

Article

Solving a System of One-Dimensional Hyperbolic Delay Differential Equations Using the Method of Lines and Runge-Kutta Methods

S. Karthick ¹, V. Subburayan ^{1,*} and Ravi P. Agarwal ^{2,3,*}

¹ Department of Mathematics, College of Engineering and Technology, SRM Institute of Science and Technology, Kattankulathur 603203, Tamilnadu, India; karthickmaths007@gmail.com

² Department of Mathematics, Texas A&M University-Kingsville, Kingsville, TX 78363, USA

³ Department of Mathematics and Systems Engineering, Florida Institute of Technology, Melbourne, FL 32901, USA

* Correspondence: subburav@srmist.edu.in or subburayan123@gmail.com (V.S.); ravi.agarwal@tamuk.edu (R.P.A.)

Abstract: In this paper, we consider a system of one-dimensional hyperbolic delay differential equations (HDDEs) and their corresponding initial conditions. HDDEs are a class of differential equations that involve a delay term, which represents the effect of past states on the present state. The delay term poses a challenge for the application of standard numerical methods, which usually require the evaluation of the differential equation at the current step. To overcome this challenge, various numerical methods and analytical techniques have been developed specifically for solving a system of first-order HDDEs. In this study, we investigate these challenges and present some analytical results, such as the maximum principle and stability conditions. Moreover, we examine the propagation of discontinuities in the solution, which provides a comprehensive framework for understanding its behavior. To solve this problem, we employ the method of lines, which is a technique that converts a partial differential equation into a system of ordinary differential equations (ODEs). We then use the Runge–Kutta method, which is a numerical scheme that solves ODEs with high accuracy and stability. We prove the stability and convergence of our method, and we show that the error of our solution is of the order $O(\Delta t + \bar{h}^4)$, where Δt is the time step and \bar{h} is the average spatial step. We also conduct numerical experiments to validate and evaluate the performance of our method.

Keywords: maximum principle; Runge–Kutta method; cubic Hermite interpolation; method of lines; delay differential equations; stable method; convergence analysis

MSC: 65M12; 65M15; 35B50; 35F10; 65M06



Citation: Karthick, S.; Subburayan, V.; Agarwal, R.P. Solving a System of One-Dimensional Hyperbolic Delay Differential Equations Using the Method of Lines and Runge-Kutta Methods. *Computation* **2024**, *12*, 64. <https://doi.org/10.3390/computation12040064>

Academic Editor: Volodimir Simulik

Received: 23 February 2024

Revised: 18 March 2024

Accepted: 25 March 2024

Published: 27 March 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Delay Differential Equations (DDEs) are a type of differential equations that incorporate delays. They are useful for modeling complex systems that exhibit time- and space-dependent behaviors, such as epidemics, neuronal activity, and wave phenomena. By introducing delays into mathematical models, researchers can capture the effects of memory, feedback, or propagation that are present in real-world situations. This can lead to more accurate and realistic representations of the dynamics of the system, as well as better predictions and control strategies. Solving DDEs requires providing values for unknown functions within defined intervals, rather than just at initial points. This is because the value of an unknown function at a certain time depends on its value at some previous time, which reflects the delay effect. The delay can be constant or variable, and it can affect one or more terms in the equation. The presence of delays can significantly alter the qualitative

behavior of the model, such as its stability, equilibrium, and periodicity. For example, consider an epidemic model that uses generalized logistic dynamics to describe the growth and decline of the susceptible and infected populations. If the duration of infection is constant, the model can exhibit periodic solutions, meaning that the populations oscillate between high and low levels. However, if the duration of infection is delayed, meaning that it depends on the past state of the system, the model can exhibit more complex behaviors, such as bifurcations, chaos, and extinction. This shows how delays can affect the outcome of the epidemic and the effectiveness of interventions [1–3]. Another example is a model that describes the activity of neurons in the brain. Neurons are cells that communicate with each other through electrical and chemical signals. The signals travel along the axons and synapses of the neurons, which introduce delays in the transmission. The delays can vary depending on the distance, the type, and the state of the neurons. The model also accounts for stochastic effects, meaning that the signals are subject to random fluctuations and noise. These effects can result from the excitation or inhibition of the neurons, which depend on the input from other neurons or external stimuli. The model can capture the inherent variability and unpredictability of the neuronal system, as well as its ability to adapt and learn [4–7].

It investigates the intricate dynamic characteristics inherent in heat exchanges, a pivotal component extensively employed in the chemical industry for thermal management. It not only delves into the theoretical foundations but also provides illuminating real-world examples. By elucidating the mathematical intricacies, the reference serves as a valuable resource for understanding and optimizing the dynamic behavior of heat exchanges, offering a comprehensive exploration of their applications in chemical engineering [8].

In a broader mathematical context, hyperbolic partial differential equations (HPDEs) are often encountered. These equations arise in various fields and play a key role in understanding and describing wave phenomena. Examples of HPDEs include the wave equation and the telegraph equation, which are used to study classical physics phenomena such as water waves, sound waves, and seismic waves [9]. These equations can also incorporate delays, which can represent the effects of dispersion, dissipation, or diffusion. Advanced numerical methods have been developed to solve hyperbolic delay partial differential equations, with special attention given to techniques such as the Forward Time Backward Space (FTBS) and Backward Time Backward Space (BTBS) methods. These methods can handle the challenges posed by the delays, such as non-linearity, instability, and boundary conditions [10,11].

Researchers have devoted extensive efforts to the analysis of convergence and numerical treatments for both ordinary delay differential equations (DDEs) and hyperbolic partial differential equations (PDEs) [12,13]. When dealing with DDEs, which involve delays in the state variables, the use of numerical techniques poses a substantial challenge. A widely used approach is the method of lines, which involves discretizing the spatial derivatives in hyperbolic equations and obtaining systems of ordinary differential equations (ODEs). The solution of these ODEs can be efficiently obtained using Runge–Kutta methods, which improve the performance of the numerical solution process. This allows for the effective approximation of DDE solutions. However, these numerical techniques are not without drawbacks. The computational demands can be high, especially for large-scale or complex problems. Furthermore, the convergence analysis for both DDEs and hyperbolic PDEs is a difficult task that requires careful attention and computational effort [14,15].

These techniques are useful for solving various problems in science and engineering, but they often require significant computational resources [16]. Another topic that has been explored in depth is the maximum principle, which reveals the implications and practical significance of applying hyperbolic, parabolic, and elliptical differential equations to various phenomena [17,18].

Implicit Runge–Kutta (IRK) methods are numerical approaches for solving ordinary differential equations (ODEs). Unlike explicit methods, IRK tackles stiff ODEs by involving algebraic equations at each stage. This makes it adept at handling problems where certain components evolve at distinct rates. In IRK, each step necessitates solving a system of equations, often using iterative methods. Its application shines in scenarios with varying timescales, like chemical reactions

or electrical circuits. The method's formulation includes coefficients dictating its accuracy and stability. Higher-order methods enhance precision, but stability analysis is crucial for reliable solutions. IRK is commonly employed in stiff systems where explicit methods become computationally demanding. It finds use in solving systems of differential-algebraic equations and time-dependent partial differential equations, offering accurate and stable results. The theoretical understanding involves delving into the coefficients role, stability analysis, convergence properties and practical implementation using iterative solvers. Implicit Runge–Kutta (IRK) methods are a sophisticated class of time discretization schemes that stand out for their advanced features compared to other methods. These schemes have higher orders of accuracy, which means they can produce more accurate solutions with fewer steps. They also exhibit desirable stability properties, which means they can handle stiff or oscillatory problems without numerical instability. Moreover, they have effective error estimators, which can provide an estimate of the local or global error of the numerical solution. These features make IRK methods suitable for optimizing the time step needed to ensure stable and accurate solutions while maintaining the dispersion and dissipation at fixed levels [19–21]. Implicit Runge–Kutta (IRK) methods are a cutting-edge group of schemes within the range of advanced time discretization schemes [22]. On a different track, Runge–Kutta methods have been carefully developed to overcome the difficulties of solving systems of ordinary differential equations that arise from discretizing the spatial derivatives in hyperbolic equations using the method of lines approach. Cubic Hermite Interpolation is a technique used to create a smooth curve between given data points, where both the function values and their derivatives are known. It ensures that the resulting curve is continuous and differentiable. This method relies on cubic polynomials to connect adjacent data points, with coefficients determined to satisfy conditions at each point. The process involves setting up and solving a system of equations to find these coefficients. The interpolation polynomial preserves both the function values and their derivatives at each data point, creating a piecewise continuous and differentiable curve. Cubic Hermite Interpolation is commonly applied in computer graphics and computer-aided design to achieve accurate and visually pleasing interpolations of complex curves. Its versatility makes it valuable in situations where precise control over both function and derivative information is essential. The main goal of these methods is to accurately adjust the time step needed to achieve stable and accurate solutions while keeping the dispersion and dissipation constants unchanged. Implicit Runge–Kutta (IRK) methods, in particular, are known for their advanced features and efficiency. They have high orders of accuracy, which make them especially relevant in situations where precision is crucial. Their stability properties ensure the generation of reliable and robust numerical solutions, complemented by error estimators for rigorous accuracy assessment. The distinctive feature of IRK methods is their skillful use of the structure derived from carefully selected time discretization formulae, which enable the customization of the method based on the specific needs and characteristics of the problem. In this investigation, our computational endeavors were facilitated by a computer boasting an Intel Core i5 processor paired with 16 GB of RAM memory. The selection of this specific hardware configuration was driven by the need for an optimal blend of processing capability and memory capacity. The Intel Core i5 processor ensured efficient execution of our simulations, while the 16 GB RAM proved instrumental in handling sizable datasets. Notably, the computing time for our experiments was impressively brief, clocking in at an elapsed time of 1.312139 s. This swift computational performance underscores the efficiency of our chosen hardware setup and lays a foundation for the subsequent exploration of our research methodology and results.

This work is organized as follows: Section 2 contains the problem statement. Section 3 presents the maximum principle and its consequence of stability results. In Section 4, we describe the time semi-discrete problem using a backward Euler scheme in temporal direction. In Section 5, we discretize the spatial domain using fourth-order Runge–Kutta method with piecewise cubic Hermite interpolation. In Section 6, we present some numerical results and compare them with the analytical solutions. Finally, conclusions are presented in Section 7.

2. Problem Statement

Works from [10,11] motivate us to study the following problem: We find $\bar{u} = (u_1, u_2, \dots, u_n), u_1, u_2, \dots, u_n \in C(\bar{D}) \cap C^{(1,1)}(D)$ such that

$$\bar{\mathcal{L}}\bar{u} := \frac{\partial \bar{u}}{\partial t} + \mathbf{A} \frac{\partial \bar{u}}{\partial x} + \mathbf{B}\bar{u}(x, t) + \mathbf{C}\bar{u}(x - \delta, t) = \bar{f}(x, t), (x, t) \in D, \tag{1}$$

$$\bar{u} = \bar{\phi}(x, t), (x, t) \in [-\delta, 0] \times [0, T], \tag{2}$$

$$\bar{u}(x, 0) = \bar{u}_0(x), x \in [0, x_f], \bar{\phi}(0, 0) = \bar{u}_0(0), \tag{3}$$

where $\bar{\mathcal{L}} = (L_1, L_2, \dots, L_n)^T$, $\bar{f} = (f_1, f_2, \dots, f_n)^T$, $\bar{\phi} = (\phi_1, \phi_2, \dots, \phi_n)^T$, $\bar{u}_0 = (u_{0,1}, u_{0,2}, \dots, u_{0,n})^T$,

$$\mathbf{A} = \begin{bmatrix} a_{11}(x, t) & 0 & \dots & 0 \\ 0 & a_{22}(x, t) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{nn}(x, t) \end{bmatrix}, \mathbf{B} = \begin{bmatrix} b_{11}(x, t) & b_{12}(x, t) & \dots & b_{1n}(x, t) \\ b_{21}(x, t) & b_{22}(x, t) & \dots & b_{2n}(x, t) \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1}(x, t) & b_{n2}(x, t) & \dots & b_{nn}(x, t) \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} c_{11}(x, t) & c_{12}(x, t) & \dots & c_{1n}(x, t) \\ c_{21}(x, t) & c_{22}(x, t) & \dots & c_{2n}(x, t) \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1}(x, t) & c_{n2}(x, t) & \dots & c_{nn}(x, t) \end{bmatrix}.$$

The above Equation (1) can be written as

$$\bar{\mathcal{L}}\bar{u} := \begin{cases} \frac{\partial \bar{u}}{\partial t} + \mathbf{A} \frac{\partial \bar{u}}{\partial x} + \mathbf{B}\bar{u} = \bar{f} - \mathbf{C}\bar{\phi}(x - \delta, t), (x, t) \in [0, \delta] \times (0, T], \\ \frac{\partial \bar{u}}{\partial t} + \mathbf{A} \frac{\partial \bar{u}}{\partial x} + \mathbf{B}\bar{u} = \bar{f} - \mathbf{C}\bar{u}(x - \delta, t), (x, t) \in (\delta, x_f] \times (0, T], \end{cases} \tag{4}$$

$$\bar{u}(0, t) = \bar{\phi}(0, t), t \in [0, T], \bar{u}(x, 0) = \bar{u}_0(x), x \in [0, x_f], \tag{5}$$

where $a_{ii} \geq \alpha_i > 0$, $b_{ii} \geq \beta_i \geq 0$, and $b_{ij} \leq 0$, $i \neq j$ and $c_{ij} \leq 0$, $D = (0, x_f] \times (0, T]$ and $\delta \leq x_f$, x_f and δ are fixed constants. Functions a_{ij} , b_{ij} , and c_{ij} , are sufficiently differentiable on their domains.

Note: If all the coefficients a_{ij} , b_{ij} , c_{ij} , f_k are continuous functions of t on a compact set, then the above system has a solution; see [23].

3. Stability Analysis

In this section, we present the maximum principle for Problems (4) and (5) which is a system of partial differential equations with initial conditions. We also present a stability result that follows from the maximum principle.

Theorem 1 (Maximum Principle). Let $\bar{\psi} = (\psi_1, \psi_2, \dots, \psi_n), \psi_1, \psi_2, \dots, \psi_n \in C(\bar{D}) \cap C^{(1,1)}(D)$ be any function satisfying $\bar{\mathcal{L}}\bar{\psi} \geq \bar{0}$, $(x, t) \in \bar{D}$, $\bar{\psi}(0, t) \geq \bar{0}$, $t \in [0, T]$, $\bar{\psi}(x, 0) \geq \bar{0}$, $x \in [0, x_f]$. Then, $\bar{\psi}(x, t) \geq \bar{0}, \forall (x, t) \in \bar{D}$.

A consequence of the above theorem is the following stability result:

Theorem 2 (Stability Result). Let $\bar{\psi} = (\psi_1, \psi_2, \dots, \psi_n), \psi_1, \psi_2, \dots, \psi_n \in C(\bar{D}) \cap C^{(1,1)}(D)$ be any function; then,

$$|\psi_k(x, t)| \leq C_1 \max \left\{ \max_t \|\bar{\psi}(0, t)\|, \max_x \|\bar{\psi}(x, 0)\|, \max_k \left\{ \sup_{(x,t) \in D} \|\mathfrak{L}_k \bar{\psi}(x, t)\| \right\} \right\},$$

$$\forall (x, t) \in \bar{D}$$

where C_1 is a constant.

3.1. Propagation of Discontinuities

We recall that System (1)–(3) consists of n partial differential equations. Now, let us focus on the k th equation of the system and fix time variable t to a constant value which we can write as

$$\mathfrak{L}_k \bar{u} = \frac{\partial u_k}{\partial t} + a_{kk} \frac{\partial u_k}{\partial x} + \sum_{l=1}^n b_{kl} u_l(x, t) + \sum_{l=1}^n c_{kl} u_l(x - \delta, t) = f_k(x, t)$$

It is assumed that $\phi_k(0, t) = u_k(0, t), \forall k = 1, \dots, n$ and $\forall t \in [0, T]$. We differentiate the equation partially with respect to x ; then,

$$\begin{aligned} a_{kk} u_{k,xx} &= f_{k,x} - u_{k,xt} - a_{kk,x} u_{k,x} - \sum_{l=1}^n [b_{kl,x} u_l + b_{kl} u_{l,x}] \\ &\quad - \sum_{l=1}^n [c_{kl,x} u_l(x - \delta, t) + c_{kl} u_{l,x}(x - \delta, t)] \\ \lim_{x \rightarrow \delta^-} a_{kk} u_{k,xx} &= f_{k,x}(\delta^-, t) - u_{k,xt}(\delta^-, t) - a_{kk,x} u_{k,x}(\delta^-, t) \\ &\quad - \sum_{l=1}^n [b_{kl,x}(\delta^-, t) u_l(\delta^-, t) + b_{kl}(\delta^-, t) u_{l,x}(\delta^-, t)] \\ &\quad - \sum_{l=1}^n [c_{kl,x}(\delta^-, t) u_l(0^-, t) + c_{kl}(\delta^-, t) u_{l,x}(0^-, t)] \\ &= f_{k,x}(\delta^-, t) - u_{k,xt}(\delta^-, t) - a_{kk,x} u_{k,x}(\delta^-, t) \\ &\quad - \sum_{l=1}^n [b_{kl,x}(\delta^-, t) u_l(\delta^-, t) + b_{kl}(\delta^-, t) u_{l,x}(\delta^-, t)] \\ &\quad - \sum_{l=1}^n [c_{kl,x}(\delta^-, t) \phi_l(0^-, t) + c_{kl}(\delta^-, t) \phi_{l,x}(0^-, t)] \end{aligned}$$

and

$$\begin{aligned} \lim_{x \rightarrow \delta^+} a_{kk} u_{k,xx} &= f_{k,x}(\delta^+, t) - u_{k,xt}(\delta^+, t) - a_{kk,x} u_{k,x}(\delta^+, t) - \sum_{l=1}^n b_{kl,x}(\delta^+, t) u_l(\delta^+, t) \\ &\quad - \sum_{l=1}^n b_{kl}(\delta^+, t) u_{l,x}(\delta^+, t) - \sum_{l=1}^n c_{kl,x}(\delta^+, t) u_l(0^+, t) \\ &\quad - \sum_{l=1}^n c_{kl}(\delta^+, t) u_{l,x}(0^+, t) \\ &= f_{k,x}(\delta^+, t) - u_{k,xt}(\delta^+, t) - a_{kk,x} u_{k,x}(\delta^+, t) \\ &\quad - \sum_{l=1}^n [b_{kl,x}(\delta^+, t) u_l(\delta^+, t) + b_{kl}(\delta^+, t) u_{l,x}(\delta^+, t)] \\ &\quad - \sum_{l=1}^n c_{kl,x}(\delta^+, t) \phi_l(0^+, t) - \sum_{l=1}^n c_{kl}(\delta^+, t) \phi_{l,x}(0^+, t) \end{aligned}$$

Hence, $a_{kk}(\delta^+, t)u_{k,xx}(\delta^+, t) \neq a_{kk}(\delta^-, t)u_{k,xx}(\delta^-, t)$. Similarly, we can show that $u_{k,xxx}(2\delta^-, t) \neq u_{k,xxx}(2\delta^+, t)$, $k = 1, 2, \dots, n$. Points $\delta, 2\delta, 3\delta, \dots$ are primary discontinuities [2].

3.2. Derivative Bounds

From differential Equations (1)–(3) that are given, we can derive the following bounds for the derivative:

Theorem 3. Let \bar{u} be the exact solution of the system of partial differential Equations (1)–(3). Then, the bound of the derivatives satisfies the following estimate: $\left| \frac{\partial^{i+j} u_k}{\partial x^i \partial t^j}(x, t) \right| \leq C$, $0 \leq i + j \leq 2$, $k = 1, 2, \dots, n$.

4. Semi-Discretization in Temporal Direction

We divide the time interval $[0, T]$ into M subintervals of equal length, and we denote the resulting time grid by $\Omega_t^M = \{t_i = i * \Delta t\}_{i=0}^M$, where $\Delta t = \frac{T}{M}$ is the time step size. Using this grid, we apply a finite difference method to discretize partial differential Equations (1)–(3) in the time variable. We assume that the initial conditions are given by $u_k^0(x) = u_{k,0}(x)$, $x \in [0, x_f]$. Moreover, we define $u_k^j(x)$ as the approximate value of $u_k(x, t_j)$ at spatial point x and time level t_j .

$$\begin{aligned} \mathfrak{L}_k^j u_k^j(x) &:= D_{k,t} u_k^j(x, t_j) + a_{kk}(x, t_j) u_{k,x}^j(x, t_j) + \sum_{l=1}^n b_{kl}(x, t_j) u_k^j(x, t_j) \\ &\quad + \sum_{l=1}^n c_{kl} u_l^j(x - \delta, t_j) = f_k(x, t_j), \end{aligned} \tag{6}$$

$$u_k^j(x) = \phi_{k(x,t_j)}, \quad x \in [-\delta, 0], \quad j = 1, 2, \dots, M,$$

where $D_{k,t} u_k^j(x, t_j) = \frac{u_k(x, t_j) - u_k(x, t_{j-1})}{\Delta t}$.

For fixed time point at $t = t_j$, the equation presented earlier can be written in the following manner:

$$\begin{aligned} \Delta t a_{kk}(x, t_j) \frac{du_k^j}{dx}(x) + (1 + \Delta t \sum_{l=1}^n b_{kl}(x, t_j)) u_k^j(x, t_j) + \Delta t \sum_{l=1}^n c_{kl} u_l^j(x - \delta, t_j) \\ = \Delta t f_k(x, t_j) + u_k^{j-1}(x, t_{j-1}), \quad j = 1, 2, \dots, M. \end{aligned} \tag{7}$$

Lemma 1. Let u_k be the solution of (1)–(3) and $u_k^j(x)$ be the solution of (6) at $t = t_j$; then, $\|u_k - u_k^j\| \leq C \Delta t$.

Proof. We let $E_{k,j}(x) = u_k(x, t_j) - u_k^j(x)$, and we let x be fixed. Then,

$$\begin{aligned} \mathfrak{L}_k^j E_{k,j}(x) &= D_{k,t} E_{k,j}(x) + a_{kk}(x, t_j) E_{k,j}(x) + \sum_{l=1}^n b_{kl}(x, t_j) E_{k,j}(x) + \sum_{l=1}^n c_{kl,i}^j E_{k,j}(x - \delta, t_j) \\ &= D_t u_k(x, t_j) - D_{k,t} u_k^j(x, t_j) + a_{kk}(x, t_j) (u_{k,x}'(x, t_j) - u_k^j(x, t_j)) \\ &\quad + \sum_{l=1}^n b_{kl}(x, t_j) (u_k(x, t_j) - u_k^j(x, t_j)) + \sum_{l=1}^n c_{kl}(x, t_j) (u_{k,x}'(x - \delta, t_j) - u_k^j(x - \delta, t_j)); \end{aligned}$$

using ([24], Lemma 4.1), we can have $E_{k,j}(x) = (D_{k,t} - \frac{\partial}{\partial t})u_k(x, t_j)$ and $|E_{k,j}(x)| \leq O(\Delta t)$, $\forall j = 1, 2, \dots, M, \forall x$, which implies $\|E_{k,j}(x)\| \leq C(\Delta t)$; therefore, $\|u_k - u_k^j(x)\| \leq C(\Delta t)$. \square

5. Fully Discretized Problem

In this section, we apply spatial discretization to the semi-discrete problem defined by Equation (7). To achieve this, we use the fourth-order Runge–Kutta method to integrate the differential equations and piecewise cubic Hermite interpolation to approximate the solution over the interval of $[0, x_f]$.

Spatial Mesh Points

In Section 3.1, it is evident that $\delta, 2\delta, \dots$ serve as primary points of discontinuity. Consequently, we partition domain $[0, x_f]$ as follows: $[0, \delta], [\delta, 2\delta], \dots, [(r - 1)\delta, r\delta],$ and $[r\delta, x_f]$. Each of these sub-domains is further subdivided into $\frac{N}{r+1}$ segments. Thus, we define $\bar{\Omega}_x^N = \{x_i\}_{i=0}^N, x_i = x_{i-1} + h_i,$ where $x_i = x_{i-1} + h_i$ and $h_i = x_i - x_{i-1}$ for $i = 1, 2, \dots, N.$ The formulation of Problem (7) can be expressed as follows:

$$f_k^*(x, u_k^j, u_k^{j-1}, t_j) = \frac{1}{\Delta t a_{kk}(x, t_j)} [\Delta t f_k(x, t_j) - (1 + \sum_{l=1}^n b_{kl}(x, t_j) \Delta t) u_k^j(x, t_j) + u_k^{j-1}(x, t_{j-1}) - \sum_{l=1}^n c_{kl}(x, t_j) \Delta t u_k^{j,l}(x)]. \tag{8}$$

For details on the numerical approach for piecewise cubic Hermite interpolation used to interpolate solution $u(x_i - \delta, t_j)$ within range $[\delta, x_f],$ one may consult [25]. Employing the fourth-order Runge–Kutta method alongside piecewise cubic Hermite interpolation in the spatial domain over $[0, x_f],$ we obtain

$$U_{r,i+1}^j = U_{r,i}^j + \frac{1}{6} [K_{r,1} + 2K_{r,2} + 2K_{r,3} + K_{r,4}], \quad i = 0, 1, \dots, N - 1, \quad j = 1, 2, \dots, M, \tag{9}$$

where, $r = 1, 2, \dots, n,$

$$K_{r1} = \frac{1}{a_{kk}(x_i, t_j) \Delta t} \{ \Delta t f_r(x_i, t_j) + U_{r,i}^{j-1}(x_i, t_{j-1}) - (1 + \sum_{l=1}^n b_{rl}(x_i, t_j) \Delta t) U_{r,i}^j(x_i, t_j) - \sum_{l=1}^n c_{rl}(x_i, t_j) \Delta t U_{r,i}^{j,l}(x_i),$$

$$K_{r2} = \frac{1}{a_{kk}(x_i + \frac{h_i}{2}, t_j) \Delta t} \{ \Delta t f_r(x_i + \frac{h_{r,i}}{2}, t_j) + (U_{r,i}^{j-1} + \frac{K_{r1}}{2}) - (1 + \sum_{l=1}^n b_{rl}(x_i + \frac{h_{r,i}}{2}, t_j) \Delta t) (U_{r,i}^j + \frac{K_{r1}}{2}) - \sum_{l=1}^n c_{rl}(x_i + \frac{h_i}{2}, t_j) \Delta t (U_{r,i}^{j,l}(x_i + \frac{h_{r,i}}{2})) \},$$

$$K_{r3} = \frac{1}{a_{kk}(x_i + \frac{h_{r,i}}{2}, t_j) \Delta t} \{ \Delta t f_r(x_i + \frac{h_{r,i}}{2}, t_j) + (U_{r,i}^{j-1} + \frac{K_{r2}}{2}) - (1 + \sum_{l=1}^n b_{rl}(x_i + \frac{h_{r,i}}{2}, t_j) \Delta t) (U_{r,i}^j + \frac{K_{r2}}{2}) - \sum_{l=1}^n c_{rl}(x_i + \frac{h_{r,i}}{2}, t_j) \Delta t (U_{r,i}^{j,l}(x_i + \frac{h_{r,i}}{2})) \},$$

$$K_{r4} = \frac{1}{a_{kk}(x_i + h_{r,i}, t_j) \Delta t} \{ \Delta t f_r(x_i + h_{r,i}, t_j) + (U_{r,i}^{j-1} + K_{r3}) - (1 + \sum_{l=1}^n b_{rl}(x_i + h_{r,i}, t_j) \Delta t) (U_{r,i}^j + K_{r3}) - \sum_{l=1}^n c_{rl}(x_i + h_{r,i}, t_j) \Delta t (U_{r,i}^{j,l}(x_i + h_{r,i})) \},$$

$$U_{r,i}^{j,l}(x) = \begin{cases} \phi_l(x_i - \delta, t_j), & \text{if } (x_i - \delta) \leq 0, \\ U_{r,p}^j A_p(x) + U_{r,p+1}^j A_{p+1}(x) + B_p(x) f_r^*(x_p, U_{r,p}^j, U_{r,p}^{j-1}, t_j) & \\ + B_{p+1}(x) f_r^*(x_{p+1}, U_{r,p+1}^j, U_{r,p+1}^{j-1}, t_j), & \text{if } (x_i - \delta) > 0, \end{cases}$$

and p is an integer such that $x_i - \delta \in (x_p, x_{p+1})$.

$$A_p(x) = \left[1 - \frac{2(x - x_p)}{x_p - x_{p+1}} \right] \left[\frac{x - x_{p+1}}{x_p - x_{p+1}} \right]^2, \quad A_{p+1}(x) = \left[1 - \frac{2(x - x_{p+1})}{x_{p+1} - x_p} \right] \left[\frac{x - x_p}{x_{p+1} - x_p} \right]^2,$$

$$B_p(x) = \frac{(x - x_p)(x - x_{p+1})^2}{(x_p - x_{p+1})^2}, \quad B_{p+1}(x) = \frac{(x - x_{p+1})(x - x_p)^2}{(x_{p+1} - x_p)^2}, \quad p = i - N.$$

Theorem 4 ([2]). Let $u_k^j(x_i)$ be the solution of problem (7) and $U_{k,i}^j$ represent the solution of problem (9). Then, $|u_k(x_i, t_j) - U_{k,i}^j| \leq C(\bar{h}^4)$ is established, where C denotes a constant and \bar{h} = maximum of h_i .

This theorem offers an estimation of error for the above method.

Theorem 5. Let $u_{k,i}^j$ be the exact solution of (1) at point (x_i, t_j) and $U_{k,i}^j$ be the numerical solution of (9); then, $\|u_k(x_i, t_j) - U_{k,i}^j\| \leq C(\Delta t + \bar{h}^4)$.

Proof. Using Lemma 1 and Theorem 4, one can prove that

$$\|u_k - U_{k,i}^j\| = \|u_k - u_{k,i}^j + u_{k,i}^j - U_{k,i}^j\| \leq \|u_k - u_{k,i}^j\| + \|u_{k,i}^j - U_{k,i}^j\| \leq C(\Delta t + \bar{h}^4).$$

□

6. Numerical Examples

In order to demonstrate the effectiveness and accuracy of the numerical methods that we developed in this paper, we present two examples in this section. We compute the maximum error of our numerical solutions by using the half mesh principle, which is a technique for refining the mesh size and comparing the solutions on different grids.

$$E_k^{N,M} = \max_{i,j} |U_{k,i}^j(\Delta x, \Delta t) - U_{k,i}^j(\Delta x/2, \Delta t/2)|, \quad 0 \leq i \leq N, \quad 0 \leq j \leq M,$$

$$D_{k,x}^N = \max_M E_k^{N,M}, \quad D_{k,t}^M = \max_N E_k^{N,M},$$

$U_{k,i}^j(\Delta x, \Delta t)$ and $U_{k,i}^j(\Delta x/2, \Delta t/2)$ stand for the numerical outcomes at node (x_i, t_j) for mesh sizes $(\Delta x, \Delta t)$ and $(\Delta x/2, \Delta t/2)$, respectively.

Example 1. We consider the first-order hyperbolic delay differential equation.

$$\frac{\partial u_k}{\partial t} + \mathbf{A} \frac{\partial u_k}{\partial x} + \mathbf{B}u_k(x, t) + \mathbf{C}u_k(x - \delta, t) = 0, \quad (x, t) \in (0, 4] \times (0, 4], \quad (10)$$

$$\bar{u}(x, t) = (0, 0), \quad (x, t) \in [-\delta, 0] \times [0, 4], \quad (11)$$

$$u_k(x, 0) = x \exp(-(6x - 1)^2/4) \times (4 - x), \quad k = 1, 2, \quad x \in [0, 4], \quad (12)$$

$$a_{11} = \frac{1 + x^3 + t^4}{1 + 2tx + 4x^2}, \quad a_{22} = \frac{1 + x^3 + t^4}{1 + 4tx + 4x^2}, \quad b_{11} = 1, \quad b_{12} = \frac{1}{2}, \quad b_{21} = 1, \quad b_{22} = \frac{1}{2},$$

$$c_{11} = -2, \quad c_{12} = -1, \quad c_{21} = -2, \quad c_{22} = -1, \quad \delta = 1.$$

We assume that $\delta = 1$ in this case. The presence of the delay term results in additional wave propagation occurring in the forward direction of x at a δ unit distance. Figures 1 and 2 show the numerical solution obtained by the proposed method and the

exact solution, respectively. We can compare the solution curves at different time levels in Figures 3 and 4. Figures 5 and 6 display the maximum error between the numerical and exact solutions at each time level. The maximum pointwise error for each case is also given in Tables 1 and 2, where we can see that the error decreases as the mesh size decreases.

Table 1. U_1 —the component maximum error for Example 1 using the conditional method.

| N and $\delta = 1$ | | | | | | |
|--------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| M ↓ | 64 | 128 | 256 | 512 | 1024 | $D_{1,t}^M$ |
| 64 | 2.3551×10^{-3} | 1.1683×10^{-3} | 5.8188×10^{-4} | 2.9037×10^{-4} | 1.4505×10^{-4} | 5.8188×10^{-4} |
| 128 | 3.9796×10^{-3} | 1.9611×10^{-3} | 9.7352×10^{-4} | 4.8503×10^{-4} | 2.4209×10^{-4} | 9.7352×10^{-4} |
| 256 | 8.9886×10^{-3} | 4.2439×10^{-3} | 2.0649×10^{-3} | 1.0186×10^{-3} | 5.0590×10^{-4} | 8.9886×10^{-3} |
| 512 | 2.0254×10^{-2} | 9.2206×10^{-3} | 4.4098×10^{-3} | 2.1579×10^{-3} | 1.0675×10^{-3} | 9.2206×10^{-3} |
| 1024 | 4.3110×10^{-2} | 1.8317×10^{-2} | 8.5104×10^{-3} | 4.1089×10^{-3} | 2.0196×10^{-3} | 8.5104×10^{-3} |
| $D_{1,x}^N$ | 8.9886×10^{-3} | 9.2206×10^{-3} | 9.7352×10^{-4} | 4.8503×10^{-4} | 5.0590×10^{-4} | - |

Table 2. U_2 —the component maximum error for Example 1 using the conditional method.

| N and $\delta = 1$ | | | | | | |
|--------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| M ↓ | 64 | 128 | 256 | 512 | 1024 | $D_{2,t}^M$ |
| 64 | 3.7129×10^{-3} | 1.8074×10^{-3} | 8.9174×10^{-4} | 4.4293×10^{-4} | 2.2074×10^{-4} | 8.9174×10^{-4} |
| 128 | 7.2100×10^{-3} | 3.4536×10^{-3} | 1.6906×10^{-3} | 8.3643×10^{-4} | 4.1602×10^{-4} | 8.3643×10^{-4} |
| 256 | 1.3467×10^{-2} | 6.3518×10^{-3} | 3.0862×10^{-3} | 1.5217×10^{-3} | 7.5559×10^{-4} | 7.5559×10^{-4} |
| 512 | 2.2033×10^{-2} | 1.0136×10^{-2} | 4.8746×10^{-3} | 2.3920×10^{-3} | 1.1850×10^{-3} | 4.8746×10^{-3} |
| 1024 | 3.4673×10^{-2} | 1.4911×10^{-2} | 7.0017×10^{-3} | 3.4002×10^{-3} | 1.6763×10^{-3} | 7.0017×10^{-3} |
| $D_{2,x}^N$ | 7.2100×10^{-3} | 6.3518×10^{-3} | 8.9174×10^{-4} | 8.3643×10^{-4} | 7.5559×10^{-4} | - |

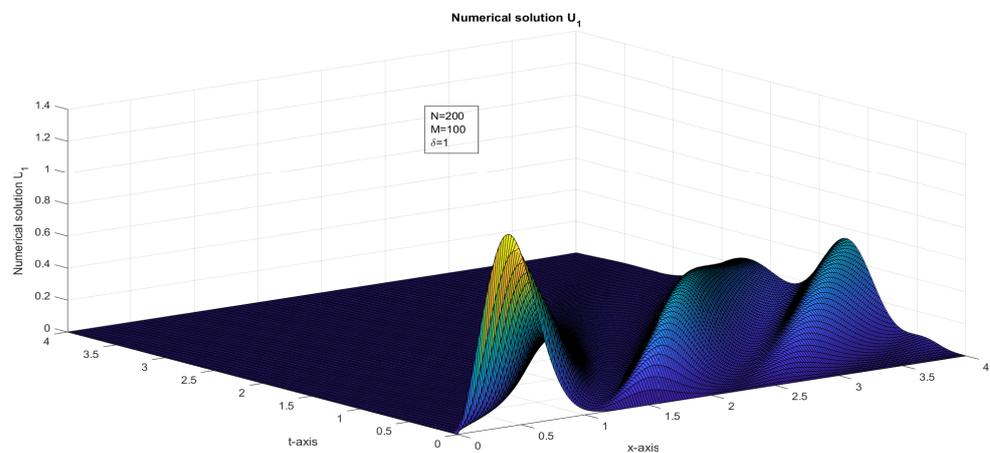


Figure 1. The surface plot of the U_1 —numerical solution of Example 1.

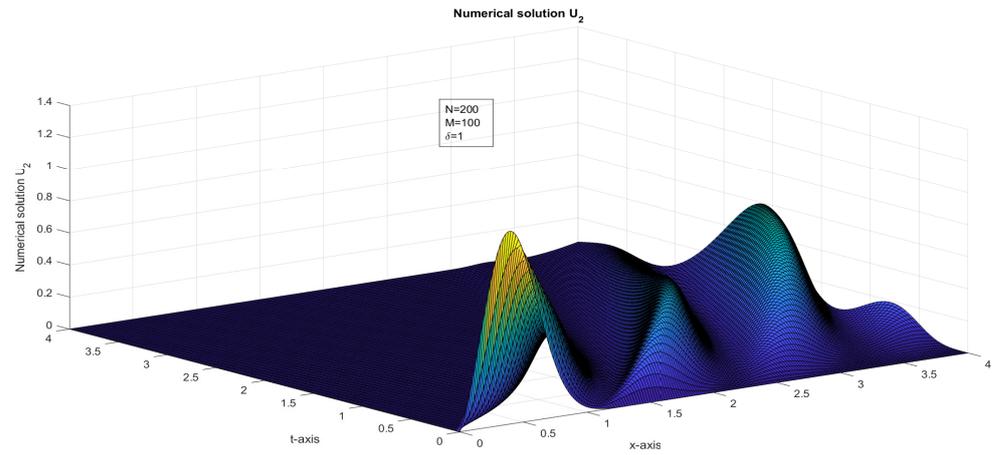


Figure 2. U_2 —numerical solution of Example 1 at different time levels.

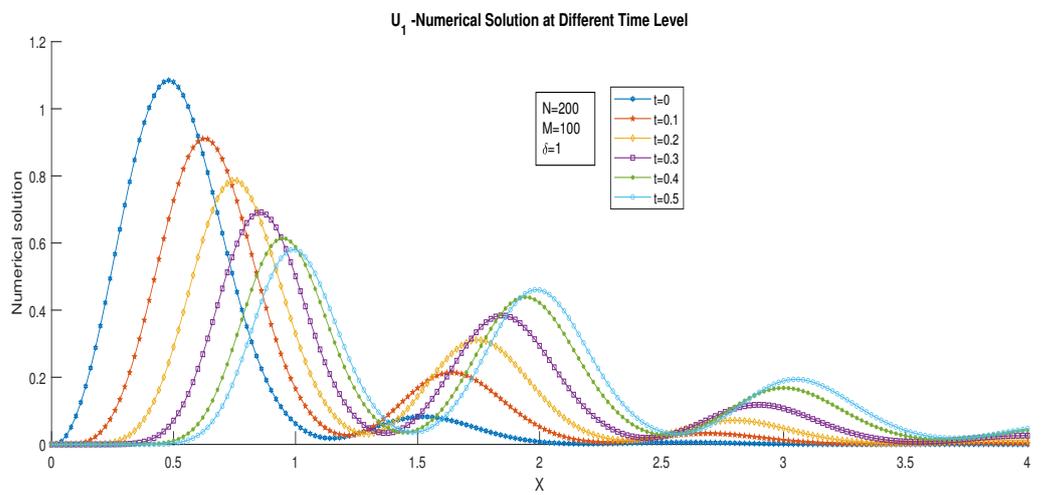


Figure 3. U_1 —numerical solution of Example 1 at different time level.

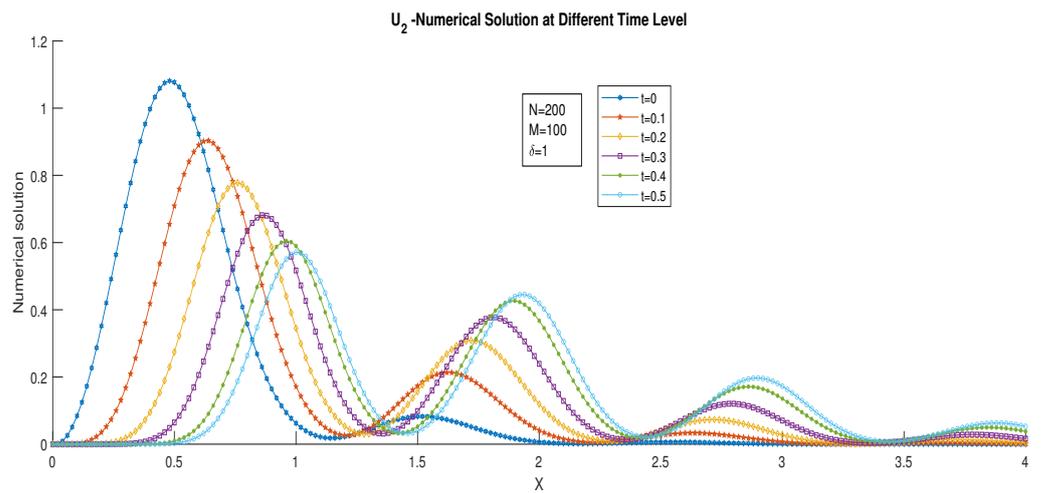


Figure 4. U_2 —numerical solution of Example 1 at different time levels.

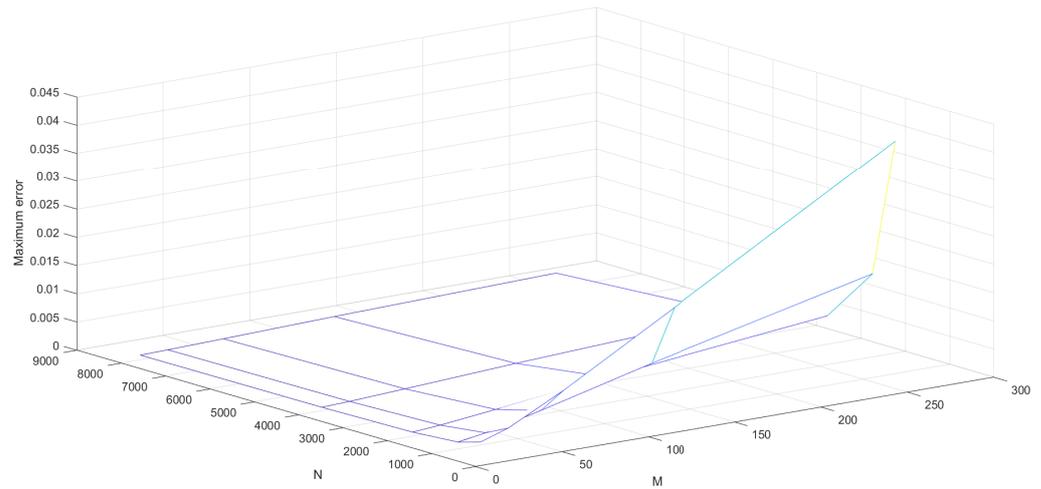


Figure 5. U_1 —Maximum point wise error of Example 1.

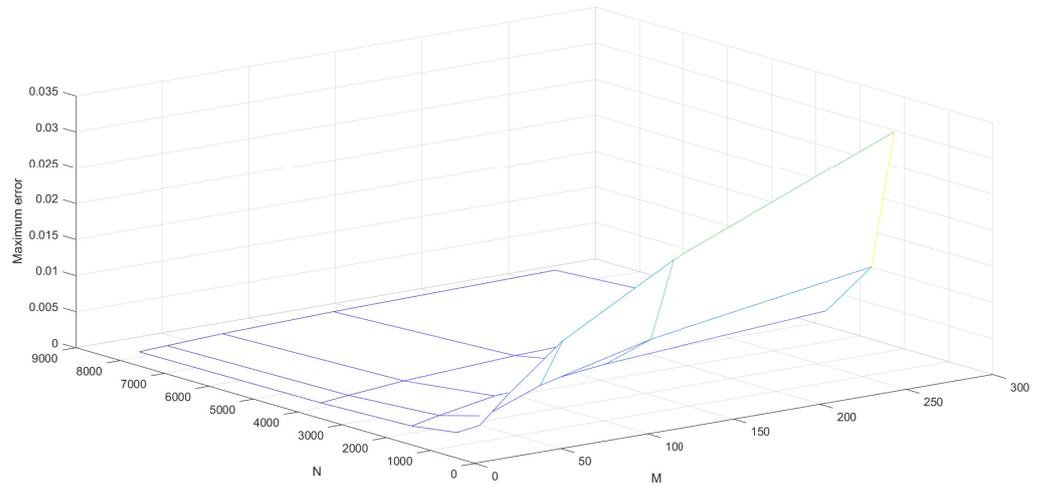


Figure 6. U_2 —maximum point wise error of Example 1.

Example 2. We consider Problems (10) and (11) with the following coefficients:

$$\bar{u}(x, 0) = \left[x \exp(-(6x - 1)^2/2), x \exp(-(4x - 1)^2/4) \right], x \in [0, 4], \tag{13}$$

$$a_{11} = \frac{3 + x^3 + t^4}{1 + 3tx + 4x^3}, a_{22} = \frac{4 + x^2 + t^4}{1 + 2tx + 4x^2}, b_{11} = \frac{1}{2}, b_{12} = -\frac{1}{4}, b_{21} = \frac{1}{2}, b_{22} = -\frac{1}{4},$$

$$c_{11} = 0, c_{12} = 0, c_{21} = 0, c_{22} = 0, \delta = 0.$$

We assume that $\delta = 0$ in this case. We observe that there is no additional wave propagation in the solution. Figures 7 and 8 show the numerical solution obtained by applying the proposed method. We can see the solution curves for different values of time in Figures 9 and 10, which demonstrates the accuracy and stability of the method. Figures 11 and 12 display the maximum error between the numerical and exact solutions at each time level. The maximum pointwise error for various values of M and N is also given in Tables 3 and 4, which confirm the convergence and consistency of the method.

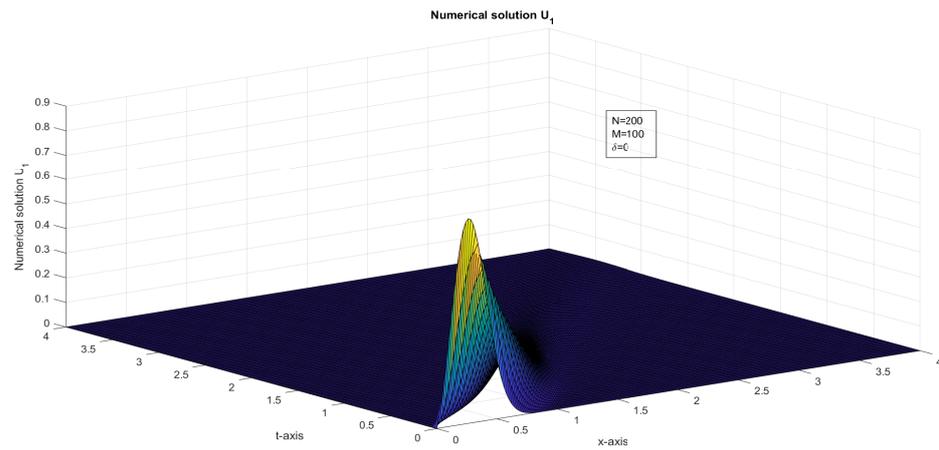


Figure 7. The surface plot of U_1 —numerical solution of Example 2.

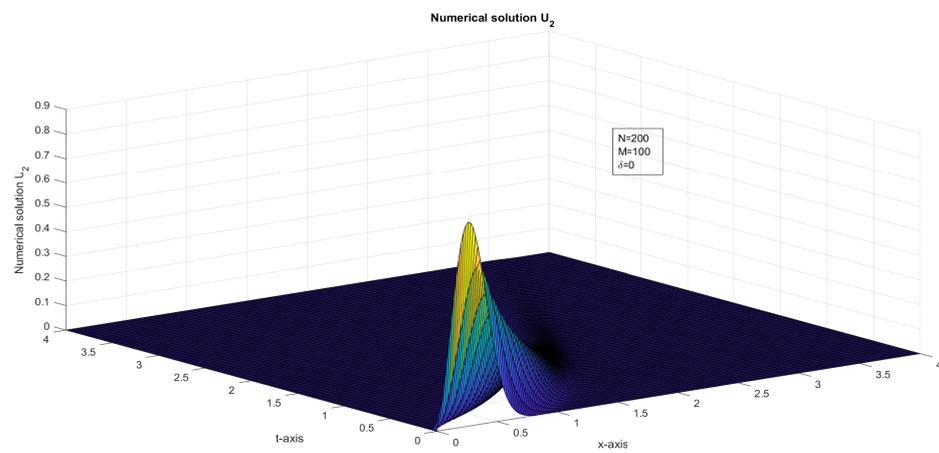


Figure 8. The surface plot of U_2 —numerical solution of Example 2.

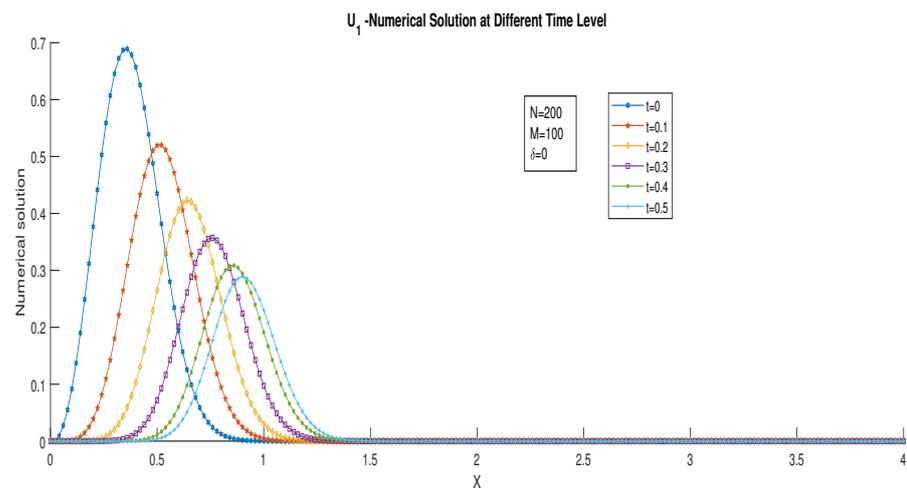


Figure 9. U_1 —numerical solution of Example 2 at different time levels.

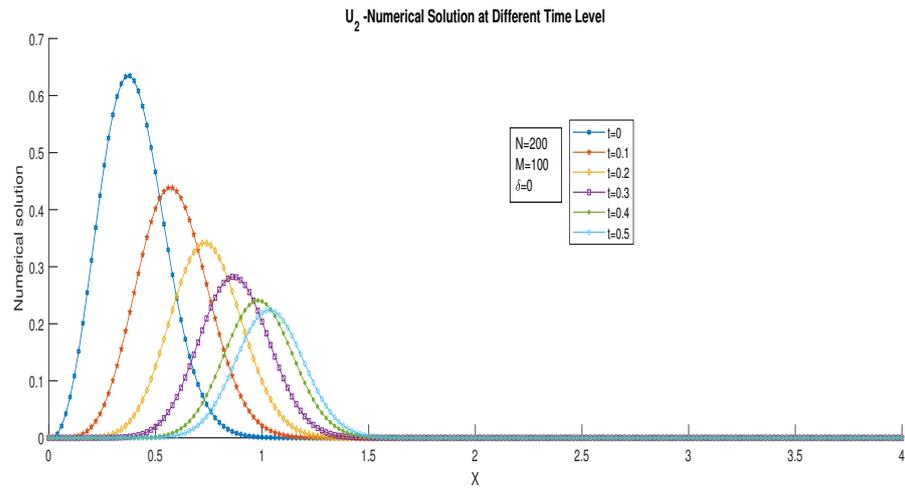


Figure 10. U_2 —numerical solution of Example 2 at different time levels.

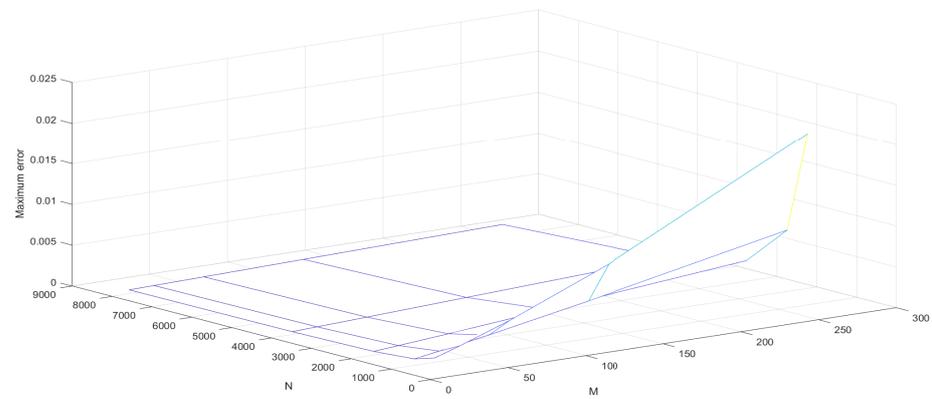


Figure 11. U_1 —maximum point wise error of Example 2.

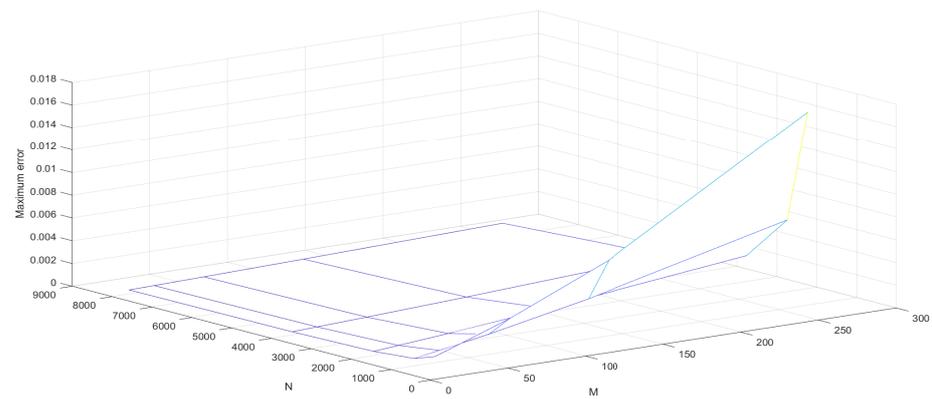


Figure 12. U_2 —maximum point wise error of Example 2.

Table 3. U_1 —the component maximum error for Example 2 using the conditional method.

| N and $\delta = 0$ | | | | | | |
|--------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| M ↓ | 64 | 128 | 256 | 512 | 1024 | $D_{1,t}^M$ |
| 64 | 1.5319×10^{-3} | 7.5986×10^{-4} | 3.7843×10^{-4} | 1.8885×10^{-4} | 9.4332×10^{-5} | 9.4332×10^{-5} |
| 128 | 2.5359×10^{-3} | 1.2540×10^{-3} | 6.2355×10^{-4} | 3.1093×10^{-4} | 1.5525×10^{-4} | 6.2355×10^{-4} |
| 256 | 4.7023×10^{-3} | 2.2893×10^{-3} | 1.1298×10^{-3} | 5.6127×10^{-4} | 2.7973×10^{-4} | 5.6127×10^{-4} |
| 512 | 9.7474×10^{-3} | 4.5751×10^{-3} | 2.2226×10^{-3} | 1.0958×10^{-3} | 5.4407×10^{-4} | 9.7474×10^{-3} |
| 1024 | 2.2033×10^{-2} | 9.5061×10^{-3} | 4.4622×10^{-3} | 2.1665×10^{-3} | 1.0678×10^{-3} | 9.5061×10^{-3} |
| $D_{1,x}^N$ | 9.7474×10^{-3} | 9.5061×10^{-3} | 6.2355×10^{-4} | 5.6127×10^{-4} | 9.4332×10^{-5} | - |

Table 4. U_2 —the component maximum error for Example 2 using the conditional method.

| N and $\delta = 0$ | | | | | | |
|--------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| M ↓ | 64 | 128 | 256 | 512 | 1024 | $D_{2,t}^M$ |
| 64 | 1.2625×10^{-3} | 6.2681×10^{-4} | 3.1231×10^{-4} | 1.5588×10^{-4} | 7.7871×10^{-5} | 7.7871×10^{-5} |
| 128 | 2.0230×10^{-3} | 1.0021×10^{-3} | 4.9875×10^{-4} | 2.4881×10^{-4} | 1.2426×10^{-4} | 4.9875×10^{-4} |
| 256 | 3.5823×10^{-3} | 1.7576×10^{-3} | 8.7066×10^{-4} | 4.3331×10^{-4} | 2.1616×10^{-4} | 8.7066×10^{-4} |
| 512 | 7.3619×10^{-3} | 3.5035×10^{-3} | 1.7114×10^{-3} | 8.4594×10^{-4} | 4.2067×10^{-4} | 8.4594×10^{-4} |
| 1024 | 1.7733×10^{-2} | 7.7339×10^{-3} | 3.6528×10^{-3} | 1.7789×10^{-3} | 8.7813×10^{-4} | 8.7813×10^{-4} |
| $D_{2,x}^N$ | 7.3619×10^{-3} | 7.7339×10^{-3} | 8.7066×10^{-4} | 8.4594×10^{-4} | 8.7813×10^{-4} | - |

7. Conclusions

This article deals with the system of first-order hyperbolic delay differential equations which include spatial delay terms. This system can model various phenomena in science, such as wave propagation, population dynamics, and neural networks. To obtain numerical solutions for this system, we adopt a semi-discretization technique in the time direction, using a backward finite difference formula on a uniform grid. This method reduces the original system to a set of algebraic equations, which have a truncation error of order $O(\Delta t)$ for a fixed x . We then discretize the resulting system further by applying the fourth-order Runge–Kutta method, which is a well-known and efficient method for solving ordinary differential equations. We also use piecewise cubic Hermite interpolation to approximate the spatial delay terms. This method offers us an overall error of order $O(\Delta t + \bar{h}^4)$, where Δt is the time step and \bar{h} is the average spatial step. We discuss how to handle Problem (1) with both smooth and non-smooth data functions, and we investigate the characteristics of the solutions. The theoretical results are also verified by numerical examples (Figures 1–12) and Tables 1–4. From these examples, we observe that, for fixed integer M , increasing the value of N leads to a decrease in the maximum error. On the other hand, for a fixed N , increasing the value of M causes the maximum error to increase. We also notice the conditional stability of the method, which requires that $\bar{h} < 1$, further the method is stable $h_i \leq C\Delta t$.

Author Contributions: Conceptualization, S.K., V.S. and R.P.A.; methodology, S.K., V.S. and R.P.A.; formal analysis, S.K., V.S. and R.P.A.; investigation, S.K., V.S. and R.P.A.; writing—original draft preparation, S.K., V.S. and R.P.A.; writing—review and editing, R.P.A., S.K. and V.S.; supervision, V.S. and R.P.A.; project administration, S.K., V.S. and R.P.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: The authors acknowledges with sincere thanks to DST-SERB for providing computational facilities from the project TAR/2021/000053. The authors wish to acknowledge the referees for their valuable comments and suggestions, which helped to improve the presentation.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Smith, H.L. *An Introduction to Delay Differential Equations with Applications to the Life Sciences*; Springer: New York, NY, USA, 2011; Volume 57, pp. 119–130.
2. Bellen, A.; Zennaro, M. *Numerical Methods for Delay Differential Equations*; Oxford University Press: Oxford, UK, 2003.
3. Kuang, Y. *Delay Differential Equations with Applications in Population Dynamics*; Academic Press: Cambridge, MA, USA, 1993.
4. Stein, R.B. A theoretical analysis of neuronal variability. *Biophys. J.* **1965**, *5*, 173–194. [[CrossRef](#)]
5. Stein, R.B. Some models of neuronal variability. *Biophys. J.* **1967**, *7*, 37–68. [[CrossRef](#)]
6. Holden, A.V. *Models of the Stochastic Activity of Neurons*; Springer Science and Business Media: Berlin/Heidelberg, Germany, 2013.
7. Rai, P.; Sharma, K.K. Numerical study of singularly perturbed differential–difference equation arising in the modeling of neuronal variability. *Comput. Math. Appl.* **2012**, *63*, 118–132. [[CrossRef](#)]
8. Varma, A.; Morbidelli, M. *Mathematical Methods in Chemical Engineering*; Oxford University Press: Oxford, UK, 1997.
9. Banasiak, J.; Mika, J.R. Singularly perturbed telegraph equations with applications in the random walk theory. *J. Appl. Math. Stoch. Anal.* **1998**, *11*, 9–28. [[CrossRef](#)]
10. Sharma, K.K.; Singh, P. Hyperbolic partial differential–difference equation in the mathematical modelling of neuronal firing and its numerical solution. *Appl. Math. Comput.* **2008**, *201*, 229–238.
11. Singh, P.; Sharma, K.K. Numerical solution of first-order hyperbolic partial differential–difference equation with shift. *Numer. Methods Partial Differ. Equ.* **2010**, *26*, 107–116. [[CrossRef](#)]
12. Al-Mutib, A.N. Stability properties of numerical methods for solving delay differential equations. *J. Comput. Appl. Math.* **1984**, *10*, 71–79. [[CrossRef](#)]
13. Loustau, J. *Numerical Differential Equations: Theory and Technique, ODE Methods, Finite Differences, Finite Elements and Collocation*; World Scientific: Singapore, 2016.
14. Warming, R.F.; Hyett, B.J. The modified equation approach to the stability and accuracy analysis of finite-difference methods. *J. Comput. Phys.* **1974**, *14*, 159–179. [[CrossRef](#)]
15. Süli, E.; Mayers, D.F. *An Introduction to Numerical Analysis*; Cambridge University Press: Cambridge, UK, 2003.
16. Singh, S.; Patel, V.K.; Singh, V.K. Application of wavelet collocation method for hyperbolic partial differential equations via matrices. *Appl. Math. Comput.* **2018**, *320*, 407–424. [[CrossRef](#)]
17. Protter, M.H.; Weinberger, H.F. *Maximum Principles in Differential Equations*; Springer Science and Business Media: Berlin/Heidelberg, Germany, 2012.
18. Bainov, D.D.; Kamont, Z.; Minchev, E. Comparison principles for impulsive hyperbolic equations of first order. *J. Comput. Appl. Math.* **1995**, *60*, 379–388. [[CrossRef](#)]
19. Jain, M.K.; Iyengar, S.R.K.; Saldanha, J.S.V. Numerical solution of a fourth-order ordinary differential equation. *J. Eng. Math.* **1977**, *11*, 373–380. [[CrossRef](#)]
20. Rana, M.M.; Howle, V.E.; Long, K.; Meek, A. A New Block Preconditioner for Implicit Runge–Kutta Methods for Parabolic PDE Problems. *SIAM J. Sci. Comput.* **2021**, *43.5*, 475–495. [[CrossRef](#)]
21. Takei, Y.; Iwata, Y. Numerical Scheme Based on the Implicit Runge–Kutta Method and Spectral Method for Calculating Nonlinear Hyperbolic Evolution Equations. *Axioms* **2022**, *11*, 28. [[CrossRef](#)]
22. Southworth, B.S.; Krzysik, O.; Pazner, W.; Sterck, H.D. Fast Solution of Fully Implicit Runge–Kutta and Discontinuous Galerkin in Time for Numerical PDEs, Part I: The Linear Setting. *SIAM J. Sci. Comput.* **2022**, *44*, A416–A443. [[CrossRef](#)]
23. Mizohata, S.; Murthy, M.V.; Singbal, B.V. *Lectures on Cauchy Problem*; Tata Institute of Fundamental Research: Mumbai, India, 1965; Volume 35.
24. Miller, J.J.H.; O’Riordan, E.; Shishkin, G.I. *Fitted Numerical Methods for Singular Perturbation Problems: Error Estimates in the Maximum Norm for Linear Problems in One and Two Dimensions*; World Scientific: Singapore, 2012.
25. Subburayan, V.; Ramanujam, N. An asymptotic numerical method for singularly perturbed convection–diffusion problems with a negative shift. *Neural Parallel Sci. Comput.* **2013**, *21*, 431–446.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.