A Differential Quadrature Algorithm for the Numerical Solution of the
Second-Order One Dimensional Hyperbolic Telegraph Equation

Ram Jiwari1∗, Sapna Pandit2, R C Mittal3
1School of Mathematics and Computer Applications, Thapar University, Patiala
2 Department of Mathematics, MNNIT Allahabad
3Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee
(Received 20 June 2011, accepted 16 January 2012)

Abstract: In this article, we proposed a numerical technique based on polynomial differential quadrature
method (PDQM) to find the numerical solutions of one dimensional hyperbolic telegraph equation. The hyper-
bolic partial differential equations model the vibrations of structures (e.g., buildings, beams, and machines)
and they are the basis for fundamental equations of atomic physics. The PDQM reduced the problem into a
system of second order linear differential equation. Then, the obtained system is changed into coupled differ-
etial equations and lastly, RK4 method is used to solve the coupled system. The accuracy of the proposed
method is demonstrated by three test examples. The numerical results are found to be in good agreement with
the exact solutions. The whole computation work is done with help of software DEV C++ and MATLAB.

Keywords: telegraph equation; differential quadrature method; uniform grid points; Gauss-Lobatto-Chebyshev
grid points; Runge-Kutta method

1 Introduction

In this article, we are dealing with the numerical approximation of the second-order one-dimensional linear hyperbolic
equation

\[ \frac{\partial^2 u}{\partial t^2} (x, t) + 2\alpha \frac{\partial u}{\partial t} (x, t) + \beta^2 u (x, t) = \frac{\partial^2 u}{\partial x^2} (x, t) + g (x, t), \quad (x, t) \in [a, b] \times [0, T], \]  

with initial conditions

\[ u (x, 0) = f_1 (x), \]  

\[ \frac{\partial u}{\partial t} (x, 0) = f_2 (x), \]  

and boundary conditions

\[ u (a, t) = \phi_1 (t), \quad t \geq 0 \]  

\[ u (b, t) = \phi_2 (t), \quad t \geq 0 \]  

where \( \alpha > \beta \geq 0 \) are known constants coefficients, \( g, f_1, f_2, \phi_1, \phi_2 \) are known functions and the function \( u \) is unknown.

Recently, it is found that telegraph equation is more suitable than ordinary diffusion equation in modeling reaction
diffusion for such branches of sciences. The hyperbolic partial differential equations model the vibrations of structures
(e.g., buildings, beams, and machines) and they are the basis for fundamental equations of atomic physics. The telegraph
equation is important for modeling several relevant problems such as signal analysis [1], wave propagation [2], random
walk theory [3], etc. The Eq. (1.1) also referred as models mixture between diffusion and wave propagation by introducing
a term that accounts for effects of finite velocity to standard heat or mass transport equation [4]. However, Eq. (1.1) is
commonly used in signal analysis for transmission and propagation of electrical signals [5] and also has applications in
other fields [6]. So, the Eq. (1.1) has great importance in science and engineering.

In recent years, much attention has been given in the literature to the development of numerical schemes of the second
order hyperbolic equations, see, for example [7-9]. The methods proposed in [7-9] are conditionally stable. After that,
Mohanty [10-11] proposed unconditionally stable schemes for the linear one dimensional hyperbolic equation (1.1). In [10], proposed a new technique to solve the linear one dimensional hyperbolic equation (1.1) which is unconditionally stable and is of second order accurate in both time and space components and in [11] the author developed a three level implicit unconditionally stable difference scheme for the equation (1.1) with variable coefficients.

Recently, the one-dimensional linear hyperbolic equation (1.1) has been considered by Lakestani and Saray [12] and Dehghan and his associates [13-16]. In [12], the authors used interpolating scaling functions to find the numerical solutions of Eq. (1.1) while in [13] the authors proposed a numerical scheme to solve the one-dimensional hyperbolic telegraph equation using collocation points and approximating the solution using thin plate splines radial basis function. Mohammedi and Dehghan [14] combined a high-order compact finite difference scheme to approximate the spatial derivative and the collocation technique for time component to find the numerical solutions of one-dimensional linear hyperbolic equation. In [15], the authors used Chebyshev cardinal functions while in [16] proposed Chebyshev Tau method for numerical solutions of hyperbolic telegraph equation (1.1).

Differential quadrature method (DQM) has been successfully applied to solve various linear and nonlinear one and two dimensional partial differential equations. Korkmaz and Dag [18] solved Schrödinger equation by DQM. In [19-20], Mittal and Jiwari solved two dimensional Burgers’ and Brusselator equations by DQM.

In the article, we proposed a numerical scheme based on polynomial differential quadrature method to find the numerical solutions of the telegraph equation (1.1) with initial conditions and Dirichlet boundary conditions. The PDQM reduced the problem into a system of second order linear differential equation. Then, the obtained system is changed into coupled differential equations and lastly, RK4 method is used to solve the coupled system. The accuracy and efficiency of the proposed method is demonstrated by three test examples. Finally, a comparison a made between the numerical solutions obtained by Chebyshev-Gauss-Lobatto grid points and the uniform grid points and finds that the Chebyshev-Gauss-Lobatto grid points give better accuracy and stable numerical solutions as comparison to the uniform grid points which agree with the theory of Shu given in his book [17].

\section{Polynomial differential quadrature method}

Differential quadrature method is a numerical technique for solving differential equations. By this method, we approximate the spatial derivatives of unknown function at any grid points using weighted sum of all the functional values at certain points in the whole computational domain. In this paper, we used one of the forms of differential quadrature method namely polynomial differential quadrature method (PDQM) to approximate the solution of the problem. Since the weighting coefficients are dependent only the spatial grid spacing, we assume that the grid points \( x_1 < x_2 < \ldots < x_N \) on the real axis. The grid points may be uniform and non-uniform. The differential quadrature discretization of the first and the second derivatives at a point \( x_i \) is given by the following equations

\[
\begin{align*}
    u_x (x_i, t) &= \sum_{j=1}^{N} a_{ij} u(x_j, t), \\
    u_{xx} (x_i, t) &= \sum_{j=1}^{N} b_{ij} u(x_j, t),
\end{align*}
\]

where \( a_{ij} \) and \( b_{ij} \) represent the weighting coefficients \([17] , i = 1, 2, \ldots, N \). The following base functions are used to obtain weighting coefficients

\[
g_k(x) = \frac{M(x)}{(x - x_k) M^{(1)}(x_k)}, \quad k = 1, 2, \ldots, N
\]

where

\[
M(x) = (x - x_1)(x - x_2)\ldots(x - x_N).
\]

\[
M^{(1)}(x_i) = \prod_{k=1,k\neq i}^{N} (x_i - x_k)
\]

using the set of base functions given in equation (2.2), the weighting coefficients of the first order derivative are found as \([17] \)

\[
a_{ij} = \frac{M^{(1)}(x_i)}{(x_i - x_j) M^{(1)}(x_j)} , \quad k = 1, 2, \ldots, N, \quad i \neq j
\]

\[
a_{ii} = - \sum_{j=1,j\neq i}^{N} a_{ij}, \quad i = 1, 2, \ldots, N
\]
and for weighting coefficients of the second order derivative, the formula is [17]

\[ b_{ij} = 2\alpha_{ij} \left( \alpha_{ii} - \frac{1}{x_i - x_j} \right), \quad i, j = 1, 2, ..., N, \quad i \neq j \]  

(2.7)

\[ b_{ii} = -\sum_{j=1, j \neq i}^{N} b_{ij}, \quad i = 1, 2, ..., N \]  

(2.8)

Similarly, Shu [17] proposed the weighting coefficients of the higher order derivative in the explicit form

\[ w^{(r)}_{ij} = r\left[ \alpha_{ij} w^{(r-1)}_{ii} - \frac{w^{(r-1)}_{ii}}{x_i - x_j} \right] \text{ for } i \neq j, \quad i, j = 1, 2, ..., N; \quad r = 2, 3, ..., N - 1 \]  

(2.9)

\[ w^{(r)}_{ii} = -\sum_{j=1, j \neq i}^{N} w^{(r)}_{ij}, \quad i = j \]  

(2.10)

where \( \alpha_{ij} \) and \( w^{(r)}_{ij} \) are the weighting coefficients of the first order derivative and \( r \)th order derivative respectively.

### 3 Numerical scheme for telegraph equation

The space derivatives in the one-dimensional linear hyperbolic equation (1.1) are approximated by the polynomial differential quadrature method. The Eq. (1.1) changed into the following form

\[ d^2 u \over dt^2 (x_i, t) + 2\alpha \frac{du}{dt} (x_i, t) + \beta^2 u (x_i, t) = \sum_{j=1}^{N} b_{ij} u (x_j, t) + g (x_i, t), \quad (x_i, t) \in [a, b] \times [0, T], \quad i = 1, 2, ..., N \]  

(3.1a)

with initial conditions

\[ u (x_i, 0) = f_1 (x_i), \]  

(3.1b)

\[ \frac{du}{dt} (x_i, 0) = f_2 (x_i), \]  

(3.1c)

and boundary conditions

\[ u (a, t) = \phi_1 (t), \quad t \geq 0 \]  

(3.1d)

\[ u (b, t) = \phi_2 (t), \quad t \geq 0 \]  

(3.1e)

The system (3.1) is system of second order differential equations. Now let

\[ \frac{du}{dt} (x_i, t) = z (x_i, t), \quad \text{then} \quad \frac{d^2 u}{dt^2} (x_i, t) = \frac{dz}{dt} (x_i, t), \]  

(3.2)

Using the assumptions of Eq. (3.2) into the Eq. (3.1), the Eq. (3.1) changed as follow

\[ \frac{dz}{dt} (x_i, t) = z (x_i, t), \]  

(3.3a)

\[ \frac{dz}{dt} (x_i, t) + 2\alpha z (x_i, t) + \beta^2 u (x_i, t) = \sum_{j=1}^{N} b_{ij} u (x_j, t) + g (x_i, t), \quad (x_i, t) \in [0, 1] \times [0, 1], \quad i = 1, 2, ..., N, \]  

(3.3b)

with initial conditions

\[ u (x_i, 0) = f_1 (x_i), \]  

(3.3c)

\[ z (x_i, 0) = f_2 (x_i), \]  

(3.3d)

and the boundary conditions (3.1d) and (3.1e).

The system (3.3) is coupled system of first order differential equations with initial and boundary conditions. The boundary conditions are handled by the PDQM and the coupled system of first order differential equations with initial conditions is solved by slandered RK4 method.
4 Selection of grid points and stability

In the numerical experiments, we have used the two types of grid points to find the numerical solutions of the Eq. (1.1)

(i) Uniform grid points
\[ x_i = x_1 + i h, \quad i = 2, 3, ..., N \]  

where \( x_1 = a \) and \( h = \frac{b-a}{N} \).

(ii) Chebyshev-Gauss-Lobatto grid points
\[ x_i = a + \frac{1}{2} \left( 1 - \cos \left( \frac{(i-1) \pi}{N-1} \right) \right) (b-a), \quad i = 1, 2, ..., N \]  

The stability of the DQM depends on the eigen-values of differential quadrature discretization matrices. These eigen-
values in turn very much depend on the distribution of grid points. It has been shown by Shu [17] in his book that the
uniform grid points distribution does not give stable solution which we have also notice in our numerical experiments.
The Tables 1-3 show that Chebyshev-Gauss-Lobatto grid points give better accuracy as comparison to uniform grid points. 
The uniform grid points do not work for time greater than \( t = 1.5 \) while nonuniform grid points work.

5 Numerical experiments

In this section, to illustrate the efficiency and accuracy of the numerical method, a set of numerical experiments have
been carried out for solving the telegraph equation. The examples are chosen such that their exact solutions are known. 
The numerical computations have been done with the help of software DEV C++ and Matlab. In each example, we have
calculated the absolute maximum error by the following formula
\[ \text{Absolute Error} = \| u_{\text{exact}} - u_{\text{num}} \|_{\infty} = \max_j |u_{j,\text{exact}} - u_{j,\text{num}}| . \]  

where \( u_{\text{exact}} \) and \( u_{\text{num}} \) denote the exact and numerical solution of the problem respectively.

Example 1 In this example, we consider the telegraph equation (1.1) with domain \([0, 1]\) and the following initial and
boundary conditions
\[ u(x, 0) = \sinh(x), \]
\[ \frac{\partial u}{\partial t}(x, 0) = -2 \sinh(x), \]
\[ u(0, t) = 0, \]
\[ u(1, t) = e^{-2t} \sinh(1), \quad t \geq 0, \]
and the function \( g(x, t) = (3 - 4 \alpha + \beta^2) e^{-2t} \sinh(x) \)

The exact solution is given by [14]
\[ u(x, t) = e^{-2t} \sinh(x). \]

The maximum absolute errors of the example are given in the Table 1 for \( \alpha = 20, \beta = 10 \) at different time by taking
time step length \( \Delta t = 0.01, 0.001 \) and 0.0001 for uniform and non uniform grid points (Chebyshev-Gauss-Lobatto). Fig
1 and Fig 2 show the absolute errors obtained by uniform and non uniform grid points at \( t = 1.0 \). The Fig 7 shows the
behavior of numerical solutions for \( \alpha = 50, \beta = 2 \) at different time.

Example 2 We consider the telegraph equation (1.1) with domain \([0, 1]\) and the following conditions
\[ u(x, 0) = \sin(x), \]
\[ \frac{\partial u}{\partial t}(x, 0) = 0, \]
\[ u(0, t) = 0, \]
\[ u(1, t) = \cos(t) \sin(1), \quad t \geq 0, \]

IJNS email for contribution: editor@nonlinearscience.org.uk
\( g(x,t) = -2\alpha \sin(t) \sin(x) + \beta^2 \cos(t) \sin(x) \)

The exact solution is given by \[14\]

\[ u(x,t) = \cos(t) \sin(x). \]

Table 2 shows the maximum absolute errors of the example for \(\alpha = 10, \beta = 5\) at different time by taking time step length \(\Delta t = 0.01, 0.001\) and \(0.0001\) for uniform and non uniform grid points (Chebyshev-Gauss-Lobatto). Fig 3 and Fig 4 show the behavior of absolute errors obtained by uniform and non uniform grid points at \(t = 1.0\). The Fig 8 shows the behavior of numerical solutions for \(\alpha = 20, \beta = 10\) at different time.

**Example 3** In this example, we consider the telegraph equation (1.1) with domain \([0, 1]\) and the following conditions

\[
\begin{align*}
\frac{\partial u}{\partial t}(x,0) &= \tan(x/2), \\
\frac{\partial^2 u}{\partial t^2}(x,0) &= \frac{1}{2} \left(1 + \tan^2\left(x/2\right)\right), \\
u(0, t) &= \tan(t/2), \\
u(1, t) &= \tan\left(\frac{1 + t}{2}\right), \quad t \geq 0,
\end{align*}
\]

\( g(x,t) = \alpha \left(1 + \tan^2\left(\frac{x + t}{2}\right)\right) + \beta^2 \tan\left(\frac{x + t}{2}\right) \)

The exact solution is given by \[14\]

\[ u(x,t) = \tan\left(\frac{x + t}{2}\right). \]

In Table 3, we present the maximum absolute errors of the example for \(\alpha = 10, \beta = 5\) at different time by taking time step length \(\Delta t = 0.01, 0.001\) and \(0.0001\) for uniform and non uniform grid points (Chebyshev-Gauss-Lobatto). The Figures 5-6 show the behavior of absolute errors obtained by uniform and non uniform grid points at time \(t = 1.0\). The Fig 9 shows the behavior of numerical solutions \(\alpha = 20, \beta = 10\) at different time.

**6 Conclusion**

In this article, we presented a numerical technique for solving the second order one dimensional linear hyperbolic telegraph equation based on polynomial differential quadrature method. The technique is easy and very suitable for computer programming and provides numerical solutions closed to the exact solutions. Finally, a comparison is made via Tables 1-3 and Figures 1-6 between the absolute errors obtained by Chebyshev-Gauss-Lobatto grid points and the uniform grid points and finds that the Chebyshev-Gauss-Lobatto grid points give better accuracy and stable numerical solutions as comparison to the uniform grid points which agree with the theory of Shu given in his book [17].

**References**


Table 1: A comparison of maximum absolute errors of Example 1 for $\alpha = 20$, $\beta = 10$ at different time and $N = 21$.

<table>
<thead>
<tr>
<th>T</th>
<th>$\Delta t = 0.001$ Uniform Grid</th>
<th>$\Delta t = 0.001$ Nonuniform grid</th>
<th>$\Delta t = 0.0001$ Uniform Grid</th>
<th>$\Delta t = 0.0001$ Nonuniform grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>2.25640×10^{-4}</td>
<td>2.22066×10^{-4}</td>
<td>2.24300×10^{-5}</td>
<td>2.21712×10^{-5}</td>
</tr>
<tr>
<td>1.0</td>
<td>5.75205×10^{-3}</td>
<td>1.41294×10^{-4}</td>
<td>3.56696×10^{-4}</td>
<td>1.41832×10^{-5}</td>
</tr>
<tr>
<td>1.5</td>
<td>2.43143×10^{-2}</td>
<td>6.93111×10^{-5}</td>
<td>1.94934×10^{-2}</td>
<td>6.91111×10^{-6}</td>
</tr>
<tr>
<td>2.0</td>
<td>2.27829×10^{-5}</td>
<td>2.16147×10^{-6}</td>
<td>2.66660×10^{-7}</td>
<td></td>
</tr>
<tr>
<td>CPU Time</td>
<td>1 second</td>
<td>2 second</td>
<td>5 second</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: A comparison of maximum absolute errors of Example 2 for $\alpha = 10$, $\beta = 5$ at different time and $N = 21$.

<table>
<thead>
<tr>
<th>T</th>
<th>$\Delta t = 0.001$ Uniform Grid</th>
<th>$\Delta t = 0.001$ Nonuniform grid</th>
<th>$\Delta t = 0.0001$ Uniform Grid</th>
<th>$\Delta t = 0.0001$ Nonuniform grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>1.66760×10^{-4}</td>
<td>1.24976×10^{-4}</td>
<td>1.67216×10^{-5}</td>
<td>1.28551×10^{-5}</td>
</tr>
<tr>
<td>0.5</td>
<td>4.30081×10^{-3}</td>
<td>3.13646×10^{-4}</td>
<td>4.71302×10^{-4}</td>
<td>3.20793×10^{-5}</td>
</tr>
<tr>
<td>1.0</td>
<td>8.47480×10^{-3}</td>
<td>5.59863×10^{-4}</td>
<td>8.62796×10^{-4}</td>
<td>5.71454×10^{-5}</td>
</tr>
<tr>
<td>1.5</td>
<td>6.69671×10^{-4}</td>
<td>6.82827×10^{-5}</td>
<td>6.83347×10^{-6}</td>
<td></td>
</tr>
<tr>
<td>CPU Time</td>
<td>1 second</td>
<td>2 second</td>
<td>5 second</td>
<td></td>
</tr>
</tbody>
</table>
Table 3: A comparison of maximum absolute errors of Example 3 for $\alpha = 20$, $\beta = 10$ at different time and $N = 21$.

<table>
<thead>
<tr>
<th>$\Delta t$</th>
<th>Uniform Grid</th>
<th>Nonuniform grid</th>
<th>Uniform Grid</th>
<th>Nonuniform grid</th>
<th>Uniform Grid</th>
<th>Nonuniform grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>$8.33780 \times 10^{-4}$</td>
<td>$7.33771 \times 10^{-4}$</td>
<td>$8.42642 \times 10^{-4}$</td>
<td>$7.13861 \times 10^{-4}$</td>
<td>$8.25562 \times 10^{-5}$</td>
<td>$7.10494 \times 10^{-5}$</td>
</tr>
<tr>
<td>0.5</td>
<td>$9.33501 \times 10^{-4}$</td>
<td>$9.23501 \times 10^{-4}$</td>
<td>$2.16334 \times 10^{-4}$</td>
<td>$9.33979 \times 10^{-5}$</td>
<td>$1.81321 \times 10^{-5}$</td>
<td>$2.20638 \times 10^{-5}$</td>
</tr>
<tr>
<td>1.0</td>
<td>$1.71143 \times 10^{-3}$</td>
<td>$1.67143 \times 10^{-3}$</td>
<td>$1.71289 \times 10^{-3}$</td>
<td>$1.68289 \times 10^{-4}$</td>
<td>$9.87860 \times 10^{-4}$</td>
<td>$1.71277 \times 10^{-4}$</td>
</tr>
<tr>
<td>1.5</td>
<td>$\ldots$</td>
<td>$5.03633 \times 10^{-3}$</td>
<td>$\ldots$</td>
<td>$5.02951 \times 10^{-4}$</td>
<td>$\ldots$</td>
<td>$5.02868 \times 10^{-5}$</td>
</tr>
<tr>
<td>CPU Time</td>
<td>1 second</td>
<td>2 second</td>
<td>5 second</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Absolute errors of Example 1 for $\alpha = 20$, $\beta = 10$, $N = 21$ and $\Delta t = 0.001$ at uniform grid points and $t = 1.0$.

Figure 2: Absolute errors of Example 1 for $\alpha = 20$, $\beta = 10$, $N = 21$ and $\Delta t = 0.001$ at uniform grid points and $t = 1.0$.

Figure 3: Absolute errors of Example 2 for $\alpha = 10$, $\beta = 5$, $N = 21$ and $\Delta t = 0.001$ at uniform grid points and $t = 1.0$.

Figure 4: Absolute errors of Example 2 for $\alpha = 10$, $\beta = 5$, $N = 21$ and $\Delta t = 0.001$ at uniform grid points and $t = 1.0$. 

IJNS homepage: http://www.nonlinearscience.org.uk/
Figure 5: Absolute errors of Example 3 for $\alpha = 20$, $\beta = 10$, $N = 21$ and $\Delta t = 0.001$ at uniform grid points and $t = 1.0$.

Figure 6: Absolute errors of Example 3 for $\alpha = 20$, $\beta = 10$, $N = 21$ and $\Delta t = 0.001$ at uniform grid points and $t = 1.0$.

Figure 7: Numerical solutions of Example 1 for $\alpha = 50$, $\beta = 2.0$ and $\Delta t = 0.001$ at different times.

Figure 8: Numerical solutions of Example 2 for $\alpha = 20$, $\beta = 10$ and $\Delta t = 0.001$ at different times.

Figure 9: Numerical solutions of Example 3 for $\alpha = 20$, $\beta = 10$ and $\Delta t = 0.001$ at different times.