure of deleting collocation points. Let $\varepsilon > 0$ be a prescribed tolerance and let $j \ge 0$,

$$l = l(\varepsilon) = \min\left(\frac{n_j}{2}, -\frac{\log \varepsilon}{\log \alpha}\right), \quad \alpha = 7 + \sqrt{192} \cong 13.928 \text{ (see [4])}.$$

Step 1. First we locate the range of the index k,

$$(k_1', l_1'), \ldots, (k_m', l_m'), \quad m = m(j, \varepsilon), \tag{3.47}$$

such that

$$|\hat{f}_{j,k}| \ge \varepsilon, \quad k_i' \le k \le l_i';\ i = 1, \ldots, m. \tag{3.48}$$

Step 2. We keep $\hat{f}_{j,k}$ in (3.27) for $k_i \le k \le l_i, k_i = k_i' - l - 3, l_i = l_i' + l + 3, i = 1, \ldots, m$; namely, we reduce $f_J(x)$ as

$$f_J(x) \approx \sum_{k=-3}^{L-1} \hat{f}_{-1,k}\phi_k(x) + \sum_{k \in K_J} \hat{f}_{j,k}\psi_{j,k}(x), \tag{3.49}$$

where $K_J = \bigcup_{1 \le i \le m}[k_i, l_i]$.

Step 3. The new collocation points and wavelet coefficients will be

$$\{x_k^{(j)}\},\ \hat{f}_{j,k}, \quad -3 \le k \le L-1, \quad \text{if } j = -1;\ k \in \mathrm{K}_j, \quad \text{if } j \ge 0. \tag{3.50}$$

The procedure of adding collocation points for higher wavelet spaces is discussed with the two-dimensional case in Section 5.

Now we discuss the projection matrix M_j of (3.42) for adaptive meshes. The project matrix M_j for W_j will consist of blocks which corresponds to the grouping of index k in (3.47):

$$M_j = \begin{bmatrix} M_1 & & & & 0 & \\ & M_2 & & & & \\ & & \bullet & & & \\ & & & \bullet & & \\ & & & & \bullet & \\ & 0 & & & M_{m-1} & \\ & & & & & M_m \end{bmatrix}. \tag{3.51}$$

To ensure the block structure of M_j, we assume that the spacing between the index groups $[k_i,\ l_i]$, $1 \le i \le m$, is bigger than 3, so that wavelet basis functions from two different index groups will not overlap. For $2 \le i \le m-1$,

$$M_i = \begin{bmatrix} 1 & -\frac{1}{14} & & & & & & \\ -\frac{1}{14} & 1 & -\frac{1}{14} & & & & 0 & \\ & -\frac{1}{14} & 1 & -\frac{1}{14} & & & & \\ & & & \bullet & \bullet & \bullet & & \\ & & & & -\frac{1}{14} & 1 & -\frac{1}{14} & \\ & 0 & & & & -\frac{1}{14} & 1 & -\frac{1}{14} \\ & & & & & & -\frac{1}{14} & 1 \end{bmatrix}_{(l_i-k_i+1)\times(l_i-k_i+1)}. \tag{3.52}$$

For $i = 1$, the exact form of M_1 will depend on which boundary functions $\psi_{j,-1}(x)$ and

$\psi_{j,0}(x)$ are included in the index (k_1, l_1); i.e.,

$$M_1 = \begin{bmatrix} 1 & \frac{9}{362} & & & & & & \\ \frac{4}{99} & 1 & -\frac{1}{14} & & & & 0 & \\ & -\frac{13}{181} & 1 & -\frac{1}{14} & & & & \\ & & -\frac{1}{14} & 1 & -\frac{1}{14} & & & \\ & & & \bullet & \bullet & \bullet & & \\ & 0 & & & & -\frac{1}{14} & 1 & -\frac{1}{14} \\ & & & & & & -\frac{1}{14} & 1 \end{bmatrix}_{(l_1-k_1+1)\times(l_1-k_1+1)} \tag{3.53a}$$

if $\psi_{j,-1}(x)$ and $\psi_{j,0}(x)$ are included in the index (k_1, l_1);

$$M_1 = \begin{bmatrix} 1 & -\frac{1}{14} & & & & & & \\ -\frac{13}{181} & 1 & -\frac{1}{14} & & & & 0 & \\ & -\frac{1}{14} & 1 & -\frac{1}{14} & & & & \\ & & -\frac{1}{14} & 1 & -\frac{1}{14} & & & \\ & & & \bullet & \bullet & \bullet & & \\ & 0 & & & & -\frac{1}{14} & 1 & -\frac{1}{14} \\ & & & & & & -\frac{1}{14} & 1 \end{bmatrix}_{(l_1-k_1+1)\times(l_1-k_1+1)} \tag{3.53b}$$

if only $\psi_{j,0}(x)$ is included;

M_1 is the same as (3.52) if no boundary wavelet is included, (3.53c)

and similarly for $i = m$, considering the index (k_m, l_m),

$$M_m = \begin{bmatrix} 1 & -\frac{1}{14} & & & & & & \\ -\frac{1}{14} & 1 & -\frac{1}{14} & & & & 0 & \\ & -\frac{1}{14} & 1 & -\frac{1}{14} & & & & \\ & & \bullet & \bullet & \bullet & & & \\ & & & -\frac{1}{14} & 1 & -\frac{1}{14} & & \\ & & & & -\frac{1}{14} & 1 & -\frac{13}{181} & \\ & 0 & & & & -\frac{1}{14} & 1 & \frac{4}{99} \\ & & & & & & \frac{9}{362} & 1 \end{bmatrix}_{(l_m-k_m+1)\times(l_m-k_m+1)} \tag{3.54a}$$

if $\psi_{j,n_j-3}(x)$ and $\psi_{j,n_j-2}(x)$ are included in the index (k_m, l_m);

$$M_m = \begin{bmatrix} 1 & -\frac{1}{14} & & & & & & \\ -\frac{1}{14} & 1 & -\frac{1}{14} & & & & 0 & \\ & -\frac{1}{14} & 1 & -\frac{1}{14} & & & & \\ & & \bullet & \bullet & \bullet & & & \\ & & & -\frac{1}{14} & 1 & -\frac{1}{14} & & \\ & & & & -\frac{1}{14} & 1 & -\frac{1}{14} & \\ & 0 & & & & -\frac{1}{14} & 1 & -\frac{13}{181} \\ & & & & & & -\frac{1}{14} & 1 \end{bmatrix}_{(l_m-k_m+1)\times(l_m-k_m+1)} \tag{3.54b}$$

if only $\psi_{j,n_j-3}(x)$ is included;

$$M_m \text{ is the same as (3.52) if no boundary wavelet is included.} \tag{3.54c}$$

4. SW-ADI METHODS

To achieve unconditional stability, one may resort to a fully implicit method for time discretization of (2.1). Unfortunately, this will result in a system of algebraic equations that is sparse, but which may require a large amount of computational effort. One remedy is to use ADI methods, which only require solving one-dimensional implicit problems for each time step.

In this section, we discuss the SW-ADI schemes and the corresponding boundary conditions which will be used in our numerical simulations for two-dimensional reaction–diffusion problems. Based on the ADI formulae, SW-ADI schemes with nonuniform meshes are developed, which is of importance in implementing wavelet approximation and the adaptive procedure.

Consider the heat conduction equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \tag{4.1}$$

in the region $\tilde{\Omega} = \Omega \times [0, T]$, where $\Omega = \{(x, y;\ 0 \le x, y \le 1\}$ with the initial condition

$$u(x, y, 0) = f(x, y), \quad (x, y) \in \Omega, \tag{4.2}$$

and boundary condition

$$u(x, y, t) = g(x, y, t), \quad (x, y, t) \in \partial\Omega \times [0, T], \tag{4.3}$$

where $\partial\Omega$ is the boundary of region Ω.

SW-ADI methods are now introduced, first, with respect to Eq. (4.1) for the heat conduction operator $D = D_x^2 + D_y^2$. The exact difference approximation of Eq. (4.1), provided that D is independent of time t, is given by

$$u_{l,m}^{n+1} = \exp(\Delta t D) u_{l,m}^{n}, \tag{4.4}$$

$$\exp\left(-\frac{\Delta t}{2}\left(D_x^2 + D_y^2\right)\right) u_{l,m}^{n+1} = \exp\left(\frac{\Delta t}{2}\left(D_x^2 + D_y^2\right)\right) u_{l,m}^{n}, \tag{4.5}$$

where (l, m) stands for the spatial position of (x, y), n represents the temporal level, and

$$D_x^2 = \frac{1}{h^2}\left(\delta_x^2 - \frac{1}{12}\delta_x^4 + \frac{1}{90}\delta_x^6 \ldots\right), \quad D_y^2 = \frac{1}{h^2}\left(\delta_y^2 - \frac{1}{12}\delta_y^4 + \frac{1}{90}\delta_y^6 \ldots\right), \tag{4.6}$$

where δ_x^2 and δ_y^2 are approximate difference operators with mesh size h to second derivatives D_x^2 and D_y^2 respectively. In the following u^n will be written in short for $u_{l,m}^n$ if there is no confusion about the notations.

Substituting (4.6) into (4.5), we have up to second-order difference,

$$\exp\left(-\frac{r}{2}\delta_x^2\right)\exp\left(-\frac{r}{2}\delta_y^2\right)u^{n+1} = \exp\left(\frac{r}{2}\delta_x^2\right)\exp\left(\frac{r}{2}\delta_y^2\right)u^n, \tag{4.7}$$

where $r = \Delta t/h^2$ and, using the Taylor expansion, it becomes

$$\left(1 - \frac{r}{2}\delta_x^2\right)\left(1 - \frac{r}{2}\delta_y^2\right)u^{n+1} = \left(1 + \frac{r}{2}\delta_x^2\right)\left(1 + \frac{r}{2}\delta_y^2\right)u^n + O(\Delta t^3 + \Delta t h^2), \tag{4.8}$$

which is the *Crank–Nicolson* (C-N) formula if $O(\Delta t^3 + \Delta t h^2)$ is dropped.

4.1. *SW-ADI Schemes*

In this section, we will consider several well-known ADI schemes to be used with our spline wavelet method to form the SW-ADI schemes. From formula (4.8), we can obtain the following ADI schemes and the details of the ADI schemes used here can be found in [12].

Peaceman-Randford (P.R) scheme. In the first step advancing from t_n to $t_n + \Delta t/2$, an implicit difference is used for $\partial^2 u/\partial x^2$ and an explicit difference is used for $\partial^2 u/\partial y^2$. In the second step advancing from $t_n + \Delta t/2$ to t_{n+1}, a reversed procedure is used. The difference approximation to Eq. (4.1) can be expressed in the form of derivative matrices of Subsection 3.3 and the Appendix,

$$\left[1 - \frac{\Delta t}{2}D_x^2\right]u^{n+1^*} = \left[1 + \frac{\Delta t}{2}D_y^2\right]u^n, \tag{4.9a}$$

$$\left[1 - \frac{\Delta t}{2}D_y^2\right]u^{n+1} = \left[1 + \frac{\Delta t}{2}D_x^2\right]u^{n+1^*}, \tag{4.9b}$$

where u^{n+1^*} is an intermediate value. It is clear that (4.9) is the same as formula (4.8) within the accuracy $O(\Delta t^3 + \Delta t h^2)$.

D'Yaknov (D.) scheme.

$$\left[1 - \frac{\Delta t}{2}D_x^2\right]u^{n+1^*} = \left[1 + \frac{\Delta t}{2}D_x^2\right]\left[1 + \frac{\Delta t}{2}D_y^2\right]u^n, \tag{4.10a}$$

$$\left[1 - \frac{\Delta t}{2}D_y^2\right]u^{n+1} = u^{n+1^*}, \tag{4.10b}$$

where u^{n+1^*} is an intermediate value.

In (4.9)–(4.10), the derivative matrices D_x^2, D_y^2 are computed by the methods given in Section 3 or the Appendix. It is known from the generation of the derivative matrices that nonuniform wavelet meshes can be used.

4.2. *Initial and Boundary Conditions*

The approximate solution u^n must satisfy the initial and boundary conditions (4.2) and (4.3), i.e.,

(i) $u^0 = f$, at all mesh points,
(ii) $u^n = g^n$, $n = 0, 1, \ldots, N$, on the boundary $\partial\Omega$.

The intermediate value u^{n+1^*} introduced in each ADI scheme above is not necessarily an approximation to the solution at any time levels. As a result, particularly with the higher order methods, the boundary values at the intermediate level must be obtained, if possible, in terms of the boundary values at t_n and t_{n+1}. The following formulae [12] give u^{n+1^*} explicitly in terms of the central difference of g^{n+1} and g^n with respect to y from (4.3):

$$u^{n+1^*} = \frac{1}{2}\left[1 - \frac{\Delta t}{2} D_y^2\right] g^{n+1} + \frac{1}{2}\left[1 + \frac{\Delta t}{2} D_y^2\right] g^n \quad (P.R.) \tag{4.11}$$

$$u^{n+1^*} = \left[1 - \frac{\Delta t}{2} D_y^2\right] g^{n+1} \quad (D.). \tag{4.12}$$

If the boundary conditions are independent of the time, the formulae giving u^{n+1^*} on the boundary $\partial\Omega$ reduce to

$$u^{n+1^*} = g \quad (P.R.) \tag{4.13}$$

$$u^{n+1^*} = \left[1 - \frac{\Delta t}{2} D_y^2\right] g \quad (D). \tag{4.14}$$

4.3. *ADI Scheme with Source Term and/or Mixed Space Derivative*

Considering the equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + S(x, t), \tag{4.15}$$

we have the second-order ADI scheme [12]

$$\left[1 - \frac{\Delta t}{2} D_x^2\right] u^{n+1^*} = \left[1 + \frac{\Delta t}{2} D_y^2\right] u^n + \frac{\Delta t}{2} S^{n+1/2}, \tag{4.16a}$$

$$\left[1 - \frac{\Delta t}{2} D_y^2\right] u^{n+1} = \left[1 + \frac{\Delta t}{2} D_x^2\right] u^{n+1^*} + \frac{\Delta t}{2} S^{n+1/2}, \tag{4.16b}$$

or

$$\left[1 - \frac{\Delta t}{2} D_x^2\right] u^{n+1^*} = \left[1 + \frac{\Delta t}{2} D_y^2\right] u^n + \frac{\Delta t}{2} S^n, \tag{4.17a}$$

$$\left[1 - \frac{\Delta t}{2} D_y^2\right] u^{n+1} = \left[1 + \frac{\Delta t}{2} D_x^2\right] u^{n+1^*} + \frac{\Delta t}{2} S^{n+1}, \tag{4.17b}$$

where $S^n = S(x, t_n)$, $S^{n+1/2} = S(x, t_n + \Delta t/2)$, and $S^{n+1} = S(x, t_{n+1})$. It is worthwhile pointing out that if the source term is dependent on the unknown function u, we need to use a Strang-type splitting scheme [13] with one-step time integrating schemes, for example, a fourth-order Runge–Kutta scheme for the source term.

For an equation with mixed space derivative, i.e.,

$$\frac{\partial u}{\partial t} = A\frac{\partial^2 u}{\partial x^2} + 2B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} \quad \text{with } A > 0, C > 0, B^2 < AC, \tag{4.18}$$

we have the ADI scheme as

$$\left[1 - \frac{\Delta t}{2}AD_x^2\right] u^{n+1^*} = \left[1 + \frac{\Delta t}{2}AD_x^2 + \Delta t C D_y^2 + 2\Delta t B D_{xy}^2\right] u^n, \tag{4.19a}$$

$$\left[1 - \frac{\Delta t}{2}CD_y^2\right] u^{n+1} = u^{n+1^*} - \frac{\Delta t}{2}CD_y^2 u^n. \tag{4.19b}$$

Here the derivative matrices D_x^2, D_y^2, and D_{xy}^2 are computed again by the method given in Section 3 and the Appendix.

5. ALGORITHMS

We discuss the numerical algorithms used to solve two-dimensional reaction diffusion problems, including the general procedure of the algorithm, the data structure, and the two-dimensional adaptive procedure. For the sake of simplicity, we only consider Eq. (4.1), the initial-boundary conditions (4.2) and (4.3), and the ADI scheme (4.10) in this section. It is similar to treat other situations of the PDEs and ADI schemes.

5.1. *Two-Dimensional Data Structure and Adaptive Procedure*

Based on the one-dimensional data structure given in Section 3, we now discuss 2D data structure and consider 2D adaptive procedure in the context of the ADI methods. For the ADI scheme only one coordinate direction, but all mesh points, will be considered in each substep. Therefore, the coordinates of all mesh points and all values of both known and unknown functions have to be stored in the forms of both, row by row and column by column, respectively, which is necessary to implement ADI sweeps. For the collocation points on each wavelet space $W_{j_x}^x \otimes W_{j_y}^y$ the active collocation points used in the computation will be stored as a sparse matrix in either compressed row storage or compressed column storage in a one-dimensional array, namely, for level $W_{j_x}^x \otimes W_{j_y}^y$. We define for each y-line (jth column of the sparse matrix):

Given jth column (mesh line along y-direction),

lencol(j, j_x, j_y)*—length of* j*th column* (y*-direction*),

ipc(j, j_x, j_y)*—pointer of first mesh on* j*th column,*

inr(j, j_x, j_y)*—row indices of* j*th column.*

and, similarly,

$$\begin{aligned}&\textit{Given } i\textit{th row } (\textit{mesh line along } x\textit{-direction}),\\ &\textit{lenrow}(i, j_x, j_y)\textit{—length of } i\textit{th row } (x\textit{-direction}),\\ &\textit{ipr}(i, j_x, j_y)\textit{—pointer of first mesh on } i\textit{th row},\\ &\textit{inc}(i, j_x, j_y)\textit{—column indices of } i\textit{th row.}\end{aligned}$$

For example, unknown function u is written as $u_r(k, j_x, j_y)$ for compressed row storage, or $u_c(k, j_x, j_y)$ for compressed column storage and index k runs over all the active points on the wavelet level (j_x, j_y), $0 \le j_x \le J_x$, $0 \le j_y \le J_y$. Therefore, with the wavelet parameter j_y, $0 \le j_y \le J_y$, or j_x, $0 \le j_x \le J_x$ given, we calculate the solution $u_c(k, j_x, j_y)$ or $u_r(k, j_x, j_y)$ for column by column or row by row along the y or x direction. Notice that a transform between $u_r(k, j_x, j_y)$ and $u_c(k, j_x, j_y)$ is also required.

Finally, we describe the procedure for 2D adaptive meshing and refinement.Given maximum wavelet level (J_x, J_y) a uniform approximation to $u(x, y)$ can be obtained by the tensor product of $(V_0^x \oplus \sum_{j_x=0}^{J_x} W_{j_x}^x) \otimes (V_0^y \oplus \sum_{j_y=0}^{J_y} W_{j_y}^y)$. The typical tensor product space is $W_{j_x}^x \otimes W_{j_y}^y$ and the wavelet coefficients are $\hat{u}_{(k,l)}^{(j_x,j_y)}$, $0 \le j_x \le J_x$, $0 \le j_y \le J_y$.

Deleting a collocation point. Given the error tolerance, we will threshold the wavelet coefficients $\hat{u}_{(k,l)}^{(j_x,j_y)}$ and discard the corresponding collocation point $(x_k^{(j_x)}, y_l^{(j_y)})$ in the following procedure. We apply the one-dimensional adaptive procedure in an x-coordinate and a y-coordinate sweep; namely, for each y-collocation point $\{y_l^{(j_y)}\}$ we apply the adaptive procedure in the space $(V_0^x \oplus \sum_{j_x=0}^{J_x} W_{j_x}^x)$ along the line $y = y_l^{(j_y)}$ and, repeating the same procedure for each x-collocation point $\{x_k^{(j_x)}\}$, we apply the adaptive procedure in the space $(V_0^y \oplus \sum_{j_y=0}^{J_y} W_{j_y}^y)$ along the line $x = x_k^{(j_x)}$. A collocation point $(x_k^{(j_x)}, y_l^{(j_y)})$ is to be deleted only if it is being flagged for deletion under the given tolerance during both an x-and y-coordinate sweep.

Adding new collocation point. In most cases, higher level wavelet space W_j^x, $j > J_x$, or W_j^y, $j > J_y$, will be introduced to create finer meshes where the solution changes rapidly. Caution is needed to create new collocation points. As shown in Fig. 2, if $\circ$ denotes old collocation points, $\times$ denotes new collocation points at $y = y^*$ during refinements along the y-direction. Sometimes, we will not have corresponding boundary points along the domain boundaries (ab and cd in Fig. 2) (marked as $\otimes$). This kind of boundary point will be added after all x- and y-direction mesh refinement. This is necessary when differentiation is to be done along the x coordinate direction at $y = y^*$. In practice, we could include all V_0^x points on $y = y^*$.

5.2. *General Steps for SW-ADI*

For convenience, we use SW-ADI3 to stand for SW-ADI with a cubic spline interpolation and SW-ADI5 for SW-ADI with a quintic spline interpolation. We will consider both of them in our numerical simulations. *Given all initial parameters*:

$$\begin{aligned}&(J_x, J_y), &&\text{wavelet levels in } x \text{ and } y \text{ directions},\\ &(L_x, L_y), &&\text{interval lengths of } x \text{ and } y \text{ directions.}\end{aligned}$$

Step 1. Generate wavelet mesh using spline wavelet transformation and the procedure given in Subsection 5.1.

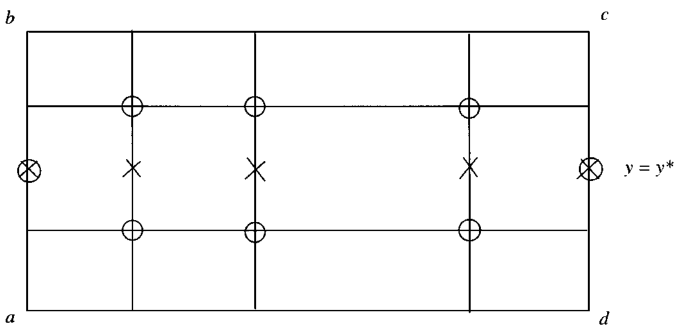

FIG. 2. Adding new collocation point (×).

Store the mesh in either compressed row x_r or compressed column x_c storage. Similarly, u_r and u_c are the storages for the solutions:

x_r,	mesh stored in compressed row storage,
x_c,	mesh stored in compressed column storage,
u_r,	solution stored in compressed row storage,
u_c,	solution stored in compressed column storage.

Given initial time $t = t_0$:

Step 2. Solving SW-ADI equations using (4.10), first calculate the right hand side

$$\left[1 + \frac{\Delta t}{2} D_y^2\right] u^n \Rightarrow \tilde{u}^n, \tag{5.1a}$$

$$\left[1 + \frac{\Delta t}{2} D_x^2\right] \tilde{u}^n \Rightarrow \utilde{u}^n, \tag{5.1b}$$

Solve the system for u^{n+1^*},

$$\left[1 - \frac{\Delta t}{2} D_x^2\right] u^{n+1^*} = \utilde{u}^n, \text{ in } x \text{ direction.} \tag{5.2a}$$

Solve the system for u^{n+1},

$$\left[1 - \frac{\Delta t}{2} D_y^2\right] u^{n+1} = u^{n+1^*}, \quad \text{in } y \text{ direction.} \tag{5.2b}$$

If $t < t_N$, $t = t + \Delta t$, repeat the above procedure.
Stop otherwise.

6. NUMERICAL RESULTS

To confirm the second-order accuracy of the algorithms presented in this paper, numerical simulations of two-dimensional reaction-diffusion system, in a rectangle domain of $\Omega = [-5,\ 15] \times [0, 2\pi]$, are performed on *Sun* Workstation *ULTRA-1*. The computational accuracy and time for uniform and nonuniform meshes, nonadaptive and adaptive cases, and different levels of wavelet approximate subspaces are investigated. The comparisons of the present method with a second-order Runge–Kutta method is also provided.

6.1. *Numerical Accuracy*

We first study the numerical accuracy of the SW-ADI algorithm for the uniform mesh case. Considering the equations

$$\begin{aligned}\frac{\partial u}{\partial t} &= \Delta u + f \\ \frac{\partial v}{\partial t} &= \Delta v - f\end{aligned} \tag{6.1}$$

in Ω, where Δ is Laplacian operator, $v = 1 - u$ with exact solution

$$\bar{u} = \begin{cases} [1 - \exp(\alpha(t - x_p))]^5, & x_p \geq t, \\ 0, & x_p < t, \end{cases} \tag{6.2}$$

where $x_p = x + \beta \sin(\omega y)$. Thus, the source term should be

$$f(x, t) = \frac{\partial}{\partial t}\bar{u} - \Delta\bar{u}, \tag{6.3}$$

where α, β, ω are given constants.

We use both cubic and quintic spline interpolation for the computation of second derivatives. Comparing ADI methods with the second-order Runge–Kutta method, we give the results in Tables 1–3 with $\alpha = 2.0$, $\beta = 0.2$, $\omega = 2.0$, where SW-ADI5 and SW-ADI3 stand for the quintic and cubic spline interpolations, respectively.
Maximum error in Tables 1–3 is calculated by

$$\varepsilon_{\max} = \max_i \left\{ \log \left| u^i_{\text{appr}} - u^i_{\text{exact}} \right| \right\},$$

where u^i_{appr} and u^i_{exact} are approximate and exact solutions, respectively. From the Table 1, we obtain second-order accuracy for ADI methods with quintic spline interpolations (SW-ADI5) and almost second-order accuracy for ADI methods with cubic spline interpolations (SW-ADI3). The Runge–Kutta method also gives results with second-order accuracy, but requires much more computational time than ADI methods.

Up to now, we find the numerical solution under the conditions of uniform meshes and without consideration of the adaptive procedure. A uniform mesh is given in Fig. 3 with wavelet level (3, 2).

TABLE 1
Numerical Results for SW-ADI5 Method

Method	Wavelet level (J_x, J_y)	Final time t	Max_error (log)	Order of accuracy
SW-ADI5	(2, 1)	0.2	−2.7852	
SW-ADI5	(3, 2)	0.2	−3.3505	1.8780
SW-ADI5	(4, 3)	0.2	−3.9509	1.9944

TABLE 2
Numerical Results for SW-ADI3 Method

Method	Wavelet level (J_x, J_y)	Final time t	Max_error (log)	Order of accuracy
SW-ADI3	(2, 1)	0.2	−0.7680	
SW-ADI3	(3, 2)	0.2	−1.2662	1.6550
SW-ADI3	(4, 3)	0.2	−1.8244	1.8543

6.2. *Nonuniform Meshes and Adaptive Procedure*

To improve computational efficiency, an adaptive procedure is implemented. As a result of an adaptive procedure, we must numerically solve discrete problems on a nonuniform mesh. We still consider Eq. (6.1). Numerical experiment indicates that second-order accuracy of the SW-ADI5 method still remains (Table 4) and that the adaptive procedure for SW-ADI3 does not show second-order accuracy, Table 5. This reflects a degeneracy of accuracy in the computation of second derivatives on a nonuniform mesh by a cubic spline. Fortunately, a quintic spline interpolation on such a nonuniform grid still gives second-order accuracy. Figure 4 gives a comparison among numerical solutions of SW-ADI5 and SW-ADI3 and an exact solution for x-cut at $y = 2.5$. It is clear from Fig. 4 that there exists an obvious difference between the approximate solution of SW-ADI3 and the exact solution.

We now consider the equation with mixed space derivative

$$\frac{\partial u}{\partial t} = A\frac{\partial^2 u}{\partial x^2} + 2B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} + f \quad \text{with } A > 0, C > 0, B^2 < AC, \tag{6.4}$$

where the exact solution is (6.2), and

$$f(x,t) = \frac{\partial u}{\partial t} - \left(A\frac{\partial^2 u}{\partial x^2} + 2B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2}\right). \tag{6.5}$$

Numerical experiment indicates that the approximation results agree well with the exact solution with the second-order accuracy (see Table 6).

6.3. *Numerical Results for Reaction–Diffusion Equations*

Finally, we consider Eq. (2.1) with source term (2.2), where the parameters $Le = 2$, $\beta = 20$, $\alpha = 0.8$, and the computational domain is $[-5, 15] \times [0, 10]$. The numerical

TABLE 3
Numerical Results for Second-Order Runge–Kutta Method

Method	Wavelet level (J_x, J_y)	Final time t	Max_error (log)	Order of accuracy
R-K2	(2, 1)	0.2	−2.4638	
R-K2	(3, 2)	0.2	−3.0858	2.0661
R-K2	(4, 3)	0.2	−3.8481	2.5326

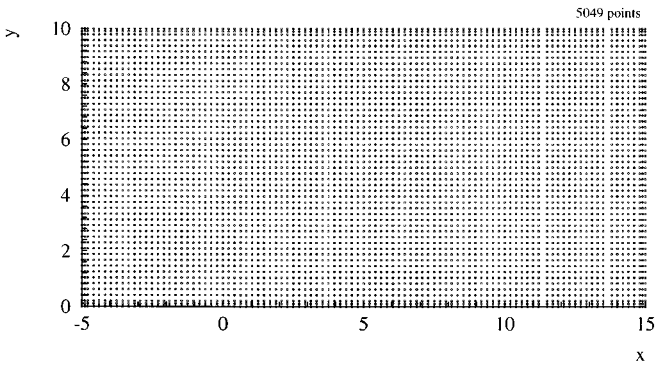

FIG. 3. A uniform mesh point distribution.

results of wavelet level (3, 2) with SW-ADI5 are obtained, where the source term is treated by a Strang-type splitting scheme [13] with a fourth-order Runge–Kutta scheme. Figures 5–9 give the nonuniform and adaptive meshes of the wavelet level (3, 2) at the five times 0, 2, 4, 6, and 8 with the control parameter $\varepsilon = 10^{-4}$. Figures 10–14 are contours of the temperature at different times ($t = 0, 2, 4, 6$, and 8). With $y = 2.5$ given, an x-cut result of the temperature solution for times $0, 1, 2, \ldots, 8$ is shown in Fig. 15. All results coincide with the results obtained by using the second-order Runge–Kutta method (see Fig. 16). According to the numerical experiment with the same computational conditions, the computational time taken by the second-order Runge–Kutta method is about three times as much as the time taken by the present SW-ADI5 method.

7. CONCLUSIONS

In this paper, we have demonstrated an efficient second-order accurate SW-ADI method for the solution of time-dependent boundary value problems of PDEs. We are extending the same approach to the full Navier–Stokes equations by combining the wavelet methods and the Beam–Warming splitting in the two-dimensional cases [14].

In some cases, if higher than second-order time accuracy is needed, the SW-ADI schemes could be used as a preconditioner in conjunction with a higher order time integrator, such as Adams–Moulton methods. Another relevant research topic will be to combine the SW-ADI and domain decomposition to form a wavelet–element method which will be applicable for more general domains. We will address these issues in later papers.

TABLE 4
Numerical Results for SW-ADI5 Method with Adaptive Procedure

Method	Wavelet level (J_x, J_y)	Final time t	Max_error (log)	Order of accuracy
SW-ADI5	(2, 1)	0.2	−2.78562	
SW-ADI5	(3, 2)	0.2	−3.35540	1.8923
SW-ADI5	(4, 3)	0.2	−3.93674	1.9312

TABLE 5
Numerical Results for SW-ADI3 Method with Adaptive Procedure

Method	Wavelet level (J_x, J_y)	Final time t	Max_error (log)	Order of accuracy
SW-ADI3	(2, 1)	0.2	−0.581356	
SW-ADI3	(3, 2)	0.2	−0.719687	0.4595
SW-ADI3	(4, 3)	0.2	−1.135147	1.3801

APPENDIX: DERIVATION OF DERIVATIVE MATRICES WITH A QUINTIC SPLINE INTERPOLATION

Let $\tau_1 < \tau_2 < \cdots < \tau_{n-1} < \tau_n$ denote nodes, where $\{\tau_2, \tau_3\}$ and $\{\tau_{n-2}, \tau_{n-1}\}$ denote not-a-knot. Assume that $P_i(x), x \in [\tau_i, \tau_{i+1}]$, $1 \le i \le n-1$ are piecewise fifth-order polynomials. Given data g_i, $1 \le i \le n$, find a quintic spline such that

$$P_i(\tau_i) = g_i, \qquad 1 \le i \le n, \tag{A.1}$$

$$P_i^{(4)}(\tau_i) = P_{i-1}^{(4)}(\tau_i), \quad 2 \le i \le n-2, \tag{A.2}$$

and at $\{\tau_2, \tau_3\}$ and $\{\tau_{n-2}, \tau_{n-1}\}$

$$\begin{aligned} P_1^{(5)}(\tau_2) &= P_2^{(5)}(\tau_2) \\ P_2^{(5)}(\tau_3) &= P_3^{(5)}(\tau_3) \\ P_{n-3}^{(5)}(\tau_{n-2}) &= P_{n-2}^{(5)}(\tau_{n-2}) \\ P_{n-2}^{(5)}(\tau_{n-1}) &= P_{n-1}^{(5)}(\tau_{n-1}), \end{aligned} \tag{A.3}$$

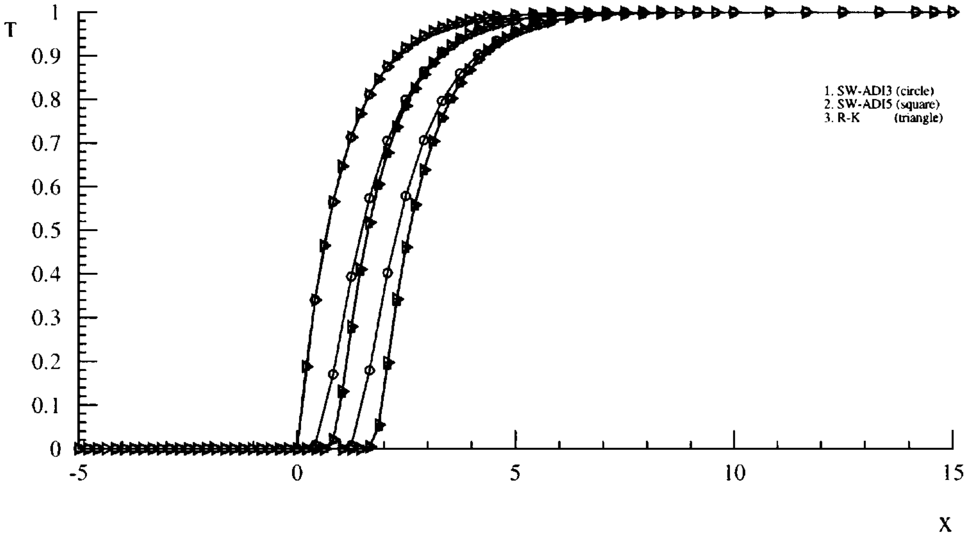

FIG. 4. Solution $u(x, y, t)$ at x-cut (comparisons).

TABLE 6
Numerical Results for SW-ADI5 Method

Method	Wavelet level (J_x, J_y)	Final time t	Max_error (log)	Order of accuracy
SW-ADI5	(2, 1)	0.2	-1.5671	
SW-ADI5	(3, 2)	0.2	-2.1519	1.9423
SW-ADI5	(4, 3)	0.2	-2.7594	2.0180

where (A.2) means that the piecewise polynomials $P_i(x)$, $1 \le i \le n-1$, are continuous up to the fourth derivatives and (A.3) indicates continuous conditions of extra derivative at the not-a-knots. We consider the unknown as

$$S_i = P_i'(\tau_i), \quad 1 \le i \le n; \quad T_i = P_i''(\tau_i), \quad 1 \le i \le n. \tag{A.4}$$

By the Newton formula, we have

$$\begin{aligned} P_i(x) = {} & P_i(\tau_i) + (x-\tau_i)[\tau_i,\tau_i]P_i + (x-\tau_i)^2[\tau_i,\tau_i,\tau_i,]P_i \\ & + (x-\tau_i)^2[\tau_i,\tau_i,\tau_i]P_i + (x-\tau_i)^3[\tau_i,\tau_i,\tau_i,\tau_{i+1}]P_i, \\ & + (x-\tau_i)^3(x-\tau_{i+1})[\tau_i,\tau_i,\tau_i,\tau_{i+1},\tau_{i+1}]P_i \\ & + (x-\tau_i)^3(x-\tau_{i+1})^2[\tau_i,\tau_i,\tau_i,\tau_{i+1},\tau_{i+1},\tau_{i+1}]P_i, \end{aligned} \tag{A.5}$$

where

$$\begin{aligned} [\tau_i]P_i &= P_i(\tau_i), \\ [\tau_i,\tau_{i+1}]P_i &= \frac{P_i(\tau_{i+1}) - P_i(\tau_i)}{\tau_{i+1}-\tau_i}, \end{aligned}$$

and

$$[\tau_i,\tau_i]P_i = \lim_{\tau\to\tau_i} \frac{P_i(\tau) - P_i(\tau_i)}{\tau - \tau_i}.$$

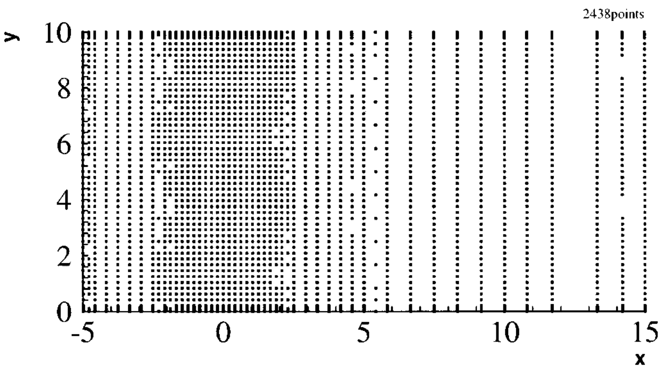

FIG. 5. Nonuniform and adaptive mesh point distributions at $t = 0$.

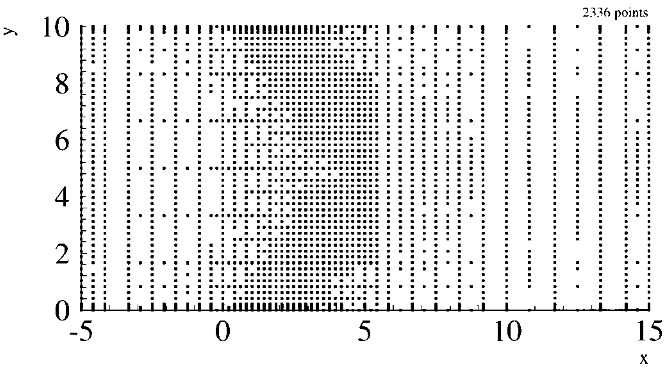

FIG. 6. Nonuniform and adaptive mesh point distributions at $t = 2$.

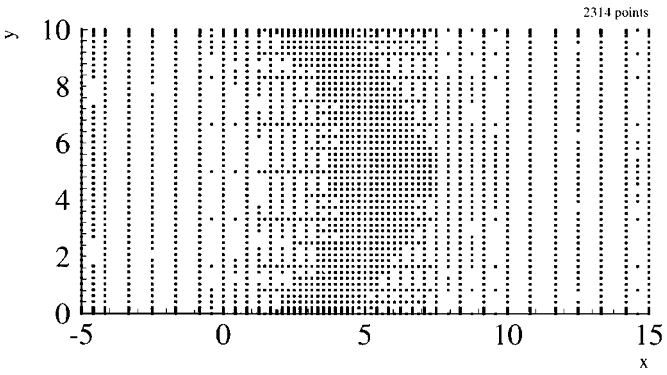

FIG. 7. Nonuniform and adaptive mesh point distributions at $t = 4$.

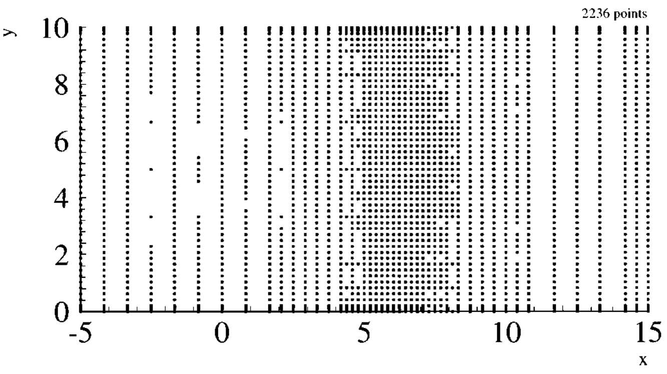

FIG. 8. Nonuniform and adaptive mesh point distributions at $t = 6$.

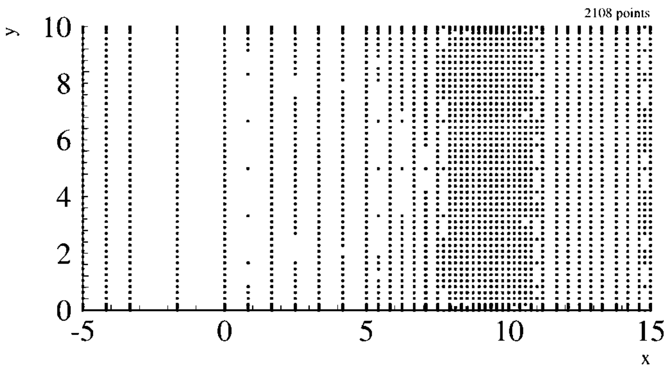

FIG. 9. Nonuniform and adaptive mesh point distributions at $t = 8$.

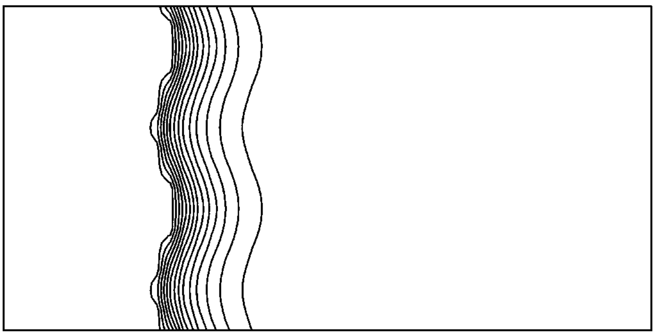

FIG. 10. Contour of temperature at $t = 0$.

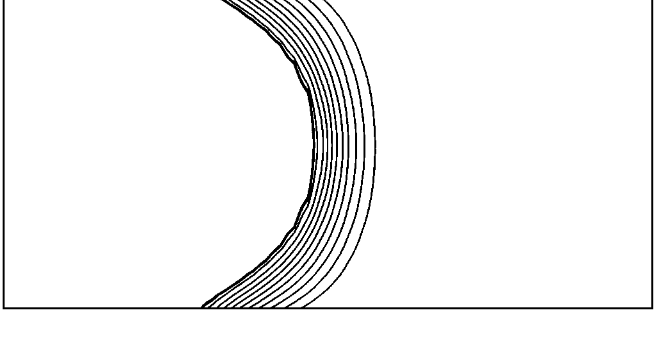

FIG. 11. Contour of temperature at $t = 2$.

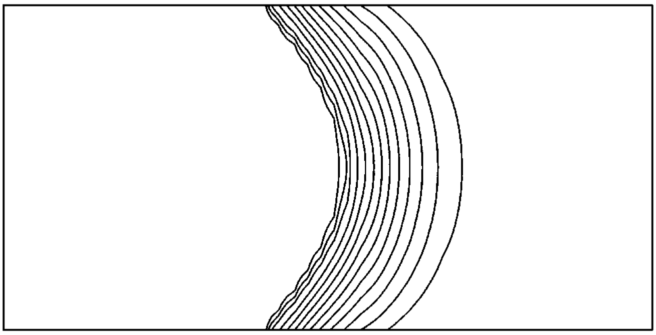

FIG. 12. Contour of temperature at $t = 4$.

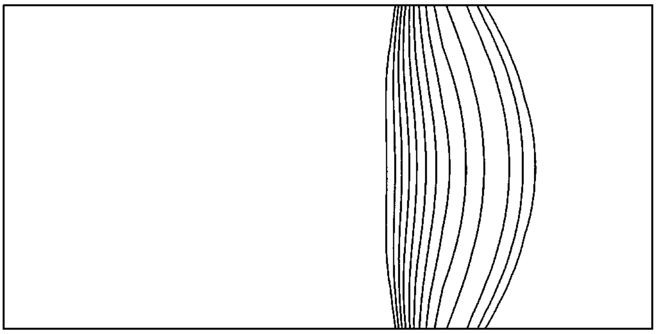

FIG. 13. Contour of temperature at $t = 6$.

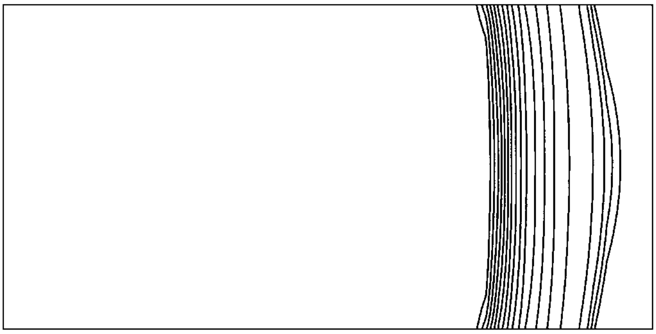

FIG. 14. Contour of temperature at $t = 8$.

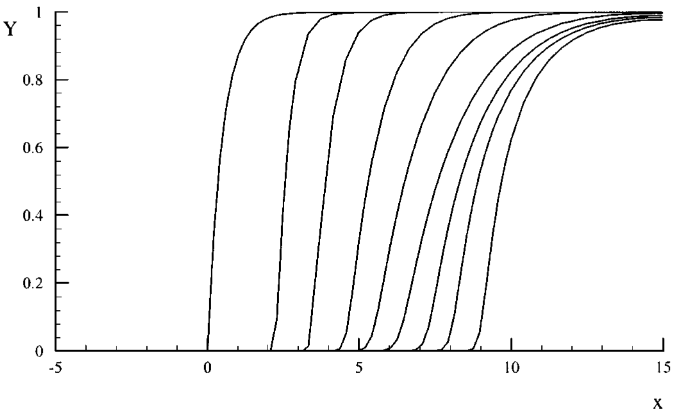

FIG. 15. Temperature of x-cut at $y = 2.5$ and $t = 0, 1, \ldots, 8$.

Assume that

$$P_i(x) = C_{1i} + C_{2i}(x - \tau_i) + C_{3i}(x - \tau_i)^2 + C_{4i}(x - \tau_i)^3 + C_{5i}(x - \tau_i)^4 + C_{6i}(x - \tau_i)^5; \tag{A.6}$$

then

$$C_{1i} = P_i(\tau_i) = g_i,$$
$$C_{2i} = P_i'(\tau_i),$$

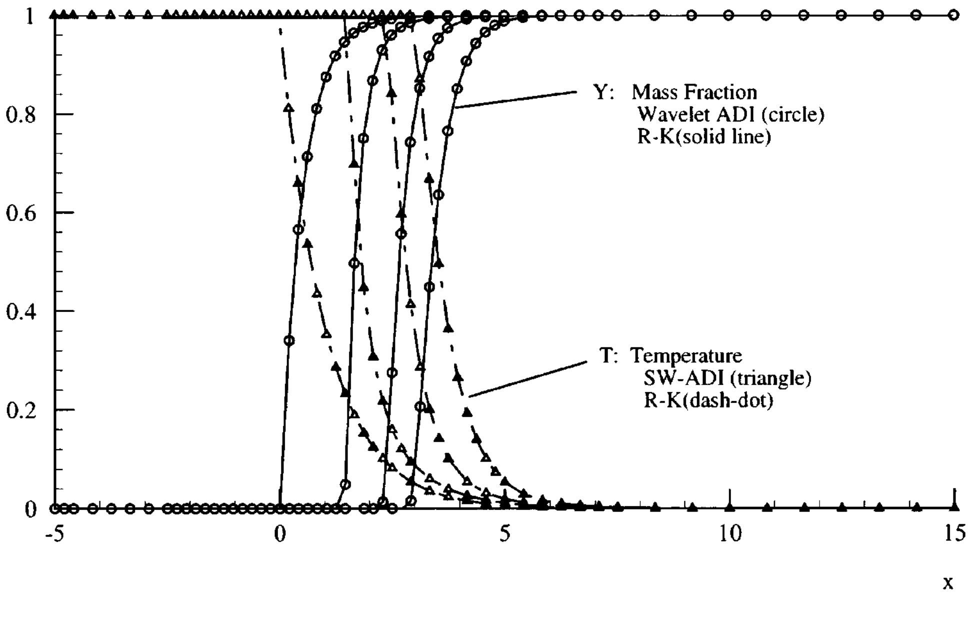

FIG. 16. Comparison between the SW-ADI and the second-order Runge–Kutta method at $t = 0, 0.5, 1, 1.5$.

$$C_{3i} = \frac{1}{2}P_i''(\tau_i) = \frac{1}{2}T_i,$$

$$C_{4i} = \frac{1}{3!}P_i'''(\tau_i) = \frac{1}{6}\left\{6[\,,\,,\,]P + 6(\tau_{i+1}-\tau_i)[\,,\,,\,,\,]P + 6(\tau_{i+1}-\tau_i)^2[\,,\,,\,,\,,\,]P\right\}$$

$$= \frac{1}{\Delta\tau_i^2}\left\{10g_i - 6S_i - 3\Delta\tau_i T_i + \Delta\tau_i T_i - 4S_{i+1}\right\},$$

$$C_{5i} = \frac{1}{4!}P_i^{(4)}(\tau_i) = \frac{1}{24}\{4\times 3\times 2[\,,\,,\,,\,]P + 48(\tau_i-\tau_{i+1})[\,,\,,\,,\,,\,]P\}$$

$$= \frac{1}{\Delta\tau_i^3}\left\{7S_{i+1} + 8S_i - 2\Delta\tau_i T_{i+1} + 3\Delta\tau_i T_i - 15g_i\right\},$$

and

$$C_{6i} = \frac{1}{5!}P_i^{(5)}(\tau_i) = \frac{1}{5!}\{5![\,,\,,\,,\,,\,]P\}$$

$$= \frac{1}{\Delta\tau_i^4}\left\{\Delta\tau_i T_{i+1} - \Delta\tau_i T_i - 3S_{i+1} - 3S_i + 6g_i\right\}.$$

Now we derive the equations among unknowns by means of continuity of the derivative at the internal knots; i.e.,

$$P_{i-1}^{(3)}(\tau_i) = P_i^{(3)}(\tau_i), \quad P_{i-1}^{(4)}(\tau_i) = P_i^{(4)}(\tau_i), \quad 3 \le i \le n-2. \tag{A.7}$$

From the definition (A.6), we obtain directly from the first part of (A.7)

$$P_{i-1}^{(3)}(\tau_i) = 3!C_{4i}, \tag{A.8a}$$

$$P_{i-1}^{(3)}(\tau_i) = 3!C_{4i-1} + 4\cdot 3\cdot 2C_{5i-1}(\tau_i-\tau_{i-1}) + 5\cdot 4\cdot 3C_{6i-1}(\tau_i-\tau_{i-1})^2. \tag{A.8b}$$

Let (A.8a) equal (A.8b); we have

$$C_{4i} = C_{4i-1} + 4C_{5i-1}(\tau_i-\tau_{i-1}) + 10C_{6i-1}(\tau_i-\tau_{i-1})^2,$$

that is,

$$\frac{1}{\Delta\tau_i^2}\left\{10g_i - 6S_i - 3\Delta\tau_i T_i + \Delta\tau_i T_{i+1} - 4S_{i+1}\right\}$$
$$= \frac{1}{\Delta\tau_{i-1}^2}\left\{10g_{i-1} - 6S_{i-1} - 3\Delta\tau_{i-1}T_{i-1} + \Delta\tau_{i-1}T_i - 4S_i\right\}$$
$$+ \frac{4}{\Delta\tau_{i-1}^3}\left\{7S_i + 8S_{i-1} - 2\Delta\tau_{i-1}T_i + 3\Delta\tau_{i-1}T_{i-1} - 15g_{i-1}t\right\}\Delta\tau_{i-1}$$
$$+ \frac{10}{\Delta\tau_{i-1}^4}\left\{\Delta\tau_{i-1}T_i - \Delta\tau_{i-1}T_{i-1} - 3S_i - 3S_{i-1} + 6g_{i-1}\right\}\Delta\tau_{i-1}^2$$

which gives

$$\frac{4}{\Delta\tau_{i-1}^2}S_{i-1} + \left(\frac{6}{\Delta\tau_{i-1}^2} - \frac{6}{\Delta\tau_i^2}\right)S_i - \frac{4}{\Delta\tau_i^2}S_{i+1} + \frac{1}{\Delta\tau_{i-1}}T_{i-1}$$

$$- \left(\frac{3}{\Delta\tau_{i-1}} + \frac{3}{\Delta\tau_i}\right)T_i \;+\; -\frac{1}{\Delta\tau_i}T_i = \frac{10g_{i-1}}{\Delta\tau_{i-1}^2} - \frac{10g_i}{\Delta\tau_i^2}. \tag{A.9}$$

Similarly, we have, by using the second part of formula (A.7),

$$P_{i-1}^{(4)}(\tau_i) = 4!C_{4i}, \tag{A.10a}$$

$$P_{i-1}^{(4)}(\tau_i) = 4!C_{5i-1} + 5\cdot 4\cdot 3\cdot 2C_{6i-1}(\tau_i - \tau_{i-1}). \tag{A.10b}$$

Let (A.10a) equal (A.10b); we have

$$C_{4i} = C_{5i-1} + 5C_{6i-1}(\tau_i - \tau_{i-1});$$

that is,

$$\frac{7S_{i+1} + 8S_i - 2\Delta\tau_i T_{i+1} + 3\Delta\tau_i T_i - 15g_i}{\Delta\tau_i^3}$$
$$= \frac{7S_i + 8S_{i-1} - 2\Delta\tau_{i-1}T_i + 3\Delta\tau_{i-1}T_{i-1} - 15g_{i-1}}{\Delta\tau_i^3}$$
$$+5\frac{\Delta\tau_{i-1}T_i - \Delta\tau_{i-1}T_{i-1} - 3S_i - 3S_{i-1} + 6g_{i-1}}{\Delta\tau_i^3},$$

which gives

$$\frac{7}{\Delta\tau_{i-1}^3}S_{i-1} + \left(\frac{8}{\Delta\tau_{i-1}^3} + \frac{8}{\Delta\tau_i^3}\right)S_i + \frac{7}{\Delta\tau_i^3}S_{i+1} \;+\; -\frac{2}{\Delta\tau_{i-1}^2}T_{i-1}$$

$$+ \left(\frac{3}{\Delta\tau_{i-1}^2} + \frac{3}{\Delta\tau_i^2}\right)T_i - \frac{2}{\Delta\tau_i^2}T_{i+1} = \frac{15g_{i-1}}{\Delta\tau_{i-1}^3} + \frac{15g_i}{\Delta\tau_i^3}. \tag{A.11}$$

Finally, we obtain the equations, by using conditions (A.3),

$$\frac{3}{\Delta\tau_1^4}S_1 + \left(\frac{3}{\Delta\tau_1^4} - \frac{3}{\Delta\tau_2^4}\right)S_2 - \frac{3}{\Delta\tau_2^4}S_3 + \frac{1}{\Delta\tau_1^3}T_i\left[\frac{1}{\Delta\tau_1^3} + \frac{1}{\Delta\tau_2^3}\right]T_2 + \frac{1}{\Delta\tau_2^3}T_3$$

$$= \frac{6g_1}{\Delta\tau_1^4} - \frac{6g_2}{\Delta\tau_2^4} \quad \text{at knot } \tau_2, \tag{A.12}$$

$$\frac{3}{\Delta\tau_2^4}S_2 + \left(\frac{3}{\Delta\tau_2^4} - \frac{3}{\Delta\tau_3^4}\right)S_3 - \frac{3}{\Delta\tau_3^4}S_4 + \frac{1}{\Delta\tau_2^3}T_2 - \left(\frac{1}{\Delta\tau_2^3} + \frac{1}{\Delta\tau_3^3}\right)T_3 + \frac{1}{\Delta\tau_3^3}T_4$$

$$= \frac{6g_1}{\Delta\tau_2^4} - \frac{6g_2}{\Delta\tau_3^4} \quad \text{at knot } \tau_3, \tag{A.13}$$

$$\begin{aligned}
&\frac{3}{\Delta\tau_{n-3}^4}S_{n-3}+\left(\frac{3}{\Delta\tau_{n-3}^4}-\frac{3}{\Delta\tau_{n-2}^4}\right)S_{n-2}-\frac{3}{\Delta\tau_{n-2}^4}S_{n-1}+\frac{1}{\Delta\tau_{n-3}^3}T_{n-3}\\
&\quad-\left(\frac{1}{\Delta\tau_{n-3}^3}+\frac{1}{\Delta\tau_{n-2}^3}\right)T_{n-2}+\frac{1}{\Delta\tau_{n-2}^3}T_{n-1}\\
&\quad=\frac{6g_{n-3}}{\Delta\tau_{n-3}^4}-\frac{6g_{n-2}}{\Delta\tau_{n-2}^4}\quad\text{at knot }\tau_{n-2},
\end{aligned}\tag{A.14}$$

$$\begin{aligned}
&\frac{3}{\Delta\tau_{n-2}^4}S_{n-2}+\left(\frac{3}{\Delta\tau_{n-2}^4}-\frac{3}{\Delta\tau_{n-1}^4}\right)S_{n-1}-\frac{3}{\Delta\tau_{n-1}^4}S_{n}+\frac{1}{\Delta\tau_{n-2}^3}T_{n-2}\\
&\quad-\left(\frac{1}{\Delta\tau_{n-2}^3}+\frac{1}{\Delta\tau_{n-1}^3}\right)T_{n-1}+\frac{1}{\Delta\tau_{n-1}^3}T_{n}\\
&\quad=\frac{6g_{n-2}}{\Delta\tau_{n-2}^4}-\frac{6g_{n-1}}{\Delta\tau_{n-1}^4}\quad\text{at knot }\tau_{n-1}.
\end{aligned}\tag{A.15}$$

In summary, we have the $2n$ equations for $2n$ unknowns $\{S_i, S_2, \ldots, S_n\}$ and $\{T_1, T_2, \ldots, T_n\}$,

$$\begin{aligned}
&\frac{3}{\Delta\tau_1^4}S_1+\left(\frac{3}{\Delta\tau_1^4}-\frac{3}{\Delta\tau_2^4}\right)S_2-\frac{3}{\Delta\tau_2^4}S_3+\frac{1}{\Delta\tau_1^3}T_1-\left(\frac{1}{\Delta\tau_1^3}+\frac{1}{\Delta\tau_2^3}\right)T_2+\frac{1}{\Delta\tau_2^3}T_3\\
&=\frac{6g_1}{\Delta\tau_1^4}-\frac{6g_2}{\Delta\tau_2^4}\quad\text{at knot }\tau_2,
\end{aligned}\tag{A.16a}$$

$$\begin{aligned}
&\frac{3}{\Delta\tau_2^4}S_2+\left(\frac{3}{\Delta\tau_2^4}-\frac{3}{\Delta\tau_3^4}\right)S_3-\frac{3}{\Delta\tau_3^4}S_4+\frac{1}{\Delta\tau_2^3}T_2-\left(\frac{1}{\Delta\tau_2^3}+\frac{1}{\Delta\tau_3^3}\right)T_3+\frac{1}{\Delta\tau_3^3}T_4\\
&=\frac{6g_1}{\Delta\tau_2^4}-\frac{6g_2}{\Delta\tau_3^4}\quad\text{at knot }\tau_3;
\end{aligned}\tag{A.16b}$$

$$\begin{aligned}
&\frac{4}{\Delta\tau_{i-1}^2}S_{i-1}+\left(\frac{6}{\Delta\tau_{i-1}^2}-\frac{6}{\Delta\tau_i^2}\right)S_i-\frac{4}{\Delta\tau_i^2}S_{i+1}+\frac{1}{\Delta\tau_{i-1}}T_{i-1}\\
&\quad-\left(\frac{3}{\Delta\tau_{i-1}}+\frac{3}{\Delta\tau_i}\right)T_i\ +\ -\frac{1}{\Delta\tau_i}T_i\\
&\quad=\frac{10g_{i-1}}{\Delta\tau_{i-1}^2}-\frac{10g_i}{\Delta\tau_i^2}\quad\text{at knots }\tau_i,\ 2\le i\le n-1,
\end{aligned}\tag{A.17a}$$

$$\frac{7}{\Delta\tau_{i-1}^3}S_{i-1} + \left(\frac{8}{\Delta\tau_{i-1}^3} + \frac{8}{\Delta\tau_i^3}\right)S_i + \frac{7}{\Delta\tau_i^3}S_{i+1} + -\frac{2}{\Delta\tau_{i-1}^2}T_{i-1}$$

$$+ \left(\frac{3}{\Delta\tau_{i-1}^2} + \frac{3}{\Delta\tau_i^2}\right)T_i - \frac{2}{\Delta\tau_i^2}T_{i+1}$$

$$= \frac{15g_{i-1}}{\Delta\tau_{i-1}^3} + \frac{15g_i}{\Delta\tau_i^3} \quad \text{at knot } \tau_i,\ 2 \le i \le n-1; \tag{A.17b}$$

$$\frac{3}{\Delta\tau_{n-3}^4}S_{n-3} + \left(\frac{3}{\Delta\tau_{n-3}^4} - \frac{3}{\Delta\tau_{n-2}^4}\right)S_{n-2} - \frac{3}{\Delta\tau_{n-2}^4}S_{n-1} + \frac{1}{\Delta\tau_{n-3}^3}T_{n-3}$$

$$- \left(\frac{1}{\Delta\tau_{n-3}^3} + \frac{1}{\Delta\tau_{n-2}^3}\right)T_{n-2} + \frac{1}{\Delta\tau_{n-2}^3}T_{n-1}$$

$$= \frac{6g_{n-3}}{\Delta\tau_{n-3}^4} - \frac{6g_{n-2}}{\Delta\tau_{n-2}^4} \quad \text{at knot } \tau_{n-2}, \tag{A.18a}$$

$$\frac{3}{\Delta\tau_{n-2}^4}S_{n-2} + \left(\frac{3}{\Delta\tau_{n-2}^4} - \frac{3}{\Delta\tau_{n-1}^4}\right)S_{n-1} - \frac{3}{\Delta\tau_{n-1}^4}S_n + \frac{1}{\Delta\tau_{n-2}^3}T_{n-2}$$

$$- \left(\frac{1}{\Delta\tau_{n-2}^3} + \frac{1}{\Delta\tau_{n-1}^3}\right)T_{n-1} + \frac{1}{\Delta\tau_{n-1}^3}T_n$$

$$= \frac{6g_{n-2}}{\Delta\tau_{n-2}^4} - \frac{6g_{n-1}}{\Delta\tau_{n-1}^4} \quad \text{at knot } \tau_{n-1}. \tag{A.18b}$$

Rearranging the order of the above equations; i.e.,

knot 2—(A.16a),
knot 2—(A.17a),
knot 3—(A.16b),
knot 3—(A.17a),
knot 4—(A.17a),
knot n-1—(A.17a);
knot 2—(A.17b),
knot 3—(A.17b),
knot n-3—(A.17b),
knot n-2—(A.17b),
knot n-2—(A.18a),
knot n-1—(A.17b),
knot n-1—(A.18b).

we have the matrix representations of (A.16)–(A.18)

$$\begin{bmatrix} A_1 & B_2 \\ B_1 & A_2 \end{bmatrix} \begin{bmatrix} S \\ T \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix}, \tag{A.19}$$

where

$$A_1 = \begin{bmatrix}
\frac{3}{\Delta\tau_1^4} & \left(\frac{3}{\Delta\tau_1^4} - \frac{3}{\Delta\tau_2^4}\right) & -\frac{3}{\Delta\tau_2^4} & & & & & \\
\frac{4}{\Delta\tau_1^2} & \left(\frac{6}{\Delta\tau_1^2} - \frac{6}{\Delta\tau_2^2}\right) & -\frac{4}{\Delta\tau_2^2} & & & & 0 & \\
 & \frac{3}{\Delta\tau_2^4} & \left(\frac{3}{\Delta\tau_2^4} - \frac{3}{\Delta\tau_3^4}\right) & -\frac{3}{\Delta\tau_3^4} & & & & \\
 & \frac{4}{\Delta\tau_2^2} & \left(\frac{6}{\Delta\tau_2^2} - \frac{6}{\Delta\tau_3^2}\right) & -\frac{4}{\Delta\tau_4^3} & & & & \\
 & & \frac{4}{\Delta\tau_3^2} & \left(\frac{6}{\Delta\tau_3^2} - \frac{6}{\Delta\tau_4^2}\right) & -\frac{4}{\Delta\tau_4^2} & & & \\
 & & & \bullet & \bullet & \bullet & & \\
 & 0 & & & \frac{4}{\Delta\tau_{n-3}^2} & \left(\frac{6}{\Delta\tau_{n-3}^2} - \frac{6}{\Delta\tau_{n-2}^2}\right) & -\frac{4}{\Delta\tau_{n-2}^2} & \\
 & & & & & \frac{4}{\Delta\tau_{n-2}^2} & \left(\frac{6}{\Delta\tau_{n-2}^2} - \frac{6}{\Delta\tau_{n-1}^2}\right) & -\frac{4}{\Delta\tau_{n-1}^2}
\end{bmatrix}$$

$$B_2 = \begin{bmatrix}
\frac{1}{\Delta\tau_1^3} & -\left(\frac{1}{\Delta\tau_1^3} + \frac{1}{\Delta\tau_2^3}\right) & \frac{1}{\Delta\tau_2^3} & & & & & \\
\frac{1}{\Delta\tau_1} & -\left(\frac{3}{\Delta\tau_1} + \frac{3}{\Delta\tau_2}\right) & \frac{1}{\Delta\tau_2} & & & & 0 & \\
 & \frac{1}{\Delta\tau_2^3} & -\left(\frac{1}{\Delta\tau_2^3} + \frac{1}{\Delta\tau_3^3}\right) & \frac{1}{\Delta\tau_3^3} & & & & \\
 & \frac{1}{\Delta\tau_2} & -\left(\frac{3}{\Delta\tau_2} + \frac{3}{\Delta\tau_3}\right) & \frac{1}{\Delta\tau_3} & & & & \\
 & & \frac{1}{\Delta\tau_3} & -\left(\frac{3}{\Delta\tau_3} + \frac{3}{\Delta\tau_4}\right) & \frac{1}{\Delta\tau_4} & & & \\
 & & & \bullet & \bullet & \bullet & & \\
 & 0 & & & \frac{1}{\Delta\tau_{n-3}} & -\left(\frac{3}{\Delta\tau_{n-3}} + \frac{3}{\Delta\tau_{n-2}}\right) & \frac{1}{\Delta\tau_{n-2}} & \\
 & & & & & \frac{1}{\Delta\tau_{n-2}} & -\left(\frac{3}{\Delta\tau_{n-2}} + \frac{3}{\Delta\tau_{n-1}}\right) & \frac{1}{\Delta\tau_{n-1}}
\end{bmatrix}$$

$$B_1 = \begin{bmatrix}
\frac{7}{\Delta\tau_1^3} & \left(\frac{8}{\Delta\tau_1^3} + \frac{8}{\Delta\tau_2^3}\right) & \frac{7}{\Delta\tau_2^3} & & & & & \\
 & \frac{7}{\Delta\tau_2^3} & \left(\frac{8}{\Delta\tau_2^3} + \frac{8}{\Delta\tau_3^3}\right) & \frac{7}{\Delta\tau_3^3} & & & 0 & \\
 & & \bullet & \bullet & \bullet & & & \\
 & & & \frac{7}{\Delta\tau_{n-4}^3} & \left(\frac{8}{\Delta\tau_{n-4}^3} + \frac{8}{\Delta\tau_{n-3}^3}\right) & \frac{7}{\Delta\tau_{n-3}^3} & & \\
 & & & & \frac{7}{\Delta\tau_{n-3}^3} & \left(\frac{8}{\Delta\tau_{n-3}^3} + \frac{8}{\Delta\tau_{n-2}^3}\right) & \frac{7}{\Delta\tau_{n-2}^3} & \\
 & & & & \frac{3}{\Delta\tau_{n-3}^4} & \left(\frac{3}{\Delta\tau_{n-3}^4} - \frac{3}{\Delta\tau_{n-2}^4}\right) & -\frac{3}{\Delta\tau_{n-2}^4} & \\
 & 0 & & & & \frac{7}{\Delta\tau_{n-2}^3} & \left(\frac{8}{\Delta\tau_{n-2}^3} + \frac{8}{\Delta\tau_{n-1}^3}\right) & \frac{7}{\Delta\tau_{n-1}^3} \\
 & & & & & \frac{3}{\Delta\tau_{n-2}^4} & \left(\frac{3}{\Delta\tau_{n-2}^4} - \frac{3}{\Delta\tau_{n-1}^4}\right) & -\frac{3}{\Delta\tau_{n-1}^4}
\end{bmatrix}$$

$$A_2 = \begin{bmatrix}
-\frac{2}{\Delta\tau_1^2} & \left(\frac{3}{\Delta\tau_1^2}+\frac{3}{\Delta\tau_2^2}\right) & -\frac{2}{\Delta\tau_2^2} & & & & \\
 & -\frac{2}{\Delta\tau_2^2} & \left(\frac{3}{\Delta\tau_2^2}+\frac{3}{\Delta\tau_3^2}\right) & -\frac{2}{\Delta\tau_3^2} & & & 0 \\
 & & \bullet & \bullet & \bullet & & \\
 & & -\frac{2}{\Delta\tau_{n-4}^2} & \left(\frac{3}{\Delta\tau_{n-4}^2}+\frac{3}{\Delta\tau_{n-3}^2}\right) & -\frac{2}{\Delta\tau_{n-3}^2} & & \\
 & & & -\frac{2}{\Delta\tau_{n-3}^2} & \left(\frac{3}{\Delta\tau_{n-3}^2}+\frac{3}{\Delta\tau_{n-2}^2}\right) & -\frac{2}{\Delta\tau_{n-2}^2} & \\
 & & & \frac{1}{\Delta\tau_{n-3}^3} & -\left(\frac{1}{\Delta\tau_{n-3}^3}+\frac{1}{\Delta\tau_{n-2}^3}\right) & \frac{1}{\Delta\tau_{n-2}^3} & \\
0 & & & & -\frac{2}{\Delta\tau_{n-2}^2} & \left(\frac{3}{\Delta\tau_{n-2}^2}+\frac{3}{\Delta\tau_{n-1}^2}\right) & -\frac{2}{\Delta\tau_{n-1}^2} \\
 & & & & \frac{1}{\Delta\tau_{n-2}^3} & -\left(\frac{1}{\Delta\tau_{n-2}^3}+\frac{1}{\Delta\tau_{n-1}^3}\right) & \frac{1}{\Delta\tau_{n-1}^3}
\end{bmatrix}$$

and

$$S = [S_1, S_2, \dots, S_n]^T, \quad T = [T_1, T_2, \dots, T_n]^T,$$
$$c = [c_1, c_2, \dots, c_n]^T = M_c g, \quad d = [d_1, d_2, \dots, d_n]^T = M_d g,$$

where

$$g = [g_1, g_2, \dots, g_n]^T,$$

$$M_c = \begin{bmatrix}
\frac{6}{\Delta\tau_1^4} & -\frac{6}{\Delta\tau_2^4} & & & & & \\
\frac{10}{\Delta\tau_1^2} & \frac{10}{\Delta\tau_2^2} & & & & & 0 \\
 & \frac{6}{\Delta\tau_2^4} & -\frac{6}{\Delta\tau_3^4} & & & & \\
 & \frac{10}{\Delta\tau_2^2} & \frac{10}{\Delta\tau_3^2} & & & & \\
 & & \frac{10}{\Delta\tau_3^2} & \frac{10}{\Delta\tau_4^2} & & & \\
 & & & \bullet & \bullet & & \\
 & 0 & & & \frac{10}{\Delta\tau_{n-3}^2} & \frac{10}{\Delta\tau_{n-2}^2} & \\
 & & & & & \frac{10}{\Delta\tau_{n-2}^2} & \frac{10}{\Delta\tau_{n-1}^2}
\end{bmatrix},$$

and

$$M_d = \begin{bmatrix}
\frac{15}{\Delta\tau_1^3} & -\frac{15}{\Delta\tau_2^3} & & & & & \\
 & \frac{15}{\Delta\tau_2^3} & \frac{15}{\Delta\tau_3^3} & & & & 0 \\
 & & \bullet & \bullet & & & \\
 & & & \frac{6}{\Delta\tau_{n-4}^2} & \frac{15}{\Delta\tau_{n-3}^2} & & \\
 & & & & \frac{15}{\Delta\tau_{n-3}^2} & \frac{15}{\Delta\tau_{n-2}^2} & \\
 & & & & \frac{6}{\Delta\tau_{n-3}^2} & \frac{6}{\Delta\tau_{n-2}^2} & \\
 & 0 & & & & \frac{15}{\Delta\tau_{n-2}^2} & \frac{15}{\Delta\tau_{n-1}^2} \\
 & & & & & \frac{16}{\Delta\tau_{n-2}^2} & \frac{6}{\Delta\tau_{n-1}^2}
\end{bmatrix}$$

Introducing two transform matrices R_1 and R_2 as

$$R_1 = \begin{bmatrix} I & -B_2 A_2^{-1} \\ 0 & I \end{bmatrix}, \quad R_2 = \begin{bmatrix} I & 0 \\ -B_1 A_1^{-1} & I \end{bmatrix},$$

we have, by multiplying (A.19) by R_1 and R_2, respectively,

$$S = D_1^{(5)} g, \tag{A.20a}$$

where

$$D_1^{(5)} = \left[A_1 - B_2 A_2^{-1} B_1\right]^{-1} \tilde{M}_c, \quad \tilde{M}_c = M_c - B_2 A_2^{-1} M_d,$$

and

$$T = D_2^{(5)} g, \tag{A.20b}$$

where

$$D_2^{(5)} = \left[A_2 - B_1 A_1^{-1} B_2\right]^{-1} \tilde{M}_d, \quad \tilde{M}_d = M_d - B_1 A_1^{-1} M_c.$$

The eigenvalue plots of the *Crank-Nicolson* method $[I - (r/2)\, D_2^{(5)}][I + (r/2)\, D_2^{(5)}]$ for both third and fifth spline interpolations are given in Figs. 17–18. The schemes are stable, since all real parts of eigenvalues are less than zero and the minimum is larger than minus one.

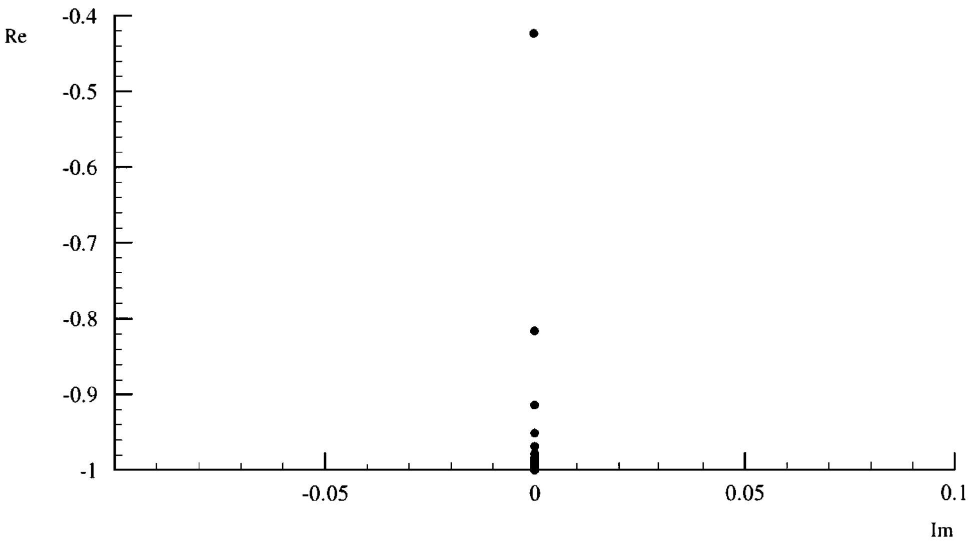

FIG. 17. Real part vs image part of eigenvalues of the C-N matrix $[I - (r/2)D_2^{(3)}][I + (r/2)D_2^{(3)}]$ for a cubic spline.

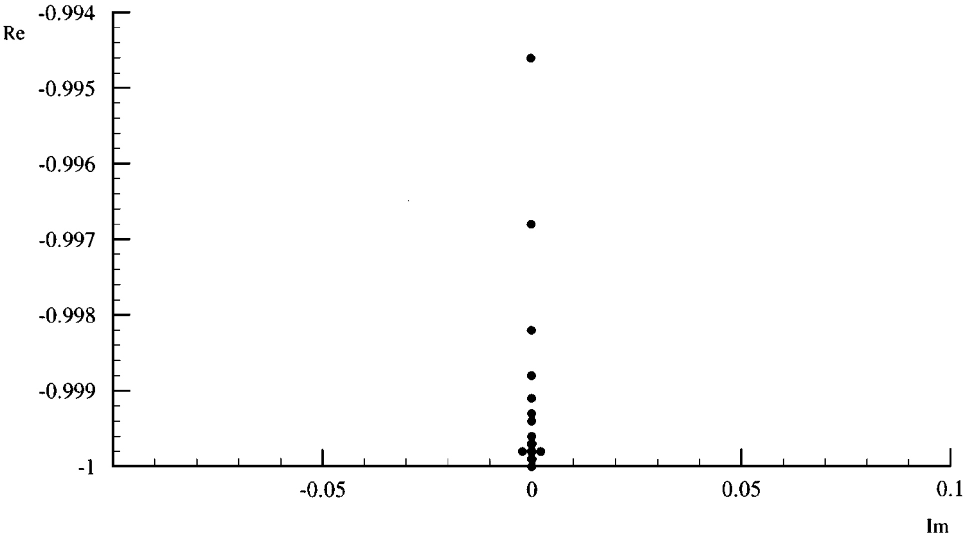

FIG. 18. Real part vs image part of eigenvalues of the C-N matrix $[I - (r/2)D_2^{(5)}][I + (r/2)D_2^{(5)}]$ for a quintic spline.

ACKNOWLEDGMENTS

This work is supported by Air Force Office of Scientific Research Grant F49620-96-1-0412. The authors thank Dr. Jianzhong Wang for many interesting discussions during the course of this work.

REFERENCES

1. G. Beylkin, R. Coifman, and V. Rokhlin, Fast wavelet transforms and numerical algorithms I, *Comm. Pure Appl. Math.* **44**, 141 (1991).
2. I. Daubechies, Orthogonal bases of compactly supported wavelet, *Comm. Pure Appl. Math.* **41**, 909 (1988).
3. B. Alpert, A class of bases in L^2 for the Sparse representation of integral operators, *SIAM J. Math. Anal.* **24**(1), 246 (1993).
4. W. Cai and J. Wang, Adaptive multiresolution collocation methods for initial boundary value problems of nonlinear PDEs, *SIAM J. Numer. Anal.* **33**(3), 937 (1996).
5. L. Jameson, *On the Wavelet-Optimized Finite Difference Method*, ICASE Report No. 94-9, NASA CR-191601.
6. L. Jameson, On the spline-based wavelet differentiation matrix, *Appl. Numer. Math.* **17**(33), 33 (1995).
7. Y. Maday, V. Perrier, and J. C. Ravel, Adaptive dynamique sur bases d'ondelettes pour l'approximation d'equations aus derivees partielles, *C.R. Acad. Sci. Paris* **312**, 405 (1991).
8. J. Froehlich and K. Schneider, An adaptive wavelet-Vagvelette algorithm for the solution of PDEs, *J. Comput. Phys.* **130**, 174 (1997).
9. B. Larrouturou, A conservative adaptive method for flame propagation, *SIAM J. Sci. Stat. Comput.* **10**(4), 742 (1989).
10. C. R. de Boor, *A Practical Guide to Spline* (Springer-Verlag, New York, 1978).
11. R. Adames, Sobolev Spaces, Academic Press, New York, 1975.
12. A. R. Mitchell and D. F. Griffith, *The Finite Difference Method in Partial Differential Equations* (Wiley, New York, 1980).
13. G. Strang, On the construction and comparison of difference schemes, *SIAM J. Numer. Anal.* **5**(3), 1968.
14. R. M. Beam and R. F. Warming, An implicit factored scheme for the compressible Navier-Stokes equations, **16**(4), 393 (1978).
15. A. Harten, Multiresolution algorithm for the numerical solution of hyperbolic conservation laws, *Comm. Pure Appl. Math.* **48**(12), 1305 (1995).