A Parallel Algorithm for the Inhomogeneous Advection Equation

M. Akram
PUCIT, University of the Punjab, Old Campus
Lahore-54000, Pakistan
m.akram@pucit.edu.pk, makrammath@yahoo.com

T. A. Cheema
Department of Mathematics
L. C. (W.) University, Lahore, Pakistan
dr_tacheema@yahoo.com

M. S. A. Taj
Department of Mathematics
COMSATS Institute of Information Technology
Defence Road, Off Raiwind Road, Lahore, Pakistan
shahadat259@yahoo.com

Abstract
In this paper a parallel algorithm is presented for the numerical solution of the advection equation \( u_t(x,t) + \alpha u_x(x,t) = s(x,t), \alpha > 0, \) \( x > 0, t > 0, \) subject to the boundary condition \( u(0,t) = f(t), t > 0, \) and the initial condition \( u(x,0) = g(x), x > 0. \)

Mathematics Subject Classification: 35L05; 65M06, 65M20, 65Y05

Keywords: Advection equation; Method of Lines; Recurrence Relation; Rational approximation; Parallel computation

1 Introduction
The advancements in computer technology have provided an impetus to solve partial differential equations (PDEs) numerically. Since it is essential for mod-

The research work of the first author is supported by PUCIT.
ern engineering analysis to have efficient computational schemes to solve complicated mathematical models of physical processes, the authors propose to study numerical techniques for approximating the solutions of various mathematical models related to first-order hyperbolic partial differential equation (advection equation or first-order wave equation).

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 independently 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.

Several authors like Arigu, Cheema, Gumel, Khaliq, and Twizell developed algorithms for obtaining numerical solutions of homogeneous advection equations in [4, 5, 9, 10]. In this paper we present a parallel algorithm for the numerical solution of non-homogeneous advection with constant coefficients. The outline of the paper is in the following way: the numerical scheme for advection equation is described in Section 3 and the parallel algorithm is presented in Section 4. In Section 5, the numerical results produced by this algorithm are given and the conclusion is given in Section 6.

2 Method of Lines and Recurrence Relation

This paper focuses on the applications of the method of lines (MOL) for solving the following first-order hyperbolic partial differential equation (advection equation or first-order wave equation)

\[ \frac{\partial u(x, t)}{\partial t} + \alpha \frac{\partial u(x, t)}{\partial x} = s(x, t), \quad \alpha > 0, x > 0, t > 0 \]  

(1)

with the boundary condition

\[ u(0, t) = f(t), \quad t > 0 \]  

(2)

and the initial condition

\[ u(x, 0) = g(x), \quad x > 0 \]  

(3)

where \( f(t), g(x) \) and \( s(x, t) \) are known, while the function \( u(x, t) \) is to be determined in some arbitrary region \( R = [0 < x \leq X] \times [t > 0] \).
Parallel algorithm for the inhomogeneous advection equation

The MOL is a method of solving PDEs by discretizing the equation with respect to all but one variable (usually time), resulting in a system of ODEs which are easily solved. Because the partial differential equation is not discrete with respect to all variables, the MOL is sometimes known as a semi-discretization method. The discretization can be done in variety of ways, but this paper is concerned with discretization using finite differences.

Using the MOL semi-discretization approach the model partial differential equation will be transformed into a system of first-order linear ordinary differential equations (ODEs), the solution of which satisfies the recurrence relation involving matrix exponential terms. The development of numerical methods is based on rational approximation to such exponentials.

The space derivative in (1) may be replaced by third-order finite difference approximation

$$\frac{\partial u(x, t)}{\partial x} \approx \frac{1}{6h} \left\{ -2u(x - 3h, t) + 9u(x - 2h, t) - 18u(x - h, t) \ight. \\
+ 11u(x, t) \right\} + \frac{h^3}{4} \frac{\partial^4 u(x, t)}{\partial x^4} + O(h^4)$$

(4)

for the mesh points \((x, t) = (x_m, t_n)\) with \(m = 3, 4, \ldots, N\).

Note that implementation of (4) requires some additional values for \(x = x_1, x_2\). Hence at these points the following third-order approximations will be used:

$$\frac{\partial u(x, t)}{\partial x} \approx \frac{1}{6h} \left\{ -3u(x - h, t) + u(x, t) + 0u(x + h, t) + 3u(x + 2h, t) \right\} \\
- u(x + 3h, t) + \frac{h^3}{4} \frac{\partial^4 u(x, t)}{\partial x^4} + O(h^4)$$

(5)

and

$$\frac{\partial u(x, t)}{\partial x} \approx \frac{1}{6h} \left\{ -u(x - 2h, t) + 2u(x - h, t) - 9u(x, t) + 10u(x + h, t) \right\} \\
- 2u(x + 2h, t) \right\} + \frac{h^3}{4} \frac{\partial^4 u(x, t)}{\partial x^4} + O(h^4).$$

(6)

Applying (1) with (4-6) to the \(N\) mesh points of the grid at time \(t\), produces a system of first-order ODE’s that can be written in matrix-vector form as

$$\frac{dU(t)}{dt} = -\alpha AU(t) + v(t), \quad t > 0$$

(7)

with initial distribution

$$U(0) = g$$

(8)
in which the matrix $A$ is of order $N$ and given by

$$A = (6h)^{-1} \begin{bmatrix} 1 & 0 & 3 & -1 & \circ \\ 2 & -9 & 10 & -2 \\ 9 & -18 & 11 \\ -2 & 9 & -18 & 11 \\ \cdots & \cdots & \cdots & \cdots \\ \circ & -2 & 9 & -18 & 11 \end{bmatrix} \tag{9}$$

$$v(t) = \left[ -\frac{3}{6h} f(t) + s_1(t), -\frac{1}{6h} f(t) + s_2(t), -\frac{2}{6h} f(t) + s_3(t), s_4(t), \cdots, s_N(t) \right]^T,$$

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

$$g = [g_1(x), g_2(x), \cdots, g_N(x)]^T.$$

Solution of (7) subject to (8) gives

$$U(t) = \exp(-\alpha t A) U(0) + \int_0^t \exp[-\alpha(s - t) A] v(s) ds. \tag{10}$$

which satisfies the recurrence relation

$$U(t + l) = \exp(-\alpha l A) U(t) + \int_t^{t+l} \exp[-\alpha A(t + l - s)] v(s) ds; t = 0, l, 2l, \cdots. \tag{11}$$

in which $l$ is a constant time step in the discretization of the time variable $t \geq 0$ at the points $t_n = nl(n = 0, 1, 2, \cdots, )$. To approximate the matrix exponential function in (11). We consider the rational approximation of the form

$$\exp(-\theta) = \frac{1 - b_1 \theta + b_2 \theta^2}{1 + a_1 \theta + a_2 \theta^2 + a_3 \theta^3} \tag{12}$$

in which $b_1 = 1 - a_1$, $b_2 = \frac{1}{2} - a_1 + a_2$, $a_3 = \frac{1}{6} - \frac{a_1}{2} + a_2$. Thus

$$\exp(-\alpha l A) = G^{-1} \left( I - (1 - a_1)\alpha l A + \left( \frac{1}{2} - a_1 + a_2 \right) \alpha^2 l^2 A^2 \right). \tag{13}$$
where
\[ G = I + a_1 \alpha l A + a_2 \alpha^2 l^2 A^2 + \left( \frac{1}{6} - \frac{a_1}{2} + a_2 \right) \alpha^3 l^3 A^3. \] (14)

The denominator of \( \exp(-\theta) \) has distinct real zeros by choice of the values \( a_1 = 1.308617, a_2 = 0.570502, a_3 = 0.082856 \) and \( L \)-stability is introduced in [13].

The integral term in (11) is approximated by a quadrature formula of the form
\[
\int_t^{t+l} \exp(-\alpha(t + l - s)A) \mathbf{v}(s) ds \approx W_1 \mathbf{v}(s_1) + W_2 \mathbf{v}(s_2) + W_3 \mathbf{v}(s_3) \] (15)

where all \( s_i (i = 1, 2, 3) \) are different and \( W_1, W_2, W_3 \) are matrices. Putting \( \mathbf{v}(s) = [1, 1, 1, ..., 1]^T, \mathbf{v}(s) = [s, s, ..., s]^T \) and \( \mathbf{v}(s) = [s^2, s^2, ..., s^2]^T \), successively, in (15) gives
\[
W_1 + W_2 + W_3 = M_1 
\]
\[
s_1 W_1 + s_2 W_2 + s_3 W_3 = M_2 
\]
\[
s_1^2 W_1 + s_2^2 W_2 + s_3^2 W_3 = M_3 
\]

where
\[
M_1 = -(\alpha A)^{-1}(\exp(-\alpha l A) - I) 
\]
\[
M_2 = -(\alpha A)^{-1} \left\{ t \exp(-\alpha l A) - (t + l) I - (\alpha A)^{-1}(\exp(-\alpha l A) - I) \right\} 
\]
and
\[
M_3 = -(\alpha A)^{-1} \left\{ t^2 \exp(-\alpha l A) - (t + l)^2 I - 2(\alpha A)^{-1}t \exp(l A) - (t + l) I - (\alpha A)^{-1}(\exp(-\alpha l A) - I) \right\}. \] (21)

Taking \( s_1 = t, s_2 = t + \frac{l}{2}, s_3 = t + l \) and then solving (16), (17), (18) simultaneously and replacing \( \exp(-\alpha l A) \) by (13) gives
\[
W_1 = \frac{l}{6} \{(I - (4 - 9a_1 + 12a_2)\alpha l A) G^{-1}, \] (22)
\[
W_2 = \frac{2l}{3} \{(I + (1 - 3a_1 + 6a_2)\alpha l A) G^{-1}, \] (23)
\[
W_3 = \frac{l}{6} \{(I - (3 - 9a_1 + 12a_2)\alpha l A + (1 - 3a_1 + 6a_2)\alpha^2 l^2 A^2 \} G^{-1}. \] (24)

Hence (11) can be written as
\[
U(t + l) = \exp(-\alpha l A)U(t) + W_1 \mathbf{v}(t) + W_2 \mathbf{v}(t + \frac{l}{2}) + W_3 \mathbf{v}(t + l). \] (25)

in which \( W_1, W_2 \) and \( W_3 \) are given by (22)–(24) respectively.
3 The Parallel Algorithm

We focused on the construction of a rational approximation with real and distinct poles. The resulting algorithm readily admits parallelization through partial fraction expansion [8]. We present the parallel algorithm using three different processors for implementing (25) which was discussed in [13] in the following form:

Processor 1

1. Input: \( l, r_1, U(0), A \)
2. Compute: \( p_1, p_4, p_7, p_{10} \) and \( I - \frac{al}{r_1} A \)
3. Decompose: \( I - \frac{al}{r_1} A = L_1U_1 \)
4. Evaluate: \( v(t), v(t + \frac{l}{2}), v(t + l) \)
5. Use: \( z_1(t) = \frac{l}{6} (p_4 v(t) + 4p_7 v(t + \frac{l}{2}) + p_{10} v(t + l)) \)
6. Solve: \( L_1U_1 y_1(t) = p_1 U(t) + z_1(t) \)

Processor 2

1. Input: \( l, r_2, U(0), A \)
2. Compute: \( p_2, p_5, p_8, p_{11} \) and \( I - \frac{al}{r_2} A \)
3. Decompose: \( I - \frac{al}{r_2} A = L_2U_2 \)
4. Evaluate: \( v(t), v(t + \frac{l}{2}), v(t + l) \)
5. Use: \( z_2(t) = \frac{l}{6} (p_5 v(t) + 4p_8 v(t + \frac{l}{2}) + p_{11} v(t + l)) \)
6. Solve: \( L_2U_2 y_2(t) = p_2 U(t) + z_2(t) \)

Processor 3

1. Input: \( l, r_3, U(0), A \)
2. Compute: \( p_3, p_6, p_9, p_{12} \) , \( I - \frac{al}{r_3} A \)
3. Decompose: \( I - \frac{al}{r_3} A = L_3U_3 \)
4. Evaluate: \( v(t), v(t + \frac{l}{2}), v(t + l) \)
5. Use: \( z_3(t) = \frac{l}{6} (p_6 v(t) + 4p_9 v(t + \frac{l}{2}) + p_{12} v(t + l)) \)
6. Solve: \( L_3U_3 y_3(t) = p_3 U(t) + z_3(t) \).
Parallel algorithm for the inhomogeneous advection equation

Hence \( U(t + l) = y_1(t) + y_2(t) + y_3(t) \). In implementing the algorithm, Processor 1 generates once only decomposition of \( I - \frac{\alpha l}{r_1} A \), Processor 2 generates decomposition of \( I - \frac{\alpha l}{r_2} A \) and Processor 3 generates decomposition of \( I - \frac{\alpha l}{r_3} A \) once only.

The parallel implementation of an algorithm involves the division of total workload into a number of smaller tasks which can be assigned to different processors and executed concurrently. This allows us to solve a large problem more quickly. The most important part in parallelization is to map out a problem on a multiprocessor environment. The choice of an approach to the problem decomposition depends upon the computational scheme.

4 Numerical Validations

In order to test the behavior of \( L_0 \)-stable scheme, two problems from the literature are considered. The algorithm is tested on a serial computer (Intel. 933 MHz, BD815 Gly, 128MB(Kingstung), HDD 20 GB (SeaCate), OS Win2000 Professional, Developer Studio) for the solutions of the advection equations.

**Example 1.** Consider the advection equation with

\[
\begin{align*}
    u_t(x, t) &= u_x(x, t) - x^2 t, & x > 0, & t > 0, \\
    u(x, 0) &= 2 + \sin(x), & 0 < x \leq 1, \\
    u(t, 0) &= 2 - \sin(t) - \frac{1}{12} t^4, & 0 < t \leq 1.
\end{align*}
\]

This problem has an analytical solution

\[
    u(x, t) = 2 + \sin(x - t) + \frac{1}{12} x^4 - \frac{1}{3} x^3 t - \frac{1}{12} (x - t)^4 \] [6].

This problem is solved for \( x = 0.2, 0.4, 0.6, 0.8, 1.0 \) with \( h=0.001 \) and \( l=0.05 \). The maximum relative errors are given in Table 1 at \( t = 10 \). Using once again the algorithm, this problem is solved for \( x = 0.2, 0.4, 0.6, 0.8, 1.0 \) with \( h, l = 0.05, 0.0125, 0.005 \) at time \( t = 2, 4, 10, 20 \). The maximum relative errors are given in Table 2.

| Table 1: Maximum relative errors for problem 1 at \( t = 10 \). |
|---|---|---|---|---|---|
| \( x \rightarrow \) | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
| New scheme | 0.2051D-6 | 0.5233D-6 | 0.7405D-6 | 0.4512D-5 | 0.1211D-5 |
Table 2: Maximum relative errors for problem 1 at $t=2, 4, 10, 20$.

<table>
<thead>
<tr>
<th>$h, l \rightarrow$</th>
<th>0.05</th>
<th>0.025</th>
<th>0.005</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t=2.0$</td>
<td>0.331D-02</td>
<td>0.512D-03</td>
<td>0.321D-04</td>
</tr>
<tr>
<td>$t=4.0$</td>
<td>0.213D-03</td>
<td>0.322D-04</td>
<td>0.431D-05</td>
</tr>
<tr>
<td>$t=10.0$</td>
<td>0.571D-06</td>
<td>0.447D-07</td>
<td>0.111D-08</td>
</tr>
<tr>
<td>$t=20.0$</td>
<td>0.497D-07</td>
<td>0.567D-08</td>
<td>0.413D-10</td>
</tr>
</tbody>
</table>

Example 2. Consider the advection equation with

$$u_t(x, t) = u_x(x, t) + (2x - x^2)e^{-t}, \quad x > 0, \quad t > 0,$$
$$u(x, 0) = x^2, \quad 0 < x \leq 1,$$
$$u(0, t) = 0, \quad 0 < t \leq 1.$$

This problem has an analytical solution $u(x, t) = x^2e^{-t}$ [6]. This problem is solved for $x = 0.2, 0.4, 0.6, 0.8, 1.0$ with $h=0.001$ and $l=0.005$. The maximum relative errors are given in Table 3 at $t = 10$.

Table 3: Maximum relative errors for problem 2 at $t = 10$.

<table>
<thead>
<tr>
<th>$x \rightarrow$</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>New scheme</td>
<td>0.1172D-6</td>
<td>0.1634D-6</td>
<td>0.5678D-6</td>
<td>0.6578D-5</td>
<td>0.7431D-5</td>
</tr>
</tbody>
</table>

5 Conclusion

In this paper a parallel algorithm has been applied to the inhomogeneous advection equations. The algorithm which may be implemented on a parallel architecture using three processors requires the application of seven diagonal. This scheme is developed for the inhomogeneous advection equation. The first-order spatial derivative is discretized to result in an approximating system of ODEs. The exact solution of this system of first order ODEs satisfies a recurrence relation involving the matrix exponential function. This function is approximated by a rational function possessing real and distinct poles which consequently readily admits a partial fraction expansion thereby allowing the distribution of the work in solving the corresponding linear algebraic systems.
on concurrent processors. The method developed does not require the use of complex arithmetic and need only real arithmetic in its implementation. This technique worked very well for the advection equations.

**Acknowledgement:** The authors wish to express their sincere thanks to Professor E. H. Twizell for his moral support.

**References**


Received: August 3, 2007