J. Indian Inst. Sci., Sep.–Oct. 2005, **85**, 253–264

A parallel algorithm for the inhomogeneous heat equations

M. AKRAM
University College of Information Technology, Punjab University, Old Campus, Lahore-54000, Pakistan.
email: m.akram@pucit.edu.pk, makrammath@yahoo.com

Received on July 4, 2005; Revised on September 19, 2005.

Abstract

In this paper, L_0-stable parallel algorithm is presented for the numerical solutions of the inhomogeneous heat equations. The algorithm is developed by approximating the second-order spatial partial derivative by finite-difference approximation and a matrix exponential function by a rational approximation having distinct real poles. Linear and nonlinear problems are solved using the algorithm and comparisons are made with results from the literature confirming the accuracy of the algorithm.

Keywords: Method of lines, rational approximate, parallel algorithm.

1. Introduction

Certain types of physical problems can be modeled by the inhomogeneous heat equation:

$$u_t = u_{xx} + s(x, t), \quad 0 < x < X; \quad t > 0,$$

$$u(0, t) = f_1(t), \quad 0 < t \leq T,$$

$$u(X, t) = f_2(t), \quad 0 < t \leq T,$$

$$u(x, 0) = g(x); \quad 0 < x < X.$$

Advances in computer technology have provided an impetus to solve such equations numerically. The method of lines (MOL) semidiscretization approach is used to transform the model partial differential equations into a system of first-order linear ordinary differential equations (ODEs). The MOL is a method of solving PDEs by discretizing the equation with respect to all but one variable (usually time). The spatial partial derivative is approximated by finite-difference approximation which produces system of ordinary differential equations (ODEs) expressible in matrix-vector form. The exact solution of the resulting system of first-order ODEs satisfies a recurrence relation. The temporal accuracy is controlled by choosing a second-order approximation to the matrix exponential function and afterwards a parallel algorithm is developed and tested on well-known problems whose exact solutions are already known in literature.

The demands of both the scientific and the commercial communities for ever-increasing computing power led to dramatic improvements in computer architecture. Initial efforts concentrated on achieving high performance on a single processor, but the more recent past

has been witness to attempts to harness multiple processors. Multiprocessor systems consist of a number of interconnected processors each of which is capable of performing complex tasks independent of the others. In a sequential algorithm all processes are performed by a single processor turn by turn but in a parallel algorithm independent parts of the program are performed by different processors simultaneously which save a lot of time.

Various sequential numerical schemes have been proposed in the literature for the solution of this problem [1–3]. Serbin [4] proposed a scheme for parallelizing certain algorithms for the inhomogeneous heat equation developed by Brenner *et al.* [5]. Serbin presented a numerical example utilizing the scheme to explore how the partial fraction approximation (PFA) compares to the usual sequential implementation for approximate solution of the semidiscrete problem. In the present paper, we develop numerical scheme based upon rational approximations to the matrix exponential function for solving inhomogeneous heat equation. The scheme is L_0-stable, second-order accurate in space and time, and uses only real arithmetic in its implementation.

The outline of the paper is in the following way: the numerical scheme for heat equation is described in Section 2; the parallel and sequential algorithms are presented in Section 3; In Section 4, the numerical results produced by this algorithm are given; nonlinear heat problem is discussed in Section 5; the conclusion is given in Section 6.

2. Derivation of the scheme

Consider the one-dimensional heat equation

$$u_t = u_{xx} + s(x, t), \quad 0 < x < X, t > 0, \tag{1}$$

subject to time-dependent boundary conditions

$$u(0, t) = f_1(t), \quad 0 < t \le T, \tag{2}$$

$$u(X, t) = f_2(t); \quad 0 < t \le T, \tag{3}$$

with initial condition

$$u(x, 0) = g(x); \quad 0 < x < X, \tag{4}$$

where $f_1(t)$, $f_2(t)$, $g(x)$ and $s(x, t)$ are known, while the function $u(x, t)$ is to be determined.

The interval $0 \le x \le X$ is divided into $N + 1$ subintervals each of width h, so that $(N + 1)h = X$ and the time variable t is discretized in the steps of length l. Thus, at each time level $t = t_n = nl$ ($n = 0, 1, 2, \ldots$), the open region $R = \{0 < x < X\} \times [t > 0]$ and its boundary ∂R consisting of lines $x = 0$ and $x = X$ and the axis $t = 0$ have been superimposed by rectangular mesh with N points within R and open point along each side of ∂R.

The solution $u(x, t)$ of (1) is sought at each point (kh, nl) in $R \times [t > 0]$, where $k = 1, 2, \ldots, N$ and $n = 0, 1, 2, \cdots$. The solution of an approximating numerical method will be denoted by $U(x, t)$. The space derivative in (1) will be replaced by the following second-order central-difference approximation given by

$$\frac{\partial^2 u(x,t)}{\partial x^2} = \frac{1}{h^2}\{u(x-h,t) - 2u(x,t) + u(x+h,t)\}$$

$$+\frac{1}{12}h^2\frac{\partial^4 u}{\partial x^4}+O(h^3),\ \text{as}\ \ h\to 0. \tag{5}$$

Applying (1) to all the interior mesh points within R at time level $t = nl$ with the space derivative replaced by (5) leads to a system of N first-order ODEs of the form

$$\frac{d\mathbf{U}(t)}{dt}=A\mathbf{U}(t)+\mathbf{v}(t),\ \ t>0 \tag{6}$$

with the initial condition

$$\mathbf{U}(0)=\mathbf{g} \tag{7}$$

in which the matrix A is of order N and given by

$$A=\frac{1}{h^2}\begin{bmatrix} -2 & 1 & & & & \mathbf{d} \\ 1 & -2 & 1 & & & \\ & \mathbf{O} & \mathbf{O} & \mathbf{O} & & \\ & & 1 & -2 & 1 & \\ \mathbf{d} & & & 2 & -2 & \end{bmatrix}. \tag{8}$$

$$\mathbf{v}(t)=[h^{-2}f_1(t)+s_1(x,t),\quad s_2(x,t),\cdots,s_{N-1}(x,t),\quad h^{-2}f_2(t)+s_N(x,t)]T$$

$$\mathbf{U}(t)=[U_1(t),\ U_2(t),\cdots,U_N(t)]^T$$

$$\mathbf{g}=[g_1(x),\quad g_2(x),\cdots,g_N(x)]T.$$

The eigenvalues of matrix A are

$$l_n=-4\sin^2\left(\frac{np}{2(N+1)}\right)<0,\ \ n=1,2,...,N \qquad (*)$$

Solving (6) subject to (7) gives

$$\mathbf{U}(t)=\exp(lA)U(0)+\int_0^t\exp((t+l-s)A)\mathbf{v}(s)ds; \tag{9}$$

which satisfies [7], the recurrence relation

$$\mathbf{U}(t+l)=\exp(lA)\mathbf{U}(t)+\int_t^{t+l}\exp((t+l-s)A)v(s)ds;\ \ t=0,\,l,\,2l,... \tag{10}$$

in which l is a constant time step in the discretization of the time variable $t \geq 0$ at the points $t_n = nl(n = 0, 1, 2, \cdots, N)$. To approximate the matrix exponential function in (10), we consider the rational approximate to $\exp(lA)$ of the form

$$\exp(lA)=\frac{I+(1-a)lA}{I-alA+(a-\frac{1}{2})(lA)^2}. \tag{11}$$

The error constant term is

$$m_2 = \frac{-1}{3} + \frac{a_1}{2}.$$

Let l_n be the eigenvalues of the matrix A given by (*). Then the amplification symbol of the numerical methods arising from (11) is

$$R(-z) = \frac{1-(1-a)z}{1+az+(a-\frac{1}{2})(z)^2} \tag{12}$$

where $z = -lRe(l) > 0$. Thus L_0-stability is granted provided $a > \frac{1}{2}$.

The integral term in (10) is approximated by a quadrature formula of the form

$$\int_t^{t+l} \exp((t+l-s)A)\mathbf{v}(s)ds \;;\; \sum_j^2 W_j \mathbf{v}(s_j)$$

where all $s_j (i = 1, 2)$ are different and weights $W_1 = W_1(lA)$ and $W_2 = W_2(lA)$ are matrices. It can easily shown be that when $\mathbf{v}(s) = [1, 1, 1, \cdots, 1]^T$,

$$W_1 + W_2 = M_1 \tag{13}$$

where $M_1 = A^{-1}(\exp(lA) - I)$ when $\mathbf{v}(s) = [s, s, s, \cdots s]^T$,

$$s_1 W_1 + s_2 W_2 = M_2 \tag{14}$$

where $M_2 = A^{-1}\{t\exp(lA) - A^{-1} + A^{-1}(\exp(lA) - I)\}$, I is the identity matrix of order N. Taking $s_1 = t$, $s_2 = t + l$ and then solving (13) and (14) simultaneously and replacing by $\exp(lA)$ gives

$$\mathbf{U}(t+1) = \exp(lA)\mathbf{U}(t) + \frac{l}{2}[S(lA)\mathbf{v}(t) + T(lA)\mathbf{v}(t+l)]; t = 0, l, 2l, ... \tag{15}$$

in which

$$S(lA) = [I - alA + (a-\tfrac{1}{2})(lA)^2]^{-1}; \tag{16}$$

$$T(lA) = [I - alA + (a-\tfrac{1}{2})(lA)^2]^{-1}[I - 2(a-\tfrac{1}{2})lA]. \tag{17}$$

3. The parallel algorithm

In order to implement (15) in parallel, the functions $\exp(lA)$, $S(lA)$ and $T(lA)$ are decomposed into their partial-fraction forms [6–8] given by

$$\exp(lA) = p_1(I - r_1 lA)^{-1} + p_2(I - r_2 lA)^{-1},$$

$$S(lA) = p_3(I - r_1 lA)^{-1} + p_4(I - r_2 lA)^{-1},$$

$$T(lA) = p_5(I - r_1 lA)^{-1} + p_6(I - r_2 lA)^{-1},$$

with

$$p_1 = \frac{1-a+r_1}{r_1-r_2}, \quad p_2 = \frac{1-a+r_2}{r_2-r_1},$$

$$p_3 = \frac{r_1}{r_1-r_2}, \quad p_4 = \frac{r_2}{r_2-r_1},$$

and

$$p_5 = \frac{1-2a+r_1}{r_1-r_2}, \quad p_6 = \frac{1-2a+r_2}{r_2-r_1},$$

where

$$r_1 = \frac{2a-1}{a+\sqrt{a^2-4a+2}}, \quad r_2 = \frac{2a-1}{a-\sqrt{(a^2-4a+2)}}.$$

The solution vector $U(t + l)$ in (15) may be obtained in parallel using two processors running concurrently as follows:

Processor 1

(1) Input l, r_1, $\mathbf{U}(0)$, A
(2) Compute p_1, p_3, p_5, $(I - r_1 lA)$
(3) Decompose $(I - r_1 lA) = L_1 U_1$
(4) Evaluate $\mathbf{v}(t)$, $\mathbf{v}(t + l)$
(5) use $\mathbf{z}_1(t) = \frac{l}{2}(p_3\mathbf{v}(t) + p_5\mathbf{v}(t + l))$
(6) Solve $L_1 U_1 \mathbf{y}_1(t) = p_1\mathbf{U}(t) + \mathbf{z}_1(t)$

Processor 2

(1) Input l, r_2, $\mathbf{U}(0)$, A
(2) Compute p_2, p_4, p_6, $(I - r_2 lA)$
(3) Decompose $I - r_2 lA = L_2 U_2$
(4) Evaluate $\mathbf{v}(t)$, $\mathbf{v}(t + l)$
(5) use $\mathbf{z}_2(t) = \frac{l}{2}(p_4\mathbf{v}(t) + p_6\mathbf{v}(t + l))$
(6) Solve $L_2 U_2 \mathbf{y}_2(t) = p_2\mathbf{U}(t) + \mathbf{z}_2(t)$

$$\mathbf{U}(t + l) = \mathbf{y}_1(t) + \mathbf{y}_2(t).$$

GOTO step (4) for next time step.

Implementing the algorithm, Processor 1 generates decomposition of $I - r_1 lA$ and Processor 2 generates decomposition of $I - r_2 lA$ simultaneously. Parallel computing is the simultaneous execution of the same task (split up and specially adapted) on multiple processors in order to obtain faster results.

3.1. *The sequential algorithm*

The numerical solutions of the inhomogeneous heat problems can also be obtained using the following sequential algorithm (SA):

1. **Input**
   A, matrix
   Real X, end point function $S(x, t)$
   Real l, time step
   Integer M, number of time steps
   Integer N, number of computational nodes
   Function $S(x, t)$, source term
   Function $g(x)$, initial condition
   Functions $f_1(t)$, $f_2(t)$ boundary conditions at $x = 0$ and $x = X$.

2. **Define a grid**
   Set $h = \frac{X}{N+1}$
   $r = \frac{1}{h^2}$

3. **Initial numerical solution**
   Set $t = 0$
   $x_0 = 0$
   $\mathbf{U}_0 = \frac{(f_1(0)+g(0))}{2}$
   for $n = 1, 2, \cdots, N$
   Set $x_n = x_{n-1} + h$
   $\mathbf{U}_n = g(x_n)$
   Set $\mathbf{U}_N = \frac{(f_2(0)+g(X))}{2}$
   $x_N = X$

4. **Begin time stepping**
   For $j = 1, 2, \cdots, M$
   DO STEPS 5–7

5. **Define tridiagonal system**
   Set $t = t + l$
   For $n = 1, 2, \cdots, N$
   Set $a_n = r$
   $b_n = -2r$
   $c_n = r$
   $d_n = \mathbf{U}_n + l.S(x, t)$
   Set $d_1 = d_1 + r^{\backprime}f_1(t)$
   $d_N = d_N + r^{\backprime}f_2(t)$

6. **Solution of a tridiagonal system**
   For $n = 2, 3, \cdots, N$
   set ratio $= \frac{a_n}{b_{n-1}}$

$b_n = b_n$ – ratio: c_{n-1}
$d_n = d_n$ – ratio: d_{n-1}
set $d_N = \frac{d_N}{b_N}$
For $n = N-1, N-2, \cdots, 1$
set $d_n = \frac{d_n - c_n d_{n+1}}{b_n}$

7. **Advance solution one time step**
   CALL TRIDI(N, a, b, c, d)
   For $n = 1, 2, \cdots, N$
   $\mathbf{U_n} = d_n$
   Set $\mathbf{U}_0 = f_1(t)$
   $U_{N+1} = f_2(t)$

8. **Output**
   For $n = 0, 1, \cdots, N+1$
   x_n, $\mathbf{U}_n$

4. Numerical tests

In order to test the behavior of L_0-stable scheme, three problems from the literature are considered. The parallel algorithm is tested on a sequential computer (Intel 933 MHz, BD815 Glly, 128MB(Kingstung), HDD 20 GB (SeaCate), OS Win2000 Professional, Developer Studio) for the solutions of the inhomogeneous heat equations. The parallel algorithm is also tested on an Alliant FX/8 for problem 3.

Problem 1. Consider the heat equation

$$u_t = u_{xx} - 2e^{x-t}, \quad 0 < x < 1, \, t > 0,$$

$$u(0, t) = e^{-t}, \quad t > 0$$

$$u(1, t) = e^{1-t}, \quad t > 0$$

$$u(x, 0) = e^{x}, \quad 0 < x < 1.$$

The analytical solution is $u(x, t) = e^{x-t}$. Using the algorithm, this problem is solved for $h = 0.125, 0.1, 0.05, 0.025$ with each $l = 0.125, 0.1, 0.05, 0.025$. The maximum absolute relative errors $j(u - U)=uj$ at time $t = 1.0$ are shown in Table I.

It is worth mentioning that the use of only real arithmetic especially in multispace dimensions can yield large saving in CPU time used.

Problem 2. Consider the heat equation

$$u_t = u_{xx} - 2, \quad 0 < x < 1, \quad t > 0,$$

$$u(0, t) = t, \quad t > 0$$

Table I
Maximum absolute relative errors at $t = 1.0$

h	0.125	0.1	0.05	0.025
$l = 0.125$	0.7412D-3	0.7212D-3	0.5671D-3	0.1232D-3
$l = 0.1$	0.5426D-3	0.4523D-3	0.23356D-3	0.1121D-3
$l = 0.05$	0.3214D-3	0.2113D-3	0.1021D-3	0.4511D-4
$l = 0.025$	0.1120D-3	0.7845D-4	0.2715D-4	0.2337D-4

$$u(1, t) = 1 + t, \quad t > 0$$

$$u(x, 0) = x^2, \quad 0 < x < 1.$$

The analytical solution is $u(x, t) = x^2 + t$. Using the algorithm, this problem is solved for $h = 0.125, 0.1, 0.05, 0.025$ with each $l = 0.125, 0.1, 0.05, 0.025$. The maximum absolute relative errors $|(u - U)/u|$ at time $t = 1{:}0$ are shown in Table II.

Problem 3. Consider the heat equation

$$u_t = u_{xx} + t^a(ax(1 - x) + 2t), \quad 0 < x < 1, \quad t > 0,$$

$$u(0, t) = 0, \; t > 0$$

$$u(1, t) = 0, \; t > 0$$

$$u(x, 0) = 0, \; 0 < x < 1.$$

This problem has analytical solution $u(x, t) = t^{a-1}x(1 - x)$. The maximum absolute errors at time $t = 1.0$, $a = 3$ and $a = 4$ are shown in Table III and IV, respectively. The results obtained using the new scheme developed in this paper are slightly more accurate than those from the scheme of Serbin [4]. It is clear that as far as efficiency is concerned, the scheme introduced in this paper is better for the model problem.

We feel that very small difference in the results of parallel and sequential algorithms would appear all the time (i.e. higher-order algorithm). In a series of numerical experiments, the interval $0 \le x \le 1$ is divided into $N + 1$ subintervals each of width h, so that $(N + 1)h = 1$, with $N = 40$, 80, 160 and 320. These values of N gave the corresponding order of matrix A and of all vectors in the parallel algorithms. The problems are investigated to time $t = 1$ using time steps each of length l, so that $Tl = 1$, with (i) $T = 50$ and (ii) $T = 200$. CPU time (in second) of the sequential and parallel algorithms of our scheme is calculated using an Alliant FX/8 (Table V). All cases are implemented on the Alliant FX/8 in parallel and sequentially. CPU time (in second) is given for each numerical experiment.

Table II
Maximum absolute relative errors at $t = 1.0$

h	0.125	0.1	0.05	0.025
$l = 0.125$	0.563D-3	0.4561D-3	0.2344D-3	0.1932D-3
$l = 0.1$	0.4710D-3	0.2365D-3	0.2112D-3	0.1103D-3
$l = 0.05$	0.3261D-3	0.2313D-3	0.1121D-3	0.7612D-4
$l = 0.025$	0.2311D-3	0.6755D-4	0.1433D-4	0.1341D-4

Table III
Maximum errors at $t = 1.0$; $a = 3$

$\frac{1}{h}$	$\frac{1}{l}$	PFA [4]	Parallel algorithm	SA[4]	Sequential algorithm
10	10	0.44770D-04	0.14120D-04	0.44770D-04	0.14120D-04
10	20	0.55086D-05	0.15421D-05	0.55086D-05	0.15421D-05
10	40	0.53259D-06	0.13221D-06	0.53259D-06	0.13220D-06
10	80	0.42228D-07	0.10423D-07	0.4229D-07	0.10424D-07
10	160	0.29254D-08	0.58115D-09	0.29244D-08	0.58109D-09
20	20	0.55182D-05	0.14633D-05	0.55182D-05	0.14630D-05
40	40	0.53516D-06	0.13222D-06	0.53516D-06	0.13221D-06
80	80	0.42488D-07	0.10671D-07	0.42500D-07	0.10670D-07
160	160	0.29400D-08	0.27861D-08	0.29308D-08	0.27845D-08
320	320	0.20733D-09	0.18543D-09	0.18953D-09	0.18200D-09

Table IV
Maximum errors at $t = 1.0$; $a = 4$

$\frac{1}{h}$	$\frac{1}{l}$	PFA [4]	Parallel algorithm	SA[4]	Sequential algorithm
20	20	0.19818D-04	0.61221D-05	0.19818D-04	0.61221D-05
40	40	0.19203D-05	0.45317D-06	0.19203D-05	0.45316D-06
80	80	0.15201D-06	0.24543D-07	0.15200D-06	0.24540D-07
160	160	0.10452D-07	0.77165D-09	0.10460D-07	0.77155D-09
320	320	0.166820D-09	0.96876D-10	0.66866D-09	0.96865D-10

Table V
CPU time in seconds

	Parallel algorithm		Sequential algorithm	
N	T = 50	T = 200	T = 50	T = 200
40	0.04	0.15	0.12	0.45
80	0.08	0.32	0.23	0.93
160	0.16	0.64	0.47	1.90
320	0.32	1.28	0.98	3.94

5. Application to a nonlinear problem

Consider the nonlinear heat equation

$$u_t = uu_x + u_{xx} + s(x, t); \quad 0 < x < X, \quad t > 0, \tag{18}$$

subject to time-dependent boundary conditions

$$u(0, t) = f_1(t); \quad 0 < t \leq T, \tag{19}$$

$$u(X, t) = f_2(t); \quad 0 < t \leq T, \tag{20}$$

with initial condition

$$u(x, 0) = g(x); \quad 0 < x < X, \tag{21}$$

where $g(x)$ is given continuous function of x. There will exist discontinuities between the initial and boundary conditions if $g(0) \neq f_1(0)$ or/and $g(X) \neq f_2(0)$. Equation (18) may be written as

$$\frac{\partial u}{\partial t} = u\frac{\partial u}{\partial x} + \frac{\partial^2 u}{\partial x^2} + s(x,t); \quad 0 < x < X, \ t > 0. \tag{22}$$

Applying the same discretization and the following finite-difference approximations

$$\frac{\partial^2 u}{\partial x^2} \; ; \; \frac{1}{h^2}\{u(x-h,t) - 2u(x,t) + u(x+h,t)\} \tag{23}$$

$$\frac{\partial u}{\partial x} \; ; \; \frac{1}{2h}\{u(x+h,t) - u(x-h,t)\} \tag{24}$$

to eqn (18) leads to a system of N first-order ODEs of the form

$$\frac{d\mathbf{U}(t)}{dt} = A\mathbf{U}(t) + \mathbf{v}(t), \ \ t > 0 \tag{25}$$

with the initial condition

$$\mathbf{U}(0) = \mathbf{g} \tag{26}$$

in which the matrix A is of order N and given by

$$A = \frac{1}{2h^2}\begin{bmatrix} -a_2 & a_2 & & & & \mathbf{d} \\ a_1 & -a_2 & a_3 & & & \\ & \mathbf{O} & \mathbf{O} & \mathbf{O} & & \\ & & a_1 & -a_2 & a_3 & \\ \mathbf{d} & & & a_1 & -a_2 & \end{bmatrix} \tag{27}$$

where

$$a_1 = 2 - \mathbf{U}(t)h$$

$$a_2 = 4$$

$$a_3 = 2 + \mathbf{U}(t)h$$

$$\mathbf{v}(t) = [(2h)^{-2}(2 - U_1(t)h)f_1(t) + s_1(x,t),\, s_2(x,t), \cdots, s_{N-1}(x,t),$$

$$(2h)^{-2}(2 + U_N(t)h)\, f_2(t) + s_N(x,t)]^T.$$

The algorithm for the nonlinear problem is implemented on an architecture consisting of at least two processors.

Problem 4. Consider the nonlinear heat equation

$$u_t = uu_x + u_{xx},\ 0 < x < 1,\ t > 0,$$

Table VI
Maximum absolute relative errors at $t = 1.0$

h	0.125	0.1	0.05	0.025
$l = 0.125$	0.456D-4	0.357D-4	0.352D-4	0.346D-4
$l = 0.1$	0.441D-4	0.333D-4	0.221D-4	0.113D-4
$l = 0.05$	0.341D-4	0.342D-4	0.221D-4	0.112D-4
$l = 0.025$	0.212D-4	0.112D-4	0.563D-5	0.456D-5

$$u(0, t) = 0, \quad t > 0$$

$$u(1, t) = 0, \; t > 0$$

$$u(x, 0) = \frac{2\pi \sin(\pi x)}{2 + \cos(\pi x)}; \quad 0 < x < 1.$$

The analytical solution is

$$u(x, t) = \frac{2\pi e^{-\pi^2 t} \sin(\pi x)}{2 + e^{-\pi^2 t} \cos(\pi x)}.$$

Using parallel algorithm, this problem is solved for $h = 0.125, 0.1, 0.05, 0.025$ with each $l = 0.125, 0.1, 0.05, 0.025$. The maximum absolute relative errors $|(u - U)/u|$ at time $t = 1.0$ are shown in Table VI. The scheme developed for linear problems carries over to nonlinear problem, although added difficulties arise in computing the solution.

6. Conclusion

An $O(h^2 + l^2)$ L_0-stable parallel algorithm has been developed for the heat equation with a known source term. The algorithm was found to be more accurate in comparison with existing algorithms from the literature, and can be implemented in parallel using a machine with two processors.

Acknowledgements

The author is highly grateful to anonymous referees for their valuable comments and suggestions for improving the paper. The author was supported by Punjab University College of Information Technology (PUCIT).

References

1. M. K. Jain, *Numerical solution of differential equations*, Wiley Eastern (1991).
2. Z. David and D. Paul, *Applied partial differential equations*, Dover (2002).
3. M. K. Jain, R. K. Jain and R. K. Mohanty, A fourth-order difference method for the one-dimentional general quasilinear parabolic partial equations, *Numerical Methods PDEs*, **6**, 311–319 (1990).
4. S. M. Serbin, A scheme for parallelizing certain algorithms for the linear inhomogeneous heat equation, *SIAM J. Sci. Stat. Comput.*, **13**, 449–458 (1992).

5. P. Brenner, M., Crouzeix and V. Thomee, Single step methods for the inhomogeneous linear differential equations in Banach space, *RAJRO Anal. Number*, **16**, 70–79 (1982).
6. E. H. Twizell, A. B. Gumel and M. A. Arigu, *Second-order*, L_0-stable methods for the heat equation with time-dependent boundary conditions, *Adv. Comp. Math.*, **6**, 333–352 (1996).
7. J. D. Lambert, *Numerical methods for ordinary differential systems: The initial-value problem*, Wiley (1991).
8. A. R. Gourlay and J. Morris, The extrapolation of first order methods for parabolic partial differential equations. II, *SIAM J. Numer. Anal.*, **17**, 641–655 (1980).