Journal of Interdisciplinary Mathematics
ISSN: 0972-0502 (Print), ISSN: 2169-012X (Online)
Vol. 29 (2026), No. 9, pp. 2779–2794
DOI : 10.47974/JIM-2705

A new three-step AGE iteration procedure based on exponential spline approximation for 1D nonlinear parabolic partial differential equations

Niranjan [§]
Department of Mathematics
Marwadi University
Rajkot 360003
Gujarat
India

Jyoti Talwar [†]
Department of Mathematics
Miranda House
University of Delhi
New Delhi 110007
Delhi
India

R. K. Mohanty *
Department of Mathematics
South Asian University
Maidan Garhi
New Delhi 110068
Delhi
India

Abstract

We report a new 3-step explicit iteration procedure arranged group wise based on compact 2-level implicit exponential spline approximation of O(2,4) for the solution of 1D nonlinear parabolic equation $-u_{rr} + g(r, t, u, u_r, u_t) = 0$, $0 < r < 1$, $t > 0$ subject to initial and boundary conditions given at $t = 0$ and $r = 0$, $r = 1$. The convergence analysis of the recommended algorithm is deliberated concisely. We also report a new 3-step explicit iteration scheme for solving nonlinear difference equation systems at each time level as a computational procedure. The proposed iteration procedure is expected to save the CPU-

[§] *E-mail:* niranjan@marwadieducation.edu.in
[†] *E-mail:* jyoti.talwar@mirandahouse.ac.in
* *E-mail:* rmohanty@sau.ac.in (Corresponding author)

© Taru Publications

time as the common term is evaluated at step-1. In application, we have solved Burgers-Fisher, Burgers-Huxley and heat equation in polar form. The computational results are compared with the exact solution, and corresponding Newton-AGE and Newton-SOR iteration methods to reveal the effectiveness of the professed iteration technique.

Subject Classification: *65M22, 65M12, 65M06.*

Keywords: *Exponential spline approximation, Nonlinear parabolic equations, 3-step AGE iteration process, Convergence analysis, Burgers-Fisher, Burgers-Huxley equations, Acceleration parameter.*

1. Introduction

Nonlinear partial differential equations naturally emerge in the mathematical modelling of many physical phenomena, particularly in the analysis of nonlinear wave propagation. Such equations play an important role across several disciplines of applied mathematics, physics, and engineering, including fluid dynamics, nonlinear optics, solid mechanics, plasma physics, and quantum field theory. The present work focuses on the study of 1D nonlinear parabolic partial differential equation (PPDE) whose general form is

$$u_{rr} = g(r, t, u, u_r, u_t), 0 < r < 1, t > 0. \quad (1.1)$$

The associated initial and boundary conditions are stated as

$$u(r, 0) = a_0(r), 0 \leq r \leq 1, \quad (1.2)$$

$$u(0, t) = b_0(t), u(1, t) = b_1(t), \mathrm{t} > 0. \quad (1.3)$$

Lately, substantial work has been undertaken to obtain the approximate solutions of PPDEs. Researchers have extensively worked on developing numerical methods based on polynomial and non-polynomial spline for approximating the solution of PPDEs. Schweikert was the first to work with Tension splines, as a method to remove extraneous inflection points while curve fitting using cubic spline. In this type of cubic spline, a non-negative parameter is introduced on every interval to represent tension. It helps preserve shape properties, like monotonicity and convexity. Kadalbajoo and Patidar [1], Mohanty and Arora [2] employed spline in tension approximation to solve singularly perturbed two-point non-linear BVPs. Rashidinia and Mohammadi [3] solved the nonlinear Klein-Gordon equation using techniques founded on tension spline formulations. Talwar, Mohanty, and Singh [4] presented approaches utilizing spline-in-compression functions to solve one dimensional parabolic equation with singular coefficients. Mittal, Ray, and Mohanty [5] proposed compact finite difference scheme in exponential form for solving two-dimensional quasi-linear parabolic partial differential equation. The scheme presented is fourth order in space and second order in time. They have applied it to solve Burgers' equation, polar-coordinate heat conduction, Navier–Stokes equations, and quasi-linear parabolic model. Obaid, Kadhim, and Abed [6] presented a comparative study of approximate methods for solving second-order linear complex partial differential equations.

Kaya and El-Sayed [7] have worked on acquiring the closed form of the exact solutions of the generalized Burger-Fisher equation. Various numerical techniques have been proposed for solving the Burgers–Fisher equation by Zhu and Kang [8], Mohammadi [9], Mittal and Tripathi [10], and Ismail, Raslan, and Rabboh [11]. Similarly, Batiha, Noorani, and Hashim [12], Altaie, Ali, and Shihab [13], Bratsos [14], and Sari, Gurarslan, and Zeytinoglu [15] proposed analytical and numerical methods for solving nonlinear differential equations, including the generalized Burgers–Huxley equation.

Also, with the advent of parallel computing, Mohanty [16] as well as Evans and Ahmad [17] developed Group Explicit Schemes for efficiently solving large linear systems. They give stable solutions, without conceding on accuracy. Many variations of the Group Explicit methods, like Alternating Group Explicit (AGE), Smart AGE, Coupled Reduced AGE have been worked upon focusing on reducing the computational cost and time. Sukon and Evans [18] worked on two parameter AGE, Mohanty and Jain [19] augmented different discretization schemes with AGE iterative methods to solve nonlinear PPDEs, Dahalan et al. [20] devised QSAGE technique to obtain the solution of fuzzy parabolic.

In the present study, in next section, we derive a two-level implicit numerical method of O(2,4) based on exponential spline approximation for the solution of nonlinear initial-boundary value problem (IBVP) (1.1)-(1.3). In Section 3, we deliberate a new 3-step iteration method for linear PPDE based on exponential spline approximation of O(2,4) and study the convergence analysis. In Section 4, the developed approach is applied to nonlinear difference equations. In Section 5, with the aid of methods discussed in this paper, we solve Burgers-Fisher, Burgers-Huxley and PPDE in polar form to demonstrate the usefulness of the formulated iteration approach. The Section 6 address the concluding remarks.

2. Exponential spline approximation and numerical procedure

The solution domain $r \in [0,1] \times (t \geq 0)$ is divided into $(M+1) \times N$ mesh points with mesh size $h = 1/(M+1) > 0$ in r-direction and $\tau > 0$ in t-direction. Let $\chi = \tau/h^2 > 0$ be the mesh ratio aspect. Let (r_m, t_n) be the internal mesh points defined in the solution domain, where $r_m = mh,\ m = 0(1)M+1$ and $t_n = n\tau, n = 0(1)N$. Let $U_m^n = u(r_m, t_n)$ corresponds to exact value of the solution $u(r,t)$ at the mesh point (r_m, t_n), and $u_m^n \approx U_m^n$.

Let $E_n(r)$ be the exponential spline function interpolating U_m^n at the point (r_m, t_n), and is specified by

$$E_n(r) = U_m^n - \frac{M_m^n}{\omega^2} + \left[\frac{u_m^n - u_{m-1}^n}{h} + \frac{M_m^n - M^n_{m-\frac{1}{2}}\cosh\theta}{\omega\theta}\right](r - r_m) + \frac{M_m^n \cosh\theta - M^n_{m-\frac{1}{2}}}{\omega^2 \sinh\theta}\sinh\omega(r - r_m)$$

$$+ \frac{M_m^n}{\omega^2}\cosh\omega(r - r_m),\ r_{m-1} \leq r \leq r_m,\ m = 1(1)M+1,\ \text{n} = 1, 2, \ldots, \text{N}, \qquad (2.1)$$

where ω is the frequency and $\theta = \omega h/2$. At $n^{th} -$ time level, the function $E_n(r)$ satisfies the following conditions:

(i) $E_n(r) \in C^2[0,1]$, and
(ii) $E_n(r_m) = U_m^n$, $E_n(r_{m-1}) = U_{m-1}^n$.

Let,

$G_m^n = g\left(r_m, t_n, U_m^n, {U_r}_m^n, {U_t}_m^n\right)$ and $G_{m-\frac{1}{2}}^n = g\left(r_{m-\frac{1}{2}}, t_n, U_{m-\frac{1}{2}}^n, {U_r}_{m-\frac{1}{2}}^n, {U_t}_{m-\frac{1}{2}}^n\right)$, $r_{m-1} \le r \le r_m$.

We consider the following two-level estimations:

$$\bar{t}_n = t_n + \frac{\tau}{2}, \tag{2.2}$$

$$\bar{U}_m^n = \frac{1}{2}(U_m^{n+1} + U_m^n), \tag{2.3}$$

$$\bar{U}_{m\pm1}^n = \frac{1}{2}\left(U_{m\pm1}^{n+1} + U_{m\pm1}^n\right), \tag{2.4}$$

$$\bar{U}_{m\pm\frac{1}{2}}^n = \frac{1}{2}\left(\bar{U}_{m\pm1}^n + \bar{U}_m^n\right), \tag{2.5}$$

$${\bar{U}_t}_m^n = \frac{1}{\tau}(U_m^{n+1} - U_m^n), \tag{2.6}$$

$${\bar{U}_t}_{m\pm1}^n = \frac{1}{\tau}\left(U_{m\pm1}^{n+1} - U_{m\pm1}^n\right), \tag{2.7}$$

$${\bar{U}_t}_{m\pm\frac{1}{2}}^n = \frac{1}{2\tau}\left[U_{m\pm1}^{n+1} + U_m^{n+1} - U_{m\pm1}^n - U_m^n\right], \tag{2.8}$$

$${\bar{U}_r}_m^n = \frac{1}{2h}(\bar{U}_{m+1}^n - \bar{U}_{m-1}^n), \tag{2.9}$$

$${\bar{U}_r}_{m\pm\frac{1}{2}}^n = \pm\frac{1}{h}\left(\bar{U}_{m\pm1}^n - \bar{U}_m^n\right), \tag{2.10}$$

$${\bar{U}_{rr}}_m^n = \frac{1}{h^2}(\bar{U}_{m+1}^n - 2\bar{U}_m^n + \bar{U}_{m-1}^n). \tag{2.11}$$

At the internal point (r_m, t_n), we denote

$$\lambda_m^n = \left(\frac{\partial g}{\partial U}\right)_m^n, \mu_m^n = \left(\frac{\partial g}{\partial U_r}\right)_m^n, \nu_m^n = \left(\frac{\partial g}{\partial U_t}\right)_m^n, \xi_m^n = \left(\frac{\partial g}{\partial t}\right)_m^n.$$

The consistency condition is given by

$$U_{m+1}^n - 2U_m^n + U_{m-1}^n = \frac{h^2}{3}\left[G_{m+\frac{1}{2}}^n + G_{m-\frac{1}{2}}^n + G_m^n\right] + O(h^6) \tag{2.12}$$

or evenly $\tanh\frac{\theta}{2} = \frac{\theta}{2}$, which on solving explicitly, we acquire $\theta = \frac{\omega h}{2} = 0.001$. For a chosen value of h we can easily estimate the value of ω.

Next, we define

$$\bar{G}_m^n = g\left(r_m, \bar{t}_n, \bar{Z}_m^n, {\bar{Z}_r}_m^n, {\bar{Z}_t}_m^n\right), \tag{2.13}$$

$$\bar{G}^n_{m\pm\frac{1}{2}} = g\left(r_{m\pm\frac{1}{2}}, \bar{t}_n, \bar{Z}^n_{m\pm\frac{1}{2}}, \bar{Z}_{r}{}^{n}_{m\pm\frac{1}{2}}, \bar{Z}_{t}{}^{n}_{m\pm\frac{1}{2}}\right). \tag{2.14}$$

With the support of (2.2) - (2.10), the approximations (2.13) and (2.14) modify to

$$\bar{G}^n_m = G^n_m + \frac{\tau}{2}\left[\xi^n_m + \lambda^n_m U_t{}^n_m + \mu^n_m U_{rt}{}^n_m\right] + \frac{h^2}{6}\mu^n_m U_{rrr}{}^n_m + \frac{\tau}{2}\nu^n_m U_{tt}{}^n_m + O(\tau^2 + \tau h^2 + h^4), \tag{2.15}$$

$$\bar{G}^n_{m\pm\frac{1}{2}} = G^n_{m\pm\frac{1}{2}} + \frac{\tau}{2}\left[\xi^n_m + \lambda^n_m U_t{}^n_m + \mu^n_m U_{rt}{}^n_m\right] + \frac{\tau}{2}\nu^n_m U_{tt}{}^n_m + \frac{h^2}{24}\left[3\lambda^n_m U_{rr}{}^n_m + \mu^n_m U_{rrr}{}^n_m + 3\nu^n_m U_{rrt}{}^n_m\right] + O(\tau^2 + \tau h^2 + h^4). \tag{2.16}$$

Next, we express

$$\hat{U}_{r}{}^{n}_{m\pm\frac{1}{2}} = \pm\frac{(\bar{u}^n_{m\pm1} - \bar{u}^n_m)}{h} \pm \frac{h}{12}\left[\bar{G}^n_m - \bar{G}^n_{m\pm\frac{1}{2}}\right]. \tag{2.17}$$

With the aid of (2.15) and (2.16), the estimation (2.17) simplifies to

$$\hat{U}_{r}{}^{n}_{m\pm\frac{1}{2}} = U_{r}{}^{n}_{m\pm\frac{1}{2}} \pm \frac{\tau}{2}U_{rt}{}^{n}_{m} + O(\tau^2 + \tau h^2 + h^4). \tag{2.18}$$

We define the estimation

$$\hat{G}^n_{i\pm\frac{1}{2}} = g\left(r_{m\pm\frac{1}{2}}, \bar{t}_n, \bar{U}^n_{m\pm\frac{1}{2}}, \hat{U}^n_{r_{m\pm\frac{1}{2}}}, \bar{U}^n_{t_{m\pm\frac{1}{2}}}\right). \tag{2.19}$$

Substituting (2.18) into (2.19) and expanding, we get

$$\hat{G}^n_{m\pm\frac{1}{2}} = G^n_{m\pm\frac{1}{2}} + \frac{\tau}{2}\left[\xi^n_m + \lambda^n_m U_t{}^n_m + \mu^n_m U_{rt}{}^n_m\right] + \frac{\tau}{2}\nu^n_m U_{tt}{}^n_m + \frac{h^2}{8}\left[\lambda^n_m U_{rr}{}^n_m + \nu^n_m U^n_{rrt_m}\right] + O(\tau^2 + \tau h^2 + h^4). \tag{2.20}$$

Let
$$\hat{U}^n_{r_m} = \bar{U}_{r}{}^{n}_{m} + b_0 h\left[\bar{G}^n_{m+\frac{1}{2}} - \bar{G}^n_{m-\frac{1}{2}}\right]. \tag{2.21}$$

Here b_0 denotes a parameter whose value is to be evaluated. With the support of (2.16), the estimation (2.21) becomes

$$\hat{U}^n_{r_m} = U^n_{r_m} + \frac{\tau}{2}U^n_{rt_m} + \frac{h^2}{6}(1 + 6b_0)U^j_{rrr_i} + O(\tau^2 + \tau h^2 + h^4). \tag{2.22}$$

For $b_0 = \frac{-1}{6}$, the estimation (2.22) shortens to

$$\hat{U}^n_{r_m} = U^n_{r_m} + \frac{\tau}{2}U^n_{rt_m} + O(\tau^2 + \tau h^2 + h^4). \tag{2.23}$$

Further, let

$$\hat{U}^n_m = \bar{U}^n_m + a_0 h^2 \bar{U}^n_{rr_m} = U^n_m + \frac{\tau}{2}U^n_{t_m} + a_0 h^2 U^n_{rr_m} + O(\tau^2 + \tau h^2 + h^4). \tag{2.24}$$

$$\hat{U}^n_{t_m} = \bar{U}^n_{t_m} + c_0\left[\bar{U}^n_{t_{m+1}} - 2\bar{U}^n_{t_m} + \bar{U}^n_{t_{m-1}}\right] = U^n_{t_m} + \frac{\tau}{2}U^n_{tt_m} + c_0 h^2 U^n_{rrt_m} + O(\tau^2 + h^4). \tag{2.25}$$

where a_0 and c_0 are parameters to be ascertained.

Finally, let

$$\hat{G}^n_m = g\left(r_m, \bar{t}_n, \hat{U}^n_m, \hat{U}^n_{r_m}, \hat{U}^n_{t_n}\right). \tag{2.26}$$

With the support of (2.23)-(2.25), the approximation (2.26) simplifies to

$$\hat{G}_m^n = G_m^n + \frac{\tau}{2}\left[\xi_m^n + \lambda_m^n U_{t_m}^n + \mu_m^n U_{rt_m}^n\right] + \frac{\tau}{2}\nu_m^n U_{tt_m}^n + h^2\left[a_0\lambda_m^n U_{rr_m}^n + c_0\nu_m^n U_{rrt_m}^n\right] + O(\tau^2 + \tau h^2 + h^4). \tag{2.27}$$

Differentiating the nonlinear PPDE (1.1) with regard to the time variable $'t'$ at the internal grid point (r_m, t_n), the outcome is

$$-\nu_m^n U_{tt_m}^n = \xi_m^n + \lambda_m^n U_{t_m}^n + \mu_m^n U_{rt_m}^n - U_{rrt_m}^n \tag{2.28}$$

From the estimates (2.3) and (2.4), we get

$$\bar{U}_{m+1}^n - 2\bar{U}_m^n + \bar{U}_{m-1}^n = U_{m+1}^n - 2U_m^n + U_{m-1}^n + \frac{\tau h^2}{2} Z_{rrt_m}^n + O(\tau h^4 + h^6) \tag{2.29}$$

With the assistance of (2.12), the trigonometric spline approximation for the nonlinear PPDE (1.1) is given by

$$\bar{U}_{m+1}^n - 2\bar{U}_m^n + \bar{U}_{m-1}^n = \frac{h^2}{3}\left[\hat{G}_{m+\frac{1}{2}}^n + \hat{G}_{m-\frac{1}{2}}^n + \hat{G}_m^n\right] + \hat{T}_m^n, \tag{2.30}$$

where the local truncation error (LTE) $\hat{T}_m^n = O(\tau^2 h^2 + \tau h^4 + h^6)$.

Now with the service of (2.12), (2.20), (2.27) - (2.29), from (2.30), the LTE is calculated as

$$\hat{T}_m^n = \frac{-h^4}{12}\left[(1 + 4a_0)\lambda_m^n U_{rr_m}^n + (1 + 4c_0)\nu_m^n U_{rrt_m}^n\right] + O(\tau^2 h^2 + \tau h^4 + h^6). \tag{2.31}$$

The exponential spline method (2.30) to be of $O(\tau^2 + \tau h^2 + h^4)$, the coefficient of h^4 in (2.31) must be zero, that is, $a_0 = c_0 = \frac{-1}{4}$. Consecutively, the LTE reduces to $\hat{T}_m^n = O(\tau^2 h^2 + \tau h^4 + h^6)$.

Now we consider the heat equation in cylindrical polar form

$$U_{rr} + \frac{1}{r}U_r = U_t + g(r,t), 0 < r < 1, t > 0 \tag{2.32}$$

Applying the method (2.30) to the linear PPDE (2.32), we arrive at

$$\bar{U}_{m+1}^n - 2\bar{U}_m^n + \bar{U}_{m-1}^n = \frac{h^2}{3}\left[-\frac{1}{r_{m+\frac{1}{2}}}\hat{U}_{r_{m+\frac{1}{2}}}^n + \hat{U}_{t_{m+\frac{1}{2}}}^n - \frac{1}{r_{m-\frac{1}{2}}}\hat{U}_{r_{m-\frac{1}{2}}}^n + \hat{U}_{t_{m-\frac{1}{2}}}^n - \frac{1}{r_m}\hat{U}_{r_m}^n + \hat{U}_m^n + \bar{g}_{m+\frac{1}{2}}^n + \bar{g}_{m-\frac{1}{2}}^n + \bar{g}_m^n\right] + \hat{T}_m^n, \tag{2.33}$$

where $\bar{g}_m^n = g(r_m, \bar{t}_n)$, $\bar{g}_{m\pm\frac{1}{2}}^n = g\left(r_{m\pm\frac{1}{2}}, \bar{t}_n\right)$.

For operator compact form, we use the approximations defined below:

$$\frac{1}{r_{m\pm\frac{1}{2}}} = \frac{1}{r_m} \mp \frac{h}{2(r_m)^2} + \frac{h^2}{4(r_m)^3} \mp O(h^3), \tag{2.34}$$

$$\bar{g}_{m\pm\frac{1}{2}}^n = \bar{g}_m^n \pm \frac{h}{2}\bar{g}_{r_m}^n + \frac{h^2}{8}\bar{g}_{rr_m}^n \pm O(h^3). \tag{2.35}$$

Substituting (2.34)-(2.35) into (2.33), neglecting LTE term, (2.33) can be re-written as a 2-level trigonometric spline scheme in compact operator form

$$\left[1+\frac{1}{12}(1-6\chi+\chi S_1)\delta_r^2+\frac{1}{12}\left(\frac{h}{2r_m}+\chi S_2\right)(2\mu_r\delta_r)\right]u_m^{n+1}$$
$$=\left[1+\frac{1}{12}(1+6\chi-\chi S_1)\delta_r^2+\frac{1}{12}\left(\frac{h}{2r_m}-\chi S_2\right)(2\mu_r\delta_r)\right]u_m^{n}+\Sigma g, \qquad (2.36)$$

where

$$\delta_r^2 u_m = u_{m+1}-2u_m+u_{m-1},\ (2\mu_r\delta_r)u_m = u_{m+1}-u_{m-1},$$

and

$$S_1=\frac{-3h^2}{2(r_m)^2},\ S_2=\frac{h}{4r_m}\left[12+\frac{3h^2}{(r_m)^2}\right],$$
$$\Sigma g=\frac{-\tau}{12}\left[12\bar{g}_m^n+h^2\left(\bar{g}_{rr_m}^n+\frac{1}{r_m}\bar{g}_{r_m}^n\right)\right].$$

The trigonometric spline scheme (2.36) is of O(2,4) and free from the terms $\frac{1}{r_{m\pm1}}$, thus the scheme (2.36) can be computed conveniently throughout the domain defined by $0<r<1, t>0$.

3. Three-step group explicit iteration method

The linear trigonometric spline scheme (2.36) can be re-written as

$$a_m u_{m-1}^{n+1}+2b_m u_m^{n+1}+c_m u_{m+1}^{n+1}=d_m, m=1(1)M, n=0,1,2\ldots N \qquad (3.1)$$

where $a_m=\frac{1}{12}(1-6\chi+\chi S_1)-\frac{1}{12}\left(\frac{h}{2r_m}+\chi S_2\right)$,

$$2b_m=1-\frac{1}{6}(1-6\chi+\chi S_1),$$
$$c_m=\frac{1}{12}(1-6\chi+\chi S_1)+\frac{1}{12}\left(\frac{h}{2r_m}+\chi S_2\right),$$
$$d_m=\left[1+\frac{1}{12}(1+6\chi-\chi S_1)\delta_r^2+\frac{1}{12}\left(\frac{h}{2r_m}-\chi S_2\right)(2\mu_r\delta_r)\right]u_m^n+\Sigma g.$$

In matrix form, the spline scheme (3.1) can be expressed as

$$\boldsymbol{PU}=\boldsymbol{Q} \qquad (3.2)$$

where

$$\boldsymbol{P}=\begin{bmatrix} 2b_1 & c_1 & & & 0\\ a_2 & 2b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{M-1} & 2b_{M-1} & c_{M-1}\\ 0 & & & a_M & 2b_M \end{bmatrix},\ \boldsymbol{U}=\begin{bmatrix} u_1^{n+1}\\ u_2^{n+1}\\ \vdots\\ \vdots\\ u_M^{n+1}\end{bmatrix} \text{ and}$$

$$\boldsymbol{Q}=\begin{bmatrix} d_1-a_1z_0^{n+1}\\ d_2\\ \vdots\\ \vdots\\ d_M-c_Mz_{M+1}^{n+1}\end{bmatrix}=\begin{bmatrix} q_1\\ q_2\\ \vdots\\ \vdots\\ q_M\end{bmatrix}$$

Now we consider odd number of internal grid points in r-directions, that is, M is odd, and let

$$\boldsymbol{P} = \boldsymbol{P}_1 + \boldsymbol{P}_2, \tag{3.3}$$

where

$$\boldsymbol{P}_1 = \begin{bmatrix} b_1 & & & & & \\ & b_2 & c_2 & & \mathbf{0} & \\ & a_3 & b_3 & & & \\ & & & \ddots & & \\ & & & & b_{M-1} & c_{M-1} \\ & \mathbf{0} & & & a_M & b_M \end{bmatrix}, \quad \boldsymbol{P}_2 = \begin{bmatrix} b_1 & c_1 & & & & \\ a_2 & b_2 & & & \mathbf{0} & \\ & & \ddots & & & \\ & & & b_{M-2} & c_{M-2} & \\ & \mathbf{0} & & a_{M-1} & b_{M-1} & \\ & & & & & b_M \end{bmatrix}$$

Assume that $\det(\boldsymbol{P_1} + \omega_0 \boldsymbol{I}) \neq 0$ and $\det(\boldsymbol{P_2} + \omega_0 \boldsymbol{I}) \neq 0$ for any $\omega_0 > 0$. Then the 3-step alternating iteration procedure for the system of equations (3.2) may be written as

$$\boldsymbol{V}^{(s)} = (\boldsymbol{P}_2 - \omega_0 \boldsymbol{I})\boldsymbol{U}^{(s)}, \text{ s} = 0, 1, 2, \ldots, \tag{3.4}$$

$$(\boldsymbol{P}_1 + \omega_0 \boldsymbol{I})\ \boldsymbol{U}^{*(s)} = \mathbf{Q} - \boldsymbol{V}^{(s)}, \text{ s} = 0, 1, 2, \ldots, \tag{3.5}$$

$$(\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})\ \boldsymbol{U}^{(s+1)} = 2\omega_0 \boldsymbol{U}^{*(s)} + \boldsymbol{V}^{(s)}, \text{ s} = 0, 1, 2, \ldots, \tag{3.6}$$

where $\boldsymbol{U}^{*(s)}$ refers to an intermediate-stage vector. The 3-step iteration process is expected to save the overall CPU-time as we are evaluating first two similar terms $\boldsymbol{V}^{(s)}$ as the right-hand side part of (3.5) and (3.6).

We denote $\alpha_m = b_m + \omega_0$ and $\beta_m = b_m - \omega_0$, then (3.4) - (3.6) explicitly written as

$$\boldsymbol{V}^{(s)} = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ \vdots \\ \vdots \\ V_{M-2} \\ V_{M-1} \\ V_M \end{bmatrix}^{(s)} = \begin{bmatrix} \beta_1 u_1^n + c_1 u_2^n \\ a_2 u_1^n + \beta_2 u_2^n \\ \vdots \\ \vdots \\ \vdots \\ \beta_{M-2} u_{M-2}^n + c_{M-2} u_{M-1}^n \\ a_{M-1} u_{M-2}^n + \beta_{M-1} u_{M-1}^n \\ \beta_M u_M^n \end{bmatrix}^{(s)}, s = 0,1,2, \ldots, \tag{3.7}$$

$$\begin{bmatrix} \alpha_1 & & & & & \\ & \alpha_2 & c_2 & & \mathbf{0} & \\ & a_3 & \alpha_3 & & & \\ & & & \ddots & & \\ & & & & \alpha_{M-1} & c_{M-1} \\ & \mathbf{0} & & & a_M & \alpha_M \end{bmatrix} \begin{bmatrix} U_1^* \\ U_2^* \\ U_3^* \\ \vdots \\ \vdots \\ U_{M-2}^* \\ U_{M-1}^* \\ U_M^* \end{bmatrix}^{(s)} = \begin{bmatrix} q_1 - V_1 \\ q_2 - V_2 \\ q_3 - V_3 \\ \vdots \\ \vdots \\ \mathrm{q}_{M-2} - V_{M-2} \\ \mathrm{q}_{M-1} - V_{M-1} \\ q_M - V_M \end{bmatrix}^{(s)}, s = 0,1,2, \ldots, \tag{3.8}$$

$$\begin{bmatrix} \alpha_1 & c_1 & & & \\ a_2 & \alpha_2 & \ddots & & \mathbf{0} \\ & \ddots & & \alpha_{M-2} & c_{M-2} \\ & & & a_{M-1} & \alpha_{M-1} \\ & \mathbf{0} & & & \alpha_M \end{bmatrix} \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \\ \vdots \\ \vdots \\ u_{M-2}^{n+1} \\ u_{M-1}^{n+1} \\ u_M^{n+1} \end{bmatrix}^{(s+1)} = 2\omega_0 \begin{bmatrix} U_1^* \\ U_2^* \\ U_3^* \\ \vdots \\ \vdots \\ U_{M-2}^* \\ U_{M-1}^* \\ U_M^* \end{bmatrix}^{(s)} + \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ \vdots \\ \vdots \\ V_{M-2} \\ V_{M-1} \\ V_M \end{bmatrix}^{(s)}, s = 0, 1, 2, \dots . \tag{3.9}$$

As $\det(\boldsymbol{P_1} + \omega_0 \boldsymbol{I}) \neq 0$ and $\det(\boldsymbol{P_2} + \omega_0 \boldsymbol{I}) \neq 0$, this implies, $(\boldsymbol{P}_1 + \omega_0 \boldsymbol{I})^{-1}$ and $(\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})^{-1}$ do exists. Therefore, (3.4) - (3.6) in explicit form can be re-written as

$$\boldsymbol{V}^{(s)} = (\boldsymbol{P}_2 - \omega_0 \boldsymbol{I})\boldsymbol{U}^{(s)}, \text{ s = 0, 1, 2,… ,}$$
$$\boldsymbol{U}^{*(s)} = (\boldsymbol{P}_1 + \omega_0 \boldsymbol{I})^{-1}(\boldsymbol{Q} - \boldsymbol{V}^{(s)}), s = 0, 1, 2, ...$$
$$\boldsymbol{U}^{(s+1)} = (\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})^{-1}\left(2\omega_0 \boldsymbol{U}^{*(s)} + \boldsymbol{V}^{(s)}\right), s = 0, 1, 2, ...$$

Thus, simplifying (3.4) - (3.6) a three-step algorithm in explicit form can be described as follows:

Stage I: For $m = 1(2)M - 2$,

$$V_m^{(s)} = [\beta_m u_m^n + c_m u_{m+1}^n]^{(s)}, \text{ s = 0, 1, 2,… ,} \tag{3.10}$$
$$V_{i+1}^{(s)} = \left[a_{i+1} u_i^j + \beta_{i+1} u_{i+1}^j\right]^{(s)}, \text{ s = 0, 1, 2,… .} \tag{3.11}$$
$$V_M^{(s)} = [\beta_M u_M^n]^{(s)}, \text{ s = 0, 1, 2,…} \tag{3.12}$$

Stage II:
$$U_1^{*(s)} = [(q_1 - V_1)/\alpha_1]^{(s)}, \text{ s = 0, 1, 2,…} \tag{3.13}$$

For $m = 2(2)M - 1$, let $\Delta = \alpha_m \alpha_{m+1} - c_m a_{m+1} \neq 0$, then

$$U_m^{*(s)} = [(q_m - V_m)\alpha_{m+1} - (q_{m+1} - V_{m+1})c_m]^{(s)}/\Delta \text{ , s = 0, 1, 2,…} \tag{3.14}$$
$$U_{m+1}^{*(s)} = [(q_{m+1} - V_{m+1})\alpha_m - (q_m - V_m)a_{m+1}]^{(s)}/\Delta \text{ , s = 0, 1, 2,…} \tag{3.15}$$

Stage III: For $m = 1(2)M - 2$, let $\Delta = \alpha_m \alpha_{m+1} - c_m a_{m+1} \neq 0$, then

$$(U_m^{n+1})^{(s)} = [(2\omega_0 U_m^* + V_m)\alpha_{m+1} - (2\omega_0 U_{m+1}^* + V_{m+1})c_m]^{(s)}/\Delta \text{ , s = 0,1, 2,…,} \tag{3.16}$$

$$(U_M^{n+1})^{(s)} = [(2\omega_0 U_M^* + V_M)]^{(s)}/\alpha_M, \text{ s = 0, 1, 2, … .} \tag{3.17}$$

Following the same methodology, we may derive the 3-stage algorithm for M is even.

Next, we address the convergence analysis of the 3-stase iteration procedure (3.4) - (3.6).

Uniting (3.4) - (3.6), we attain

$$\boldsymbol{U}^{(s+1)} = \boldsymbol{A}_\omega \boldsymbol{U}^{(s)} + \boldsymbol{B}_\omega, \tag{3.18}$$

where

$$\boldsymbol{A}_\omega = (\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})^{-1}(\boldsymbol{P}_1 - \omega_0 \boldsymbol{I})(\boldsymbol{P}_1 + \omega_0 \boldsymbol{I})^{-1}(\boldsymbol{P}_2 - \omega_0 \boldsymbol{I}). \tag{3.19}$$

is identified as the 3-phase explicit iteration matrix.

Let $\boldsymbol{\epsilon}^{(s)} = \boldsymbol{u}^{(s)} - \boldsymbol{U}$ be the error vector defined at at s^{th} iteration. As (3.18) is a linear iteration process, the consequent error equation is given by

$$\boldsymbol{\epsilon}^{(s+1)} = \boldsymbol{A}_\omega \boldsymbol{\epsilon}^{(s)} \tag{3.20}$$

Let $\rho(\boldsymbol{A}_\omega)$ is the spectral radius of $\boldsymbol{A}_\omega$. It is required to prove that $\rho(\boldsymbol{A}_\omega) < 1$ to ensure the convergence.

Let

$$\boldsymbol{A}_\omega^* = (\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})\boldsymbol{A}_\omega(\boldsymbol{P}_2 + \boldsymbol{I}\omega_0)^{-1}$$
$$= (\boldsymbol{P}_1 - \omega_0 \boldsymbol{I})(\boldsymbol{P}_1 + \omega_0 \boldsymbol{I})^{-1}(\boldsymbol{P}_2 - \omega_0 \boldsymbol{I})(\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})^{-1}. \tag{3.21}$$

From (3.21), we observe that $\boldsymbol{A}_\omega$ is similar to $\boldsymbol{A}_\omega^*$. Thus $\rho(\boldsymbol{A}_\omega) = \rho(\boldsymbol{A}_\omega^*)$.

With the support of matrix norm property, we have the inequality

$$||\boldsymbol{A}_\omega^*||_2 \leq ||(\boldsymbol{P}_1 - \omega_0 \boldsymbol{I})(\boldsymbol{P}_1 + \omega_0 \boldsymbol{I})^{-1}||_2 . ||(\boldsymbol{P}_2 - \omega_0 \boldsymbol{I})(\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})^{-1}||_2 \tag{3.22}$$

Let $\lambda_m, m = 1(1)M$ be the M eigenvalues of $\boldsymbol{P}_1$, then $\lambda_1 = b_1$ and

$$\lambda_m{}^2 - (b_m + b_{m+1})\lambda_m + (b_m b_{m+1} - a_{m+1}c_m) = 0, m = 2(1)M - 1. \tag{3.23}$$

Moreover $R_e(\lambda_m) > 0$, this implies

$$||(\boldsymbol{P}_1 - \omega_0 \boldsymbol{I})(\boldsymbol{P}_1 + \omega_0 \boldsymbol{I})^{-1}||_2 = \max_{\lambda_m} \left|\frac{\lambda_m - \omega_0}{\lambda_m + \omega_0}\right| < 1. \tag{3.24}$$

Similarly,

$$||(\boldsymbol{P}_2 - \omega_0 \boldsymbol{I})(\boldsymbol{P}_2 + \omega_0 \boldsymbol{I})^{-1}||_2 < 1. \tag{3.25}$$

Thus from (3.22), we conclude

$$\rho(\boldsymbol{A}_\omega) = \rho(\boldsymbol{A}_\omega^*) = ||\boldsymbol{A}_\omega^*||_2 < 1. \tag{3.26}$$

Hence, the explicit iteration procedure (3.18) converges.

Whenever $g(r, t, u, u_r, u_t)$ exhibits non-linearity, the approximation (2.30) gives rise to a non-linear system, which can be handled using a 3-stage Newton iterative procedure similar to method (3.4)-(3.6) designed for linear equation described by [19].

4. Numerical Illustrations

To assess the sustainability of the recommended 3-stage iteration process, we solve the following three benchmark problems, whose analytical solutions are known. The homogenous function, boundary and initial conditions can be evaluated by relying on the exact solution as a reference for testing. For all problems we have computed ω_{opt} values and compared the performance of the suggested procedures with the SoR iteration method described by Varga [21] and other available iteration methods. We have used 5-inner iterations for computing nonlinear difference equations. Throughout the computation, we have used $\mathbf{0}$ as the initial input, and iterations continued until the error tolerance fell below or equalled 10^{-12}. All computational tasks were carried out using MATLAB programs.

Example 1: (Burger-Huxley Equation)

$$u_{rr} = u_t + uu_r + u^3 - (1+\gamma)u^2 + \gamma u,\ 0 < r < 1,\ t > 0. \tag{4.1}$$

The explicit analytical expression of the solution is given by $u(r,t) = \frac{\gamma\{1+\tanh b_0(r-b_1 t)\}}{2}$, where $b_0 = \frac{\gamma}{4}$, $b_1 = \gamma - 1$. The maximum absolute errors (MAEs), iteration counts and the value of ω_{opt} are tabulated at $t = 1$ for $\tau = 0.01, \gamma = 0.001$ in Table 1 and at different time-levels in Table 2.

Table 1

Example 1: The MAEs

M	Proposed Method			Newton AGE Method [16]			Newton SOR Method [21]			MAE
	iter	ω_{opt}	CPU time (in sec)	Iter	ω_{opt}	CPU time (in sec)	Iter	ω_{opt}	CPU time (in sec)	
10	4	1.211	0.0282	4	0.541	0.0316	6	1.123	0.0401	3.375×10^{-8}
20	4	0.781	0.0294	4	0.253	0.0338	7	1.165	0.0512	3.318×10^{-8}
30	4	0.452	0.0323	4	0.152	0.0402	8	1.261	0.0697	3.294×10^{-8}
40	5	0.384	0.0381	5	0.111	0.0471	9	1.332	0.0793	3.282×10^{-8}
60	6	0.267	0.0410	6	0.102	0.0518	10	1.407	0.1020	3.268×10^{-8}
80	8	0.206	0.0525	8	0.092	0.0661	12	1.481	0.1191	3.262×10^{-8}
100	9	0.161	0.0588	9	0.071	0.0792	13	1.544	0.1356	3.259×10^{-8}

Table 2

Example 1: The MAEs

r	t	Proposed method	Ref. [11]	Ref. [12]	Ref. [14]	Ref. [15]
0.1	0.05	7.637×10^{-9}	1.937×10^{-7}	1.874×10^{-8}	1.874×10^{-8}	7.727×10^{-9}
	0.10	1.118×10^{-8}	3.874×10^{-7}	3.748×10^{-8}	3.748×10^{-8}	1.129×10^{-8}
	1.00	1.585×10^{-8}	3.875×10^{-6}	3.748×10^{-7}	3.748×10^{-7}	1.686×10^{-8}
0.5	0.05	1.6234×10^{-8}	1.937×10^{-7}	1.8740×10^{-8}	1.873×10^{-8}	1.735×10^{-8}
	0.10	2.792×10^{-8}	3.874×10^{-7}	3.748×10^{-8}	3.747×10^{-8}	2.883×10^{-8}
	1.00	4.594×10^{-8}	3.875×10^{-6}	3.748×10^{-7}	3.721×10^{-7}	4.684×10^{-8}
0.9	0.05	7.637×10^{-9}	1.937×10^{-7}	1.874×10^{-8}	1.872×10^{-8}	7.727×10^{-9}
	0.10	1.118×10^{-8}	3.874×10^{-7}	3.748×10^{-8}	3.741×10^{-8}	1.129×10^{-8}
	1.00	1.585×10^{-8}	3.875×10^{-6}	3.748×10^{-7}	3.684×10^{-7}	1.686×10^{-8}

Example 2: (Burger-Fisher Equation)

$$u_{rr} = u_t + uu_r + u^2 - u,\ 0 < r < 1, t > 0. \tag{4.2}$$

The exact form of the analytic solution is given by $u(r,t) = \frac{1}{2}[1 + \tanh b_0(r - b_1 t)]$, where $b_0 = \frac{-1}{4}$ and $b_1 = \frac{5}{2}$. The MAEs, iteration counts and the value of ω_{opt} are tabulated in Table 3 at $t = 1$ for $\tau = 0.01$. For order of convergence (C_O), the MAEs are also listed in Table 4 at t=1 for a fixed value of τ=$1.6h^2$.

Table 3

Example 2: The MAEs

M	Proposed Method			Newton AGE Method [16]			Newton SOR Method [21]			MAE
	Iter	ω_{opt}	CPU time (in sec)	Iter	ω_{opt}	CPU time (in sec)	Iter	ω_{opt}	CPU time (in sec)	
11	04	1.207	0.0290	04	0.537	0.0327	06	1.130	0.0412	5.483×10^{-9}
21	04	0.775	0.0305	04	0.248	0.0343	07	1.171	0.0523	5.426×10^{-9}
31	05	0.447	0.0337	05	0.147	0.0411	08	1.281	0.0704	5.382×10^{-9}
41	05	0.376	0.0396	06	0.106	0.0485	10	1.344	0.0804	5.360×10^{-9}
61	06	0.258	0.0454	07	0.102	0.0525	11	1.412	0.1035	5.346×10^{-9}
81	07	0.197	0.0547	08	0.091	0.0670	12	1.491	0.1202	5.330×10^{-9}
101	08	0.156	0.0601	09	0.067	0.0801	13	1.554	0.1368	5.317×10^{-9}

Table 4

Example 2: The MAEs and C_O.

$h = 1/(M+1)$	Ref. [8]	Ref. [9]	Ref. [10]	Proposed Method	C_O
1/8	3.501×10^{-5}	3.799×10^{-5}	1.717×10^{-6}	8.873×10^{-7}	
1/16	8.473×10^{-6}	9.157×10^{-6}	1.072×10^{-7}	5.545×10^{-8}	4.0
1/32	2.085×10^{-6}	2.251×10^{-6}	6.689×10^{-9}	3.443×10^{-9}	4.0
1/64	5.115×10^{-7}	5.338×10^{-7}	4.454×10^{-10}	2.148×10^{-10}	4.0

Example 3: (Heat equation in polar form)

$$\frac{\partial^2 u}{\partial r^2}+\frac{\beta}{r}\frac{\partial u}{\partial r}-\frac{\beta}{r^2}u=\frac{\partial u}{\partial t}+g(r,t), 0<r<1, t>0. \tag{4.3}$$

The solution admits the following exact analytical representation: $u(r,t)=e^{-t}\cosh r$. The MAEs, iteration counts and the values of ω_{opt} are tabulated in Table 5 at $t=1$ for $\tau=0.01$, $\beta=1$. and for the C_O the MAEs are computed in Table 6 for $\beta=1$ and $\beta=2$ at t=1 for a constant value of $\tau=2h^2$.

Table 5

Example 3: The MAEs at $t=1$ for $\tau=0.01$, $\beta=1$

M	Proposed Method			AGE Method [16]			SOR Method [21]			MAE
	Iter	ω_{opt}	CPU time (in sec)	Iter	ω_{opt}	CPU time (in sec)	Iter	ω_{opt}	CPU time (in sec)	
10	11	0.891	0.0421	12	0.855	0.0563	14	1.073	0.0160	2.894×10^{-6}
20	20	1.521	0.0463	21	1.481	0.0847	27	1.257	0.0201	3.874×10^{-6}
30	28	2.392	0.0501	30	2.134	0.0683	38	1.398	0.0766	4.023×10^{-6}
40	37	3.122	0.0547	39	2.640	0.0774	51	1.501	0.0871	4.102×10^{-6}
60	54	4.623	0.0685	56	4.635	0.0870	74	1.626	0.1645	4.318×10^{-6}
80	72	7.012	0.0914	73	6.368	0.1140	99	1.703	0.1740	4.469×10^{-6}
100	88	7.713	0.1321	90	7.857	0.1753	120	1.753	0.1904	4.458×10^{-6}

Table 6

Example 3: The MAEs for $\beta=1$ and 2, and $\tau=2h^2$.

h	$\beta=1$		$\beta=2$	
	MAE	C_O	MAE	C_O
1/20	9.130×10^{-7}	---	1.131×10^{-6}	---
1/40	5.942×10^{-8}	3.940	7.311×10^{-8}	3.951
1/80	3.841×10^{-9}	3.951	4.648×10^{-9}	3.974
1/160	2.403×10^{-10}	3.997	2.898×10^{-10}	4.002

5. Conclusion

In this research, we have presented an efficient iterative process based on trigonometric spline method for solving 1D nonlinear PPDEs. The proposed scheme offering a compact and stable numerical framework of O(2,4). A major contribution is the introduction of a three-phase group explicit iteration algorithm, which effectively solves the resulting nonlinear difference equations at each time level. This iterative strategy ensures reliable convergence while significantly reducing computational overhead compared to conventional methods such as Newton-AGE and Newton-SOR. The method's performance has been thoroughly assessed through several benchmark problems, including the Burgers-Huxley and Burgers-Fisher equations and singular PPDEs. The numerical results consistently demonstrate the accuracy, stability, and computational efficiency of the proposed approach. Given these encouraging outcomes, the method holds strong potential for extension to higher-dimensional or coupled systems, with further improvements possible through adaptive time-stepping, hybrid solvers, or parallel implementations.

Conflict of Interest: No conflicts of interest are declared by the authors.

Data availability statement: As the study did not generate or analyze empirical datasets, there are no data to make publicly available.

References

[1] M. K. Kadalbajoo and K. C. Patidar, "Tension spline for the solution of self-adjoint singular perturbation problems," *Int. J. Comput. Math.*, vol. 79, no. 7, pp. 849–865 (2002). doi: 10.1080/0020716021000013316.

[2] R. K. Mohanty and U. Arora, "A family of non-uniform mesh tension spline methods for singularly perturbed two point singular boundary value problems with significant first derivatives," *Appl. Math. Comput.*, Vol. 172, no. 1, pp. 531-544 (2006), doi: 10.1016/j.amc.2005.02.023.

[3] J. Rashidinia and R. Mohammadi, "Tension spline approach for the numerical solution of nonlinear Klein–Gordon equation," *Comput. Phys. Comm.*, vol. 181, no. 1, pp. 78–91 (2010), doi: 10.1016/j.cpc.2009.09.007.

[4] J. Talwar, R. K. Mohanty, and S. Singh, "A new spline in compression approximation for one space dimensional quasilinear parabolic equations on a variable mesh," *Appl. Math. Comput.*, vol. 260, pp. 82–96 (2015), doi: 10.1016/j.amc.2015.03.057.

[5] K. Mittal, R. K. Ray, and R. K. Mohanty, "Two-level implicit high accuracy method in exponential form for 2D quasi-linear parabolic

equations on irrational domain," *Numer. Methods Partial Differ. Equ.*, vol. 41, no. 6 (2025), doi: 10.1002/num.70049.

[6] R. A. Obaid, M. A. Kadhim, and A. M. Abed, "A comparative study of approximate solutions to solving second-order linear complex partial differential equations," *J. Interdiscip. Math.,* vol. 29, no. 5-B, pp. 1475–1486 (2026), doi: 10.47974/JIM-2535.

[7] D. Kaya and S. M. El-Sayed, "A numerical simulation and explicit solutions of the generalized Burgers–Fisher equation," *Appl. Math. Comput.,* vol. 152, no. 2, pp. 403–413 (2004), doi: 10.1016/S0096-3003(03)00565-4.

[8] C. G. Zhu and W. S. Kang, "Numerical solution of Burgers–Fisher equation by cubic B-spline quasi-interpolation," *Appl. Math. Comput.,* vol. 216, no. 9, pp. 2679–2686 (2010), doi: 10.1016/j.amc.2010.03.113.

[9] R. Mohammadi, "Spline solution of the generalized Burgers'-Fisher equation," *Appl. Anal.,* vol. 91, no. 12, pp. 2189–2215 (2012), doi: 10.1080/00036811.2011.596479.

[10] R. C. Mittal and A. Tripathi, "Numerical solutions of generalized Burgers–Fisher and generalized Burgers–Huxley equations using collocation of cubic B-splines," *Int. J. Comput. Math.,* vol. 92, no. 5, pp. 1053–1077 (2015), doi: 10.1080/00207160.2014.920834.

[11] H. N. A. Ismail, K. Raslan, and A. A. A. Abd Rabboh, "Adomian decomposition method for Burgers–Huxley and Burgers–Fisher equations," *Appl. Math. Comput.,* vol. 159, no. 1, pp. 291–301 (2004), doi: 10.1016/j.amc.2003.10.050.

[12] B. Batiha, M. S. M. Noorani, and I. Hashim, "Application of variational iteration method to the generalized Burgers-Huxley equation," *Chaos, Solitons and Fractals*, vol. 36, pp. 660-663 (2008), doi: 10.1016/j.chaos.2006.06.080.

[13] H. O. Altaie, A. R. A. Ali, and B. N. Shihab, "Analytical solutions for nonlinear differential equations via differential transform technique," *J. Interdiscip. Math.,* vol. 28, no. 3-B, pp. 1155–1159 (2025), doi: 10.47974/JIM-2117.

[14] A. G. Bratsos, "A fourth order improved numerical scheme for the generalized Burgers–Huxley equation," *Amer. J. Comput. Math.,* vol. 1, no. 3, pp. 152–158 (2011), doi: 10.4236/ajcm.2011.13017.

[15] M. Sari, G. Gurarslan, and A. Zeytinoglu, "High-order finite difference schemes for numerical solutions of the generalized Burgers-

Huxley equation," *Numer. Meth. Partial Diff. Eq.*, vol. 27, pp. 1313-1326 (2011), doi: 10.1002/num.20585.

[16] R. K. Mohanty, "On the use of AGE algorithm with a new high accuracy Numerov-type variable mesh discretization for 1D non-linear parabolic equations," *Numer. Algorithms*, vol. 54, pp. 379–393 (2010), doi: 10.1007/s11075-009-9341-9.

[17] D. J. Evans and A. R. Ahmad, "The smart alternating group explicit (SMAGE) method," *Int. J. Comput. Math.*, vol. 65, pp. 263–271 (1997), doi: 10.1080/00207169708804615.

[18] K. S. Sukon and D. J. Evans, "Two parameter AGE (TAGE) method for the solution of a tridiagonal linear system of equations," *Int. J. Comput. Math.*, vol. 60, pp. 265–278 (1996), doi: 10.1080/00207169608804491.

[19] R. K. Mohanty and M. K. Jain, "High accuracy cubic spline alternating group explicit methods for 1D quasi-linear parabolic equations," *Int. J. Comput. Math.*, vol. 86, pp. 1556–1571 (2009), doi: 10.1080/00207160801923049.

[20] A. A. Dahalan, M. S. Muthuvalu, E. Aruchunan, J. Sulaiman, and W. R. W. Din, "Application of implicit scheme with AGE iterative method for solving fuzzy parabolic equation," *AIP Conf. Proc.*, vol. 1830, Art. no. 020001 (2017), doi: 10.1063/1.4980864.

[21] R. S. Varga, *Matrix iterative analysis*. Englewood Cliffs, NJ: Prentice-Hall Inc. (1962).

Received December, 2025