



## A numerical solution of two-dimensional hyperbolic telegraph equation based on moving least square meshless method and radial basis functions

Sepideh Niknam and Hojatollah Adibi\*

Department of Applied Mathematics, Central Tehran Branch, Islamic Azad University, Tehran, Iran.

---

### Abstract

In this research, a linear combination of moving least square (MLS) and local radial basis functions (LRBFs) is considered within the framework of the meshless method to solve the two-dimensional hyperbolic telegraph equation. Besides, the differential quadrature method (DQM) is employed to discretize temporal derivatives. Furthermore, a control parameter is introduced and optimized to achieve minimum errors via an experimental approach. Illustrative examples are provided to demonstrate the applicability and efficiency of the method. The results prove the superiority of this method over using MLS and LRBF individually.

---

**Keywords.** Meshless method, Moving least square, Local radial basis function, Two-dimensional hyperbolic telegraph equation, Differential quadrature method.

**2010 Mathematics Subject Classification.** 65L05, 34K06, 34K28.

### 1. INTRODUCTION

Meshless (mesh-free) methods, as numerical techniques are used to analyze a wide range of industrial and engineering applications. In these methods, only scattered nodes are required to approximate unknown functions [6]. The moving least square (MLS) method and radial basis functions (RBFs) are originally proposed to make scattered data approximation. In MLS, a minimization scheme is utilized to optimize the level of accuracy. This imposes particular limitations including complex calculations and Delta Kronecker-free features. Moreover, MLS employs a special procedure to consider internal and external nodes, which might lead to complicated conditions [10]. In order to overcome difficulties of approximating functions by MLS near edge-points, Salkauskas [17] proposed a small support domain for weight functions using a thin plate and other RBFs. In addition to the aforementioned disadvantage, the algebraic set of equations for obtaining MLS approximation may not be solved. One of the ways of eliminating this problem was suggested as using orthogonal polynomials [11, 25, 26].

Besides, RBF is an interpolation method used for scattered data approximation. While a great number of investigations have been developed to approximate the solution of partial differential equations, including FDM [22], FEM [27], BEM [1], and FVM [14], researchers' attention has recently drawn to RBF. Ronald Hardy first introduced RBF methods in 1968 [7]. Then, Richard Franke [18], Charles Micchelli [13], and Edward Kansa respectively in 1979, 1986, and 2000 extended the RBF's theory. Various advantages of the RBF method have been reported frequently in previous studies. However, it has been mentioned in [24] that increasing the number of collocation points leads to an ill-conditioned set of equations.

In 2015, Baiyu Wang [24] proposed a new method in which a linear combination of local MLS and LRBF shape functions was employed to approximate functions using the same support domain. In this method, temporal derivatives were discretized by the finite difference method (FDM). The effectiveness of this scheme was examined by providing examples of one and two-dimensional heat transfer equations.

---

Received: 23 October 2020 ; Accepted: 07 August 2021.

\* Corresponding author. Email: adibih@aut.ac.ir.

On the other hand, there are numerous methods to approximate a certain function with respect to time including FDM. An alternative way is DQM, first proposed and successfully employed in [21]. This method had been frequently applied to spatial derivatives of differential equations before. However, in [21], DQM was step-by-step applied over a time span. Later, this method was used in [15] to solve the dynamic equation of particle delivery through a carbon Nano-tube.

Furthermore, the telegraph equation has been repeatedly studied in previous works. In [8], the polynomial differential quadrature method was used to solve the two-dimensional hyperbolic telegraph equation under Dirichlet and Neumann boundary conditions. Dehghan and Shokri introduced an additional meshless method for solving this equation using radial basis functions of thin plate splines at collocation points [2]. In [3], the advantages of local weak and strong forms of the meshless method were combined. In this method, the local Petrov–Galerkin weak form was applied only to the nodes on the Neumann boundaries while a meshless collocation method based on the strong form of the equation was applied to the interior nodes and those located on the Dirichlet boundaries. Moreover, Dehghan and Shokri proposed a new numerical method to solve the one-dimensional hyperbolic telegraph equation using collocation points and approximating the solution using thin-plate splines radial basis function [4]. All these attempts resulted in the acceptable accuracy of the solutions.

In the current work, a linear combination of LRBFs and MLS shape functions is employed to overcome the drawbacks of either individually using RBF or MLS. Then, step-by-step DQM is applied to temporal derivatives to achieve more accuracy in comparison with FDM. This method is specifically utilized to solve two-dimensional telegraph equation as a second-order hyperbolic partial differential equation [4] and [16]:

$$\frac{\partial^2 u}{\partial t^2} + 2\alpha(x, y) \frac{\partial u}{\partial t} + \beta^2(x, y)u = A(x, y) \frac{\partial^2 u}{\partial x^2} + B(x, y) \frac{\partial^2 u}{\partial y^2} + f(x, y, t) \quad (1.1)$$

Where,  $\alpha(x, y)$ ,  $\beta(x, y)$ ,  $A(x, y)$ , and  $B(x, y)$  are known coefficients that are positive throughout the domain.

Eq. (1.1) represents two dimensional telegraph equation provided that  $\alpha$  and  $\beta$  are positive constants ( $\alpha \geq \beta > 0$ ) and,  $A = B = 1$ .

This paper includes five sections introduced as follows. In section 2, preliminaries of MLS, RBF and their local approximation techniques as well as time discrete scheme utilizing DQM are provided. Section 3 describes solution procedure. In section 4, some test examples are solved to demonstrate the efficiency of the method. Finally, section 5 includes conclusion.

## 2. SPATIAL AND TEMPORAL DISCRETE SCHEMES

In this section, an overview of RBF, MLS and step-by-step DQM are provided. Then, the solution procedure of the two dimensional telegraph equation is presented in section 3.

**2.1. A brief review of RBF and LRBF.** Radial basis function (RBF) method is an efficient and a truly mesh-free technique for interpolation of multidimensional scattered data for solving partial differential equations (PDEs). The RBF approximation of a function  $u(x)$ , may be written as:

$$u(\mathbf{x}) \simeq \sum_{j=1}^N \lambda_j \varphi(\mathbf{x}, \mathbf{x}_j) + \psi(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Omega \subset \mathbb{R}^d \quad (2.1)$$

where  $N$  is the number of data points throughout the domain,  $\mathbf{x} = (x_1, x_2, \dots, x_d)$ ,  $d$  stands for the dimension of the problem,  $\lambda_j$  are coefficients to be determined,  $\varphi$  is the RBF and  $\psi(\mathbf{x})$  is called balancing polynomial. In this article, multi-quadric form of RBF is chosen as:  $\varphi(x, x_j) = \varphi(r_j) = \sqrt{c^2 + r_j^2}$  where  $r_j = \|\mathbf{x} - \mathbf{x}_j\|$  is the Euclidean norm and  $c$  is constant and is selected in this paper as:  $c = 3.6$ . Eq. (2.1) may be written without the balancing polynomial  $\psi(x)$ , if  $\varphi$  is unconditionally positive definite (e.g., Gaussian or inverse multi-quadrics) [2]. More details are provided in [2]. In this paper,  $\psi(\mathbf{x}) = 0$ .

In order to determine unknown coefficients  $\lambda_j$ ,  $N$  collocation points are needed:  $\mathbf{x}_i (i = 1, 2, 3, \dots, N)$  to form a



linear set of  $N$  equations as follows.

$$u(\mathbf{x}_i) = u_i = \sum_{j=1}^N \lambda_j \varphi(\mathbf{x}_i, \mathbf{x}_j) \quad , \quad i = 1, 2, 3, \dots, N \tag{2.2}$$

Where,  $u_i$  is the value of  $u(\mathbf{x})$  at the point  $\mathbf{x}_i$ . By solving Eq. (2.2) for  $\lambda_j$ , the RBF interpolation function  $u(\mathbf{x})$  is attained which can be written for a two dimensional space as:

$$u(x, y) \simeq \sum_{j=1}^N \lambda_j \sqrt{c^2 + (x - x_j)^2 + (y - y_j)^2} \quad i = 1, 2, 3, \dots, N \tag{2.3}$$

For large number of collocation points, the algebraic set of equations always becomes ill-conditioned. Moreover, accuracy of the solution sensitively depends upon selecting the free parameter  $c$  [24]. In order to resolve these issues, for approximating the function at a certain point  $\mathbf{x}$ , only a few nodes neighboring the point are employed. The domain which contains all these nodes is called the influence domain of the point  $\mathbf{x}$ . Influence domains can be chosen in an arbitrary shapes including circle, rectangle, etc.

The local RBF (LRBF) method utilizes only the nodal points in the influence domain of a point at the general coordinate  $\mathbf{x}$  in each computing step, to have a better-conditioned linear system [4, 9, 19, 20, 23]. In LRBF, the procedure is the same as RBF except for that instead of using all  $N$  points through the domain, only  $n$  points of the influence domain of the point  $\mathbf{x}$  is used. Hence, the interpolation LRBF  $u^L(\mathbf{x})$  is obtained as:

$$u^L(x, y) \simeq \sum_{j=1}^n \lambda_j^L \sqrt{c^2 + (x - x_j)^2 + (y - y_j)^2} \tag{2.4}$$

Where,  $\mathbf{x}_j = (x_j, y_j) (i = 1, 2, 3, \dots, n)$  is  $j$ -th point in the influence domain of the point  $\mathbf{x} = (x, y)$ . In addition, the superscript,  $L$ , represents the local RBF interpolation.

Since solving Eq. (2.2) results in finding  $\lambda_j$  in terms of  $u_i$  (for  $i = 1, 2, 3, \dots, N$ ),  $\lambda_j^L$  in Eq. (2.4) can be similarly obtained in terms of  $u_i$  (for  $i = 1, 2, 3, \dots, n$ ). Thus, Eq. (2.4) can be rearranged and collected with respect to  $u_j$  and be rewritten in terms of the nodal values  $u_j = u(x_j, y_j)$  as:

$$u^L(x, y) = \sum_{j=1}^n u_j \varphi_j^L(r_j) \tag{2.5}$$

Where,  $r_j = \sqrt{(x - x_j)^2 + (y - y_j)^2}$  and  $\varphi_j^L(r_j)$  is the coefficient of  $u_j$  in Eq. (2.4) and called LRBF shape function corresponding to the point  $\mathbf{x}_j = (x_j, y_j)$ .

RBF method has been proved to be a promising tool for solving differential equations. It has been successfully combined with other methods, e.g. boundary knot method [5]. The following subsections demonstrate how this method can be combined with moving least square meshless method.

**2.2. Moving Least Square Approximation.** Let  $u(\mathbf{x})$  be a multivariate function defined in  $d$  dimensional domain  $\Omega$ . Assuming that the influence domain of  $\mathbf{x} \in \Omega$  contains a set of  $n$  local nodes, the approximation of  $u(\mathbf{x})$  at a certain point  $\mathbf{x}$  is represented as  $u^h(\mathbf{x})$

$$u^h(\mathbf{x}) = \sum_{j=1}^m p_j(\mathbf{x}) a_j(\mathbf{x}) \equiv \mathbf{P}^T(\mathbf{x}) \mathbf{a}(\mathbf{x}) \tag{2.6}$$

Where  $p_j(\mathbf{x})$  is a polynomial in terms of  $\mathbf{x}$  and the  $j$ -th component of polynomial vector  $\mathbf{P}(\mathbf{x})$  such that  $\mathbf{P}^T(\mathbf{x}) = \{p_1(\mathbf{x}), p_2(\mathbf{x}), p_3(\mathbf{x}), \dots, p_m(\mathbf{x})\}$ . In addition,  $\mathbf{a}(\mathbf{x})$  is unknown coefficient vector, arbitrarily function of  $\mathbf{x}$ , given by:  $\mathbf{a}^T(\mathbf{x}) = \{a_1(\mathbf{x}), a_2(\mathbf{x}), a_3(\mathbf{x}), \dots, a_m(\mathbf{x})\}$  to be determined later. For a 1D space,  $\mathbf{P}(\mathbf{x})$  can be displayed as:

$$\mathbf{P}^T(\mathbf{x}) = \{1, x, x^2, \dots, x^m\} \tag{2.7}$$



And for a 2D space,

$${}^{\circ}{}^{\circ} \mathbf{P}^T({}^{\circ}{}^{\circ} \mathbf{x}) = {}^{\circ}{}^{\circ} \mathbf{P}^T(x, y) = \{1, x, y, xy, x^2, y^2, \dots, x^k, y^k\}. \quad (2.8)$$

Where,  $k < m$  in a way that the vector  ${}^{\circ}{}^{\circ} \mathbf{P}^T(x, y)$  contains  $m$  components.

To calculate  ${}^{\circ}{}^{\circ} \mathbf{a}({}^{\circ}{}^{\circ} \mathbf{x})$ , a functional of weighted residual,  $J$ , at the nodes located in the influence domain of  ${}^{\circ}{}^{\circ} \mathbf{x}$  is constructed and then minimized as follows.

$$J = \sum_{i=1}^n \hat{W}(\|{}^{\circ}{}^{\circ} \mathbf{x} - {}^{\circ}{}^{\circ} \mathbf{x}_i\|) [{}^{\circ}{}^{\circ} \mathbf{P}^T(x_i) {}^{\circ}{}^{\circ} \mathbf{a}(x) - u_i]^2 \quad (2.9)$$

Where,  $\hat{W}(\|{}^{\circ}{}^{\circ} \mathbf{x} - {}^{\circ}{}^{\circ} \mathbf{x}_i\|)$  is the weight function. The minimization process necessitates:

$$\frac{\partial J}{\partial a_j({}^{\circ}{}^{\circ} \mathbf{x})} = 0 \quad j = 1, 2, 3, \dots, m \quad (2.10)$$

which results in a linear set of equations. Solving this set leads to finding  $a_j({}^{\circ}{}^{\circ} \mathbf{x})$  ( $j = 1, 2, 3, \dots, m$ ) in terms of  $u_i$  ( $i = 1, 2, 3, \dots, n$ ). Hence, Eq. (2.6) can be rewritten and collected with respect to  $u_j$  resulting in the following MLS approximation [24]:

$$u^h(x, y) = \Phi^T(\mathbf{x}) \mathbf{U} = \sum_{j=1}^n u_j \varphi_j^M({}^{\circ}{}^{\circ} \mathbf{x}) \quad (2.11)$$

Where,  $\mathbf{U}$  is the vector of nodal values of the function  $u(\mathbf{x})$ ,  $\mathbf{U}^T = \{u_1, u_2, u_3, \dots, u_n\}$  and  $\varphi_j^M(\mathbf{x})$  is the coefficient of  $u_j$  in Eq. (2.6) and is called the  $j$ -th element of MLS shape function vector  $\Phi$  which can be expressed as [24]:

$$\Phi^T = \{\varphi_1^M(\mathbf{x}), \varphi_2^M(\mathbf{x}), \varphi_3^M, \dots, \varphi_n^M(\mathbf{x})\} = \mathbf{P}^T(\mathbf{x}) \mathbf{A}^{-1}(\mathbf{x}) \mathbf{B}(\mathbf{x}) \quad (2.12)$$

In Eq. (2.12),  $\mathbf{A}(\mathbf{x})$  is called the MLS moment matrix given by [3]:

$$\begin{aligned} \mathbf{A}(\mathbf{x}) &= \mathbf{P}^T(\mathbf{x}) \mathbf{W}(\mathbf{x}) \mathbf{P}(\mathbf{x}) \\ \mathbf{B}(\mathbf{x}) &= \mathbf{P}^T(\mathbf{x}) \mathbf{W}(\mathbf{x}) \end{aligned} \quad (2.13)$$

Where,  $\mathbf{W}(\mathbf{x})$  is the diagonal matrix of the vector  $\{\hat{W}(\|\mathbf{x} - \mathbf{x}_1\|), \hat{W}(\|\mathbf{x} - \mathbf{x}_2\|), \hat{W}(\|\mathbf{x} - \mathbf{x}_3\|), \dots, \hat{W}(\|\mathbf{x} - \mathbf{x}_n\|)\}^T$

The weighted function  $\hat{W}$  has been proposed in various types including cubic and quartic spline as well as exponential weight functions. In general,  $\hat{W}$  shows two specific behavior. First is that it devotes larger weighting to the points farther to the point  $\mathbf{x}$ , at which the function is approximated. Second is that it causes the points smoothly enter and leave the support domain when  $\mathbf{x}$  moves [12]. These properties may give proper guidelines for designing and choosing weighted function. In this paper, the weighted function  $\hat{W}$  is assumed to be:

$$\begin{aligned} \hat{W}(\mathbf{x} - \mathbf{x}_i) &\equiv \hat{W}(\bar{d}) = \begin{cases} 1 - 6\bar{d}^2 + 8\bar{d}^3 - 3\bar{d}^4, & \bar{d} \leq 1 \\ 0 & \bar{d} > 1 \end{cases} \\ \bar{d} &= \frac{\|\mathbf{x} - \mathbf{x}_i\|_2}{d_w} = \frac{\sqrt{(x - x_i)^2 + (y - y_i)^2}}{d_w} \end{aligned} \quad (2.14)$$

Where,  $d_w$  is the radius of support domain. For solving a specific problem,  $d_w$  must be chosen in a way that the number of points in support domain of a certain point,  $\mathbf{x}$ , equals to the number of unknown coefficients  $\lambda_j$ . Otherwise, the resulting set of algebraic equations.

In this work, the radius of the influence domain ( $h$ ) is selected in a way that the constructed MLS shape functions are compatible and consistent with the order of polynomials included in the formulation. To ensure that the number of points for approximating a function at a certain point matches the order of polynomials,  $h$  was experimentally selected as  $h = 1.5\delta$ , where  $\delta$  is horizontal (or vertical) distance between two adjacent nodes. This selection leads to the solvability of the set of equations (2.10).



**2.3. The combined shape function.** For avoiding disadvantages of individually using LRBF or MLS, a linear combination of these two mentioned methods' shape functions is employed in this research via introducing a control parameter,  $v \in [0, 1]$ . This technique was first proposed in [24] and applied on two heat transfer equations successfully. Thus, the shape function,  $\phi_j$ , as well as the approximation of an arbitrary function,  $u(x, y)$ , are expressed as:

$$\begin{aligned} \phi_j &= \nu\varphi_j^M + (1 - \nu)\varphi_j^L \\ u(x, y) &= \sum_{j=1}^n \phi_j(x, y)u_j \end{aligned} \tag{2.15}$$

Since Eq.(2.15) is used to approximate unknown function of the telegraph equation in terms of nodal values, spatial derivatives of shape functions  $\phi_j(x, y)$  are required. The procedure of calculating these derivatives are frequently reported in previous works including [3].

**2.4. Time discrete scheme.** For approximating temporal derivatives of 2D telegraph equation, step-by-step DQM is employed. This method was effectively examined in [21] and [15]. Hence, the procedure of this method is summarized as follows [15].

First in this method, the time domain  $([0, T])$  is divided into  $b$  blocks (time spans) each of which is discretized with  $N_t$  nodes according to the following distribution:

$$\tau_{m,l} = \left\{ \frac{m-1}{b} + \frac{1}{2b} \left[ 1 - \cos\left(\frac{(l-1)\pi}{N_T-1}\right) \right] \right\} T, \begin{cases} m = 1, 2, 3, \dots, b \\ l = 1, 2, 3, \dots, N_t \end{cases} \tag{2.16}$$

Where,  $\tau_{m,l}$  indicates the  $l$ -th node in  $m$ -th block. On the basis of polynomial DQM, temporal derivatives of a certain function  $f(x, y, t)$  at  $t = \tau_{m,l}$  are approximated as:

$$\begin{cases} \dot{f}|_{t=\tau_{m,l}} \cong \sum_{i=1}^{N_t} g_{l,i}^{(1)} f_{m,i} \\ \ddot{f}|_{t=\tau_{m,l}} \cong \sum_{i=1}^{N_t} g_{l,i}^{(2)} f_{m,i} \end{cases} \tag{2.17}$$

Where,  $f_{m,i}$  is the value of the function  $f(x, y, t)$  in  $m$ -th block at  $i$ -th node or simply at  $t = \tau_{m,i}$ . The constants,  $g_{k,i}^{(s)} (s = 1, 2)$ , can be obtained from the following recursive formulas:

$$g_{l,i}^{(s)} = n(g_{l,l}^{(s-1)} g_{l,i}^{(1)} - \frac{g_{l,i}^{(s-1)}}{\tau_l - \tau_i}), \begin{cases} i, l = 1, 2, 3, \dots, N_t \\ i \neq l \end{cases} \tag{2.18}$$

$$g_{l,l}^{(s)} = - \sum_{j=1, j \neq l}^{N_t} g_{l,j}^{(s)}, l = 1, 2, 3, \dots, N_t \tag{2.19}$$

$$g_{l,i}^{(1)} = \frac{\hat{M}(\tau_l)}{(\tau_l - \tau_i)\hat{M}(\tau_i)}, \begin{cases} i, l = 1, 2, 3, \dots, N_t \\ i \neq l \end{cases} \tag{2.20}$$

Where,  $\hat{M}(\tau_l) = \prod_{j=1, j \neq l}^{N_t} (\tau_l - \tau_j)$ .

$g_{k,i}^{(1)}$  and  $g_{k,i}^{(2)}$  are calculated by Eqs. (2.18)-(2.20). Note that, in each block, initial conditions are final temporal conditions of the previous block. In other words, for  $m \geq 2$ :

$$\begin{cases} f|_{t=\tau_{m,1}} = f|_{t=\tau_{m-1,N_t}} \\ \dot{f}|_{t=\tau_{m,1}} = \dot{f}|_{t=\tau_{m-1,N_t}} \end{cases} \tag{2.21}$$

Which results in:

$$\begin{cases} f_{m,1} = f_{m-1,N_t} \\ \sum_{i=1}^{N_t} g_{l,i}^{(1)} f_{m,i} = \sum_{i=1}^{N_t} g_{N_t,i}^{(1)} f_{m-1,i}, m = 2, 3, \dots, b \end{cases} \tag{2.22}$$



For  $m = 1$ , the initial conditions are the same as the main initial conditions of the problem. Hence, in each block (for  $m = 1, 2, 3, \dots, b$ ), the main differential equation must be discretized and solved at  $t = \tau_{m,3}, \tau_{m,4}, \tau_{m,5}, \dots, \tau_{m,N_t}$  under initial conditions Eq. (2.22).

### 3. SOLUTION PROCEDURE

In section 2, the spatial and temporal approximation used in this paper were explained. Now, Eq. (1.1) is considered to be discretized throughout the domain  $\Omega = \{(x, y), L_x^0 < x < L_x^1, L_y^0 < y < L_y^1\}$  surrounded by the boundary  $\partial\Omega$  under initial conditions and Dirichlet boundary equations listed as:

$$\frac{\partial^2 u}{\partial t^2} + 2\alpha(x, y) \frac{\partial u}{\partial t} + \beta^2(x, y)u = A(x, y) \frac{\partial^2 u}{\partial x^2} + B(x, y) \frac{\partial^2 u}{\partial y^2} + f(x, y, t)$$

$$u(x, y, 0) = g_0(x, y), \quad (x, y) \in \Omega \quad (3.1)$$

$$\frac{\partial u(x, y, t)}{\partial t} \Big|_{t=0} = g_1(x, y), \quad (x, y) \in \Omega$$

$$u(x, y, t) = h(x, y, t), \quad (x, y) \in \partial\Omega \quad (3.2)$$

Before discretizing Eq. (1.1), Eq. (2.15) must be rewritten in a general form. Supposing that  $N$  is total number of scattered points (nodes) through the domain  $\Omega$ , Eq. (2.15) can be displayed as:

$$u(x_i, y_i, t) = \sum_{j=1}^N \psi_{i,j} u_j(t) \quad (3.3)$$

In which,  $(x_i, y_i)$  is  $i$ -th node in global numbering. Furthermore,  $\psi_{i,j} = \phi_j(x_i, y_i)$  if  $j$ -th node is in influence domain of  $i$ -th node. Otherwise,  $\psi_{i,j} = 0$ . Moreover, the spatial derivatives of  $u(x, y, t)$  at  $i$ -th node  $((x, y) = (x_i, y_i))$  are given as follows.

$$\frac{\partial^2 u(x, y, t)}{\partial x^2} \Big|_{(x,y)=(x_i,y_i)} = \sum_{j=1}^N \psi_{i,j}^{xx} u_j(t)$$

$$\frac{\partial^2 u(x, y, t)}{\partial y^2} \Big|_{(x,y)=(x_i,y_i)} = \sum_{j=1}^N \psi_{i,j}^{yy} u_j(t) \quad (3.4)$$

Now, Eq. (1.1) at  $i$ -th point, which is inside the domain  $\Omega$ , can be written as:

$$\sum_{j=1}^N \psi_{i,j} \ddot{u}_j(t) + 2\alpha_i \psi_{i,j} \dot{u}_j(t) + \beta_i^2 \psi_{i,j} u_j(t) - A_i \psi_{i,j}^{xx} u_j(t) - B_i \psi_{i,j}^{yy} u_j(t) - f_i = 0 \quad (3.5)$$

Where  $\alpha_i, \beta_i, A_i, B_i$  and  $f_i$  are respectively  $\alpha(x_i, y_i), \beta(x_i, y_i), A(x_i, y_i), B(x_i, y_i)$  and  $f(x_i, y_i)$ . If  $i$ -th node is on the boundary  $\partial\Omega$ , the boundary conditions Eq. (3.2) must be set:

$$u_i = h_i(t) \quad (3.6)$$

Eq. (3.5) or (3.6) must be written for  $i = 1, 2, 3, \dots, N$  and then all equations are collected and expressed in the following matrix form:

$$\mathbf{M}\ddot{\mathbf{X}}(t) + \mathbf{C}\dot{\mathbf{X}}(t) + \mathbf{K}\mathbf{X}(t) - \mathbf{F} = 0 \quad (3.7)$$

In Eq. (3.7),  $\mathbf{X}$  and  $\mathbf{F}$  are respectively the vectors of unknowns and constants:

$$\mathbf{X}^T = \{u_1, u_2, u_3, \dots, u_N\},$$

$$\mathbf{F}^T = \{f_1, f_2, f_3, \dots, f_N\}, \quad (3.8)$$

In Eq. (3.8), if  $i$ -th node is on the boundary  $\partial\Omega$ , the corresponding element of the vector,  $\mathbf{F}$  must be replaced with  $h_i$  using Eq. (3.6).



In addition,  $\mathbf{M}, \mathbf{C}$ , and  $\mathbf{K}$  are matrices whose components at  $i - th$  row are as follows: If  $i - th$  node is inside the domain, then:

$$\begin{aligned} M_{ij} &= \psi_{i,j} \\ C_{ij} &= 2\alpha_i \psi_{i,j} \\ K_{ij} &= \beta_i^2 \psi_{i,j} - A_i \psi_{i,j}^{xx} - B_i \psi_{i,j}^{yy} \end{aligned} \tag{3.9}$$

If  $i - th$  node is on the boundary  $\partial\Omega$  :

$$\begin{aligned} M_{ij} &= 0 \\ C_{ij} &= 0 \\ K_{ij} &= 1 \end{aligned} \tag{3.10}$$

Now, for solving the linear initial value set of ordinary differential equations (3.7), step-by-step DQM is employed using Eqs. (2.16) -(2.22). First, the coefficients  $g_{l,i}^{(1)}$  and  $g_{l,i}^{(2)}$  are calculated by Eqs. (2.18). Then the time span,  $[0, T]$ , is discretized with  $b$  blocks and  $N_t$  nodes in each block according to Eq. (2.16). In each block based on Eq. (2.17), Eq. (3.7) can be approximated at the time  $\tau_{m,l}$  for  $l = 3, 4, 5, \dots, N_t$ :

$$\mathbf{M} \sum_{i=1}^{N_t} g_{l,i}^{(2)} \mathbf{X}_{m,i} + \mathbf{C} \sum_{i=1}^{N_t} g_{l,i}^{(1)} \mathbf{X}_{m,i} + \mathbf{K} \mathbf{X}_{m,l} - \mathbf{F} = 0 \tag{3.11}$$

Afterwards, two initial conditions are added to Eq. (3.11) to form a linear algebraic set of  $N, N_t$  equations. Solving this set leads to the solution of Eq. (3.7) within the  $m - th$  block. Repeating this procedure for  $m = 1, 2, 3, \dots, b$ , completely covers the time span.

#### 4. NUMERICAL EXPERIMENTS

This section provides numerical examples and results are compared with those reported in previous studies to reveal the superiority of this method over similar ones. In order to evaluate this method, the following errors are defined [20]:

$$L_\infty = \max_i |u_i^{exact} - u_i^{numerical}| \tag{4.1}$$

$$L_2 = \sqrt{\sum_{i=1}^N (u_i^{exact} - u_i^{numerical})^2}$$

$$RMS = \sqrt{\sum_{i=1}^N \frac{1}{N^2} (u_i^{exact} - u_i^{numerical})^2}$$

$$\epsilon = \sqrt{\sum_{i=1}^N \frac{(u_i^{exact} - u_i^{numerical})^2}{u_i^{exact^2}}}$$

Where,  $L_\infty$  and  $\epsilon$  are respectively absolute and the relative error. In the following examples, the domain is supposed to be  $[0, a] \times [0, a]$  and uniformly discretized with  $\bar{N}$  nodes in  $x$  and  $y$  directions leading to total number of  $\bar{N}^2$  nodes. Thus, the distance between two adjacent nodes is  $\delta = \frac{a}{\bar{N} - 1}$ . In addition, the radius of the influence domain,  $h$ , is selected as  $h = 1.5\delta$ . Furthermore, the number of blocks ( $b$ ) and temporal nodes of each block ( $N_t$ ) are displayed as  $(b, N_t)$ . Note that in all examples,  $c = 3.6$  and  $m = 2$ .



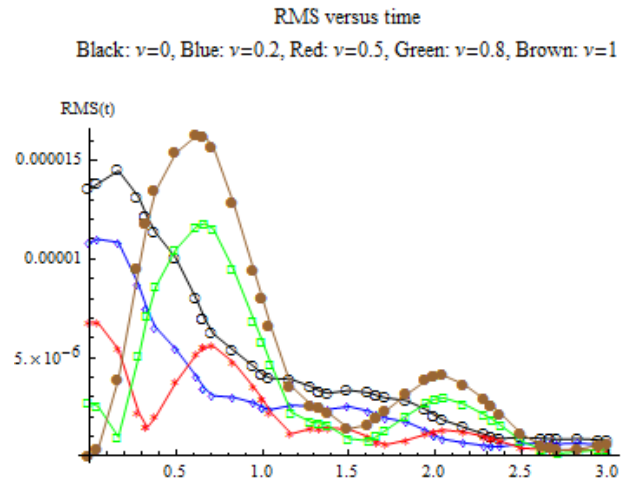


FIGURE 1. RMS error of example 4.1 through time for different values of  $\nu$ . Black:  $\nu = 0$ , Blue:  $\nu = 0.2$ , Red:  $\nu = 0.5$ , Green:  $\nu = 0.8$ , Brown:  $\nu = 1$ .

**Example 4.1.** In the first example,  $\alpha = \beta = 1$ , and  $A = B = 1$ . Spatial and temporal domains are  $(x, y) \in [0, 1] \times [0, 1]$  and  $T = 3$ . The initial and boundary conditions are given by

$$\begin{aligned}
 u(x, y, 0) &= \sinh(x) \sinh(y) \\
 u_t(x, y, t)|_{t=0} &= -\sinh(x) \sinh(y) \\
 u(0, y, t) &= u(x, 0, t) = 0 \\
 u(1, y, t) &= e^{-t} \sinh(1) \sinh(y) \\
 u(x, 1, t) &= e^{-t} \sinh(1) \sinh(x)
 \end{aligned} \tag{4.2}$$

The exact solution is  $u(x, y, t) = e^{-t} \sinh(x) \sinh(y)$ . The solution parameters are set as:  $\bar{N} = 11$ ,  $(b, N_t) = (9, 5)$ . Figure (1) shows the variations of RMS error versus time for several values of  $\nu$ . It is evident that for  $\nu = 0.5$ , the solution provides better approximation with the lowest maximum error. Hence,  $\nu = 0.5$  is set for obtaining the next results. Moreover, it is evident that the RMS decreases as the time elapses. This is one of the advantages of this method since in time-marching methods, e.g. finite difference method, the error is accumulated and subsequently increases as the time passes. However in step-by-step DQM, the time span is discretized similar to boundary value problems and the whole discretized domain is solved simultaneously whereas in time-marching methods the solution at each time step is obtained directly from previous time steps resulting in accumulating the error over time. Hence, in this method, the errors do not increase necessarily and may even decrease.

In Figure (2), maximum  $L_\infty$  for each point is plotted over whole time span,  $[0, 3]$  conceding the accuracy of the solution throughout the domain. This is similarly certified using Figure (3) depicting exact and numerical solution at various times. To receive a better consideration of accuracy, different errors obtained by this method are tabulated in Table (1). It is inferred that the present results are almost one order of magnitude more accurate than those reported in [5].





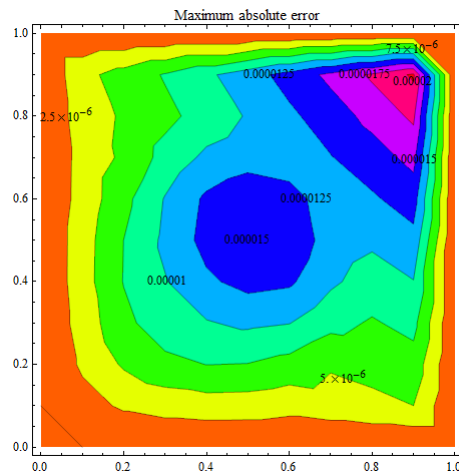


FIGURE 2. Maximum absolute error over time span throughout the domain in example 4.1.

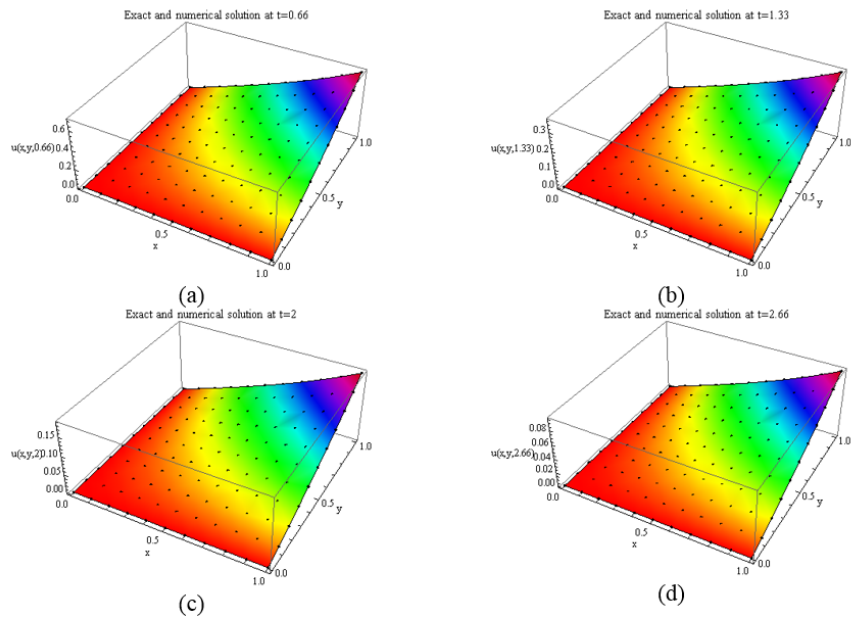


FIGURE 3. Exact and numerical solution throughout the domain in example 4.1 at various times: (a)  $t = 0.66$ , (b)  $t = 1.33$ , (c)  $t = 2$ , (d)  $t = 2.66$  ; Black points: numerical solution, 3D plot: exact solution.

TABLE 1. Comparison of the present solution of example 4.1 with the corresponding results in [5].

$t$	$L_2$	$L_\infty$	$\epsilon$	RMS	RMS reported in [5] by IMQ
0.5	$4.14 \times 10^{-5}$	$1.16 \times 10^{-5}$	$1.42 \times 10^{-5}$	$3.76 \times 10^{-6}$	$1.76 \times 10^{-5}$
1	$3.18 \times 10^{-5}$	$6.41 \times 10^{-6}$	$1.80 \times 10^{-5}$	$2.89 \times 10^{-6}$	$6.96 \times 10^{-6}$
2	$1.39 \times 10^{-5}$	$3.33 \times 10^{-6}$	$2.15 \times 10^{-5}$	$1.26 \times 10^{-6}$	$3.05 \times 10^{-5}$
3	$2.79 \times 10^{-6}$	$5.06 \times 10^{-7}$	$1.17 \times 10^{-5}$	$2.53 \times 10^{-7}$	$4.63 \times 10^{-5}$



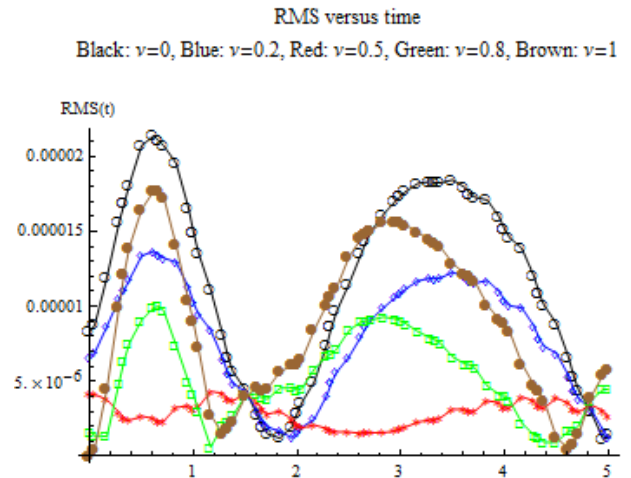


FIGURE 4. RMS error of example 4.2 through time for various values of  $\nu$ . Black:  $\nu = 0$ , Blue:  $\nu = 0.2$ , Red:  $\nu = 0.5$ , Green:  $\nu = 0.8$ , Brown:  $\nu = 1$ .

**Example 4.2.** In the this example,  $\alpha = \beta = 1$ , and  $A = B = 1$ . In addition,  $(x, y) \in [0, 1] \times [0, 1]$  and  $T = 5$ . The initial and boundary conditions are as follows.

$$\begin{aligned}
 u(x, y, 0) &= \sin(x) \sin(y) \\
 u_t(x, y, t)|_{t=0} &= 0 \\
 u(0, y, t) &= u(x, 0, t) = 0 \\
 u(1, y, t) &= \cos(t) \sin(1) \sin(y) \\
 u(x, 1, t) &= \cos(t) \sin(1) \sin(x)
 \end{aligned} \tag{4.3}$$

The exact solution is  $u(x, y, t) = \cos(t) \sin(x) \sin(y)$ . To solve this initial-boundary value problem,  $\bar{N}$  and  $(b, N_t)$  are respectively taken 11, (9, 5). Figure (4) illustrates RMS error through time for various values of  $\nu$ . Similar to example 4.1, it can be realized that applying  $\nu = 0.5$  results in a more accurate solution. Therefore,  $\nu$  is set to be 0.5. Moreover, maximum value of  $L_\infty$  all over the domain is depicted in Figure (5). In Figure (6), exact and numerical solution at different times are shown through the domain. Figures (4) and (5) demonstrate the capability of the proposed method for solving two dimensional telegraph equation.

Additionally, various errors, defined in Eq. (4.1) are calculated and given in Tables (2) and (3). Comparing the results with those presented in [8], [3] and [5] using these tables reveals higher accuracy of this method. On the other hand in [8] and [3],  $\bar{N} = 21$ . While in [8] standard RK4 method with  $\Delta t = 0.001$  was employed to solve time-dependent set of ordinary differential equations (ODEs), an iterative process with  $\Delta t = 0.1$  was used in [3] to solve ODE set. Hence, in the current paper, total number of spatial and temporal nodes are fewer than those in the aforementioned researches whereas the obtained accuracy is higher. This proves the superiority of the present method over some previous works.



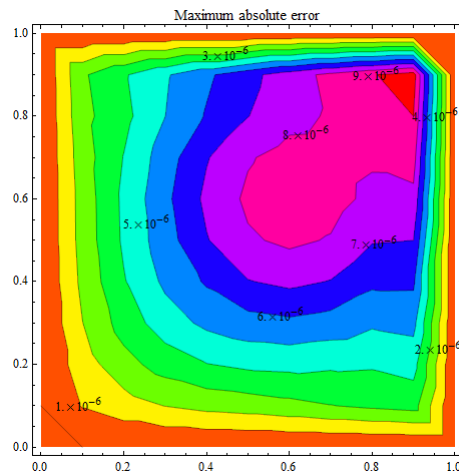


FIGURE 5. Maximum absolute error over time span throughout the domain in example 4.2.

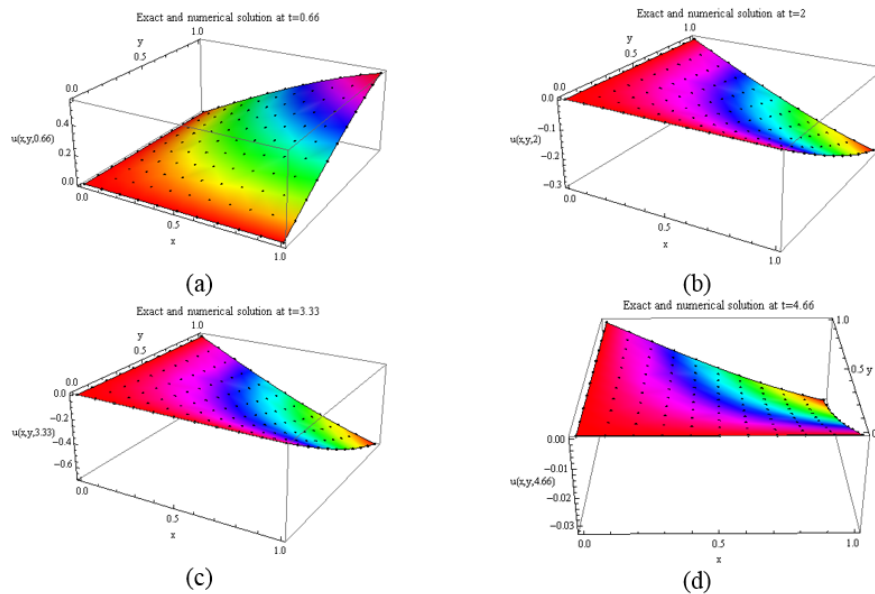


FIGURE 6. Exact and numerical solution throughout the domain in example 4.2 at various times: (a)  $t=0.66$ , (b)  $t=2$ , (c)  $t=3.33$ , (d)  $t=4.66$ ; Black points: numerical solution, 3D plot: exact solution.

TABLE 2. Comparison of the errors in example 4.2 with the corresponding results in [8] and [5].

$t$	$L_2$	RMS	RMS reported in [8]	RMS reported in [5] via IMQ
1	$3.18 \times 10^{-5}$	$3.18 \times 10^{-6}$	$4.27 \times 10^{-6}$	$7.24 \times 10^{-5}$
2	$2.29 \times 10^{-5}$	$2.08 \times 10^{-6}$	$3.94 \times 10^{-6}$	$9.37 \times 10^{-5}$
3	$2.01 \times 10^{-5}$	$1.83 \times 10^{-6}$	$7.65 \times 10^{-7}$	$5.46 \times 10^{-5}$
4	$3.58 \times 10^{-5}$	$3.25 \times 10^{-6}$	-	-
5	$2.89 \times 10^{-5}$	$2.63 \times 10^{-6}$	$4.18 \times 10^{-6}$	-



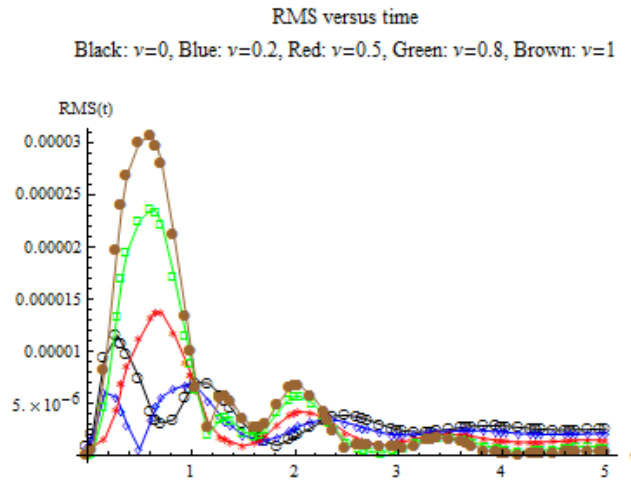


FIGURE 7. RMS error of example 4.3 with respect to time for different values of  $\nu$ . Black:  $\nu = 0$ , Blue:  $\nu = 0.2$ , Red:  $\nu = 0.5$ , Green:  $\nu = 0.8$ , Brown:  $\nu = 1$ .

TABLE 3. The results of example 4.2 compared with those reported in [8] and [3].

t	$L_\infty$	$\varepsilon$	Relative error reported in [8]	Relative error reported in [3] by MLWS-MLS	Relative error reported in [3] by MLPG-MLS
1	$6.24 \times 10^{-6}$	$2.09 \times 10^{-5}$	$2.66 \times 10^{-5}$	$5.60 \times 10^{-5}$	$2.24 \times 10^{-5}$
2	$5.15 \times 10^{-6}$	$1.78 \times 10^{-5}$	$3.19 \times 10^{-5}$	$8.17 \times 10^{-5}$	$5.00 \times 10^{-5}$
3	$7.57 \times 10^{-6}$	$6.59 \times 10^{-6}$	$2.60 \times 10^{-6}$	$1.90 \times 10^{-4}$	$8.00 \times 10^{-5}$
4	$6.20 \times 10^{-6}$	$1.77 \times 10^{-5}$	-	-	-
5	$6.29 \times 10^{-6}$	$3.30 \times 10^{-5}$	$4.98 \times 10^{-5}$	-	-

**Example 4.3.** Similar to foregoing examples,  $\alpha = \beta = A = B = 1, (x, y) \in [0, 1] \times [0, 1]$  and  $T = 5$ . The initial and boundary conditions are written as:

$$\begin{aligned}
 u(x, y, 0) &= \log(1 + x + y) \\
 u_t(x, y, t)|_{t=0} &= \frac{1}{1 + x + y} \\
 u(0, y, t) &= \log(1 + y + t) \\
 u(x, 0, t) &= \log(1 + x + t) \\
 u(1, y, t) &= \log(2 + y + t) \\
 u(x, 1, t) &= \log(2 + x + t)
 \end{aligned} \tag{4.4}$$

The exact solution is  $u(x, y, t) = \log(1 + x + y + t)$ . In this example,  $\bar{N} = 9$  and  $(b, N_t) = (15, 5)$ . In order to clarify the role of  $v$  in attaining an appropriate approximation, RMS error versus time for  $v = 0, 0.2, 0.5, 0.8, 1$  is plotted in Figure (7). According to this figure,  $v$  is taken to be 0.2 as this value offers higher accuracy. Furthermore, Figures (8) and (9) indicate that the numerical results agree well with the exact solution. To have a precise image of accuracy level, Tables (4) and (5) compare the results with obtained errors in [8],[3], and [5]. It can be inferred that the proposed method provides more accurate solution using fewer spatial and temporal nodes and consequently less computational cost.



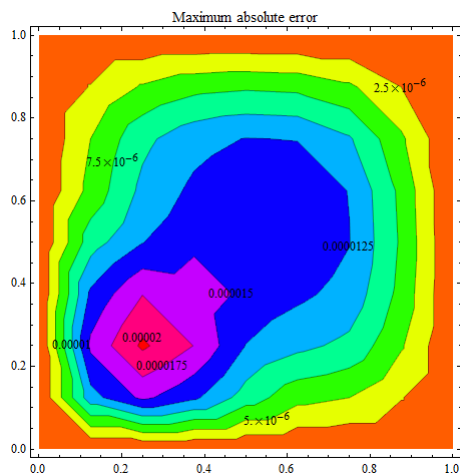


FIGURE 8. Maximum absolute error over time span throughout the domain in example 4.3.

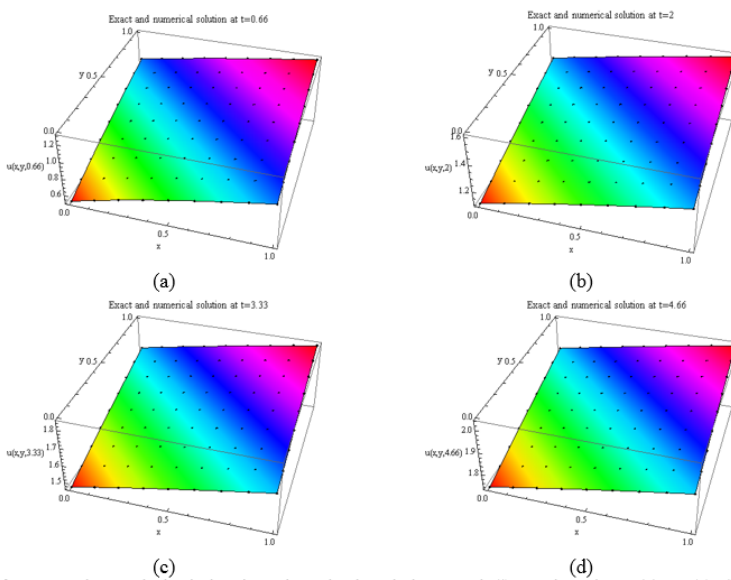


FIGURE 9. Exact and numerical solution throughout the domain in example 4.3 at various times: (a)  $t=0.66$ , (b)  $t=2$ , (c)  $t=3.33$ , (d)  $t=4.66$ ; Black points: numerical solution, 3D plot: exact solution.

TABLE 4. Comparison of the errors in example 4.3 with the corresponding results in [8] and [5].

$t$	$L_2$	RMS	RMS reported in [8]	RMS reported in [5] via IMQ
1	$6.19 \times 10^{-5}$	$6.87 \times 10^{-6}$	$5.39 \times 10^{-5}$	$1.14 \times 10^{-5}$
2	$2.45 \times 10^{-5}$	$2.72 \times 10^{-6}$	$4.95 \times 10^{-5}$	$2.39 \times 10^{-5}$
3	$1.84 \times 10^{-5}$	$2.04 \times 10^{-6}$	$4.96 \times 10^{-5}$	$3.18 \times 10^{-5}$
4	$2.09 \times 10^{-5}$	$2.33 \times 10^{-6}$	-	-
5	$1.97 \times 10^{-5}$	$2.19 \times 10^{-6}$	$4.42 \times 10^{-5}$	-



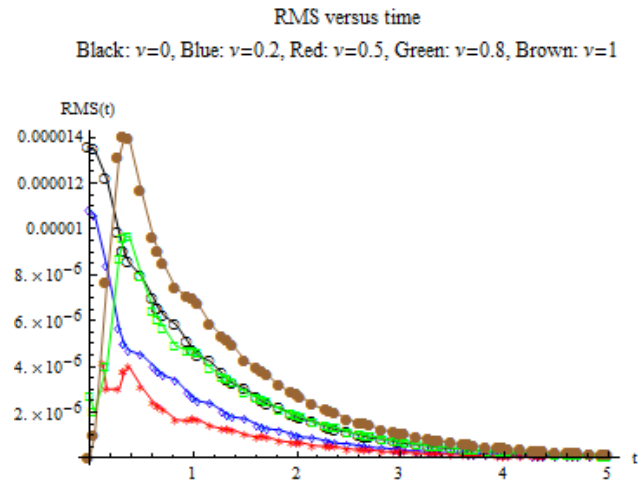


FIGURE 10. RMS error of example 4.4 with respect to time for different values of  $\nu$ . Black:  $\nu=0$ , Blue:  $\nu=0.2$ , Red:  $\nu=0.5$ , Green:  $\nu=0.8$ , Brown:  $\nu=1$ .

TABLE 5. The results of example 4.3 compared with those reported in [8] and [3].

t	$L_\infty$	$\varepsilon$	Relative error reported in [8]	Relative error reported in [3] by MLWS-MLS	Relative error reported in [3] by MLPG-MLS
1	$1.43 \times 10^{-5}$	$6.26 \times 10^{-6}$	$5.34 \times 10^{-5}$	$9.09 \times 10^{-5}$	$7.19 \times 10^{-5}$
2	$6.72 \times 10^{-6}$	$1.96 \times 10^{-6}$	$3.33 \times 10^{-5}$	$8.70 \times 10^{-4}$	$8.78 \times 10^{-5}$
3	$4.76 \times 10^{-6}$	$1.27 \times 10^{-6}$	$2.77 \times 10^{-5}$	$9.93 \times 10^{-4}$	$4.80 \times 10^{-4}$
4	$5.75 \times 10^{-6}$	$1.30 \times 10^{-6}$	-	$4.70 \times 10^{-3}$	$6.09 \times 10^{-4}$
5	$5.46 \times 10^{-6}$	$1.12 \times 10^{-6}$	$2.27 \times 10^{-5}$	$7.30 \times 10^{-3}$	$9.49 \times 10^{-4}$

**Example 4.4.** In this example, the following singular problem is investigated:  $\alpha = \frac{1}{x^2}, \beta = \frac{1}{x}, A = B = 1+x^2, (x, y) \in [0, 1] \times [0, 1]$  and  $T = 5$ . The initial and boundary conditions are written as:

$$\begin{aligned}
 u(x, y, 0) &= \sinh(x) \sinh(y) \\
 u_t(x, y, t)|_{t=0} &= -\sinh(x) \sinh(y) \\
 u(0, y, t) &= u(x, 0, t) = 0 \\
 u(1, y, t) &= \sinh(1)e^{-t} \sinh(y) \\
 u(x, 1, t) &= \sinh(1)e^{-t} \sinh(x)
 \end{aligned}
 \tag{4.5}$$

The exact solution is  $u(x, y, t) = e^{-t} \sinh(x) \sinh(y)$ . In this example,  $\bar{N} = 11$  and  $(b, N_t) = (15, 5)$ . Similar to previous examples, RMS error versus time for  $\nu = 0, 0.2, 0.5, 0.8, 1$  is depicted in Figure (10) suggesting  $\nu$  be selected as 0.2. Besides, Figures (11) and (12) reveal the agreement of numerical solution with the exact one. Moreover, Table (6) displays various errors and compares them with those given in [2] where finite difference scheme was used with  $N_t = 1001$  and  $\bar{N} = 11$ . It is induced that this method is capable of solving 2D telegraph equation more accurately with significantly less computational cost.



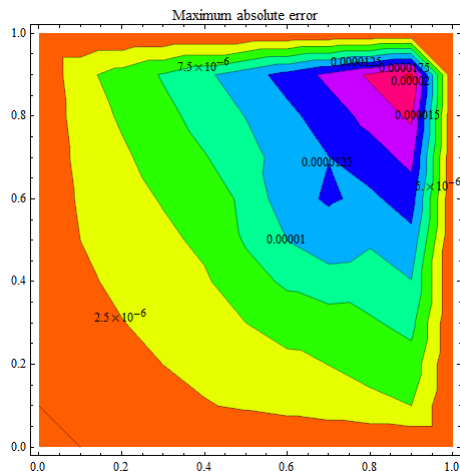


FIGURE 11. Maximum absolute error over time span throughout the domain in example 4.4.

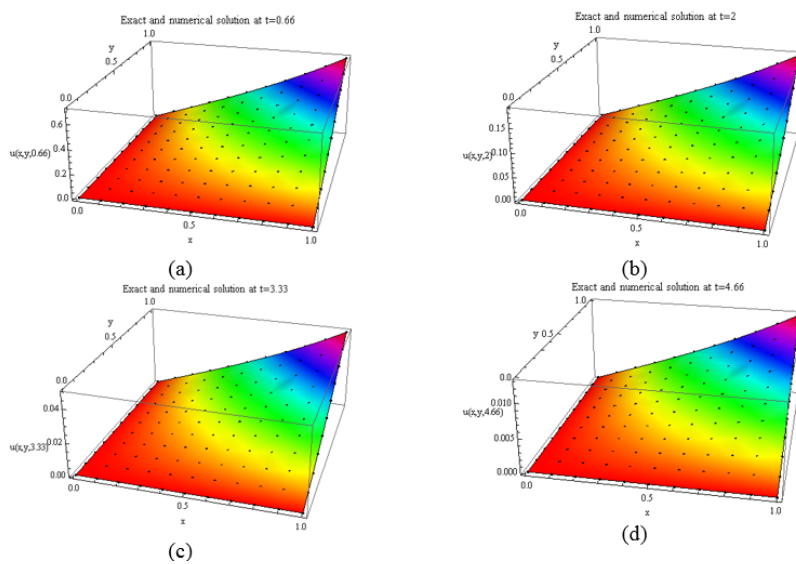


FIGURE 12. Exact and numerical solution throughout the domain in example 4.4 at various times: (a)  $t=0.66$ , (b)  $t=2$ , (c)  $t=3.33$ , (d)  $t=4.66$ ; Black points: numerical solution, 3D plot: exact solution.

TABLE 6. Comparison of the errors in example 4.4 with the corresponding results in [2].

t	$L_\infty$	$L_\infty$ reported in [18]	$L_2$	$L_2$ reported in [2]	$\epsilon$	RMS	RMS reported in [2]
1	$4.55 \times 10^{-6}$	$4.06 \times 10^{-5}$	$1.88 \times 10^{-5}$	$1.70 \times 10^{-4}$	$1.07 \times 10^{-5}$	$1.71 \times 10^{-6}$	$1.55 \times 10^{-5}$
2	$1.74 \times 10^{-6}$	$2.10 \times 10^{-5}$	$7.69 \times 10^{-6}$	$8.57 \times 10^{-5}$	$1.18 \times 10^{-5}$	$6.99 \times 10^{-7}$	$7.79 \times 10^{-6}$
3	$6.68 \times 10^{-7}$	$9.35 \times 10^{-6}$	$3.02 \times 10^{-6}$	$3.71 \times 10^{-5}$	$1.26 \times 10^{-5}$	$2.74 \times 10^{-7}$	$3.37 \times 10^{-6}$
4	$2.52 \times 10^{-7}$	$3.79 \times 10^{-6}$	$1.17 \times 10^{-6}$	$1.51 \times 10^{-5}$	$1.33 \times 10^{-5}$	$1.06 \times 10^{-7}$	$1.37 \times 10^{-6}$
5	$9.54 \times 10^{-8}$	$1.48 \times 10^{-6}$	$4.48 \times 10^{-7}$	$5.95 \times 10^{-6}$	$1.39 \times 10^{-5}$	$4.07 \times 10^{-8}$	$5.41 \times 10^{-7}$



## 5. CONCLUSION

This paper deals with solving two-dimensional hyperbolic telegraph equation. The solution method is based on a linear combination of shape functions arising in the moving least square (MLS) meshless method and local radial basis functions (LRBFs). In order to obtain higher accuracy in comparison with some previous works, the step-by-step differential quadrature method (DQM) is employed to solve the time-dependent ODE set. Moreover, a control parameter is introduced to be experimentally optimized to achieve minimum errors. Besides, numerical examples are provided to validate the applicability and efficiency of the method. The results demonstrate that by using the proposed method, higher accuracy, as well as less computational cost, are achieved. This proves that the presented method is a trustworthy and promising tool for investigating the telegraph equation.

## REFERENCES

- [1] C. A. Brebbia, J. C. F. Teles, and L.C. Wrobel, *Boundary element techniques*, Berlin: Springer, 1984.
- [2] M. Dehghan and A. Shokri, *A Meshless Method for Numerical Solution of a Linear Hyperbolic Equation with Variable Coefficients in Two Space Dimensions*, Numer. Methods Partial Differential Eq., *25* (2009), 494-506.
- [3] M. Dehghan and A. Ghesmati, *Combination of meshless local weak and strong (MLWS) forms to solve the two dimensional hyperbolic telegraph equation*, Engineering Analysis with Boundary Elements, *34* (2010), 324-336.
- [4] M. Dehghan and A. Shokri, *A numerical method for solving the hyperbolic telegraph equation*, Numerical Methods Partial Differential Eq, *24* (2008), 1080-1093.
- [5] M. Dehghan and R. Salehi, *A method based on meshless approach for the numerical solution of the two-space dimensional hyperbolic telegraph equation*, Math. Meth. Appl. Sci.
- [6] F. Gao and C. Chi, *Unconditionally stable difference schemes for a one-space-dimensional linear hyperbolic equation*, Appl. Math. Comput, *187* (2007), 1272-1276.
- [7] R. L. Hardy, *Multiquadric equations of topography and other irregular surfaces*, Journal of Geophysical Research, *76*(8) (1971), 1905-1915.
- [8] R. Jiwari, S. Pandit, and R. C. Mittal, *A differential quadrature algorithm to solve the two dimensional linear hyperbolic telegraph equation with Dirichlet and Neumann boundary conditions*, Applied Mathematics and Computation, *218* (2012), 7279-7294.
- [9] S. Kazem, J. A. Rad, and K. Parand, *A meshless method on non-Fickian flows with mixing length growth in porous media based on radial basis functions: a comparative study*, Comput. Math. Appl, *64* (2012), 399-412.
- [10] D. Levin, *The approximation power of moving least-squares*, Math. Comput, *67* (1998), 1517-1531.
- [11] K. M. Liew, C. Yumin, and S. Kitipornchai, *Boundary Element-free method (BEFM) for two-dimensional elastodynamic analysis using Laplace transforms*, Int. J. Numer. Methods. Eng, *64*(12) (2005), 1610-1627.
- [12] G. R. Liu, *Moving Beyond the Finite Element Method*, Second Edition, Boca Raton, CRC Press, 2009.
- [13] C. Micchelli, *Interpolation of scattered data: Distance matrices and conditionally positive definite functions*, Constructive Approximation, *2* (1986), 11-22.
- [14] S. Patankar, *Numerical heat transfer and fluid flow*, USA, Taylor & Francis, 1980.
- [15] B. Rezapour and M. A. Fariborzi Araghi, *Nanoparticle delivery through single walled carbon nanotube subjected to various boundary conditions*, Microsyst. Technol, *25* (2019), 1345-1356.
- [16] D. Rostamy, M. Emamjomea, and S. Abbasbandy, *A meshless technique based on the pseudospectral radial basis functions method for solving the two-dimensional hyperbolic telegraph equation*, Phys. J. Plus, (2017), 132-263.
- [17] K. Salkauskas, *Moving least squares interpolation with thin-plate splines and radial basis functions*, Comput. Math. Appl, *24* (1992), 177-185.
- [18] S. A. Sarra, *A numerical study of the accuracy and stability of symmetric and asymmetric RBF collocation methods for hyperbolic PDEs*, Numerical Method Partial Differential Equations, *24*(2) (2008), 670-686.
- [19] A. Shokri and M. Dehghan, *Meshless method using radial basis functions for the numerical solution of two-dimensional complex Ginzburg–Landau equation*, Comput. Model Eng. Sci, *34* (2012), 333-358.
- [20] A. Shokri and M. Dehghan, *A Not-a-Knot meshless method using radial basis functions and predictor–corrector scheme to the numerical solution of improved Boussinesq equation*, Comput. Phys. Commun, *181* (2010), 1990-2000.





- [21] C. Shu and K. S. Yao, *Block-marching in time with DQ discretization: an efficient method for time-dependent problems*, *Comput. Methods Appl. Mech. Eng.*, 191 (2002), 4587-4597.
- [22] G. D. Smith, *Numerical solution of partial differential equations: finite difference methods*, USA, Oxford University Press, 1986.
- [23] M. Tatari and M. Dehghan, *A method for solving partial differential equations via radial basis functions: application to the heat equation*, *Eng. Anal. Bound. Elem.*, 34 (2010), 206-212.
- [24] B. Wang, *A local meshless method based on moving least squares and local radial basis functions*, *Engineering Analysis with Boundary Elements*, 50 (2015), 395-401.
- [25] Z. Zhang, L. Yumin, K. M. Cheng, and Y. Y. Lee, *Analyzing 2D fracture problems with the improved element-free Galerkin method*, *Eng. Anal. Boundary Elem.*, 32 (2008), 241-250.
- [26] Z. Zhang, P Zhao, and K. M. Liew, *Improved element-free Galerkin method for two- dimensional potential problems*, *Eng. Anal. Boundary Elem.*, 33 (2009), 547-554.
- [27] O. C. Zienkiewicz and R. L. Taylor, *The finite element method*, Bristol, Butterworth, 2000.

