Communications in Mathematics and Applications
Vol. 16, No. 2, pp. 679–691, 2025
ISSN 0975-8607 (online); 0976-5905 (print)
Published by RGN Publications
DOI: 10.26713/cma.v16i2.2912

http://www.rgnpublications.com

Research Article

Higher-Order Numerical Techniques for Solving the Nonlinear Fisher Equation are Based on the Runge-Kutta Method

Richa Kumari[1], Amrita[2] and Vikash Vimal*[3]

[1] *Department of Physics, Patna University, Ashok Rajpath, Patna 800005, Bihar, India*
[2] *Department of Physics, Patna Women's College (Autonomous), Patna University, Patna 800005, Bihar, India*
[3] *Department of Mathematics and Computing Technology, National Institute of Technology, Patna 800005, Bihar, India*
***Corresponding author:** vikashvimalnitjsr@gmail.com

Received: October 16, 2024 **Revised:** March 22, 2025 **Accepted:** April 6, 2025

Abstract. This paper presents higher-order numerical methods for solving nonlinear Fisher equations. These types of equations arise in various fields of sciences and engineering, the main application of this equation has been found in the biomedical sciences. The solution of this equation helps to determine the size of the brain tumor. In this paper explores the utilization of advanced numerical techniques, such as the method of lines and higher-order strong stability preserving schemes of order four and stage seven, to approximate solutions to the Fisher equation with higher-order accuracy. These schemes are explicitly designed and easy to implement, especially for addressing nonlinear problems. Their stability-preserving nature ensures only mild restrictions on time steps. This scheme is then tested on two examples and the results show that it is more efficient methods and requires less computing time. Various test problems are examined to verify the scheme's performance, including a comparison of l_2 and l_∞ errors with the exact solution, leading to high accuracy.

Keywords. Fisher's problems, Method of lines, Finite difference methods, Strong stability preserving Runge-Kutta methods

Mathematics Subject Classification (2020). 26A33, 65L60, 42C05, 65L05

1. Introduction

We consider the well-known one-dimensional nonlinear Fisher equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \alpha u(1-u), \quad 0 \le x \le 1,\ t > 0, \tag{1.1}$$

where $u(x,t)$ represents the population density.

The Fisher equation was initially formulated in 1937, and equation (1.1) is referenced from (Fisher [9]). The details of the analysis and explanation of equation (1.1) can be found in [15]. Due to this it is called the *Fisher-Kolmogorov-Petrovsky-Piscounov* (Fisher-KPP) equation. However, it is widely known as Fisher equation. Over time, the Fisher equation has become really important for studying how things spread out, like helpful genes, populations, and how stuff moves in nature and biology. Many scientists are studying different things using the Fisher equation, e.g., Alshammari and Mashat [2], Ammerman and Cavalli-Sforza [3], Bramson [6], Canosa [7], Frank-Kamenetskii [10], Shah [24], Tang and Weber [29], and Tyson and Brazhnik [30]. The Fisher equation's math properties and discussions, along with numerous of numerical methods, are explained in the literature with references for more information, see, Bastani and Salkuyeh [4], Chandrakera *et al*. [8], Jiwari and Mittal [14], Macías-Díaz *et al*. [16], Mickens [17], Mittal and Jiwari [18], Mittal and Kumar [19], Qiu and Sloan [22], Verma *et al*. [31], Wang [35]. Research works of Parambu *et al*. [21], Shampine [25], and Shu [26] provide useful information about the background and uses of this equation in different scientific areas.

In mathematical science, we develop a method for obtaining numerical solutions of a one-dimensional nonlinear reaction-diffusion equation using the SSPRK-74 technique (Gottlieb *et al*. [11], Spiteri and Ruuth [27]). We achieve this by transforming the partial differential equation into an ordinary differential equation in time through the application of the method of lines (Oymak and Selçuk [20]). The method of lines, which is a technique for finding numerical solutions of partial differential equations, plays an essential role in preserving the accuracy and stability of the developing solution. The ordinary differential equations resulting from the discretization of the Navier-Stokes equations are integrated using an implicit method, Adams-Moulton, which is integrated with the widely recognized ODE solver Hindmarsh [13].

In this paper, we introduce a numerical approach for solving Fisher's equation. We combine the *Method of Lines* (MOL) in spatial dimensions with the strong stability preserving Runge-Kutta method (SSP-RK74) in time dimensions. SSPRK-74 method is a advanced techniques for solving high-order time discretization methods. In Section 2, we introduce Fisher equation in one dimension with initial and boundary conditions. In Section 3, involves semi-discretizing the derived equation in the spatial dimension using MOL and fully discretizing it by implementing the SSP-RK74 method on the resulting ODE system. In Section 4, we describe numerical experiments of test examples and compare the numerical solutions with a few existing methods. Our method demonstrates higher accuracy compared to the existing methods. In Section 5, conclusion. This paper presents an efficient method for addressing Fisher's equation, providing valuable perspectives for various scientific and engineering applications.

2. Problem Statement

We consider the well-known one-dimensional nonlinear Fisher equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \alpha u(1-u), \quad 0 \le x \le 1,\ t > 0, \tag{2.1}$$

with initial condition

$$u(x,0) = u_0(x), \quad 0 \le x \le 1,$$

and the boundary conditions

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

$$u(1,t) = f_2(t), \quad 0 \le t \le T.$$

In this context, α represents the reactive factor, T signifies the final time, and $u_0(x)$, f_1, and f_2 denote given functions that are sufficiently smooth, collectively defining initial and boundary conditions for the mathematical model.

3. Numerical Scheme

The *Method of Lines* (MOL) is a widely recognized technique used for solving time-dependent partial differential equations. Initially, the partial differential equations are converted into ordinary differential equations using the MOL. Then the set of ordinary differential equations is solved by applying the SSP-RK74 scheme for integration. To discretize the solution domain for equation (2.1), we apply a uniform mesh approach. The spatial interval $[0,1]$ is divided into M equal sub-intervals, each with a width of Δx, where Δx is calculated as $\Delta x = \frac{1}{M}$. We then define spatial points x_m as $x_m = m\Delta x$ for m is ranging from 0 to M.

3.1 Semi-discretization: Method of Lines (MOL)

Rothe [23] introduced the *Method of Lines* (MOL), and in a subsequent works by Bonkile [5], Parambu *et al*. [21] utilized MOL to transform the PDEs into a set of ODEs, effectively addressing the Burger's equation and Stefan problem. Unsteady nonlinear partial differential equation undergoes spatial discretization to create a semi-discrete MOL scheme. This includes discretizing the reaction term $\frac{\partial u}{\partial t}$ with a second-order central method and using central difference to discretize the diffusion term $\frac{\partial^2 u}{\partial x^2}$,

$$\frac{\partial u}{\partial x} \approx \frac{u_{m+1}(t) - u_{m-1}(t)}{2h}, \tag{3.1}$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{m+1}(t) - 2u_m(t) + u_{m-1}(t)}{h^2}, \quad h = \Delta x, \tag{3.2}$$

$$\frac{du_m}{dt} = \left(\frac{u_{m+1}(t) - 2u_m(t) + u_{m-1}(t)}{h^2}\right) + (\alpha u_m(1-u_m)), \tag{3.3}$$

where $m = 1,2,3,\dots,M-1$.

This can be expressed in the form of a discrete operator, the right-hand side of equation (3.3),

$$\frac{du_m}{dt} = L(u_m), \tag{3.4}$$

where m is ranging from 1 to $M-1$, and L is basically a nonlinear difference operator.

3.1.1 SSP-RK74

Table 1. Butcher tableau of SSP-RK74 scheme (Spiteri and Ruuth [27])

$a_{i,k}$	1.00000000000000						
	0.20161507213829	0.79838492786171					
	0.19469598207921	0.00000000000000	0.80530401792079				
	0.58143386885601	0.00000000000000	0.00000000000000	0.41856613114399			
	0.01934367892154	0.00000000000000	0.00000000000000	0.00000000000000	0.98065632107846		
	0.00000000000000	0.00000000000000	0.00000000000000	0.00000000000000	0.00000000000000	1.00000000000000	
	0.06006304558847	0.00000000000000	0.30152730794242	0.10518998496676	0.01483791154585	0.00000000000000	0.51838174995650
$b_{i,k}$	0.3011872706068						
	0.00000000000000	0.24040865318216					
	0.00000000000000	0.00000000000000	0.24249212077315				
	0.00000000000000	0.00000000000000	0.00000000000000	0.12603810060080			
	0.00000000000000	0.00000000000000	0.00000000000000	0.00000000000000	0.29529398308716		
	0.00000000000000	0.00000000000000	0.00000000000000	0.00000000000000	0.00000000000000	0.30111872706068	
	0.00000000000000	0.00000000000000	0.09079551914158	0.02888359354880	0.00000000000000	0.00000000000000	0.15609445267839

The purpose of SSP-RK74 is to achieve high-order accuracy in time integration while maintaining strong stability properties. The SSP-RK74 scheme utilizes seven stages ($s = 7$) and achieves a fourth-order accuracy ($k = 4$). The primary characteristic of SSP methods is that the number of stages ($s = 7$) exceeds the order ($k = 4$) of accuracy in terms of the time variable for the method (Gottlieb *et al*. [11]). Let's express an s-stage explicit Runge-Kutta method in the following manner,

$$U^{(0)} = U^n, \tag{3.5}$$

$$U^{(i)} = \sum_{k=0}^{i-1} (a_{i,k} U^{(k)} + \Delta t b_{i,k} L(U^{(k)})), \quad i = 1,2,3,\dots,s, \tag{3.6}$$

$$U^{n+1} = U^{(m)}. \tag{3.7}$$

The SSP-RK74 scheme is characterized by coefficients $a_{i,k}$ that satisfy the conditions $a_{i,k} \geq 0$ and $a_{i,k} = 0$ only if $b_{i,k} = 0$ (Shu [26]). This scheme also possesses a *Courant-Friedrichs-Lewy* (CFL) coefficient of 3.32094921415661. Additionally, it's required that the sum of coefficients $\sum_{k=0}^{i-1} a_{i,k} = 1$ holds for $i = 1,2,3,\dots,s$. To descretize the temporal domain $[0,T]$ into N equivalent sub-intervals with a uniform mesh size, we assume $\Delta t = T/N = k$ and utilize $t_n = n\Delta t$. We perform the integration of equation (3.4) from t_n to $t_n + \Delta t$ using the following steps for $n = 0,1,2\dots,N$, resulting in the complete determination of the solution $u(x,t)$ at a specific time level. Table (1) gives the values of a_{ik} and b_{ik} coefficients,

$$u_m^{(0)} = u_m^n, \tag{3.8}$$

where u_m^n is a initial condition

$$\begin{aligned} u_m^{(1)} &= a_{10} u_m^{(0)} + \Delta t b_{10} L(u_m^{(0)}) \\ &= u_m^{(0)} + \Delta t (0.3011872706068) L(u_m^{(0)}) \\ &= u_m^{(0)} + k(0.3011872706068) \left[\frac{u_{m+1}^{(0)} - 2u_m^{(0)} + u_{m-1}^{(0)}}{h^2} + \alpha u_m^{(0)} (1 - u_m^{(0)}) \right], \end{aligned} \tag{3.9}$$

$$u_m^{(2)} = \sum_{k=0}^{1} (a_{2k} u_m^{(0)} + \Delta t b_{2k} L(u_m^{(k)}))$$

$$= (0.20161507213829)u_m^{(0)} + (0.79838492786171)u_m^{(1)} + \Delta t(0.24040865318216)L(u_m^{(1)})$$
$$= (0.20161507213829)u_m^{(0)} + (0.79838492786171)u_m^{(1)}$$
$$+ k(0.24040865318216)\left[\frac{u_{m+1}^{(1)} - 2u_m^{(1)} + u_{m-1}^{(1)}}{h^2} + \alpha u_m^{(1)}(1-u_m^{(1)})\right], \tag{3.10}$$

$$u_m^{(3)} = \sum_{k=0}^{2}(a_{3k}u_m^{(k)} + \Delta t b_{3k}L(u_m^{(k)}))$$
$$= 0.19469598207921u_m^{(0)} + 0.80530401792079u_m^{(2)} + \Delta t \cdot 0.24249212077315L(u_m^{(2)})$$
$$= (0.19469598207921u_m^{(0)} + 0.80530401792079u_m^{(2)})$$
$$+ k(0.24249212077315)\left[\frac{u_{m+1}^{(2)} - 2u_m^{(2)} + u_{m-1}^{(2)}}{h^2} + \alpha u_m^{(2)}(1-u_m^{(2)})\right], \tag{3.11}$$

$$u_m^{(4)} = \sum_{k=0}^{3}(a_{4k}u_m^{(k)} + \Delta t b_{4k}L({u_m}^{(k)}))$$
$$= (0.58143386885601)u_m^{(0)} + (0.41856613114399)u_m^{(3)} + \Delta t(0.12603810060080)L(u_m^{(3)})$$
$$= (0.58143386885601)u_m^{(0)} + (0.41856613114399)u_m^{(3)}$$
$$+ k(0.12603810060080)\left[\frac{u_{m+1}^{(3)} - 2u_m^{(3)} + u_{m-1}^{(3)}}{h^2} + \alpha u_m^{(3)}(1-u_m^{(3)})\right], \tag{3.12}$$

$$u_m^{(5)} = \sum_{k=0}^{4}(a_{5k}u_m^{(k)} + \Delta t b_{5k}L({u_m}^{(k)}))$$
$$= (0.01934367892154)u_m^{(0)} + (0.98065632107846)u_m^{(4)} + (0.2952939830871)kL(u_m^{(4)})$$
$$= (0.01934367892154)u_m^{(0)} + +(0.98065632107846)u_m^{(4)}$$
$$+ k(0.2952939830871)\left[\frac{u_{m+1}^{(4)} - 2u_m^{(4)} + u_{m-1}^{(4)}}{h^2} + \alpha u_m^{(4)}(1-u_m^{(4)})\right], \tag{3.13}$$

$$u_m^{(6)} = \sum_{k=0}^{5}(a_{6k}u_m^{(k)} + \Delta t b_{6k}L({u_m}^{(k)}))$$
$$= u_m^{(5)} + (0.30111872706068)kL(u_m^{(5)})$$
$$= u_m^{(5)} + k(0.30111872706068)\left[\frac{u_{m+1}^{(5)} - 2u_m^{(5)} + u_{m-1}^{(5)}}{h^2} + \alpha u_m^{(5)}(1-u_m^{(5)})\right], \tag{3.14}$$

$$u_m^{(n+1)} = \sum_{k=0}^{6}(a_{7k}u_m^{(k)} + \Delta t b_{7k}L({u_m}^{(k)}))$$
$$= (0.06006304558847)u_m^{(0)} + (0.30152730794242)u_m^{(2)} + (0.10518998496676)u_m^{(3)}$$
$$+ (0.01483791154585)u_m^{(4)} + (0.51838174995650)u_m^{(6)}$$
$$+ (0.09079551914158)kL(u_m^{(2)}) + (0.02888359354880)kL(u_m^{(3)})$$
$$+ (0.15609445267839)kL(u_m^{(6)})$$
$$= (0.06006304558847)u_m^{(0)} + (0.30152730794242)u_m^{(2)} + (0.10518998496676)u_m^{(3)}$$

$$
\begin{aligned}
&+(0.01483791154585)u_m^{(4)}+(0.51838174995650)u_m^{(6)}\\
&+k(0.09079551914158)\left[\frac{u_{m+1}^{(2)}-2u_m^{(2)}+u_{m-1}^{(2)}}{h^2}+\alpha u_m^{(2)}(1-u_m^{(2)})\right]\\
&+k(0.02888359354880)\left[\frac{u_{m+1}^{(3)}-2u_m^{(3)}+u_{m-1}^{(3)}}{h^2}+\alpha u_m^{(3)}(1-u_m^{(3)})\right]\\
&+k(0.15609445267839)\left[\frac{u_{m+1}^{(6)}-2u_m^{(6)}+u_{m-1}^{(6)}}{h^2}+\alpha u_m^{(6)}(1-u_m^{(6)})\right],
\end{aligned}
\tag{3.15}
$$

for $m=1,2,3,\dots,M-1$. In the succeeding iteration, we have utilized $u_m^{(0)}=u_m^{n+1}$, with n ranging from 0 to $N-1$.

4. Numerical Experiment

Presented are the numerical results of the SSP-RK74 method applied to various instances of the Fisher equation (2.1) using MATLAB. With the help of the exact solution, we measured the accuracy of the numerical method. Assess the accuracy and efficiency of the proposed method by evaluating the l_2 and l_∞ error norms,

$$
l_2=\left[\frac{1}{M}\sum_{m=0}^{M}(U_m-u_m)^2\right]^{1/2},\quad l_\infty=\max_{0\le m\le M}|U_m-u_m|,
$$

where u_m is numerical solution and U_m as the exact solution corresponding to the node at position x_m.

Example 4.1. Consider the Fisher's equation

$$u_t=u_{xx}+\alpha u(1-u),$$

subject to the initial condition

$$u(x,0)=\frac{1}{(1+e^{\sqrt{\frac{\alpha}{6}}x})^2},$$

where the exact solution is presented in [18] given by

$$u(x,t)=\frac{1}{(1+e^{\sqrt{\frac{\alpha}{6}}x-\frac{5}{6}\alpha t})^2}.$$

Example 4.2. Consider the following generalized Fisher's equation in domain $[0,1]$:

$$u_t=u_{xx}+u(1-u^\alpha),$$

with initial condition

$$u(x,0)=\left\{\frac{1}{2}\tanh\left(-\frac{\alpha}{2\sqrt{2\alpha+4}}x\right)+\frac{1}{2}\right\}^{\frac{2}{\alpha}}.$$

The exact solution is presented in [18] by

$$u(x,0)=\left\{\frac{1}{2}\tanh\left(-\frac{\alpha}{2\sqrt{2\alpha+4}}\left(x-\frac{\alpha+4}{\sqrt{2\alpha+4}}t\right)\right)+\frac{1}{2}\right\}^{\frac{2}{\alpha}}.$$

Tables 2 and 3 provide numerical and exact solutions at different times for various values of 'α' for examples one and two. We compare the numerical and exact solutions for $\alpha=6$ and 1 using at two distinct time steps, namely $\Delta t = 0.0001$ and $\Delta t = 0.000005$, with a specific focus on Examples (4.1) and (4.2). Figures 1, 2, 3 and 4 represent numerical solutions compared to the exact solution, while Figures 5, 6, 7 and 8 display the absolute error graphs in relation to the exact solution for different values of α, with Δt values of 0.0001 and 0.000005. Tables 4 and 5 provide the l_2 and l_∞ errors for examples one and two. From these tables and graphs, we observed that the proposed method yields more accurate values for all time steps.

Table 2. Numerical and exact solution of Example 4.1 at $\alpha = 6$ and $M = 20$

		$\Delta t = 0.0001$		$\Delta t = 0.000005$	
x	T	Numerical solution	Exact solution	Numerical solution	Exact solution
0.25	0.4	0.725864	0.725824	0.725827	0.725824
	0.6	0.883457	0.883437	0.883439	0.883437
	0.8	0.954581	0.954573	0.954573	0.954573
	1.0	0.982921	0.982919	0.982919	0.982919
0.5	0.4	0.668474	0.668428	0.668433	0.668428
	0.6	0.854063	0.854038	0.854041	0.854038
	0.8	0.942245	0.942235	0.942235	0.942235
	1.0	0.978150	0.978147	0.978147	0.978147
0.75	0.4	0.604245	0.604195	0.604200	0.604195
	0.6	0.818422	0.818393	0.818395	0.818393
	0.8	0.926752	0.926740	0.926740	0.926740
	1.0	0.972075	0.972071	0.972071	0.972071

Table 3. Numerical and exact solution of Example 4.2 at $\alpha = 1$ and $M = 20$

		$\Delta t = 0.0001$		$\Delta t = 0.000005$	
x	T	Numerical solution	Exact solution	Numerical solution	Exact solution
0.25	0.4	0.310883	0.310875	0.310875	0.310875
	0.6	0.357842	0.357834	0.357834	0.357834
	0.8	0.406437	0.406428	0.406429	0.406428
	1.0	0.455748	0.455739	0.455739	0.455739
0.5	0.4	0.283306	0.283298	0.283298	0.283298
	0.6	0.328835	0.328827	0.328827	0.328827
	0.8	0.376535	0.376526	0.376526	0.376526
	1.0	0.425517	0.425509	0.425509	0.425509
0.75	0.4	0.256840	0.256832	0.256833	0.256832
	0.6	0.300644	0.300635	0.300636	0.300635
	0.8	0.347114	0.347106	0.347106	0.347106
	1.0	0.395420	0.395411	0.395412	0.395411

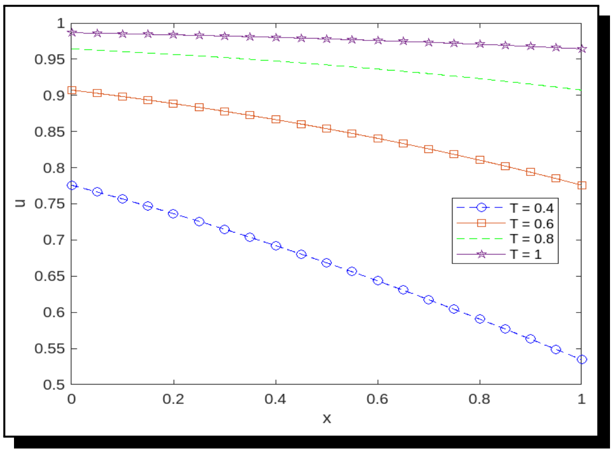

Figure 1. Comparison of numerical solutions of Example 4.1 at time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 6$, $\Delta t = 0.0001$, and $M = 20$

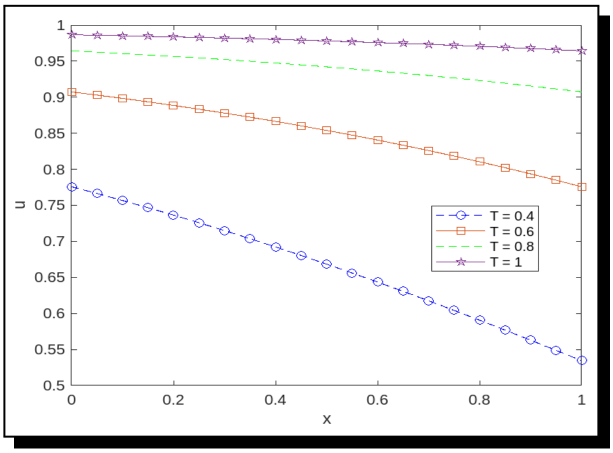

Figure 2. Comparison of numerical solutions of Example 4.1 at time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 6$, $\Delta t = 0.000005$, and $M = 20$

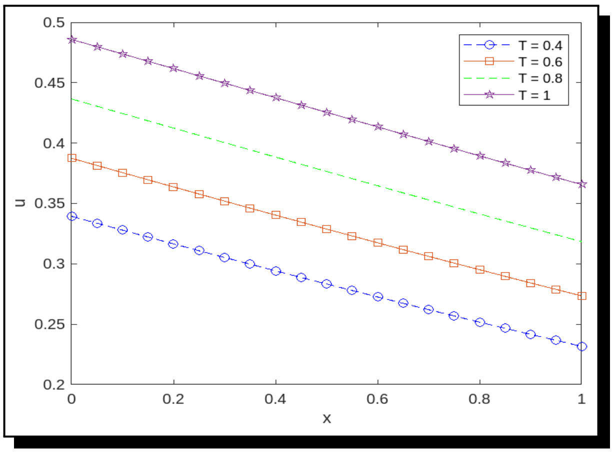

Figure 3. Comparison of numerical solutions of Example 4.2 at time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 1$, $\Delta t = 0.0001$, and $M = 20$

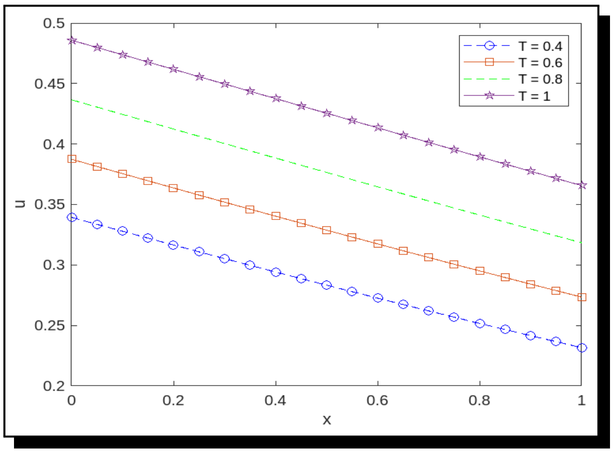

Figure 4. Comparison of numerical solutions of Example 4.2 at time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 1$, $\Delta t = 0.000005$, and $M = 20$

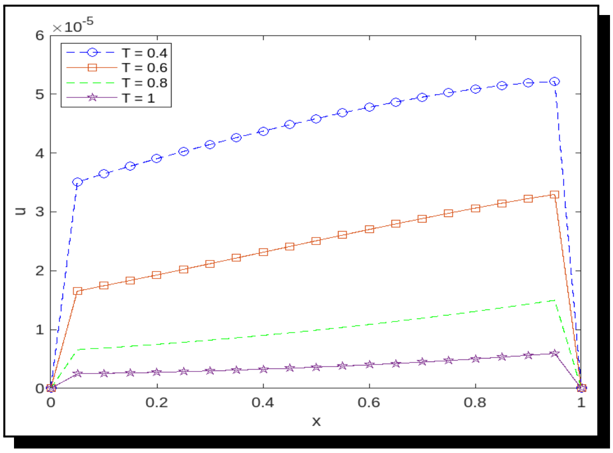

Figure 5. Absolute error comparison of Example 4.1 at different time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 6$, $\Delta t = 0.0001$, and $M = 20$

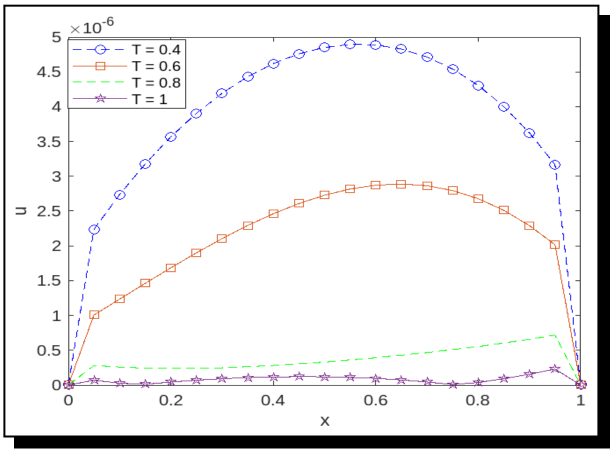

Figure 6. Absolute error comparison of Example 4.1 at different time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 6$, $\Delta t = 0.000005$, and $M = 20$

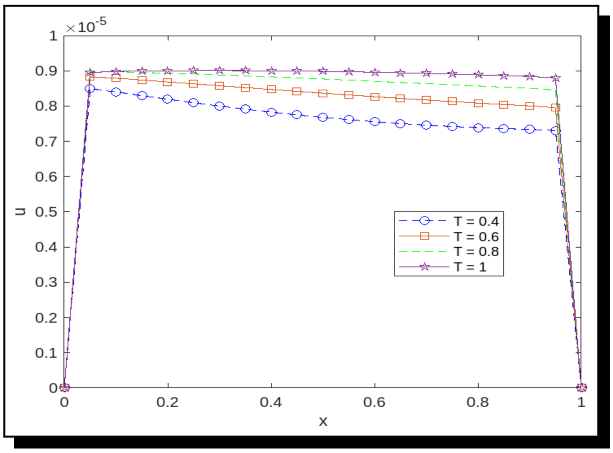

Figure 7. Absolute error comparison of Example 4.2 at different time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 1$, $\Delta t = 0.0001$, and $M = 20$

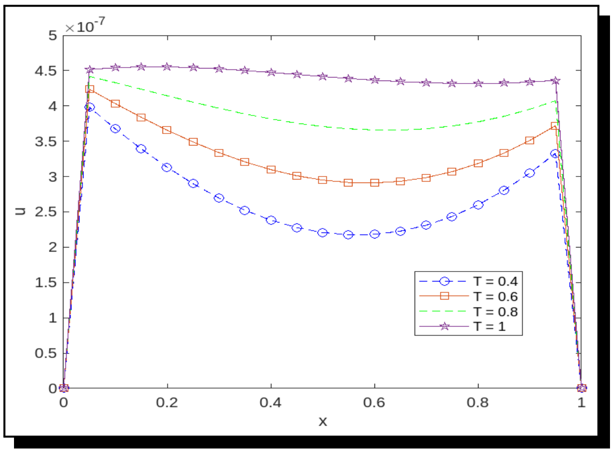

Figure 8. Absolute error comparison of Example 4.2 at different time levels $T = 0.4, 0.6, 0.8$ and 1 for $\alpha = 1$, $\Delta t = 0.000005$, and $M = 20$

Table 4. Errors of Example 4.1 at $\alpha = 6$

	$\Delta t = 0.0001$		$\Delta t = 0.000005$	
T	l_2	l_∞	l_2	l_∞
0.4	4.42530E-05	5.21652E-05	4.04398E-06	4.89540E-06
0.6	2.48444E-05	3.29639E-05	2.28567E-06	2.88950E-06
0.8	1.02717E-05	1.49734E-05	4.06231E-07	7.17900E-07
1.0	3.87287E-06	5.95150E-06	9.76728E-08	9.76728E-08

Table 5. Errors of Example 4.2 at $\alpha = 1$

	$\Delta t = 0.0001$		$\Delta t = 0.000005$	
T	l_2	l_∞	l_2	l_∞
0.4	7.58056E-06	8.4923E-06	2.73113E-07	3.981E-07
0.6	8.17259E-06	8.8286E-06	3.27646E-07	4.237E-07
0.8	8.76592E-06	8.9761E-06	3.82155E-07	4.418E-07
1.0	8.72275E-06	9.0070E-06	4.31856E-07	4.554E-07

Table 6. Comparison of numerical solution for Example 4.1 at $\Delta t = 0.000005$, $T = 0.1$ and $\alpha = 6$

x	BDF1 [32]	BDF2 [32]	BDF3 [32]	SSP43 [34]	Present solution	Exact solution
0.1	0.35841806	0.35842071	0.35842016	0.35842328	0.35842348	0.35842691
0.2	0.32997086	0.32997260	0.32997468	0.32997524	0.32997541	0.32998421
0.3	0.30230060	0.30230157	0.30230568	0.30230424	0.30230437	0.30231742
0.4	0.27558402	0.27558442	0.27558983	0.27558708	0.27558720	0.27560315
0.5	0.24997987	0.24997993	0.24998585	0.24998256	0.24998266	0.25000000
0.6	0.22562504	0.22562500	0.22563062	0.22562757	0.22562766	0.22564477
0.7	0.20263156	0.20263170	0.20263624	0.20263415	0.20263425	0.20264943
0.8	0.18108475	0.18108534	0.18108812	0.18108759	0.18108770	0.18109917
0.9	0.16104234	0.16104363	0.16104409	0.16104557	0.16104569	0.16105159

Table 8. Comparison of numerical solution for Example 4.1 at $\Delta t = 0.000005$, $T = 0.1$ and $\alpha = 1$

x	BDF1 [33]	BDF2 [33]	Present solution	Exact solution
0.1	0.26073733	0.26073824	0.26073858	0.26073843
0.2	0.25042002	0.25042078	0.25042105	0.25042110
0.3	0.24031064	0.24031127	0.24031151	0.24031169
0.4	0.23041738	0.23041791	0.23041811	0.23041838
0.5	0.22074766	0.22074815	0.22074834	0.22074865
0.6	0.21130823	0.21130872	0.21130891	0.21130920
0.7	0.20210505	0.20210558	0.20210578	0.20210601
0.8	0.19314332	0.19314394	0.19314417	0.19314428
0.9	0.18442751	0.18442824	0.18442850	0.18442843

Table 9. Numerical and exact results for Example 4.1 are compared at $\Delta t = 0.00005$, $M = 20$ for $\alpha = 6$

T	x	DQM [4], [18]	Present method	Exact solution
0.5	0.25	0.81847	0.818409	0.818393
	0.75	0.72592	0.725845	0.725824
1.0	0.25	0.98293	0.982920	0.982919
	0.75	0.97208	0.972073	0.972071

5. Conclusions

This paper presents a new approach to solving nonlinear partial differential equations. The method of lines is used to discretize the equation, and the resulting system of ordinary differential equations is solved using the strong stability preserving time-steping Runge-Kutta (SSP-RK74) method. To evaluate the accuracy and efficiency of the technique, two test examples involving the Fisher equation were examined. The numerical solutions were compared with exact solution at various values of the parameter 'α' within the system of ODEs. The norms l_2 and l_∞ were also used to measure the absolute errors of the numerical solutions. This technique can be applied to numerically solve higher-dimensional nonlinear partial differential equations.

Acknowledgment

The authors would like to thank the editor and the anonymous referees for the helpful suggestions to improve the quality of the manuscript.

Competing Interests

The authors declare that they have no competing interests.

Authors' Contributions

All the authors contributed significantly in writing this article. The authors read and approved the final manuscript.

References

[1] K. M. Agbavon, A. R. Appadu and M. Khumalo, On the numerical solution of Fisher's equation with coefficient of diffusion term much smaller than coefficient of reaction term, *Advances in Difference Equations* **2019** (2019), article number 146, DOI: 10.1186/s13662-019-2080-x.

[2] B. S. Alshammari and D. S. Mashat, Numerical study of Fisher's equation by finite difference schemes, *Applied Mathematics* **8**(8) (2017), 1100 – 1116, DOI: 10.4236/am.2017.88083.

[3] A. J. Ammerman and L. L. Cavalli-Sforza, Measuring the rate of spread of early farming in Europe, *Man* **6**(4) (1971), 674 – 688, DOI: 10.2307/2799190.

[4] M. Bastani and D.K. Salkuyeh, A highly accurate method to solve Fisher's equation, *Pramana* **78** (2012), 335 – 346, DOI: 10.1007/s12043-011-0243-8.

[5] M. Bonkile, A. Awasthi and S. Jayaraj, A numerical implementation of higher order time integration algorithm on unsteady Burgers' equation, in: *Proceedings of the International Conference On Mathematical Modeling and Computer Simulation*, Indian Institute of Technology Madras, Chennai, India (2014).

[6] M. D. Bramson, Maximal displacement of branching Brownian motion, *Communications on Pure and Applied Mathematics* **31**(5) (1978), 531 – 581, DOI: 10.1002/cpa.3160310502.

[7] J. Canosa, Diffusion in nonlinear multiplicative media, *Journal of Mathematical Physics* **10**(10) (1969), 1862 – 1868, DOI: 10.1063/1.1664771.

[8] V. Chandrakera, A. Awasthib and S. Jayaraja, A numerical treatment of Fisher equation, *Procedia Engineering* **127** (2015), 1256 – 1262, DOI: 10.1016/j.proeng.2015.11.481.

[9] R. A. Fisher, The wave of advance of advantageous genes, *Annals of Eugenics* **7**(4) (1937), 355 – 369, DOI: 10.1111/j.1469-1809.1937.tb02153.x.

[10] D. A. Frank-Kamenetskii, *Diffusion and Heat Transfer in Chemical Kinetics*, 2nd enlarged and revised edition, Plenum Press, New York – London, xxvi + 574 pages (1969).

[11] S. Gottlieb, C.-W. Shu and E. Tadmor, Strong stability-preserving high-order time discretization methods, *SIAM Review* **43**(1) (2001), 89 – 112, DOI: 10.1137/S003614450036757X.

[12] S. Hasnain and M. Saqib, Numerical study of one dimensional Fishers KPP equation with finite difference schemes, *American Journal of Computational Mathematics* **7**(01) (2017), 70 – 83, DOI: 10.4236/ajcm.2017.71006.

[13] A. C. Hindmarsh, *Toward a Systematized Collection of ODE Solvers*, No. UCRL-87465; CONF-820810-6, Lawrence Livermore National Laboratory, CA (USA), (1982), URL: https://ntrl.ntis.gov/NTRL/dashboard/searchResults/titleDetail/DE82012217.xhtml.

[14] R. Jiwari and R. C. Mittal, A higher order numerical scheme for singularly perturbed Burger-Huxley equation, *Journal of Applied Mathematics and Informatics* **29**(3-4) (2011), 813 – 829, URL: https://koreascience.kr/article/JAKO201121559392442.pdf.

[15] A. Kolmogorov, Étude de l'équation de la diffuṡion avec croissance de la quantité de matière et son application à un problème biologigue, *Bulletin de l'Universite d'Etat a Moscou Serie Internationale Section* **1** (1937), 12 pages.

[16] J. E. Macías-Díaz, I. E. Medina-Ramírez and A. Puri, Numerical treatment of the spherically symmetric solutions of a generalized Fisher–Kolmogorov–Petrovsky–Piscounov equation, *Journal of Computational and Applied Mathematics* **231**(2) (2009), 851 – 868, DOI: 10.1016/j.cam.2009.05.008.

[17] R. E. Mickens, A best finite-difference scheme for the fisher equation, *Numerical Methods for Partial Differential Equations* **10**(5) (1994), 581 – 585, DOI: 10.1002/num.1690100505.

[18] R. C. Mittal and R. Jiwari, Numerical study of Fisher's equation by using differential quadrature method, *International Journal of Information and System Sciences* **5**(1) (2009), 143 – 160.

[19] R. C. Mittal and S. Kumar, Numerical study of Fisher's equation by wavelet Galerkin method, *International Journal of Computer Mathematics* **83**(3) (2006), 287 – 298, DOI: 10.1080/00207160600717758.

[20] O. Oymak and N. Selçuk, Method-of-lines solution of time-dependent two-dimensional Navier-Stokes equations, *International Journal for Numerical Methods in Fluids* **23**(5) (1996), 455 – 466, DOI: 10.1002/(SICI)1097-0363(19960915)23:5%3C455::AID-FLD435%3E3.0.CO;2-J.

[21] R. A. V. Parambu, A. Awasthi, V. Vimal and N. Jha, A numerical implementation of higher-order time integration method for the transient heat conduction equation with a moving boundary based on boundary immobilization technique, *AIP Conference Proceedings* **2336**(1) (2021), 030012, DOI: 10.1063/5.0045874.

[22] Y. Qiu and D. M. Sloan, *Numerical solution of Fisher's equation using a moving mesh method*, *Journal of Computational Physics* **146**(2) (1998), 726 – 746, DOI: 10.1006/jcph.1998.6081.

[23] E. Rothe, Zweidimensionale parabolische Randwertaufgaben als Grenzfall eindimensionaler Randwertaufgaben, *Mathematische Annalen* **102**(1) (1930), 650 – 670, DOI: 10.1007/BF01782368.

[24] N. A. Shah, An analytical view of fractional-order Fisher's type equations within Caputo operator, *Mathematical Problems in Engineering* **2021**(1) (2021), 5516392, DOI: 10.1155/2021/5516392.

[25] L. F. Shampine, ODE solvers and the method of lines, *Numerical Methods for Partial Differential Equations* **10**(6) (1994), 739 – 755, DOI: 10.1002/num.1690100608.

[26] C.-W. Shu, Total-variation-diminishing time discretizations, *SIAM Journal on Scientific and Statistical Computing* **9**(6) (1988), 1073 – 1084, DOI: 10.1137/0909073.

[27] R. J. Spiteri and S. J. Ruuth, A new class of optimal high-order strong-stability-preserving time discretization methods, *SIAM Journal on Numerical Analysis* **40**(2) (2002), 469 – 491, DOI: 10.1137/S0036142901389025.

[28] R. J. Spiteri and S. J. Ruuth, Non-linear evolution using optimal fourth-order strong-stability-preserving Runge-Kutta methods, *Mathematics and Computers in Simulation* **62**(1-2) (2003), 125 – 135, DOI: 10.1016/S0378-4754(02)00179-9.

[29] S. Tang and R. Weber, Numerical study of Fisher's equation by a Petrov-Galerkin finite element method, *The ANZIAM Journal* **33**(1) (1991), 27 – 38, DOI: 10.1017/S0334270000008602.

[30] J. J. Tyson and P. K. Brazhnik, On traveling wave solutions of Fisher's equation in two spatial dimensions, *SIAM Journal on Applied Mathematics* **60**(2) (2000), 371 – 391, URL: http://www.jstor.org/stable/118487.

[31] A. Verma, R. Jiwari and M. E. Koksal, Analytic and numerical solutions of nonlinear diffusion equations via symmetry reductions, *Advances in Difference Equations* **2014** (2014), article number 229, DOI: 10.1186/1687-1847-2014-229.

[32] V. Vimal, R. K. Sinha and L. Pannikkal, Numerical methods for solving nonlinear Fisher equation using backward differentiation formula, *Applications and Applied Mathematics* **19**(4) (2024), Article 4, URL: https://digitalcommons.pvamu.edu/aam/vol19/iss4/4.

[33] V. Vimal, R. K. Sinha and P. Liju, An unconditionally stable numerical scheme for solving nonlinear Fisher equation, *Nonlinear Engineering* **13**(1) (2024), 20240006, DOI: 10.1515/nleng-2024-0006.

[34] V. Vimal, R. Kumari and R. K. Sinha, Higher-order numerical technique based on strong stability preserving method for solving nonlinear Fisher equation, *Communications in Mathematics and Applications* **15**(3) (2024), 997 – 1010, DOI: 10.26713/cma.v15i3.2806.

[35] X. Y. Wang, Exact and explicit solitary wave solutions for the generalised Fisher equation, *Physics Letters A* **131**(4-5) (1988), 277 – 279, DOI: 10.1016/0375-9601(88)90027-8.